

---

# BAZE DE DATE

---

Autori: Prof.univ.dr. LUNGU Ion

Asist.univ.drd. BOTHA Iuliana

## CUPRINS

- ✚ Unitatea de învățare 1: ORGANIZAREA DATELOR ÎN MEMORIA EXTERNĂ
- ✚ Unitatea de învățare 2: BAZE DE DATE – NOȚIUNI FUNDAMENTALE
- ✚ Unitatea de învățare 3: BAZE DE DATE RELAȚIONALE (BDR)
- ✚ Unitatea de învățare 4: METODOLOGIA DE REALIZARE A BAZELOR DE DATE
- ✚ Unitatea de învățare 5: BAZE DE DATE ORIENTATE OBIECT ( BDOO)

## Introducere

Cursul de **Baze de date** se adresează studenților înscriși la programul de studiu ID, organizat de facultatea de Cibernetică, Statistică și Informatică Economică și face parte din planul de învățământ aferent secției de Informatică Economică, anul 2, semestrul 1.



**OBIECTIVUL GENERAL** al cursului de **Baze de date** vizează pregătirea studenților în domeniul proiectării bazelor de date și al utilizării acestora în diferite domenii economico-sociale.

**OBIECTIVELE SPECIFICE PRINCIPALE** ale acestui curs, concretizate în competențele dobândite după parcurgerea și asimilarea lui sunt următoarele:

- ✚ familiarizarea cu posibilitățile de utilizare ale bazelor de date indiferent de tipul acestora;
- ✚ dobândirea abilității de a proiecta baze de date relaționale;
- ✚ familiarizarea cu principalele caracteristici ale bazelor de date orientate obiect.

Pentru o bună înțelegere a noțiunilor teoretice și practice prezentate în acest curs, este necesară parcurgerea anterioară a disciplinelor *Structuri de date*, *Programarea calculatoarelor*, *Sisteme de operare*.

Cursul de **Baze de date** este *structurat* în 5 unități de învățare (capitole), fiecare dintre acestea cuprinzând câte o lucrare de verificare, care va fi transmisă tutorelui alocat, precum și într-un număr de 14 laboratoare (activități asistate) la care prezența va fi obligatorie.

Pentru ca procesul de instruire să se desfășoare într-un mod riguros, dar și atractiv, se poate utiliza un set de *resurse suplimentare* în format multimedia.

**EVALUAREA CUNOȘTINȚELOR**, respectiv stabilirea notei finale, se va realiza în felul următor:

- evaluarea finală – 50%
- evaluarea continuă – testele de la sfârșitul fiecărei unități de învățare, precum și un proiect la seminar – 50%

*Notă.* Structura obligatorie pentru proiect va fi:

- 1 pagina care să conțină: Tema, Descrierea problemei, Schema conceptuală a BD;
- comenzi SQL, care să acopere întreaga materie parcursă.

## Unitatea de învățare 1: ORGANIZAREA DATELOR ÎN MEMORIA EXTERNĂ

---

### Cuprins

- 1.1. Obiective
- 1.2. Evoluția organizării datelor
- 1.3. Organizarea datelor în fișiere
- 1.4. Modele de structurare a datelor în baze de date
- 1.5. Protecția bazelor de date
- 1.6. Rezumat
- 1.7. Teste de autoevaluare
- 1.8. Răspunsuri și comentarii la testele de autoevaluare
- 1.9. Bibliografia unității de învățare 1

### 1.1. Obiective



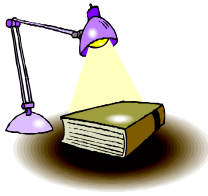
După studiul acestei unități de învățare vei avea cunoștințe despre:

- *necesitatea organizării datelor în memoria externă, comparativ cu memoria internă;*
- *evoluția organizării datelor pornind de la fișiere (primul mod de organizare a datelor în memoria externă) și ajungându-se la ultimele tipuri de baze de date, cele orientate obiect;*
- *noțiunea de fișier și cele mai importante aspecte de organizare a datelor: operații, structura, modul de organizare, modul de acces;*
- *organizarea datelor în baze de date conform modelelor logice de date fundamentale: ierarhic, rețea, relațional, orientat obiect;*
- *protecția bazelor de date, tratată prin cele două aspecte ale sale: securitatea, integritatea.*

*Durata medie a unei unități de studiu individual – 4 ore*



## 1.2. Evoluția organizării datelor



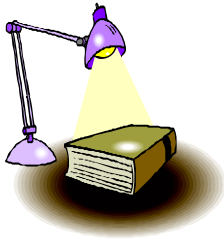
- Organizarea datelor înseamnă definirea și structurarea datelor în memoria internă sau în memoria externă (în colecții) și stabilirea legăturilor între date, conform unui model de date.
- Sintetizând, evoluția organizării datelor în memoria externă a avut în vedere câteva aspecte

Aspecte Etape(1)	Mod de organizare a datelor (2)	Structura de date (3)	Mod de prelucrare (4)	Redundanță (5)	Software utilizat (6)
1. Înainte de 1965	Fișiere secvențiale	Logică coincide cu fizică	Pe loturi (batch)	Mare, necontrolat	Operații simple de I/E (limbaje asamblare și universal)
2. Anii '60	Fișiere secvențiale, indexate, directe	Logică și fizică	Loturi, on-line	Mare, necontrolat	Chei simple de acces (limbaje universale)
3. Anii '70	Baze de date arborescente, rețea	Logică, fizică, conceptuală	Loturi, conversațional	Scade, controlat	Chei multiple de acces, legături între date, protecția (SGBD)
4. Sfârșitul anilor '70 până acum	Baze de date relaționale	Logică, fizică, conceptuală	Conversațional, interactiv	Mică, controlat	Limbaje de regăsire, protecție, concurență (SGBD)
5. Sfârșitul anilor '80 până acum	Baze de date orientate obiect	Logică, fizică, conceptuală	Interactiv	Minimă, controlat	Limbaje din programarea OO (SGBD)

*Notă. Necesitatea organizării datelor în memoria externă rezultă analizând câteva criterii de comparație dintre memoria internă și cea externă:*

CRITERIU	MEM. INTERNĂ	MEM. EXTERNĂ
Cost	Mare	Mic
Viteză	Mare	Mică
Capacitate(volum date)	Mică	Mare
Persistență	Nu	Da
Organizare date	Variabile, constante, masive, pointeri etc.	Fișiere, baze de date

### 1.3. Organizarea datelor în fișiere



- *Fisierul* este o colecție organizată de date unite după criterii comune calitative, de prelucrare și scop.
- Toate limbajele de programare universale lucrează cu această noțiune, pentru organizarea datelor în memoria externă. Sistemele de bază de date lucrează cu această noțiune și în plus o dezvoltă în noțiunea de bază de date.

Prezentăm, în continuare, câteva noțiuni fundamentale utilizate în organizarea datelor în fișiere. De aceste noțiuni se ține seama și la baze de date.

1. Caracteristici ale unui fișier:
  - *actualizarea* se referă la trei operații : adaugarea, modificarea, ștergerea de înregistrări;
  - *natura* datelor din fișier trebuie să fie omogenă (să se refere la aceeași entitate din lumea reală);
  - *prelucrarea* datelor din fișier se referă la tipul și frecvența operațiilor efectuate pe înregistrări;
  - *volumul* de date din fișier (se măsoară în număr de octeți).
2. Structura unui fișier:
  - *partea de identificare* este dată de etichetele plasate la începutul și sfârșitul fișierului;
  - *partea de date* este colecția omogenă de date ce aparțin aceleiași entități din lumea reală structurată astfel:  
fișier → înregistrări → câmpuri → valori.
3. Modul de organizare reprezintă modul de dispunere a înregistrărilor pe suportul fizic și presupune reguli de memorare a datelor.

*Categoriile* de moduri de organizare a fișierelor:

#### a) **Standard**

- Este cea mai veche și există pe toate tipurile de calculatoare.
- Înregistrarea este formată dintr-un șir de caractere dispus pe o linie acceptată de periferic.
- Toate limbajele recunosc fișiere standard de intrare și ieșire.

#### b) **Clasică** (elementară)

- Organizarea se face pe medii magnetice sau optice.

– *Tipuri:*

##### ➤ SECVENȚIALĂ

- Înregistrările sunt dispuse în fișier una după alta fără nici o ordine prestabilită.
- Localizarea unei înregistrări se face prin parcurgerea tuturor înregistrărilor anterioare ei (secvențial).
- Toate sistemele de operare și limbajele de programare acceptă organizarea secvențială.

##### ➤ RELATIVĂ

- Înregistrările sunt dispuse în fișier una după alta și numerotate (de către sistem) de la 0 sau 1 la câte sunt (număr de realizare).
- Localizarea unei înregistrări se poate face secvențial sau direct prin numărul de realizare.
- INDEXAT-SECVENȚIALĂ
  - Înregistrările sunt dispuse în fișier în ordine strict crescătoare după o cheie (face parte din înregistrare).
  - Cheia este unul sau mai multe câmpuri care identifică în mod unic o înregistrare.
  - Fișierului îi este atașat o tabelă de indecși care face legătura între valoarea cheii și adresa fizică a înregistrării.
  - Localizarea unei înregistrări se poate face secvențial dar și direct prin cheie:
    - se compară cheia înregistrării căutate cu indecșii din tabela de index și se localizează direct partea fizică a fișierului în care se află înregistrarea căutată;
    - în partea fizică localizată se face o căutare secvențială a înregistrării dorite.
- c) Specială (complexă)**
  - Se bazează pe modurile de organizare clasice.
  - Sunt utilizate în baze de date și în sisteme de fișiere.
  - *Tipuri (câteva):*
    - PARTIȚIONAREA
      - Înregistrările din fișier sunt grupate în partiții sub un nume.
      - În cadrul unei partiții înregistrările sunt organizate secvențial.
      - Se utilizează pentru bibliotecile de programe
    - MULTIINDEXAREA
      - Este o extindere a indexării prin utilizarea mai multor chei alese de programator.
      - Spațiul ocupat este mai mare.
      - Se utilizează pentru fișiere care necesită regăsiri intense multicriteriale.
    - INVERSĂ
      - Presupune existența a două fișiere: de bază și invers.
      - Fișierul de bază conține datele propriu-zise și are organizare secvențială. El este fișierul în care se caută.
      - Fișierul invers este construit din cel de bază (printr-o tehnică de inversare) și are organizare relativă. El este fișierul prin intermediul căruia se caută.
      - Spațiul ocupat necesar este cam de 3,5 ori mai mare față de cât ocupă fișierul de bază.

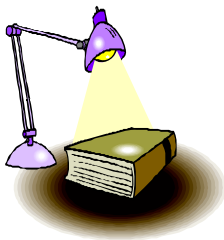
4) Modul de acces reprezintă modul în care se determină locul ocupat de o înregistrare într-un fișier și depinde de modul de organizare.

*Tipuri de moduri de acces pentru fișiere:*

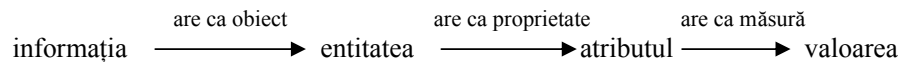
- SECVENȚIAL presupune că localizarea unei înregistrări se face prin parcurgerea tuturor înregistrărilor care o preced.
  - Este permis accesul secvențial pentru toate tipurile de fișiere.

- Se recomandă pentru fișierele din care sunt necesare, la o prelucrare, peste 50% din numărul total de înregistrări.
- Pentru optimizare se recomandă ordonarea fișierului.
- DIRECT presupune că localizarea unei înregistrări se face cu ajutorul unei chei definite de programator.
- DINAMIC presupune că la o singură deschidere de fișier se pot localiza, alternativ și repetat, înregistrări în acces secvențial și direct.

## 1.4. Modele de structurare a datelor în BD



- *Informația*, care se reprezintă în calculator în memoria internă sau externă, se poate defini structural după schema:



- *Modelul de structură* se referă la descrierea tuturor atributelor unei entități în interdependență.
- Valorile atributelor se materializează prin *date*, care dau o reprezentare simbolică a informațiilor.
- *Modelul de date* este ansamblul de concepte și instrumente necesar pentru a realiza structura colecțiilor de date (schema) (fig. 2.1.).

Modelul de date este *compus* din:

- concepte;
- un formalism pentru a descrie datele (structura de date);
- un ansamblu de operatori pentru a le manipula (datele).

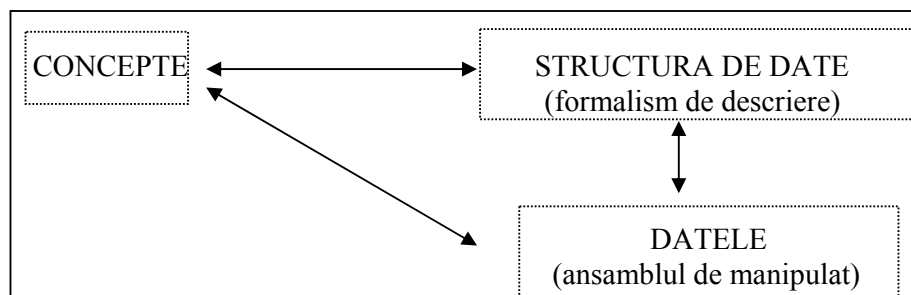


Fig. 1.1. Modelul de date

În literatura de specialitate [RICC01], [RAGE00] sunt prezentate trei **tipuri de modele** de date pentru baze de date. Prezentăm, pe scurt, aceste tipuri, împreună cu câteva caracteristici pentru fiecare:

- *modelul conceptual* (la analiză)
  - descrie sistemul în termeni pe înțelesul utilizatorului;

- se folosește pentru specificația inițială a datelor și pentru comunicarea cu utilizatorii;
- este un mod de stabilire a unei legături între dezvoltatori și utilizatori.
- *modelul logic* (la proiectare)
  - specifică structura bazei de date (colecțiile de date și legăturile dintre ele);
  - se scrie într-o formă care să poată fi folosită pentru crearea bazei de date;
  - se folosește pentru definirea și întreținerea bazei de date de către SGBD și pentru formularea cerințelor de regăsire de către utilizatori.
- *modelul fizic* (sarcina SGBD)
  - descrie modul în care modelul logic al datelor va fi reprezentat la stocare, în memoria externă (fișiere, indecși, discuri etc.);
  - de obicei, este generat automat, de către SGBD, pornind de la modelul logic.

**Schema** [*conceptuală*] reprezintă descrierea fenomenelor din realitatea înconjurătoare prin entități și atribute (tipurile de date), împreună cu toate corelațiile (legăturile) dintre ele (constrângerile).

Definirea schemei este o activitate de *modelare* pentru că traduce în termeni abstracți entitățile lumii reale.

Schema BD *se reprezintă* cu ajutorul unui model de date implementat prin intermediul unui SGBD adecvat.

Trebuie să se facă *distincție clară* între structura bazei de date (schema, de exemplu: *stud(cod:N,4; nume:C,15)* ) și conținutul ei (instanța, de exemplu: *22 Nedelcu Anda*). Schema și instanțele sunt stocate în baza de date, și anume prima în dicționar, iar cea de-a doua în datele propriu-zise.

În cele ce urmează ne vom ocupa, în cea mai mare parte, de modelul logic de date pentru baze de date.

**Elementele** (componentele) oricărui model de date pentru baze de date sunt:

**1. Definirea structurii** modelului (partea structurală):

- definirea entităților și a atributelor asociate;
- definirea legăturilor (asocierea) dintre entități.

Asocierea poate fi de *tipul* :

- unu la unu (1:1)
- unu la mulți (1:M)
- mulți la mulți (M:N)
- neprecizat explicit

Definirea structurii de date se face cu un LDD (Limbaaj de Descriere a Datelor) dintr-un SGBD.

**2. Operatorii modelului** (partea de manipulare) care acționează asupra structurilor de date, dar și asupra datelor, pentru operații de prelucrare (compunere, actualizare etc.).

Operatorii se implementează cu ajutorul unui LMD (Limbaaj de Manipulare a Datelor) dintr-un SGBD.

**3. Regulile de integritate** (partea de coerență) sunt restricții stabilite la descrierea datelor, care le asigură acestora menținerea corectitudinii și dau logica modelului.

Restricțiile se implementează cu un LDD din SGBD.

**Tipuri** de modele logice de date pentru BD:

- *fundamentale*: ierarhice, rețea, relaționale, orientate obiect;



- *derivate* (extinse din cele fundamentale): distribuite, multimedia etc.

Prezentăm, în continuare, principalele caracteristici pentru modelele de date fundamentare (de bază) pentru baze de date.

## I. Modelul ierarhic

### 1. Definirea structurii modelului ierarhic.

a) **Definirea entităților** se face prin noțiune de *tip de înregistrare* (clasă de entități), care este formată din *caracteristici* (câmpuri).

*Realizarea* (instanța) unui tip de înregistrare este dată de ansamblul valorilor pentru câmpurile acesteia (înregistrarea).

b) **Definirea legăturilor** dintre entități se face *fizic* și conduce la structura de *tip ierarhic* (arborescent) reprezentată sub forma unei *diagrame* (fig. 2.2.).

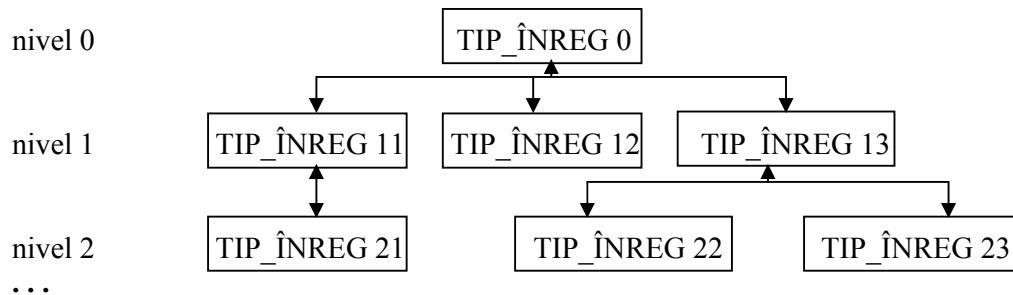


Fig. 1.2. Diagrama de structură ierarhică

### Caracteristici ale structurii ierarhice (arborescente):

- Fiecare nod corespunde unui tip de înregistrare și fiecare drum corespunde unei legături (asocieri).
- Orice acces la un nod se face prin vârful ierarhiei, numit *rădăcină*, pe o singură cale.
- Un nod subordonat (copil) nu poate avea decât un singur superior (părinte).
- Un superior poate avea unul sau mai mulți subordonați.
- Legătura copil-părinte este doar de tip 1:1 (la o realizare copil corespunde o singură realizare părinte).
- Legătura părinte-copil poate fi de tip 1:1 sau 1:M.
- În structură există un singur nod rădăcină și unul sau mai multe noduri dependente situate pe unul sau mai multe niveluri.
  - ierarhie de tipuri de înregistrări se numește *tip arbore*.
  - *realizare* a unui tip arbore este formată dintr-o singură realizare a tipului de înregistrare rădăcină împreună cu o mulțime ordonată formată din una sau mai multe realizări ale fiecărui tip de înregistrare de pe nivelurile inferioare.
- Ordonarea realizărilor dintr-un arbore conduce la o *secvență ierarhică*.

### 2. Operatorii modelului ierarhic

- *Localizarea unui arbore* în BD: se localizează o realizare a tipului de înregistrare rădăcină.
- *Trecerea de la un arbore la altul* în BD: se trece de la o realizare a tipului de înregistrare rădăcină (secvență ierarhică) la o altă realizare a aceluiași tip de înregistrare rădăcină.

- *Trecerea de la o realizare* (înregistrare) *la alta* într-un arbore (secvență ierarhică) : se poate face trecerea pe același nivel sau pe niveluri diferite între tipuri de înregistrări legate între ele.
- *Actualizarea într-un arbore* : adăugarea, modificarea sau ștergerea unei înregistrări. Operația este greoaie și consumatoare de resurse calculator (spațiu și timp) pentru că antrenează automat toate înregistrările din arbore care se înlănțuie cu înregistrarea actualizată.

*Notă.* Operatorii de mai sus sunt la nivel de înregistrare; acționează pe o înregistrare și produc tot o înregistrare.

Există și operatori la nivel de mulțime de înregistrări care se implementează în LMD mult mai greu.

### 3. Restricțiile de integritate ale modelului ierarhic.

- Realizarea subordonat este totdeauna asociată unei singure realizări superior.
- Dacă un tip de înregistrare nu are realizări atunci nici tipurile înregistrări descendente nu au realizări.

*Notă.* Efectele restricțiilor de integritate sunt:

- complicarea operației de actualizare;
- dacă o realizare a unui nod subordonat trebuie să fie asociată cu mai multe realizări ale nodului părinte atunci ea trebuie multiplicată (crește redundanța);
- dau logica și corența modelului arborescent.

### ***Caracterizarea generală a modelului ierarhic.***

1. Modelul a fost *propus* de către IBM și a fost primul utilizat pentru BD.
2. *Structura* modelului este simplă (graf orientat) familiară specialiștilor în informatică.
3. *Implementarea* modelului se face fizic și condiționează performanțele BD:
  - prin pointeri, utilizând diferite metode (liste simplu înlănțuite, vectori de pointeri, chei primare etc.);
  - secvențial, fiecare realizare a unui arbore corespunde unei înregistrări (logice) dintr-un fișier secvențial.
4. *Aplicabilitatea* modelului se regăsește cu succes în tehnologia construcțiilor de mașini, dar și în alte domenii.
5. *Limitele* modelului ierarhic:
  - numărul de ierarhii posibile crește combinatoric cu numărul înregistrărilor, în legătura părinte-copil;
  - actualizarea datelor este greoaie și consumatoare de resurse calculator;
  - nivelul logic nu este separat clar de cel fizic (exemplu: indecșii pot fi structuri logice dar și fizice);
  - nu se poate realiza legătura de tip M:N
6. *Exemplu* de SGBD ierarhic este IMS (Information Management System) realizat de către IBM.
7. *Baza de date ierarhică* este o mulțime ordonată de realizări ale unui tip arbore.

## **II. Modelul rețea**

### 1. Definirea structurii modelului rețea.

- a) **Definirea entităților** se face prin noțiunea de *tip de înregistrare*, care este formată din *caracteristici* (câmpuri).

*Realizarea* (instanța) unui tip de înregistrare este dată de ansamblul valorilor pentru câmpurile acestuia (înregistrarea).

b) **Definirea legăturilor** dintre entități se face *fizic* și conduce la o structură de tip *rețea*, reprezentată sub forma unei *diagrame* (fig. 2.3.) numită și *schema* (conceptuală).

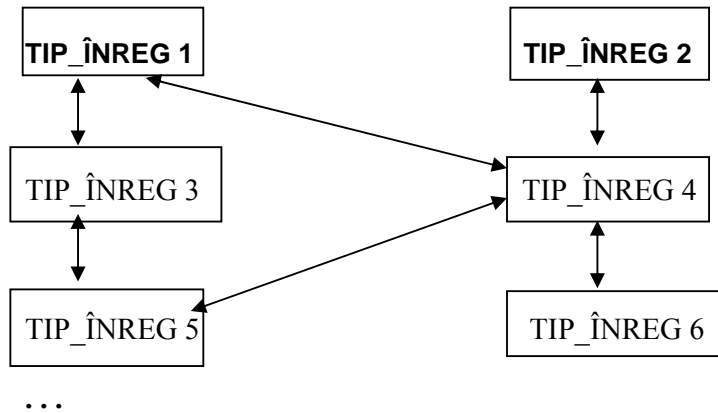


Fig. 1.3. Diagrama de structură rețea

#### Caracteristici ale structurii rețea :

- Un nod poate avea oricâți superiori și oricâți subordonați.
- Pot exista mai multe tip de înregistrare rădăcină.
- La un subordonat se poate ajunge pe mai multe căi.
- Este o extensie a structurii arborescente în care graful nu mai este limitat.
- Legăturile în model pot fi 1:1, 1:M, M:N, ciclice.
- Pentru exprimarea legăturilor se folosește conceptul de *tip set*. Un tip set este format dintr-un singur tip de *nod proprietar* și unul sau mai multe tipuri de *noduri membre* dependente de cel proprietar.
- *Realizarea* tipului set este o colecție de înregistrări care are o realizare proprietar și un număr de realizări membre asociate.

#### 2. Operatorii modelului rețea.

- *Localizarea unui tip set*: se identifică o realizare a unui nod proprietar.
- *Trecerea* de la o înregistrare la alta în cadrul unui set: proprietar-membru, membru-proprietar, membru-membru.
- *Actualizarea într-un set*: adăugarea, modificarea, ștergerea unor membri.
- *Actualizarea într-o rețea*: adăugarea, modificarea, ștergerea unor seturi.  
*Notă.* Operațiile de actualizare sunt greoaie și mari consumatoare de resurse calculator. Aceasta, deoarece sunt antrenate întotdeauna înregistrările dintr-unul sau mai multe seturi.
- *Localizarea unei înregistrări* pe baza valorii unui câmp (cheie).

#### 3. Restricțiile de integritate ale modelului rețea.

- O înregistrare nu poate fi membră a două realizări ale aceluiași tip set. Înregistrarea se va multiplica.
- O înregistrare poate să aparțină mai multor tipuri set prin multiplicare.
- Un tip de înregistrare poate fi nod proprietar într-un set și nod membru în alt set.
- Un set poate avea un singur nod proprietar.

*Notă.* Efectele restricțiilor de integritate sunt:

- dau logica și coerența modelului rețea;
- complică operația de actualizare;
- introduce o redundanță controlată.

***Caracterizarea generală a modelului rețea.***

1. A fost *propus* de CODASYL ca soluție pentru a se elimina limitele modelului ierarhic.
2. Se poate utiliza pentru domenii variate din lumea reală, deoarece permite reprezentarea unor *structuri complexe*.
3. *Implementarea* modelului se face fizic și condiționează performanțele BD:
  - prin pointeri structurați în liste înălțuite, de diferite tipuri (owner, prior, next);
  - prin hartă de biți (matrice cu înregistrări, iar la intersecție sunt legăturile: 1-da, 0-nu).
4. *Limitele* modelului rețea:
  - complexitatea modelului îl face dificil de implementat;
  - legăturile din model nu sunt întotdeauna foarte clare;
  - prelucrarea înregistrărilor se face secvențial;
  - actualizarea este greoaie și consumatoare de resurse calculator;
  - reproiectarea este dificilă datorită complexității ridicate.
5. *Exemple* de SGBD rețea : IDMS (Integrated Database Management System), Socrate.
6. *Baza de date rețea* este o mulțime oarecare de tipuri de înregistrări structurate pe tipuri set.

### III. Modelul relațional

1. *Definirea structurii modelului relațional.*

**a) Definirea entităților** se face sub forma unor tablouri bidimensionale numite *tabele* sau *relații* de date.

*Conceptele* utilizate sunt:

- *domeniu* este un ansamblu de valori caracterizat printr-un nume. el poate fi explicit (se enumera valorile posibile, de exemplu  $D1: \{m, f\}$ ) sau implicit (se precizeaza proprietatile valorilor, de exemplu  $D1: \{a/a \in N\}$ ).
  - *tabela/relația* este un subansamblu al produsului cartezian al mai multor domenii, caracterizat printr-un nume.
  - *atributul* este coloana unei tabele, caracterizata printr-un nume.
  - *tuplul* este linia dintr-o tabelă și nu are nume.
- Notă.* ordinea liniilor (tupluri) și coloanelor (attribute) dintr-o tabelă nu trebuie să prezinte nici-o importanță.
- *schema tablei* este numele tablei, urmat între paranteze rotunde de lista atributelor, iar pentru fiecare atribut se precizeaza domeniul asociat.
  - *cheia* este un atribut sau un ansamblu de attribute care au rolul de a identifica un tuplu dintr-o tabela. *tipuri* de chei: primare/alternate, simple/comune, externe.
  - *schema relațională* este schema tablei + cheile + restricțiile de integritate.

**Exemplu.** Fie tabela STUDENT cu attributele: NUME din domeniul D1 (numele de persoane), ANSTUDIU din domeniul D2(anii de studiu dintr-o facultate), ANNAȘTERE din domeniul D3(anii calendaristici).

STUDENT

NUME: D1	ANSTUDIU : D2	ANNAȘTERE: D3
POPA A.	2	1982
PETRE M.	2	1982
ILIE C.	3	1981

Schema tabelii : STUDENT(NUME: D1, ANSTUDIU: D2, ANNAȘTERE: D3).

*Notă.* Numărul de domenii este mai mic sau egal cu numărul de atribute pentru o tabelă (mai multe atribute pot lua valori din același domeniu).

**b) Definirea legăturilor** dintre entități se face *logic* construind asocieri între tabele cu ajutorul unor atribute de legătură.

Legăturile se pot reprezenta sub forma unei *diagrame de structură* (fig. 2.4.) numită și *schema BD*.

**Exemplu.** Fie o BD privind desfacerea în care am identificat tabelele: BENI, CONTR, PROD. Legăturile dintre tabele sunt:

- un beneficiar poate încheia mai multe contracte (1:M);
- un produs se poate livra prin mai multe contracte (1:M);
- un beneficiar poate cumpăra mai multe produse și un produs se poate livra către mai mulți beneficiari (M:N).

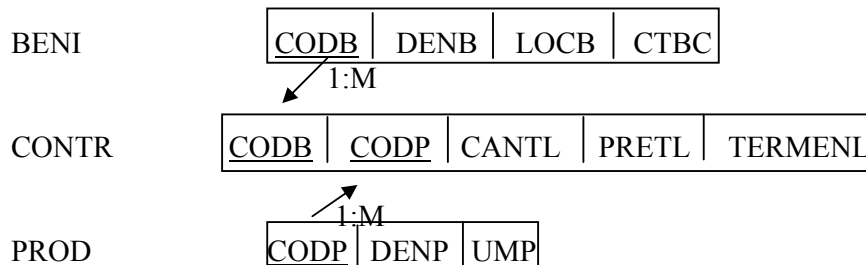


Fig. 1.4. Schema BD

*Notă.* Legătura M:N nu apare direct în schemă.

Atributele subliniate sunt chei.

**Caracteristici ale structurii relaționale:**

- Atributele implicate în realizarea legăturilor se găsesc fie în tabelele asociate, fie în tabele distincte construite special pentru legături.
- Atributul din tabela inițială se numește *cheie externă* iar cel din tabela finală este *cheie primară*.
- Legăturile posibile sunt 1:1, 1:M, M:N.
- Potențial, orice tabelă se poate lega cu orice tabelă, după orice atribute.
- Legăturile se stabilesc la momentul descrierii datelor (prin LDD) cu ajutorul restricțiilor de integritate.

Practic, se stabilesc și legături dinamice la momentul execuției.

1. Operatorii modelului relațional.

a) Operatori din **algebra relațională**:

- standard: selecția proiecția, joncțiunea, reuniunea, diferența, produsul cartezian, intersecția, diviziunea;

- extensii : complementarea, descompunerea (splitarea) etc.

*Notă.* Operatorii algebrei relaționale acționează la nivel de tabelă în expresii relaționale. Operanții sunt tabele iar rezultatul este întotdeauna o tabelă.

b) Operatorii din **calculul relațional**:

- orientați pe tuplu: conectivele (conjuncția  $\wedge$ , disjuncția  $\vee$ , negația  $\neg$ ), cuantificatorii (existențial  $\exists$ , universal  $\forall$ );
- orientați pe domeniu: idem ca mai sus.

*Notă.* Pentru operatorii calculului relațional operandul poate fi tuplu sau domeniu.

2. Restricțiile de integritate ale modelului relațional.

a) **Structurale** sunt cele care se definesc prin compararea unor valori din tabele:

- *cheie unică*: într-o tabelă nu trebuie să existe mai multe tupluri cu aceeași valoare pentru ansamblul cheie;
- *referențială*: într-o tabelă T1 care referă o tabelă T2, valorile cheii externe trebuie să figureze printre valorile cheii primare din T2 sau să ia valoarea NULL (neprecizat);
- *entității*: într-o tabelă, atributele din cheia primară nu trebuie să ia valoarea NULL.

*Notă.* Cele trei restricții de mai sus sunt *minimale*.

Pe lângă acestea, există o serie de alte restricții structurale care se referă la dependențele dintre date: funcționale, multivaloare, joncțiune etc. (sunt luate în considerare la tehnicile de proiectare BD relaționale).

b) **Semantice** sunt cele care se definesc prin comportamentul datelor și țin cont de valorile din BD:

- restricția *de domeniu*: domeniul corespunzător unui atribut dintr-o tabelă trebuie să se încadreze între anumite valori;
- restricții *temporare*: valorile anumitor atribute se compară cu niște valori temporare (rezultate din calcule etc.).

*Notă.* Restricțiile semantice fiind foarte generale se gestionează fie la momentul descrierii datelor (de exemplu prin clauza CHECK), fie în afara modelului la momentul execuției (de exemplu prin instrucțiunea IF) .

### ***Caracterizarea generală a modelului relațional.***

1. A fost *propus* de către IBM și a revoluționat reprezentarea datelor în BD făcând trecerea la o nouă generație (a doua).

2. Modelul este *simplu*, are o solidă *fundamentare teoretică* fiind bazat pe teoria seturilor (ansamblurilor) și pe logica matematică.

3. Pot fi reprezentate toate tipurile de structuri de date de mare complexitate, din *diferite domenii* de activitate.

4. *Implementarea* modelului se face logic prin atribute având rol de chei.

5. *Limitele* modelului relațional:

- prea marea simplitate a modelului îl face dificil de aplicat pentru noile tipuri de aplicații (multimedia, internet etc.);
- nu asigură o independență logică totală a datelor de aplicație;
- pentru aplicații de volum și complexitate foarte mari nu mai face față;
- poate introduce o redundanță prea mare (la proiectare prin tehnica de normalizare).

6. *Baza de date relațională* este un ansamblu de tabele prin care se reprezintă atât datele cât și legăturile dintre ele.

#### IV. Modelul orientat obiect

##### 1. Definirea structurii modelului orientat obiect (OO).

a) **Definirea obiectelor** se face cu ajutorul conceptului de *clasă de obiecte* care este definită din entitatea regăsită în lumea reală. Se pune accentul atât pe date cât și pe comportamentul acestora, ambele încapsulate în obiect.

Conceptele utilizate sunt:

- *clasele (tipurile) de obiecte* sunt un tip abstract de date prin care se definește structura obiectelor (*proprietățile*) și comportamentul (*metodele*) acestora.
  - *obiectele* reprezintă o colecție de proprietăți care se referă la aceeași entitate. obiectul are:
    - un nume prin care este referit ;
    - un identificator unic atribuit de sistem;
    - implementare care este privată;
    - interfață care este publică.
  - *metoda* reprezintă operațiile permise asupra obiectului, deci comportamentul (funcționalitatea) acestuia.
  - *mesajul* reprezintă cereri adresate obiectelor pentru a returna o valoare sau o stare.
  - *caracteristici* (principii) fundamentale (de baza) ale obiectelor:
    - *încapsurarea* : descrierea obiectelor se face astfel încât nu se poate avea acces din afara obiectului la datele sale;
    - *polimorfismul* : diferite obiecte pot răspunde diferit la aceleași mesaje;
    - *moștenirea* : capacitatea unui obiect de a-și deriva datele și funcționalitatea din alt obiect.
  - *instanța* unei clase reprezintă realizarea unei clase, dată de valorile variabilelor aferente.
- b) **Definirea legăturilor** între obiecte se realizează implicit prin modul de construire (definire) al obiectelor.

*Tipurile de legături* în modelul OO sunt:

- *ierarhice*, caracterizate prin:
  - clasa de obiecte este structura de bază a modelului;
  - fiecare obiect are un identificator unic;
  - toate obiectele sunt membri ai unei clase;
  - clasele sunt structurate în ierarhii având caracteristica de moștenire;
  - prin obiecte se pot defini orice tip de date (text, grafic, imagine, sunet, video etc.);
  - ansamblul claselor de obiecte structurate în ierarhii alcătuiesc schema BD.
- *de referință* caracterizate prin:
  - se realizează pe baza identificatorului unic de obiect;
  - pot fi de următoarele *feluri*:
    - simple de asociere : referirea unui obiect de către alt obiect;
    - de compunere (tip parte-întreg) : obiectele care reprezintă componente ale unui întreg sunt asociate cu obiectul ce reprezintă întregul;
    - de agregare : obiectele independente sunt agregate succesiv pentru a forma un întreg.
  - permit definirea și manipularea de obiecte compuse din alte obiecte. Obiectele compuse rezultate au o structură ierarhică dar nu au caracteristică de moștenire.

##### 2. Operatorii modelului OO:

- La baza operațiilor din model stau mesajele ca unic mod de a *comunica* obiectele între ele.

- *Actualizarea metodelor* : adaugare, modificare, ștergere de metode.
- *Actualizarea proprietăților* : adaugare, modificare, ștergere de date.
- *Actualizarea claselor* : adaugare, modificare, ștergere de clase.
- Realizarea *legăturilor între clase* : compunere, partiționare etc..
- *Actualizarea instanțelor* : prin metode care schimbă starea internă a obiectului.

3. Restricțiile de integritate ale modelului OO:

- Orice obiect trebuie să respecte restricțiile impuse la definirea clasei din care face parte (protocol de obiect).
- Identificatorul obiectului asigură integritatea referirii la el (se atribuie și se șterge automat o dată cu obiectul).
- Accesul la obiecte este limitat la folosirea protocolului de mesaje definit pentru clasa din care face parte obiectul.

***Caracterizarea generală a modelului OO.***

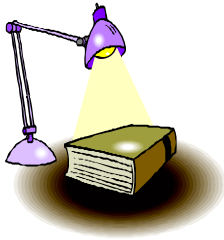
1. Modelul OO marchează trecerea la o *a treia generație* de BD. El aduce bazelor de date un plus de deschidere, independența logică a datelor de aplicații, reutilizarea obiectelor etc.
2. Modelul OO dă bune rezultate pentru probleme foarte *mari și complexe* (principiile complexității sunt incluse în model).
3. În structură sunt acceptate toate tipurile de date cunoscute, deci se poate aplica pentru *toate domeniile* de activitate.
4. *Limitele* modelului OO:
  - nu există un model OO unanim acceptat de specialiști. Există totuși organisme internaționale de standardizare – exemplu OMG (Object Management Group) – precum și modele OO standardizate – exemplu ODMG (Object Data Management Group);
  - imaturitatea și naturalețea dezarmantă a tehnologiei OO;
  - elemente încă insuficient testate pentru SBDOO: controlul integrității, volume foarte mari, regăsirea etc.
5. *Exemple* de SGBDOO: O2 (Franța), Jasmine (SUA) etc.
6. *Baza de date orientată obiect* este o mulțime de clase de obiecte persistente (în memoria externă), organizată coerent și ordonată în ierarhii, partajată pentru utilizatorii concurenți.

**Note**

- 1) O *comparație* între modelul relațional și orientat obiect vezi în Anexa 1.
- 2) După prezentarea de până acum a primelor două capitole, putem da o **definiție completă și explicativă** a noțiunii de *bază de date*, ca fiind un ansamblu de colecții de date:
  - *organizat* , pe niveluri de organizare a datelor (conceptual, logic, fizic), așa cum reiese din arhitectura unui SBD pe niveluri;
  - *coerent* , conform restricțiilor de integritate și a legăturilor dintre date, care rezultă din modelul logic de date aferent;
  - *structurat* , conform unui model de date pentru baze de date (unul fundamental sau derivat);
  - *cu o redundanță minimă și controlată* , care este asigurată prin modelul de date implementat și prin tehnicile de proiectare ale BD;
  - *accesibil mai multor utilizatori în timp util* , adică mai mulți utilizatori, concomitent, pot obține informațiile dorite atunci când are nevoie de ele.



## 1.5. Protecția BD



- *Activitatea de protecție a bazelor de date* este deosebit de importantă atât pentru modul de lucru pe calculatoare independente cât și pentru modul de lucru în rețea de calculatoare.
- Asigurarea protecției pentru o bază de date revine atât în sarcina SGBD, prin funcția de administrare (tot mai mult), cât și în sarcina administratorului bazei de date.
- *Protecția bazei de date* este un ansamblu de măsuri necesare asigurării securității și integrității datelor.

În cele ce urmează, vom prezenta o sinteză a principalelor aspectelor care apar pentru a se asigura protecția bazelor de date. Lucruri suplimentare vor fi prezentate în capitolele aferente SGBD-ului Oracle, iar exemplificarea se poate găsi, pe larg, în referința bibliografică [VELU02].

**Securitatea datelor** semnifică interzicerea accesului la date pentru utilizatorii neautorizați.

**Integritatea datelor** înseamnă corectitudinea datelor încărcate, precum și manipularea lor astfel încât să se respecte restricțiile de integritate ale modelului de date implementat.

În continuare, vom prezenta, pe scurt, cele două aspecte care compun activitatea de protecție a unei baze de date. La fiecare aspect vom sintetiza activitățile aferente.

### INTEGRITATEA datelor.

1. Integritatea semantică constă în prevenirea introducerii unor date incorecte în BD și în prevenirea realizării unor prelucrări eronate. Acest lucru se asigură prin respectarea restricțiilor de integritate. Acestea pot fi *implicite* (asigurate automat de SGBD) și *explicite* (asigurate prin proceduri incluse în programele de aplicație).
2. Controlul concurenței la date constă în garantarea coerenței (corectitudinii) și simultaneității datelor în cazul prelucrării *tranzacțiilor* (unitatea logică de prelucrare) prin tehnici specifice (blocarea, interblocarea etc.).
3. Salvarea și restaurarea. *Salvarea* este operația de stocare a datelor în copii de siguranță prin tehnici specifice (copiere, jurnalizare etc.). Ea se poate face automat de către SGBD (cel mai des) sau manual de către administratorul BD. *Restaurarea* este operația de refacere a consistenței BD, pornind de la datele salvate, minimizând prelucrările pierdute. Restaurarea se poate face automat de către SGBD (cel mai des) sau manual de către administratorul BD.

### SECURITATEA datelor.

1. Autorizarea și controlul accesului la date constă în identificarea utilizatorilor și restricționarea accesului acestora, pentru diferite operații de prelucrare.
2. Viziunile (views) sunt partiții logice ale BD definite pentru diferiți utilizatori.
3. Procedurile speciale sunt rutine, oferite de SGBD, care efectuează anumite operații asupra datelor și care sunt accesibile anumitor utilizatori.
4. Criptarea este operația de codificare a datelor în vederea stocării sau transmiterii datelor. În acest sens, se folosesc o mulțime de tehnici specifice: parole, algoritmi de criptare/decriptare, rutine speciale etc.

## 1.6. Rezumat

După ce s-a definit noțiunea de *organizare a datelor*, s-a prezentat *evoluția* acesteia, pornind de la fișiere (primul mod de organizare a datelor în memoria externă) și ajungându-se la ultimele tipuri de baze de date, cele orientate obiect. În această evoluție, s-au urmărit câteva aspecte (caracteristici) comparative: structura de date, modul de prelucrare a datelor, redundanța datelor, software-ul utilizat.

Pentru a se fundamenta *necesitatea* organizării datelor în memoria externă, comparativ cu memoria internă, s-a prezentat un tabel în care se găsesc cinci criterii: cost, viteză, capacitate, persistență, organizare.

În continuare, se definește noțiunea de *fișier* și, acesteia, i se prezintă câteva dintre cele mai importante aspecte de organizare a datelor: operații, structura, modul de organizare, modul de acces.

O parte consistentă din capitol este destinată organizării datelor în baze de date conform unor *modele de date*.

Mai întâi sunt prezentate principalele *noțiuni* necesare: informația, modelul de structură, modelul de date, tipuri de modele de date, schema conceptuală, elementele componente ale unui model logic de date (definirea structurii, operatorii, regulile de integritate).

După aceea sunt prezentate *cele patru modele* logice de date fundamentale (de bază), întâlnite la organizarea datelor în baze de date: ierarhic, rețea, relațional, orientat obiect. Fiecare model este prezentat în același mod: elementele componente, caracterizare generală. În acest fel se pot urmări, comparativ, cum se regăsesc noțiunile prezentate la începutul capitolului, în fiecare model în parte.

Capitolul se încheie cu o sinteză privind *protecția bazelor de date*. Sunt luate în considerare cele două aspecte ale protecției: securitatea, integritatea. Fiecare dintre acestea este descompusă în principalele subactivități care trebuie să fie asigurate într-un sistem de bază de date. de către SGBD și administratorul bazei de date.

## 1.7. Teste de autoevaluare



1. În modelul relațional pentru baze de date:
  - a) există noțiunea de tip înregistrare
  - b) atributele care au rol în realizarea legăturilor între tabele se numesc chei compuse
  - c) legăturile între tabele se descriu în LMD
  - d) există noțiunea de schema tabelii
  - e) proiecția, negația și existența sunt operatori din algebra relațională
2. După modelul de date implementat, bazele de date sunt:
  - a) ierarhice, locale
  - b) distribuite, generalizate
  - c) relaționale, de conducere
  - d) orientate obiect, specializate
  - e) rețea, de documentare
3. La organizarea datelor în memoria externă, noțiuni corespunzătoare sunt:
  - a) câmp – valoare
  - b) caracteristică – câmp
  - c) caracteristică – înregistrare
  - d) câmp – înregistrare
  - e) colecția de date – fișier
4. În modelul ierarhic pentru baze de date:
  - a) definirea obiectelor se face prin clase de obiecte
  - b) se admit legături 1:1, 1:M, M:N
  - c) există secvențe ierarhice de realizări
  - d) există operatorul de actualizare într-un arbore
  - e) o realizare copil este întotdeauna asociată unei singure realizări părinte
5. În modelul orientat obiect pentru baze de date:
  - a) obiectul are identificator, proprietăți
  - b) obiectul are interfață, implementare
  - c) există caracteristicile fundamentale ale obiectelor: succesiunea și încapsularea
  - d) există restricția de integritate: orice clasă respectă regulile impuse obiectului din care face parte
  - e) la baza operațiilor (operatorilor) stau metodele
6. La evoluția organizării datelor în memoria externă se au în vedere aspectele:
  - a) modul de organizare a datelor
  - b) nivelurile de structurare a datelor
  - c) modul de prelucrare a datelor
  - d) redundanța programelor
  - e) aplicațiile software aferente

### 1.8. Răspunsuri și comentarii la testele de autoevaluare



1. d; 2. -; 3. b, e; 4. c, d, e; 5. a, b, c; 6. a, b, c, e.

### 1.9. Bibliografia unității de învățare 1



- [RAGE00] R.Ramkrishnan, J.Gehrke – “Database Management Systems”, ed.Mc Graw Hill, 2000
- [RICC01] G.Riccardo – “Principles of Database Systems - with Internet and Java Applications”, ed.Addison Wesley, 2001
- [VELU00] M.Velicanu, I.Lungu ș.a. – “Sisteme de gestiune a bazelor de date”, ed.Petrion, 2000
- [VELU02] M.Velicanu, I.Lungu, M.Muntean, M.Iorga, S.Ionescu – “Oracle – platformă pentru baze de date”, ed. Petrion, 2002
- [LUBO95] I. Lungu, C. Bodea ș. a. - “Baze de date – organizare, proiectare și implementare “, ed. ALL, 1995
- [CONN00] T.Connolly – “Database Systems”, ed.Mc Graw Hill, 2000

## Unitatea de învățare 2:

### BAZE DE DATE – NOȚIUNI FUNDAMENTALE

---

#### Cuprins

- 2.1. Obiective
- 2.2. Baza de date – concepte
- 2.3. Administrarea bazelor de date
- 2.4. Rezumat
- 2.5. Teste de autoevaluare
- 2.6. Răspunsuri și comentarii la testele de autoevaluare
- 2.7. Bibliografia unității de învățare 2

#### 2.1. Obiective



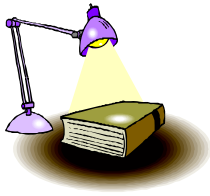
După studiul acestei unități de învățare vei avea cunoștințe despre:

- *generații de baze de date;*
- *noțiuni de bază: domeniul de valori, caracteristica, familia de caracteristici, colecția de date, baza de date;*
- *definiția bazei de date, văzută ca un ansamblu de date în memoria externă cu următoarele caracteristici: organizat, pe trei niveluri (conceptual, logic, fizic); structurat, conform unui model de date; coerent, prin restricțiile de integritate și protecția datelor; cu o redundanță minimă și controlată, prin implementarea unui model de date și prin aplicarea unei tehnici de proiectare; accesibil mai multor utilizatori în timp util;*
- *activitatea de administrare a unei baze de date, punctându-se sarcinile administratorului BD, precum și instrumentele necesare pentru a fi la dispoziția acestuia pentru a îndeplini aceste sarcini.*

***Durata medie a unei unități de studiu individual – 2 ore***



## 2.2. Baza de date – concepte



1. Evoluție
2. Elementele componente ale unei baze de date
3. Conceptul de bază de date

### Evoluție

Într-un calculator datele sunt stocate atât în memoria internă (temporar) cât și în memoria externă (persistent).

În memoria externă, evoluția modului de memorare a datelor a fost determinată de următoarele aspecte:

- accesul cât mai rapid și ușor la date;
- stocarea unui volum cât mai mare de date;
- creșterea complexității datelor;
- perfecționarea echipamentelor de culegere, stocare, transmitere și prelucrare a datelor.

Bazele de date, ca mod de organizare a datelor în memoria externă, au evoluat din fișiere printr-un proces de integrare a lor (fișierele și legăturile dintre ele) și ținând cont de cerințele aplicațiilor informatice.

Sunt numeroase aspecte care privește trecerea de la fișiere la baze de date (vezi Anexa A2) dar cel mai important este modelul de structurare a datelor în memoria externă (vezi capitolul 2) care diferă pentru cele două noțiuni.

De la apariția lor și până astăzi, bazele de date au parcurs următoarele generații (determinate în principal de modelul de date implementat):

- generația I: BD arborescente și rețea (până la sfârșitul anilor '70);
- generația a-II-a: BD relaționale (sfârșitul anilor '70 și până acum);
- generația a-III-a: BD orientate obiect (sfârșitul anilor '80 și până acum).

### Elementele componente ale unei baze de date

Am arătat faptul că bazele de date au evoluat din fișiere (vezi și capitolul 2). Acest lucru se regăsește și în noțiunile utilizate în cele două moduri de organizare a datelor în memoria externă, așa cum reiese din tabelul următor. De remarcat faptul că atât fișierele cât și bazele de date se construiesc pornind de la lumea reală înconjurătoare.

Citirea tabelului se va face de la stânga la dreapta, iar noțiunile corespunzătoare de pe cele două rânduri pot fi considerate similare.

Astfel vom spune: o bază de date este formată din mai multe colecții de date. Fiecare dintre acestea are atașată o familie de caracteristici, care este formată din mai multe caracteristici, care iau valori dintr-un domeniu de valori.

În mod similar se citește rândul de jos: un sistem de fișiere este format din mai multe fișiere. Fiecare dintre acestea are atașată o înregistrare, care este formată din mai multe câmpuri, care iau valori.

Baza de date	Colecții de date	Familie de caracteristici	Caracteristici	Domenii de valori
Sistem de fișiere	Fișiere	Înregistrare	Câmpuri	Valori

Ne vom ocupa, în continuare, de noțiunile aferente bazelor de date, urmând ca de fișiere să ne ocupăm în capitolul următor.

Domeniul de valori este dat de mulțimea valorilor posibile pentru o caracteristică (exemplu: culorile posibile pentru un automobil).

Caracteristica semnifică definirea și descrierea unui anumit aspect (proprietăți) dintr-o entitate a lumii reale (exemplu: marca auto)

Familia de caracteristici este ansamblul caracteristicilor care se referă la aceeași entitate din lumea reală (exemplu: mulțimea caracteristicilor prin care se poate descrie un automobil {NUMĂR, MARCA, CAPACITATE\_CILINDRICĂ, CULOARE}).

Colecția de date(entitatea) este o familie de caracteristici asupra căreia se aplică un predicat (care conduce la o relație de ordine între caracteristici și la obținerea informațiilor cu un anumit scop) căruia i se afectează anumite legături.

### Conceptul de bază de date

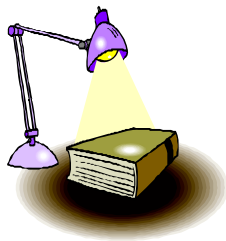
Definirea noțiunii de bază de date o putem face din mai multe puncte de vedere. Astfel vom avea, mai întâi, în vedere componentele sale și apoi vom ține cont de organizarea datelor în memoria externă.

*Baza de date* este un ansamblu de colecții de date aflate în interdependență, împreună cu descrierea datelor și a legăturilor dintre ele.

*Baza de date* este un ansamblu de date în memoria externă cu următoarele caracteristici: - organizat, pe trei niveluri (conceptual, logic, fizic);

- structurat, conform unui model de date;
- coerent, prin restricțiile de integritate și protecția datelor;
- cu o redundanță minimă și controlată, prin implementarea unui model de date și prin aplicarea unei tehnici de proiectare;
- accesibil mai multor utilizatori în timp util.

## 2.3. Administrarea bazelor de date



- *Administratorul BD* este format din una sau mai multe persoane cu experiență în analiză și proiectare, care se ocupă de organizarea și întreținerea BD.

*Nivelurile de administratori* de baze de date, după activitățile desfășurate, sunt:

- administrator global realizează:
  - schema (conceptuală) a BD, pornind de la cerințele aplicației;
  - organizarea datelor la nivel logic (colaborează).
- administrator de aplicație realizează:
  - schema externă (logică) pentru o aplicație, pornind de la cerințele de prelucrare ale aplicației;
  - organizarea datelor la nivel fizic (colaborare).
- administratorul bazei de date realizează:
  - schema internă (fizică) a datelor;
  - reorganizarea bazei de date;

- gestionează funcționarea BD.

*Sarcinile administratorului BD* sunt structurate după activitățile de realizare a unei BD:

- la analiza și proiectarea BD:
  - definește obiectivele BD;
  - colaborează la formularea cerințelor aplicației;
  - definește dicționarul BD(schema, restricțiile de integritate etc.);
  - colaborează la schema externă și la cea internă;
  - concepe protecția datelor.
- la implementarea BD:
  - elaborează documentație;
  - definește regulile de implementare și dare în folosință a BD;
  - asigură încărcarea BD din diferite surse de date.
- la exploatarea BD:
  - monitorizează accesul la date;
  - asigură protecția datelor;
  - întreține funcționarea BD la parametrii proiectați.

*Instrumente* la dispoziția administratorului BD pentru a-și îndeplini sarcinile:

- instrumentele oferite de SGBD pentru: reorganizarea BD, refacerea BD, analize statistice, gestionarea dicționarului de date, protecția datelor.
- instrumente specifice create de administratori și programatori.

*Notă.* Un *exemplu de administrare* a unei baze de date relaționale se găsește în [VELU02], cu exemplificare pe sistemul Oracle. Ca o concluzie, se poate spune că în acest moment, activitatea de administrare a unei baze de date a fost mult automatizată. Acest lucru înseamnă că multe dintre sarcinile de administrare au fost preluate de SGBD, degrevându-l pe administrator.



## 2.4. Rezumat

Cunoașterea conceptelor într-un domeniu de activitate este de mare importanță pentru profesioniști. Acest lucru este valabil și pentru cei care doresc să lucreze cu baze de date. Conceptele aferente acestui domeniu au fost create de cercetători, în mare parte, încă de la apariția primei generații de baze de date, ținându-se cont de evoluția din fișiere, ca mod de organizare a datelor în memoria externă.

Principalele *concepte prezentate* sunt: generații de baze de date, domeniul de valori, caracteristica, familia de caracteristici, colecția de date, baza de date.

*Baza de date* este un ansamblu de date în memoria externă cu următoarele caracteristici: - organizat, pe trei niveluri (conceptual, logic, fizic);

- structurat, conform unui model de date;
- coerent, prin restricțiile de integritate și protecția datelor;
- cu o redundanță minimă și controlată, prin implementarea unui model de date și prin aplicarea unei tehnici de proiectare;
- accesibil mai multor utilizatori în timp util.

Pe scurt, este prezentată activitatea de *administrare* a unei baze de date, punctându-se: ce este un administrator, care sunt nivelurile de administrare, care sunt sarcinile administratorului BD, precum și instrumentele necesare a fi la dispoziția sa pentru a îndeplini aceste sarcini.

## 2.5. Teste de autoevaluare



7. Baza de date este un ansamblu de date:
  - a) organizat, structurat
  - b) cu o redundanță minimă și necontrolată
  - c) accesibil mai multor utilizatori în timp util
  - d) coerent, modular
  - e) distribuit uniform
  
8. Pentru o bază de date:
  - a) structura conceptuală se deduce din cea logică
  - b) structura externă se deduce din cea conceptuală
  - c) structura globală se deduce din cea conceptuală
  - d) structura globală se deduce din cea fizică
  - e) structura internă se deduce din cea fizică
  
9. Concepte specifice unei baze de date sunt:
  - a) colecția de proprietăți
  - b) colecția de date
  - c) familia de caracteristici
  - d) familia de înregistrări
  - e) caracteristica

## 2.6. Răspunsuri și comentarii la testele de autoevaluare



1. a, c; 2. b; 3. b, c, e;

## 2.7. Bibliografia unității de învățare 2



- [LUBO95] I. Lungu, C. Bodea ș. a. - “Baze de date – organizare, proiectare și implementare “, ed. ALL, 1995
- [VELU02] M.Velicanu, I.Lungu, M.Muntean, M.Iorga, S.Ionescu – “Oracle – platformă pentru baze de date”, ed. Petron, 2002

## Unitatea de învățare 3: BAZE DE DATE RELAȚIONALE (BDR)

---

### Cuprins

- 3.1. Obiective
- 3.2. Conceptul de BDR
- 3.3. Optimizarea BDR prin tehnica normalizării
- 3.4. Algebra relațională și calculul relațional
- 3.5. Rezumat
- 3.6. Teste de autoevaluare
- 3.7. Răspunsuri și comentarii la testele de autoevaluare
- 3.8. Bibliografia unității de învățare 3

### 3.1. Obiective



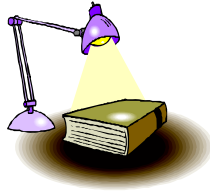
După studiul acestei unități de învățare vei avea cunoștințe despre:

- *bazele de date relaționale (BDR);*
- *realizarea unei aplicații cu BDR, ceea ce presupune obligatoriu proiectarea bazei de date;*
- *diferite tehnici de proiectare a BDR, dintre care cea mai utilizată este normalizarea;*
- *cele cinci forme normale (FN1 la FN5) care se construiesc în procesul de proiectare a unei BDR;*
- *două domenii importante din teoria relațională: calculul relațional și algebra relațională, împreună cu cei mai importanți operatori aferenți acestora.*

***Durata medie a unei unități de studiu individual – 8 ore***



### 3.2. Conceptul de baze de date relaționale (BDR)

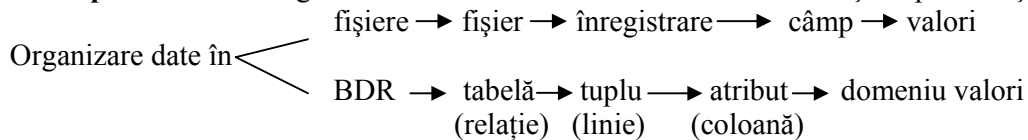


- Bazele de date relaționale (BDR) utilizează modelul de date relațional și noțiunile aferente.
- BDR au o solidă fundamentare teoretică, în special prin cercetările de la IBM conduse de E.F.Codd.
- BDR este un ansamblu organizat de *tabele* (relații) împreună cu *legăturile* dintre ele.

**Câteva dintre avantajele BDR față de fișiere:**

CRITERIU	BDR	FIȘIERE
Independența datelor	logică și fizică	fizică
Niveluri de structurare	conceptual, logic și fizic	logic și fizic
Deschidere și portabilitate	mare	mică
reprezentarea și utilizarea datelor	simplificat prin model	complicat
structura de date se păstrează	în dicționarul BD	în programe.

**Concepte** utilizate la organizarea datelor în memoria externă în BDR și respectiv fișiere:

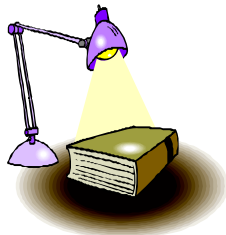


**Condiții minimale** pentru ca un SGBD să fie relațional:

- să implementeze modelul de date relaționale prin LDD și LMD;
- să implementeze cel puțin un limbaj relațional.

*Notă.* Condițiile complete ca un SGBD să fie relațional sunt date de cele 13 reguli ale lui CODD.

### 3.3. Proiectarea BDR prin tehnica normalizării



- **Aspecte generale privind proiectarea BD**
- **Noțiuni privind normalizarea**
- **Formele normale**

#### a) Aspecte generale privind proiectarea BD

**Proiectarea** unei BD este un proces în care se utilizează diferite modele, tehnici și instrumente pentru a transpune un domeniu din lumea reală înconjurătoare, în termenii organizării datelor aferente, pe calculator.

Pornind de la lumea reală se **identifică și specifică** cerințele aplicației (domeniul de interes) prin: documentare, interviu, terminologie, comunicare etc.

Rezultatul acestei activități este obținerea elementelor necesare pentru organizarea informației din domeniul respectiv. Acest lucru se realizează [RICC01] cu ajutorul unor **concepțe și instrumente** adecvate:

- *Schemele BD:*
  - Schema conceptuală specifică structura organizațională și conținutul informațional al sistemului și reprezintă un mod de comunicare între proiectanții și utilizatorii BD;
  - Schema logică definește informația într-o manieră care poate fi folosită pentru crearea BD.  
Schema externă organizează informația într-o ordine care permite *accesul utilizatorilor* la BD (construcția viziunilor).  
*Notă.* Cele două scheme pot fi separate sau integrate într-una singură deoarece ambele se materializează prin instrucțiuni (nivelul logic) dintr-un limbaj din SGBD, destinate unor utilizatori (extern).
  - Schema fizică specifică reprezentarea informației în calculator în termeni fizici (cum este stocată informația în memoria externă, pe suportul tehnic de informație).
- Obiectele din lumea reală se reprezintă prin *entități* care au *caracteristici* (attribute).
- Obiectele din lumea reală au *asocieri* (legături) de diferite tipuri cu alte obiecte, fiecare având un anumit rol în aceste legături.

#### b) Noțiuni privind normalizarea

**Aspectul dinamic** al structurii de date este avut în vedere și rezolvat de BDR. Acest lucru înseamnă că modelul de date relațional permite schimbarea în timp a structurii de date fără schimbarea programelor de aplicație (independența logică).

**Tehnica de normalizare** este utilizată în activitatea de proiectare a structurii BDR și constă în eliminarea unor anomalii (neajunsuri) de actualizare din structură.

**Anomaliile de actualizare** sunt situații nedorite care pot fi generate de anumite tabele în procesul proiectării lor:

- *anomia de ștergere* semnifică faptul că ștergând un tuplu dintr-o tabelă, pe lângă informațiile care trebuie șterse, se pierd și informațiile utile existente în tuplul respectiv;
- *anomaliile de adăugare* semnifică faptul că nu pot fi incluse noi informații necesare într-o tabelă, deoarece nu se cunosc și alte informații utile (de exemplu valorile pentru cheie);
- *anomia de modificare* semnifică faptul că este dificil de modificat o valoare a unui atribut atunci când ea apare în mai multe tupluri.

*Notă.* Anomaliile de actualizare sunt rezolvate de către teoria relațională, lucru care nu se întâmplă la alte tipuri de BD.

**Normalizarea** este o teorie construită în jurul conceptului de *forme normale* (FN), care ameliorează structura BD prin înlăturarea treptată a unor neajunsuri și prin imprimarea unor facilități sporite privind manipularea datelor.

*Notă.* Teoria normalizării a fost concepută inițial de către E.F.Codd, ulterior aducându-și contribuția și alți cercetători.

Normalizarea utilizează ca metodă **descompunerea** (top-down) unei tabele în două sau mai multe tabele, păstrând informații (attribute) de legătură.

#### c) Formele normale

O tabelă (relație) este într-o **formă normală** (FN) dacă satisface anumite restricții, care arată că FN(i+1) este preferată față de FN(i), i=1,4.

Sunt cinci forme normale (FN1 la FN5) și una suplimentară (BCNF) care revizuieste FN3.  
O **bază de date este în** FN<sub>i</sub>,  $i=1,5$  dacă toate tabelele sale sunt în FN<sub>i</sub>.  
**BD inițială** va fi formată dintr-o singură colecție de date, care apoi se descompune în mai multe tabele, parcurgând formele normale.

#### FN1

O **tabelă este în FN1** dacă toate atributele ei conțin valori elementare (nedecompozabile), adică fiecare tuplu nu trebuie să aibă date la *nivel de grup* sau *repetitiv*.  
Structurile de tip arborescent și rețea *se transformă* în tabele cu atribute elementare.  
O tabelă în FN1 prezintă încă o serie de *anomalii* de actualizare datorită eventualelor dependențe funcționale incomplete.  
*Fiecare* structură repetitivă generează (prin descompunere) o nouă tabelă, iar atributele la nivel de grup se înlătură, rămânând doar cele elementare.

#### FN2

O **tabelă este în FN2** dacă și numai dacă este în FN1 și fiecare atribut noncheie al tabelii este *dependent funcțional complet* de cheie.  
Un atribut B al unei tabele *depinde funcțional* de atributul A al aceleiași tabele, dacă fiecărei valori a lui A îi corespunde o singură valoare a lui B, care îi este asociată în tabelă.  
Un atribut B este *dependent funcțional complet* de un ansamblu de atribute A în cadrul aceleiași tabele, dacă B este dependent funcțional de întreg ansamblul A (nu numai de un atribut din ansamblu).  
O tabelă în FN2 prezintă încă o serie de *anomalii* de actualizare, datorită eventualelor dependențe tranzitive.  
*Eliminarea* dependențelor incomplete se face prin descompunerea tabelii inițiale în două tabele, ambele conținând atributul intermediar (B).

#### FN3

O **tabelă este în FN3** dacă și numai dacă este în FN2 și fiecare atribut noncheie depinde în mod *netranzitiv* de cheia tabelii.  
Într-o tabelă T, fie A,B,C trei atribute cu A cheie. Dacă B depinde de A ( $A \longrightarrow B$ ) și C depinde de B ( $B \longrightarrow C$ ) atunci C depinde de A în mod *tranzitiv*.  
*Eliminarea* dependențelor tranzitive se face prin descompunerea tabelii inițiale în două tabele, ambele conținând atributul intermediar (B).  
O tabelă în FN3 prezintă încă o serie de *anomalii* de actualizare, datorate eventualelor dependențe multivaloare.  
*Notă.* O definiție mai riguroasă pentru FN3 a fost dată prin forma intermediară BCNF (Boyce Codd Normal Form): o tabelă este în BCNF dacă fiecare determinant este un candidat cheie. Determinantul este un atribut elementar sau compus față de care alte atribute sunt complet dependente funcțional.

#### FN4

O **tabelă este în FN4** dacă și numai dacă este în FN3 și nu conține două sau mai multe dependențe *multivaloare*.  
Într-o tabelă T, fie A,B,C trei atribute. În tabela T se menține dependența multivaloare A dacă și numai dacă mulțimea valorilor lui B ce corespunde unei perechi de date (A,C), depinde numai de o valoare a lui A și este independentă de valorile lui C.

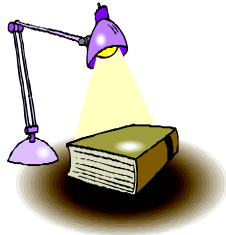
#### FN5

O tabelă este în FN5 dacă și numai dacă este în FN4 și fiecare *dependență joncțiune* este generată printr-un candidat cheie al tabelii.

În tabela T (A,B,C) se menține *dependența joncțiune* (AB, AC) dacă și numai dacă T menține dependența multivaloare  $A \twoheadrightarrow B$  sau C.

*Notă.* Dependența multivaloare este caz particular al dependenței joncțiune. Dependența funcțională este caz particular al dependenței multivaloare.

### 3.4. Algebra relațională și calculul relațional



- Pentru manipularea datelor conform modelului relațional se folosesc *operatorii* din algebra și calculul relațional [RICC01].
- Pentru implementarea acestor operatori există *comenzi specifice* în LMD din SGBDR. Aceste comenzi sunt utilizate în operații de regăsire (interogare).

După *tehnica folosită* la manipulare, LMD sunt bazate pe:

- calculul relațional (QUEL în Ingres, ALPHA propus de Codd);
- algebra relațională (ISBL, RDMS);
- transformarea (SQL, SQUARE);
- grafică (QBE, QBF).

#### Calculul relațional

Calculul relațional se bazează pe *calculul predicatelor* de ordinul întâi (domeniu al logicii) și a fost propus de E.F.Codd.

*Predicatul* este o relație care se stabilește între anumite elemente și care poate fi confirmată sau nu.

*Predicatul de ordinul 1* este o relație care are drept argumente variabile care nu sunt predicate.

*Variabila* poate fi de tip tuplu (valorile sunt dintr-un tuplu al unei tabeli) sau domeniu (valorile sunt dintr-un domeniu al unei tabeli).

*Cuantificatorii* (operatorii) utilizați în calculul relațional sunt: universal ( $\forall$ ) și existențial ( $\exists$ ).

*Construcția de bază* în calculul relațional este expresia relațională de calcul tuplu sau domeniu (funcție de tipul variabilei utilizate).

*Expresia relațională de calcul* este formată din:

- operația de efectuat;
- variabile (tuplu respectiv domeniu);
- condiții (de comparație, de existență);
- formule bine definite (operanzi-constante, variabile, funcții, predicate; operatori);
- cuantificatori.

*Notă.* Exemplu de implementare a calculului relațional în limbajul QUEL vezi în cartea [VELU00].

#### Algebra relațională

*Algebra relațională* este o colecție de operații formale aplicate asupra tabelilor (relațiilor), și ea a fost concepută de E.F.Codd.

Operațiile sunt aplicate în *expresiile algebrice relaționale* care sunt cereri de regăsire.

Acestea sunt compuse din operatorii relaționali și operanzi.

*Operanzii* sunt întotdeauna tabele (una sau mai multe).

*Rezultatul* evaluării unei expresii relaționale este format dintr-o singură tabelă.

Algebra relațională are cel puțin puterea de regăsire a calcului relațional. O expresie din calculul relațional se poate transforma într-una *echivalentă* din algebra relațională și invers.

**Operatorii** din algebra relațională pot fi grupați în două categorii (R1, R2, R3 sunt relații (tabele)):

### 1. Operatori pe mulțimi

REUNIUNEA:  $R3 = R1 \cup R2$ , unde R3 va conține tupluri din R1 sau R2 luate o singură dată.

DIFERENȚA:  $R3 = R1 \setminus R2$ , unde R3 va conține tupluri din R1 care nu se regăsesc în R2

PRODUSUL CARTEZIAN:  $R3 = R1 \times R2$ , unde R3 va conține tupluri construite din perechi ( $x_1x_2$ ), cu  $x_1 \in R1$  și  $x_2 \in R2$ .

INTERSECȚIA:  $R3 = R1 \cap R2$ , unde R3 va conține tupluri care se găsesc în R1 și R2 în același timp.

### 2. Operatori relaționali speciali

SELECȚIA: din R1 se obține o subtabelă R2, care va conține o submulțime din tuplurile inițiale din R1 ce satisfac un predicat (o condiție).

Numărul de atribute din R2 = numărul de atribute din R1

Numărul de tupluri din R2 < numărul de tupluri din R1.

PROIECȚIA: din R1 se obține o subtabelă R2, care va conține o submulțime din atributele inițiale din R1 și fără tupluri duplicate

Numărul de atribute din R2 < numărul de atribute din R1

Numărul de tupluri din R2  $\leq$  numărul de tupluri din R1.

JONCȚIUNEA: derivație a produsului cartezian, ce presupune utilizarea unui calificator care să permită compararea valorilor unor atribute din R1 și R2, iar rezultatul în R3.

R1 și R2 trebuie să aibă unul sau mai multe atribute comune care au valori comune.

*Note.*

- 1) Codd a introdus șase operatori *de bază* (reuniunea, diferența, produsul cartezian, selecția, proiecția, joncțiunea) și doi operatori *derivați* (intersecția și diviziunea).
- 2) Ulterior au fost introduși și alți operatori derivați (*speciali*).
- 3) Algebra relațională este prin definiție *neprocedurală* (descriptivă), iar calculul relațional permite o manieră de căutare *mixtă* (procedurală/neprocedurală).

## Transformarea

Oferă o putere de regăsire *echivalentă* cu cea din calculul și algebra relațională.

Se bazează pe *transformarea* (mapping) unui atribut sau grup de atribute într-un atribut dorit prin intermediul unor relații.

*Rezultatul* este o relație (tabelă) care se poate utiliza într-o altă transformare.

## Grafica

Oferă *interactivitate* mare pentru construirea cererilor de regăsire.

Utilizatorul specifică cerea *alegând* sau completând un ecran structurat grafic.

Poate fi *folosit* de către toate categoriile de utilizatori în informatică.



### 3.5. Rezumat

Se definește *noțiunea* de bază de date relațională (BDR) ținând cont de modelul de date relațional. Se face o *comparație* a noțiunilor utilizate în BDR cu cele utilizate la lucrul cu fișiere.

Realizarea unei aplicații cu BDR este o activitate complexă, care presupune obligatoriu *proiectarea* bazei de date. În acest sens se folosesc diferite tehnici, dintre care cea mai utilizată este *normalizarea*. Scopul tehnicii de normalizare este de a elimina, în procesul de proiectare, *anomaliile de actualizare*.

Sunt prezentate cele cinci *forme normale* (FN1 la FN5) care se construiesc în procesul de proiectare a unei BDR.

În continuare, se prezintă un *exemplu practic* de proiectare a unei BDR prin tehnica de normalizare, ajungându-se la o schemă conceptuală. Sunt exemplificate doar primele trei forme normale, deoarece acestea sunt cele mai utilizate în aplicațiile practice.

În finalul capitoului se prezintă o sinteză a două domenii importante din teoria relațională: *calculul relațional* și *algebra relațională*. Fiecare dintre cele două domenii este definit pe scurt, iar apoi sunt prezentați cei mai importanți *operatori* aferenți. Sunt citate exemple de limbaie relationale clasificate după tehnica folosită la implementare.

### 3.6. Teste de autoevaluare



1. O tabelă este în FN3 dacă:
  - a) este în FN2 și fiecare atribut noncheie depinde în mod netranzitiv de cheia tablei
  - b) este în FN1 și fiecare atribut cheie depinde tranzitiv de atributele noncheie
  - c) este în FN2 și are dependențe complete
  - d) este în FN1 și are dependențe funcționale incomplete
  - e) este în FN2 și are cel puțin o dependență funcțională completă între atributele noncheie și cheia tablei
2. Calculul relațional:
  - a) are drept construcție de bază expresia de calcul tuplu sau expresia de calcul domeniu
  - b) este implementat în limbajul ISBL
  - c) conține operatorul de joncțiune
  - d) stă la baza limbajelor procedurale universale
  - e) folosește noțiunea de formulă bine definită
3. O tabelă este în FN1 dacă:
  - a) există atribute la nivel de grup
  - b) există atribute repetitive
  - c) nu există atribute la nivel de grup
  - d) există atribute decompozabile
  - e) nu există atribute repetitive

### 3.7. Răspunsuri și comentarii la testele de autoevaluare



1. a; 2. a, e; 3. c, e;

### 3.8. Bibliografia unității de învățare 3



- [CONN00] T.Connolly – “Database Systems”, ed.Mc Graw Hill, 2000
- [DATE94] J.Date – “ An introduction to Database Systems”, ed. Addison Wesley, 1994
- [RICC01] G.Riccardo – “Principles of Database Systems - with Internet and Java Applications”, ed.Addison Wesley, 2001
- [VELU00] M.Velicanu, I.Lungu ș.a. – “Sisteme de gestiune a bazelor de date”, ed.Petrion, 2000
- [VELU01] M.Velicanu, I.Lungu, M.Muntean – “Dezvoltarea aplicațiilor cu baze de date în Visual Foxpro”, ed. ALL, 2001

## Unitatea de învățare 4: METODOLOGIA DE REALIZARE A BAZELOR DE DATE

---

### Cuprins

- 4.1. Obiective
- 4.2. Organizarea unei baze de date
- 4.3. Obiectivele urmărite în realizarea unei baze de date
- 4.4. Etape în realizarea unei baze de date
- 4.5. Utilizarea metodologiei UML în proiectarea bazelor de date
- 4.6. Proiectarea bazelor de date relaționale – o extensie UML
- 4.7. Eficiența bazelor de date
- 4.8. Rezumat
- 4.9. Teste de autoevaluare
- 4.10. Răspunsuri și comentarii la testele de autoevaluare
- 4.11. Bibliografia unității de învățare 4

### 4.1. Obiective



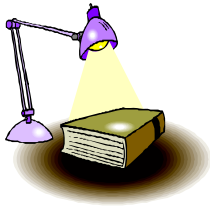
După studiul acestei unități de învățare vei avea cunoștințe despre:

- *activitatea de realizare a unei baze de date;*
- *organizarea activității de realizare a unei baze de date (intrările, memorarea datelor, protecția datelor, legăturile datelor) astfel încât să se poată decide dacă merită sau nu demararea acțiunii respective;*
- *metodologiile de realizare a bazelor de date;*
- *etapele (activitățile) care trebuie parcurse în vederea realizării unei baze de date: analiza de sistem, proiectarea noului sistem, realizarea componentelor logice, punerea în funcțiune, dezvoltarea;*
- *eficiența activității de realizare a bazelor de date, importantă pentru evaluarea investiției efectuate.*

***Durata medie a unei unități de studiu individual – 12 ore***

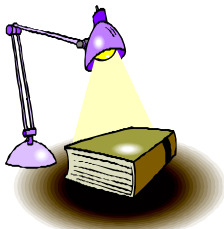


## 4.2. Organizarea unei baze de date



- Activitatea de realizare a unei baze de date trebuie să înceapă cu *organizarea* acesteia, adică pregătirea acțiunii. În cadrul acestei pregătiri, ținând cont de obiectivele urmărite pentru realizarea bazei de date, se va face un fel de inventar a ceea ce este un minim necesar pentru a se putea realiza baza de date. Se poate astfel, din start, hotărî dacă activitatea va demara imediat sau dacă mai sunt necesare alte acțiuni pregătitoare.
  
- Prezintă, în continuare, câteva *aspecte* mai importante urmărite la organizarea unei baze de date:
  1. *Organizarea intrărilor* de date, se referă la:
    - sursa datelor (documente primite, fișiere etc.);
    - actualizarea datelor (moduri, momente, reorganizări etc.);
    - controlul intrărilor de date (natură, conținut, format, periodicitate etc.).
  2. *Organizarea memorării* datelor, se referă la:
    - principii de memorare (memoria internă/externă, forma de stocare, legături între date, tehnici utilizate etc.);
    - optimizarea memorării (redundanță, spațiu, alocare, codificare, reorganizare etc.).
  3. *Organizarea protecției* datelor sub cele două aspecte (securitatea și integritatea), se referă la:
    - instrumente oferite de către SGBD (autorizare acces, fișiere jurnal, arhivări etc.);
    - metode proprii (parole, rutine speciale, antivirus etc.).
  4. *Organizarea legăturilor* bazei de date cu alte aplicații, se referă la: conversii, standarde, compatibilității, interfețe etc.

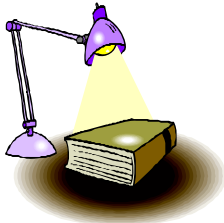
## 4.3. Obiectivele urmărite la realizarea unei BD



- Atunci când dorim să realizăm o bază de date trebuie să știm clar ce avem de făcut, adică să stabilim obiectivele activității noastre. În acest sens, câteva dintre cele mai importante *obiective*, le prezentăm în continuare. Acestea, ar trebui să constituie un set minim de obiective, de care să se țină cont la realizarea unei baze de date:
  1. *Partiționarea* semnifică faptul că aceleași date trebuie să poată fi folosite în moduri diferite de către diferiți utilizatori.
  2. *Deschiderea* se referă la faptul că datele trebuie să fie ușor adaptabile la schimbările care pot apărea (actualizarea structurii, tipuri noi de date etc.).
  3. *Eficiența* are în vedere stocarea și prelucrarea datelor, care trebuie să se facă la costuri cât mai scăzute, costuri care să fie inferioare beneficiilor obținute.
  4. *Reutilizarea* înseamnă faptul că fondul de date existent trebuie să poată fi reutilizat în diferite aplicații informatice.

5. *Regăsirea* este o activitate frecventă pe bazele de date și de aceea cererile de regăsire trebuie să poată fi adresate ușor de către toate categoriile de utilizatori, după diferite criterii.
6. *Accesul* înseamnă modul de localizare a datelor și acest lucru trebuie să poată fi realizat prin diferite moduri de acces, rapid și ușor.
7. *Modularizarea* presupune faptul că realizarea BD trebuie să se poată face modular pentru generalitate și posibilitatea lucrului în echipă.
8. *Protecția* bazei de date trebuie asigurată sub ambele aspecte: securitatea și integritatea datelor.
9. *Redundanța* se asigură în limite acceptabile prin implementarea unui model de date pentru baze de date și prin utilizarea unei tehnici de proiectare a BD. Se asigură astfel, o redundanță minimă și controlată.
10. *Independența* datelor față de programe trebuie asigurată atât la nivel logic cât și fizic.

#### 4.4. Etape în realizarea unei BD



- *Activitățile* (etapele) parcurse pentru realizarea unei BD sunt: analiza de sistem, proiectarea noului sistem, realizarea componentelor logice, punerea în funcțiune, dezvoltarea.

##### 1. *Analiza de sistem.*

*Scopul* acestei activități este de a evidenția cerințele aplicației și resursele utilizate (studiul), precum și de a evalua aceste cerințe prin modelare (analiza).

STUDIUL situației existente se realizează prin:

- definirea caracteristicilor generale ale unității;
- identificarea activităților desfășurate;
- identificarea resurselor existente (informaționale, umane, energetice, echipamente, financiare etc.);
- identificarea necesităților de prelucrare.

ANALIZA sistemului existent urmează investigării (studiului) și utilizează informațiile obținute de aceasta.

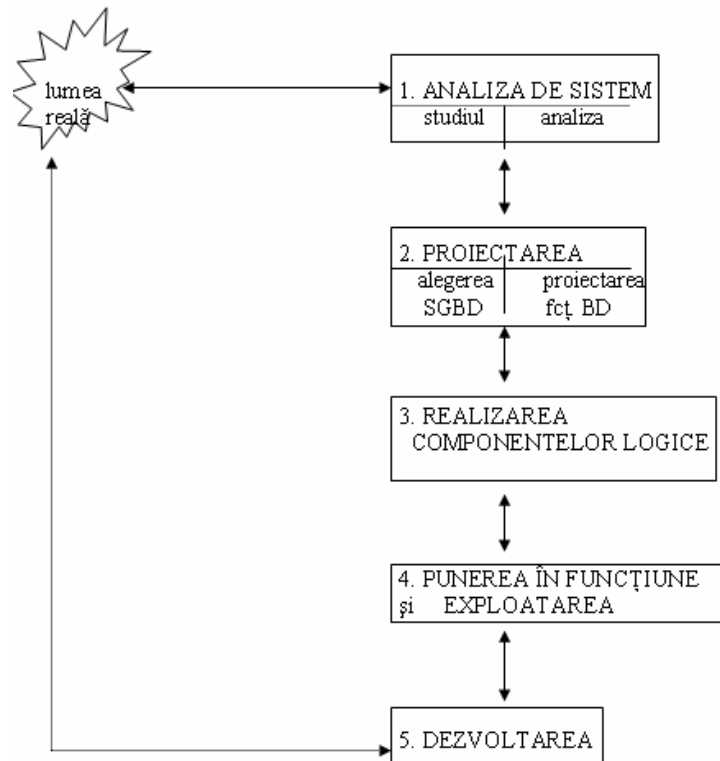


Fig. 4.1. Activitățile parcurse pentru realizarea unei baze de date

Analiza este o activitate de modelare (conceptuală) [RICC01] și se realizează sub **trei aspecte**: structural, dinamic și funcțional.

a) *Analiza structurală* evidențiază, la nivel conceptual, modul de structurare a datelor și a legăturilor dintre ele. Cea mai utilizată tehnică este *entitate-asociere*. Aceasta conține:

- 1) Identificarea entităților: fenomene, procese, obiecte concrete sau abstracte (substantivele din prezentarea activității descrise) (exemple de entități: Persoane, Produse, Beneficiari).
- 2) Identificarea asocierilor dintre entități ca fiind legăturile semnificative de un anumit tip (verbele din prezentarea activității descrise).
- 3) Identificarea atributelor ce caracterizează fiecare entitate în parte (exemple de atribute: Marca, Nume, Adresă).
- 4) Stabilirea atributelor de identificare unică a realizărilor entității, drept chei.

*Rezultatul* analizei structurale: modelul static (structural) numit și diagrama entitate-asociere.

*Notă.* Diagrama entitate-asociere (Entity-Relationship) poate fi generată cu produse software tip CASE (Computer Aided Software Engineering), ca de exemplu Oracle Designer. Pornind de la o astfel de diagramă, se pot construi, în activitatea de proiectare, schemele relațiilor (tabelor). Pentru această translație există un set de 14 reguli [vezi RICC01] care pot fi aplicate.

b) *Analiza dinamică* evidențiază comportamentul elementelor sistemului la anumite evenimente. Una din tehnicile utilizate este diagrama *stare-tranziție*. Aceasta presupune:

- 1) Identificarea stărilor în care se pot afla componentele sistemului.
- 2) Identificarea evenimentelor care determină trecerea unei componente dintr-o stare în alta.
- 3) Stabilirea tranzițiilor admise între stări.
- 4) Construirea diagramei stare-tranziție.

*Rezultatul* analizei dinamice: modelul dinamic

c) *Analiza funcțională* evidențiază modul de asigurare a cerințelor informaționale (fluxul prelucrărilor) din cadrul sistemului, prin care intrările sunt transformate în ieșiri. Cea mai utilizată tehnică este *diagrama de flux* al datelor. Conform acestei tehnici se delimitează:

- 1) Aria de cuprindere a sistemului.
- 2) Se identifică sursele de date.
- 3) Se identifică modul de circulație și prelucrare a datelor.
- 4) Se identifică apoi rezultatele obținute.

*Rezultatul* analizei funcționale: modelul funcțional.

*Notă.* Exemplu privind activitatea de analiză, sub toate cele trei aspecte prezentate mai sus, vezi în cartea [LUVE00].

## 2. *Proiectarea structurii bazei de date*

Pornind de la modelele realizate prin activitatea de analiză, se poate proiecta structura BD, parcurgând două subactivități: alegerea SGBD-ului, proiectarea funcțiilor BD.

a) **Alegerea SGBD-ului** se face ținând cont de două aspecte: cerințele aplicației (utilizatorului) și performanțele tehnice ale SGBD-ului.

*Cerințele aplicației* se referă la:

- volumul de date estimat a fi memorat și prelucrat în BD;
- complexitatea problemei de rezolvat;
- ponderea și frecvența operațiilor de intrare/ieșire;
- condiții privind protecția datelor;
- operații necesare pe baza de date (încărcare/validare, actualizare, regăsire etc.);
- particularități ale activității pentru care se realizează baza de date.

*Performanțele tehnice* ale SGBD-ului se referă la:

- modelul de date pe care-l implementează;
- ponderea utilizării SGBD-ului pe piață și tendința;
- configurația de calcul minimă cerută;
- limbajele de programare din SGBD;
- facilități de utilizare oferite pentru diferite categorii de utilizatori;
- limitele SGBD-ului;
- optimizări realizate de SGBD;
- facilități tehnice:
  - lucrul cu mediul distribuit și concurența de date;
  - elemente de multimedia;
  - elemente de CASE;
  - interfețe de comunicare;
  - autodocumentarea;
  - instrumente specifice oferite.

b) **Proiectarea funcțiilor BD** (vezi și [RAGE00]) se realizează prin: proiectarea schemelor BD, proiectarea modulelor funcționale specializate.

Proiectarea schemelor BD se realizează pornind de la rezultatele modelării conceptuale (analiza de sistem) și ținând cont de modelul de date implementat de SGBD-ul ales. Se vor proiecta schemele: conceptuală, externă, internă.

*Proiectarea schemei conceptuale* pornește de la identificarea setului de date necesar sistemului. Aceste date sunt apoi integrate și structurate într-o schemă (exemplu: pentru BD relaționale cea mai utilizată tehnică este *normalizarea*). Pentru acest lucru se parcurg **pașii**:

1. Stabilirea schemei conceptuale inițiale care se deduce din modelul entitate-asociere (vezi analiza structurală). Pentru acest lucru, se transformă fiecare

entitate din model într-o colecție de date (fișier), iar pentru fiecare asociere se definesc cheile aferente. Dacă rezultă colecții izolate, acestea se vor lega de alte colecții prin chei rezultând asocieri (1:1, 1:M, M:N).

2. Ameliorarea progresivă a schemei conceptuale prin eliminarea unor anomalii (exemplu: cele cinci forme normale pentru BD relaționale).
3. Stabilirea schemei conceptuale finale trebuie să asigure un echilibru între cerințele de actualizare și performanțele de exploatare (exemplu: o formă normală superioară asigură performanțe de actualizare, dar timpul de răspuns va fi mai mare).

*Proiectare schemei externe* are rolul de a specifica viziunea fiecărui utilizator asupra BD. Pentru acest lucru, din schema conceptuală se identifică datele necesare fiecărei viziuni. Datele obținute se structurează logic în subscheme ținând cont de facilitățile de utilizare și de cerințele utilizator. Schema externă devine operațională prin construirea unor viziuni (view) cu SGBD-ul și acordarea drepturilor de acces. Datele într-o viziune pot proveni din una sau mai multe colecții și nu ocupă spațiul fizic.

*Proiectarea schemei interne* presupune stabilirea structurilor de memorare fizică a datelor și definirea căilor de acces la date. Acestea sunt specifice fie SGBD-ului (scheme de alocare), fie sistemului de operare. Proiectarea schemei interne înseamnă realizarea **operațiilor**:

1. estimarea *spațiului fizic* pentru BD și definirea unui model fizic de alocare (a se vedea dacă SGBD-ul permite explicit acest lucru);
2. definirea unor *indecși* pentru accesul direct, după cheie, la date;  
Un ghid privind activitatea de indexare [RAGE00] pentru baze de date, poate include *pașii*:
  - necesitatea indexării: se indexează dacă se reduce timpul de căutare sau dacă este necesară ordonarea datelor;
  - alegerea cheii de indexare: atributele din clauzele WHERE / FOR sunt candidate pentru chei;
  - cheile construite din atribute multiple: mai multe atribute separate (cheie multiplă) sau concatenate (cheie compusă), într-o anumită ordine;
  - tipul de indexare: arbori (pentru cereri cu egalitate), hash-cod (pentru cereri cu joncțiune) etc.;
  - costul întreținerii indecșilor este dat de două aspecte: indecșii se șterg dacă operațiile care-i implică sunt rare (se reduce astfel spațiul) și indecșii se păstrează dacă operațiile care-i implică sunt dese (se reduce astfel timpul).
3. construirea unor *clustere*, care înseamnă regruparea fizică a datelor din BD, pentru a permite un acces mai rapid pentru anumiți utilizatori. Datele pot proveni din una sau mai multe colecții. Clusterelor generează avantaj la regăsire și dezavantaj la actualizare, în ceea ce privește timpul de răspuns.

Proiectarea modulelor funcționale ține cont de concepția generală a BD, precum și de schemele proiectate anterior. În acest sens, se proiectează:

- fluxul informațional;
- modulele de încărcare și manipulare a datelor;
- interfețele specializate;
- integrarea elementelor proiectate cu organizarea și funcționarea BD.

### 3. *Realizarea componentelor logice*

Componentele logice ale unei BD sunt programele de aplicație dezvoltate, în cea mai mare parte, în SGBD-ul ales.

Programele se realizează conform modulelor funcționale proiectate în etapa anterioară.

Componentele logice țin cont de ieșiri, intrări, prelucrări și colecțiile de date.



În paralel cu dezvoltarea programelor de aplicații se întocmesc și documentațiile diferite (tehnică, de exploatare, de prezentare).

#### 4. *Punerea în funcțiune și exploatarea*

Se testează funcțiile BD mai întâi cu date de test, apoi cu date reale.

Se încarcă datele în BD și se efectuează procedurile de manipulare, de către beneficiar cu asistența proiectantului.

Se definitivează documentațiile aplicației.

Se intră în exploatare curentă de către beneficiar conform documentației.

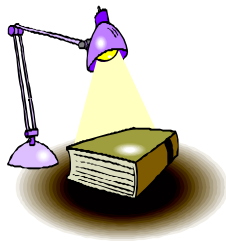
#### 5. *Dezvoltarea sistemului*

Imediat după darea în exploatare a BD, în mod continuu, pot exista factori perturbatori care generează schimbări în BD.

Schimbările necesare trebuie preluate de BD ușor și “din mers”.

Factorii pot fi: organizatorici, datorți progresului tehnic, rezultați din cerințele noi ale beneficiarului, din schimbarea metodologiilor etc.

### 4.5. Utilitatea metodologiei UML în proiectarea bazelor de date



- Complexitatea deosebită a sistemelor actuale a determinat abordarea pe scară largă a principiilor orientării obiectuale nu doar în ceea ce privește programarea ci și în proiectarea lor, lucru care a avut consecințe și asupra proiectării bazelor de date. În general modelarea orientată obiect (gen UML) este privită ca apanajul doar al proiectării (sau dezvoltării) aplicațiilor orientate obiect.

Această abordare se dovedește însă **limitată** având în vedere două aspecte :

**1.** *Pe de o parte proiectarea aplicațiilor (orientate obiect) cu baze de date separate de proiectarea bazei de date va produce un **nivel suplimentar de mapare** a entităților care figurează în cele două arii **la nivel conceptual**.* Proiectarea unui sistem integrat din care să rezulte concertat ce obiecte vor fi stocate persistent în baza de date, ce obiecte vor face parte din package -urile sau modulele aplicațiilor rezidente sub forma componentelor și modul cum vor interacționa între ele se poate dovedi o alternativă mai bună. Singurul substrat suplimentar (care poate fi modelat și el prin stereotipuri UML) ar fi cel al mapării entităților conceptuale în structurile oferite de modelul logic al bazei de date pentru stocare și regăsire.

**2.** *Pe de altă parte modelele logice ale bazelor de date relationale comerciale propun astăzi o serie de concepte noi cum sunt procedurile stocate și declansatoarele care nu fac parte din modelele relationale tradiționale și de obicei nu sunt luate în considerare în faza proiectării bazei de date .* Valorificarea direct în proiectarea conceptuală a acestora se poate face folosind concepte orientate obiect și mecanismele de extensibilitate ale unui limbaj de modelare cum este UML.

**Motivațiile** pentru care metodologia UML ar fi potrivită și pentru specificarea modelelor implicate în proiectarea bazelor de date (fata metodele tradiționale) ar putea fi rezumate astfel:

- modelul structural al UML încorporează toate conceptele modelului entitate –relatie (cel mai popular model pentru proiectarea conceptuală a bazelor de date) plus conceptele specifice modelelor semantice (generalizare, agregare) și abordării orientate obiect (mostenire, polimorfism, supraîncărcare). Limbajul UML posedă

mecanismele de extensibilitate (restricții, stereotipuri, note, etichete) care fac posibilă specializarea modelului în funcție de aspectele particulare ale sistemelor cu baze de date (flexibilitate sporită în modelare);

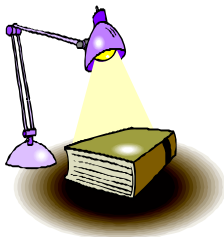
- conceptele limbajului UML propun un vocabular comun între modelele rezultate din proiectarea sistemului ca ansamblu și cele din proiectarea specifică bazei de date;
- metodologia UML (procesul unificat) favorizează munca în echipă: scopul folosirii UML pentru modelarea proceselor afacerii, dezvoltării aplicațiilor și modelării bazei de date este integrarea echipelor de dezvoltare pentru a împiedica construirea unei arhitecturi fără implicarea corespunzătoare în acest proces a tuturor echipelor care concurează la realizarea sistemului în ansamblu;
- integrarea instrumentelor de dezvoltare din diferitele arii de interes din care rezultă arhitectura sistemului pe baza unei platforme conceptuale comune. Instrumentele CASE bazate pe metodologii obiectuale ofereau inițial suport pentru proiectarea generală a sistemului și mai puțin pentru proiectarea de detaliu. Cu alte cuvinte arhitectura sistemului era sprijinită prin instrumente *upper CASE* destul de slab integrate cu cele *lower CASE*.

Adoptarea UML ca limbaj de modelare a produs în acest sens două mutații:

- apariția de noi instrumente CASE care oferă suport pentru întreg ciclul de dezvoltare până la obținerea unei versiuni complete a sistemului (gen Rational Rose);
- integrarea de noi facilități dedicate modelării în mediile de dezvoltare integrate comerciale (gen suitele de dezvoltare de la Borland, Microsoft sau Oracle) cu posibilități de proiectare vizuală din care să rezulte componentele conținând codul sursă sau scripturile pentru generarea schemelor bazelor de date.

*Implicarea metodologiei UML în proiectarea bazelor de date nu constituie un pas artificial ci o integrare naturală cu procesul mai larg al construirii sistemelor informaționale pe principii obiectuale.*

#### 4.6. Proiectarea bazelor de date relaționale – o extensie UML



- Diagrama claselor prezintă un mecanism neutru de implementare pentru modelarea aspectelor ce țin de stocarea datelor sistemului. Clasele persistente, atributele acestora și relațiile dintre ele pot fi implementate direct într-o bază de date orientată obiect. *Cu toate acestea, în prezent, bazele de date relaționale rămân modalitatea de stocare a datelor cea mai uzuală.*

Diagrama claselor din UML permite modelarea unor aspecte specifice proiectării bazelor de date relaționale, dar nu acoperă în întregime această problematică, de notat faptul că nu prevede noțiunea de atribute cheie care sunt mecanismul principal de relaționare a tabelor. Pentru a surprinde mai bine aceste aspecte este bine să se apeleze la diagrama entitate asocieră (ER), în completarea setului de diagrame propus de UML. Diagrama claselor poate fi utilizată pentru a modela logic baza de date, independent de faptul că se alege o implementare relațională sau orientată obiect, prin clase reprezentându-se tablele, iar prin atribute coloanele acestora. Dacă se alege pentru implementare un mediu relațional, atunci diagrama claselor poate fi ușor translatată într-o diagramă logică entitate asocieră. Claselor persistente și atributelor acestora le corespund entitățile logice și atributele lor, iar relațiilor dintre clase le corespund relații între entități.

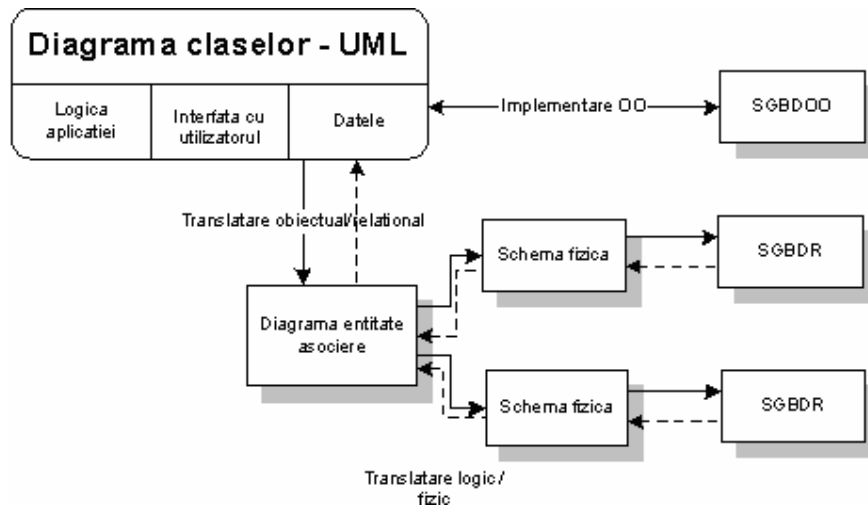
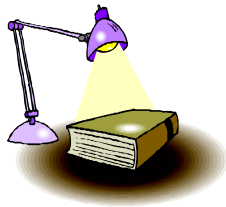


Fig. 4.2. Proiectarea BDR cu ajutorul diagramei entitate asociere

Odată întocmită această diagramă se poate trece la proiectarea bazei de date relaționale conform tehnicii normalizării.

#### 4.7. Eficiența bazelor de date



- *Efectele* directe și indirecte ale bazelor de date conduc spre eficiență, mai ales pentru aplicațiile mari și complexe, acolo unde alte produse informatice nu fac față.

Efectele utilizării BD conduc spre **avantajele** oferite de acestea:

- elimină redundanțele necontrolate;
- crește productivitatea activității informatice;
- oferă suport informațional optim pentru conducere și execuție;
- permite interogarea datelor de către toate categoriile de utilizatori;
- conduce spre aplicații deschise.

**Efortul** depus pentru realizarea și întreținerea unei BD este considerabil și de aceea investiția necesară trebuie fundamentată pentru a putea fi recuperată.

**Fundamentarea** investiției pentru o aplicație cu BD se face prin studiul de fezabilitate (analiza preliminară) care va conține un *studiu al costurilor* și o *evaluare a efortului* de realizare și întreținere.

**Etape** care trebuie parcurse pentru stabilirea eficienței BD:

1. *Studiul costurilor inițiale* (de realizare)
  - costul software-ului necesar;
  - costul hardware-ului necesar;
  - costul impus de trecerea de la sistemul existent la SBD;
  - costul resurselor necesare (umane, materiale, informaționale etc.);
  - costul consultanței tehnice.
2. *Evaluarea costurilor* de funcționare a BD
  - costul de punere în funcțiune;

- costul de exploatare;
  - costul de întreținere și dezvoltare.
3. *Calcul de rentabilitate*
- utilizarea unor metode specifice (exemplu: Analiza Cost Beneficiu)
  - utilizarea unor instrumente hardware și software specializate pentru BD (cresc performanța, scad portabilitatea).

#### 4.8. Rezumat

Activitatea de realizare a unei baze de date este complexă și cere un efort considerabil. De aceea, ea trebuie realizată sistematic, după o metodologie adecvată.

Înainte de a se realiza o bază de date, se are în vedere *organizarea* acestei activități (intrările, memorarea datelor, protecția datelor, legăturile datelor) astfel încât să se poată decide dacă merită sau nu demararea acțiunii respective.

După luarea deciziei că baza de date este posibil a fi realizată, se fixează câteva dintre *obiectivele* importante ale acestei activități: partiționarea, deschiderea, eficiența, reutilizarea, regăsirea, accesul, modularizarea, protecția, redundanța, independența.

În continuare, pentru realizarea bazei de date, se poate urma o metodologie anume sau se pot parcurge *etapele (activitățile)* prezentate: analiza de sistem, proiectarea noului sistem, realizarea componentelor logice, punerea în funcțiune, dezvoltarea.

Atât pe parcursul realizării unei baze de date cât și după aceea, trebuie avută în vedere *eficiența* acestei activități. Acest lucru este important pentru că investiția pentru realizarea unei baze de date este mare și ea se recuperează în timp. Eficiența bazei de date va avea în vedere aspecte: studiul costurilor inițiale, evaluarea costurilor de funcționare, calcule de rentabilitate etc.

#### 4.9. Teste de autoevaluare



1. În metodologia de realizare a BD diagrama entitate-asociere rezultă în urma:
  - a) analizei funcționale
  - b) proiectării dinamice
  - c) analizei structurale
  - d) analizei dinamice
  - e) proiectării logice
  
2. Eficiența unei baze de date poate fi evaluată prin:
  - a) studiul organigramei întreprinderii
  - b) studiul costurilor inițiale
  - c) calcule de rentabilitate
  - d) calcule relaționale
  - e) algebra relațională
  
3. În metodologia de realizare a BD, construirea unei viziuni (view) se face prin:
  - a) proiectarea schemei externe
  - b) proiectarea schemei conceptuale
  - c) proiectarea schemei interne
  - d) analiza logică
  - e) analiza globală
  
4. Obiective urmărite în realizarea unei BD sunt:
  - a) aceleași date nu pot fi utilizate în moduri diferite
  - b) creșterea prețului de cost pentru memorarea datelor
  - c) cererile de regăsire trebuie să fie doar prestabilite
  - d) aplicațiile cu baze de date trebuie să poată reutiliza un fond de date existent deja
  - e) redundanța minimă dar necontrolată

#### 4.10. Răspunsuri și comentarii la testele de autoevaluare



1. c; 2. b, c; 3. a; 4. d;

#### 4.11. Bibliografia unității de învățare 4



- [SIKO97] A.Silbershartz, H.Korth ș.a. – “Database system concepts”, ed.McGraw-Hill, 1997
- [DATE94] J.Date – “ An introduction to Database Systems”, ed. Addison Wesley, 1994
- [MACI01] L.Maciaszek – “Requirements analysis and system design - developing Information systems with UML”, ed.Addison Wesley, 2001
- [DIAZ00] O.Diaz – “Advanced database technology and design”, ed.Piattini, 2000
- [RAGE00] R.Ramkrishnan, J.Gehrke – “Database Management Systems”, ed.Mc Graw Hill, 2000
- [RICC01] G.Riccardo – “Principles of Database Systems - with Internet and Java Applications”, ed.Addison Wesley, 2001
- [VELU00] M.Velicanu, I.Lungu ș.a. – “Sisteme de gestiune a bazelor de date”, ed.Petrion, 2000
- [VELU01] M.Velicanu, I.Lungu, M.Muntean – “Dezvoltarea aplicațiilor cu baze de date în Visual Foxpro”, ed. ALL, 2001
- [VELU02] M.Velicanu, I.Lungu, M.Muntean, M.Iorga, S.Ionescu – “Oracle – platformă pentru baze de date”, ed. Petrion, 2002

## Unitatea de învățare 5: BAZE DE DATE ORIENTATE OBIECT ( BDOO)

---

### Cuprins

- 5.1. Obiective
- 5.2. Conceptul de baze de date orientate obiect
- 5.3. Elemente de proiectare a BDOO
- 5.4. Tehnici de implementare utilizate în BDOO
- 5.5. Rezumat
- 5.6. Teste de autoevaluare
- 5.7. Răspunsuri și comentarii la testele de autoevaluare
- 5.8. Bibliografia unității de învățare 5

### 5.1. Obiective



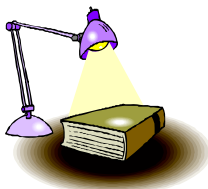
După studiul acestei unități de învățare vei avea cunoștințe despre:

- *bazele de date orientate obiect (BDOO) în contextul trecerii de la sistemele relaționale la a treia generație de baze de date;*
- *cele mai importante caracteristici specifice unei BDOO și principalele concepte utilizate la realizarea unei astfel de baze de date;*
- *proiectarea unei BDOO, conform unei anumite metodologii, care precizează niște pași care trebuie parcurși;*
- *o serie de tehnici utilizate pentru implementarea într-o BDOO a principalelor elemente care dau facilitățile acestor sisteme.*

*Durata medie a unei unități de studiu individual - 3 ore*



### 5.2. Conceptul de baze de date orientate obiect



- *Limitele sistemelor relaționale, în special cele referitoare la volume mari de date și complexitatea ridicată a datelor, au determinat evoluția spre sistemele orientate obiect.*

- *Dezvoltarea* BDOO (generația a treia de BD) a fost favorizată (începând cu anii '90) de următoarele *aspecte*: noul context informatic, noile tipuri de baze de date, evoluția limbajelor de programare.
- *Tehnologia* orientată obiect constă în totalitatea conceptelor, metodelor, proceselor și instrumentelor utilizate pentru construcția sistemelor cu obiecte.
- *Aplicarea* tehnologiei orientată obiect la SBD a însemnat *avantaje* privind îmbunătățirea structurii de date (modelul orientat obiect) și a celorlalte componente ale sistemului.

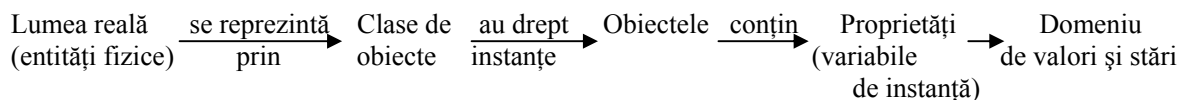
a) Caracteristici ale BDOO

1. Sunt *tratate în mod unitar* (ca obiecte): datele, programele, comunicația, de unde rezultă independența totală între ele.
2. *Comunicația și distribuirea* sunt asigurate atât între date, cât și între programe.
3. *Structura de date* este simplificată foarte mult, de unde rezultă ușurința în utilizare și portabilitatea ridicată a sistemelor rezultate.
4. Lucrul cu obiecte ne apropie firesc de *lumea reală*, în care se găsesc obiecte, care au proprietăți, asupra cărora acționăm noi.
5. Pot fi abordate foarte *multe domenii* din lumea reală în care se regăsesc cele mai *diferite tipuri de date*.
6. *Se asigură*: accesul neprocedural, comunicația, portabilitatea, deschiderea aplicațiilor cu baze de date.

b) Notiuni utilizate

Rămân valabile toate *noțiunile* prezentate la definirea structurii modelului OO: clase (tipuri) de obiecte, obiecte, metode, mesaje, instanțe, caracteristici (principii) fundamentale ale obiectelor. *Programul* este o secvență liniară de definiții și apeluri de obiecte și clase de obiecte. *BDOO* este o mulțime de clase de obiecte persistente (în memoria externă), organizată coerent și ordonată în ierarhii, partajată pentru utilizatorii concurenți. *SGBDOO* este suportul software complet care asigură descrierea (structura obiectului), manipularea și protecția datelor organizate după modelul OO.

*Sinteza conceptelor* din BDOO:



Un obiect se identifică prin:

- nume
- identificator
- metode
- mesaje
- implementare (privată)
- interfață (publică)
- caracteristici fundamentale

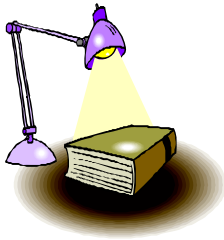
*Paralelă* între noțiunile utilizate în sistemele relaționale și orientate obiect:

<b>Sistem orientat</b>	Ierarhia de	Clasă de	Obiect	Variabilă de	Identificator
------------------------	-------------	----------	--------	--------------	---------------



<b>obiect</b>	clase	obiecte		instanță	
<b>Sistem relațional</b>	Schema BD	Tabelă	Tuplu	Atribut	Cheie

### 5.3. Elemente de proiectare a BDOO



- Dezvoltarea tehnologiei OO include și o serie de *metodologii* utilizate pentru realizarea aplicațiilor cu BDOO (exemplu: RUP).
- Diferitele metodologii precizează *etape* (pași) necesare a fi parcurse pentru realizarea unei BDOO.

Sintetizand, etapele din diferitele metodologii conduc la urmatoarele **activități**: analiza, proiectarea, elaborarea programelor, implementarea și exploatarea .

#### a) Analiza

- *Rolul* activității de analiză este de a preciza universul de discurs (se identifică entitățile fizice și operațiile necesare).
- *Pașii* parcurși la analiză sunt:
  - se delimitează universul de discurs pornind de la lumea inconjuratoare;
  - se identifică obiectele pornind de la universul de discurs;
  - se clasifică obiectele după proprietăți și comportament obținându-se clasele de obiecte;
  - se identifică secvențele de mesaje către obiecte (legăturile dintre obiecte și clase de obiecte);
  - se determină secvențele de activități și ciclul de viață al obiectelor (modelarea).
- *Parcurgerea* pașilor de mai sus se face prin reluări succesive a unor pași precedenți.
- *Rezultatul* analizei este un ansamblu de specificații scrise în comportamentul cerut.
- *Note*. Referitor la modelele utilizate pentru analiza BDOO pot fi formulate următoarele afirmații [RICC01]:
  1. Modelul *entitate-asociere* (utilizat la BDR) a fost dezvoltat cu ierarhii de clase de entități. Acesta este un suport natural pentru conceptele orientate obiect.
  2. Modelele OO pentru analiză descriu *conținutul* informației dar și *comportamentul* ei (metodele care se folosesc pentru manipularea obiectelor) Un model standard este ODL (Object Definition Language – produs de ODMG).
- *Schema ODL* este o colecție de definiții de interfețe (se poate mapa în C++, Java).
- *Caracteristici* ale schemei ODL:
  - Reprezentarea claselor se face prin:
    - Definirea clasei: nume, set de proprietăți;
    - Definirea proprietăților: atribut, metodă sau legătură.
  - Specificarea tipurilor de legături:
    - Legătura binară se reprezintă prin două proprietăți (câte una în fiecare clasă);

- Legătura de un grad ridicat nu se poate reprezenta direct ci prin cele binare;
  - Legătura se bazează pe identificatorul unic de obiect (OID) care este independent de valorile obiectului;
  - Legăturile nu pot fi denumite (în BDR da).
- Se admit structuri ierarhice de clase, cu proprietatea de moștenire.

#### b) Proiectarea

- *Rolul* activității de proiectare este de a furniza metode pentru reprezentarea obiectelor identificate la analiză.
- **Metode** pentru proiectarea BDOO: diagrama obiectelor, specificațiile obiectelor.

**1. Diagrama obiectelor:** numele proprietăților obiectelor se scriu într-un dreptunghi, iar sub acesta se scrie cu litere mari numele obiectului.

*Numele proprietăților* se scriu cu litere mici, iar dacă o proprietate este la randul ei obiect atunci se scrie cu litere mari.

Orice proprietate, care nu este obiect, poate lua o mulțime de valori (*domeniul de valori*) și se descrie prin: tip, lungime, restricții.

Dacă *proprietatea este un obiect* atunci domeniul este reprezentat de o mulțime de instanțe de obiecte (obiecte particulare).

*Viziunea* (view) reprezintă ansamblul proprietăților văzute de un utilizator din totalul celor existente într-un obiect.

Metoda diagramei dă o *imagine grafică* a structurii obiectelor și a legăturilor dintre ele.

**2. Specificațiile obiectelor:** liste de propoziții explicative care conțin informații despre structura obiectelor și a legăturilor dintre ele.

Listele, ordonate după diferite criterii, pot fi de *forma*:

- definiții ale obiectelor ;
- enumerarea obiectelor și a proprietăților acestora ;
- enumerarea domeniilor și definiții ale acestora;
- descrieri semantice ale obiectelor;
- restricții asupra obiectelor.

#### **Notă:**

În procesul de proiectare a BDOO obiectele pot fi **grupate**:

- *simple* sunt cele care au proprietăți cu o singură valoare (domeniu cu o valoare) și au proprietăți care nu sunt obiecte
- *compozite* sunt cele care au cel puțin o proprietate cu valori multiple și au proprietăți care nu sunt obiecte
- *compuse* sunt cele care au cel puțin o proprietate care este obiect;
- *asociate* sunt obiectele independente care au rolul de a stabili o legătură între alte obiecte (apar ca proprietăți ale obiectelor care le leagă);
- *agregate* sunt obiectele care reprezintă grupuri de entități și conțin proprietăți de grup ce se transmit tuturor instanțelor subclasselor (moștenirea).

#### c) Elaborarea programelor

Programele pentru definirea și manipularea obiectelor se scriu într-un SGBDOO (exemplu O2, Jasmine etc.) , care conțin limbaje de programare orientate obiect.

*Elaborarea* programelor se desfășoară pe baza specificațiilor obținute prin activitatea de proiectare a BDOO.

*Programele* sunt secvențe liniare de descrieri de obiecte și de apeluri de obiecte (mesaje).

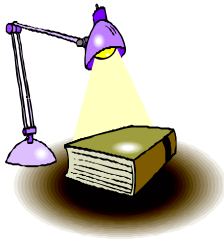
În SBDOO se folosește intens *reutilizarea* programelor.

#### d) Implementarea și exploatarea

Implementarea și exploatarea sunt activități care au *rolul* de a pune în funcțiune o BDOO la parametrii proiectați.

În cazul unor disfuncționalități *se revine* la o activitate precedentă și se produce o nouă versiune a BDOO.

### 5.4. Tehnici de implementare utilizate în BDOO



- La realizarea unei BDOO se va ține cont de mai multe elemente, care trebuie implementate prin diferite *tehnici*. Câteva dintre aceste tehnici sunt prezentate în continuare. Vom specifica denumirea tehnicii și apoi o scurtă explicație la fiecare.

1. *Gestiunea mediului* limbajului de programare: trebuie asigurată tratarea conceptelor modelului orientat obiect prin limbaje de programare corespunzătoare.
2. *Reprezentarea obiectelor*: modelul utilizat poate fi simplu (obiecte mici și puține) sau complex (obiecte mari și multe — structurate).
3. *Gestiunea persistenței* : se descriu și manipulează date permanente (persistente), în același mod cu cele temporare. Persistența se asigură prin :
  - tipaj, adică numai anumite tipuri (clase) de obiecte pot deveni persistente;
  - conectivitate, adică numai obiectele care rezultă (direct sau indirect) din alte obiecte persistente vor deveni persistente;
  - stocaj, adică toate obiectele plasate într-un spațiu explicit declarat, devin persistente.
4. *Gestiunea memoriei* : obiectele pot fi memorate atât în memoria internă cât și în cea externă. Identificatorul (logic sau fizic) de obiect joacă un rol important privind localizarea obiectului în memorie.
5. *Extensibilitatea*: trebuie să se permită noi structuri de date în BDOO creată deja.
6. *Distribuirea*: trebuie să se permită lucrul distribuit în rețea și să se separe prelucrarea (consumă memorie internă ) de stocarea (consumă memorie externă) obiectelor.
7. *Gestiunea versiunilor*: trebuie păstrate stările succesive ale obiectelor și evoluția lor în timp.
8. *Gestiunea obiectivelor* traditionale ale unei BD.

## 5.5. Rezumat

Se definește noțiunea de baze de date orientate obiect (BDOO) în contextul trecerii de la sistemele relaționale la a treia generație de baze de date.

Sunt prezentate câteva caracteristici specifice unei BDOO și principalele concepte utilizate la realizarea unei astfel de baze de date.

Proiectarea unei BDOO se face conform unei anumite metodologii, care precizează niște pași care trebuie parcurși. Sinteza acestor pași ne conduce spre o serie de activități care se regăsesc în procesul de realizare a unei BDOO: analiza, proiectarea, elaborarea programelor, implementarea și exploatarea. Fiecare dintre aceste activități este prezentată pe scurt, precizându-se rolul ei, o descriere, precum și eventualele metode utilizate.

În finalul capitolului sunt prezentate câteva dintre tehnicile utilizate pentru implementarea într-o BDOO, a principalelor elemente care dau facilitățile acestor sisteme.

## 5.6. Teste de autoevaluare



5. Caracteristici ale BDOO sunt:
  - a) permite abordarea unui domeniu limitat de tipuri de date
  - b) structura de date este mult complicată
  - c) independența datelor față de programe este parțială
  - d) se tratează în mod unitar datele, programele și comunicația
  - e) asigură comunicația atât între date cât și între programe
  
6. La proiectarea unei BDOO se pot utiliza metodele:
  - a) diagrama obiectelor
  - b) specificațiile obiectelor
  - c) normalizarea
  - d) diagramele de dependență
  - e) distribuirea obiectelor
  
7. Tehnici de implementare a BDOO sunt:
  - a) portabilitatea, partiționarea
  - b) gestiunea mediului, reprezentarea obiectelor
  - c) gestiunea persistenței, gestiunea memoriei
  - d) extensibilitatea, distribuirea majoră
  - e) flexibilitatea, portabilitatea

## 5.7. Răspunsuri și comentarii la testele de autoevaluare



1. d, e; 2. a, b; 3. b, c.

### 5.8. Bibliografia unității de învățare 5



- [EMBL98] D.Embley – “Object database development”, ed.Addison Wesley, 1998
- [DATE94] J.Date – “ An introduction to Database Systems”, ed. Addison Wesley, 1994
- [MACI01] L.Maciaszek – “Requirements analysis and system design - developing Information systems with UML”, ed.Addison Wesley, 2001
- [RICC01] G.Riccardo – “Principles of Database Systems - with Internet and Java Applications”, ed.Addison Wesley, 2001
- [DIAZ00] O.Diaz – “Advanced database technology and design”, ed.Piattini, 2000
- [VELU02] M.Velicanu, I.Lungu, M.Muntean, M.Iorga, S.Ionescu – “Oracle – platformă pentru baze de date”, ed. Petrion, 2002
- [VELU01] M.Velicanu, I.Lungu, M.Muntean – “Dezvoltarea aplicațiilor cu baze de date în Visual Foxpro”, ed. ALL, 2001