ACADEMIA DE STUDII ECONOMICE BUCUREŞTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ŞI INFORMATICĂ ECONOMICĂ
CATEDRA DE INFORMATICĂ ECONOMICĂ

BAZE DE DATE

Autori: Prof.univ.dr. LUNGU Ion

Asist.univ.drd. BOTHA Iuliana

CUPRINS

Introducere

Cursul de **Baze de date** se adresează studenților înscrişi la programul de studiu ID, organizat de facultatea de Cibernetică, Statistică şi Informatică Economică şi 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 şi al utilizării acestora în diferite domenii economico-sociale.

OBIECTIVELE SPECIFICE PRINCIPALE ale acestui curs, concretizate în competențele dobândite după parcurgerea şi 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 şi 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 şi într-un număr de 14 laboratoare (activități asistate) la care prezența va fi obligatorie.

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

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

- evaluarea finală – 50%
- evaluarea continuă – testele de la sfârşitul fiecărei unități de învățare, precum şi 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

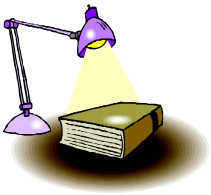
1.1. Obiective

După studiul acestei unități de învățare vei avea cunoştințe despre:

- *necesitatea organizării datelor în memoria externă, comparativ cu memoria internă;*
- *evoluția organizării datelor pornind de la fişiere (primul mod de organizare a datelor în memoria externă) şi ajungându-se la ultimele tipuri de baze de date, cele orientate obiect;*
- *noțiunea de fişier şi 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 şi structurarea datelor în memoria internă sau în memoria externă (în colecții) şi stabilirea legăturilor între date, conform unui model de date.

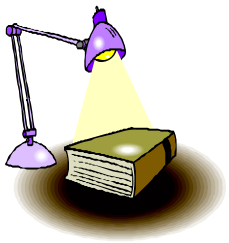
- Sintetizând, *evoluția* organizării datelor în memoria externă a avut în vedere câteva *aspecte*

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	Fişiere secvențiale	Logică coincide cu fizică	Pe loturi (batch)	Mare, necontrolat	Operații simple de I/E (limaje asamblare şi universal)
2. Anii ‘60	Fişiere secvențiale, indexate, directe	Logică şi fizică	Loturi, on-line	Mare, necontrolat	Chei simple de acces (limbaje universale)
3. Anii ‘70	Baze de date arborescente, rețea	Logică, fizică, conceptuală	Loturi, conversați-onal	Scade, controlat	Chei multiple de acces, legături între date, protecția (SGBD)
4. Sfărşitul anilor ’70 până acum	Baze de date relaționale	Logică, fizică, conceptuală	Conversați-onal, interactiv	Mică, controlat	Limbaje de regăsire, protecție, concurență (SGBD)
5. Sfârşitul 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 şi 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.	Fişiere, baze de date

1.3. Organizarea datelor în fişiere



- *Fisierul* este o colecţie organizată de date unite după criterii comune calitative, de prelucrare şi scop.

- Toate limbajele de programare universale lucrează cu această noţiune, pentru organizarea datelor în memoria externă. Sistemele de bază de date lucrează cu această noţiune şi în plus o dezvoltă în noţiunea de bază de date.

Prezentăm, în continuare, câteva noţiuni fumdamentale utilizate în organizarea datelor în fişiere. De aceste noţiuni se ţine seama şi la baze de date.

1. Caracteristici ale unui fişier:
– *actualizarea* se referă la trei operaţii : adaugarea, modificarea, ştergerea de înregistrări;
– *natura* datelor din fişier trebuie să fie omogenă (să se refere la aceeaşi entitate din lumea reală);
– *prelucrarea* datelor din fişier se referă la tipul şi frecvenţa operaţiilor efectuate pe înregistrări;
– *volumul* de date din fişier (se măsoară în număr de octeţi).

2. Structura unui fişier:
– *partea de identificare* este dată de etichetele plasate la începutul şi sfârşitul fişierului;
– *partea de date* este colecţia omogenă de date ce aparţin aceleaşi entităţi din lumea reală structurată astfel:

fişier ⟶ înregistrări ⟶ câmpuri ⟶ valori.

3. Modul de organizare reprezintă modul de dispunere a înregistrărilor pe suportul fizic şi presupune reguli de memorare a datelor.

Categorii de moduri de organizare a fişierelor:

a) **Standard**
– Este cea mai veche şi există pe toate tipurile de calculatoare.
– Înregistrarea este formată dintr-un şir de caractere dispus pe o linie acceptată de periferic.
– Toate limbajele recunosc fişiere standard de intrare şi ieşire.

b) **Clasică** (elementară)
– Organizarea se face pe medii magnetice sau optice.
– *Tipuri:*
  - SECVENŢIALĂ
    - Înregistrările sunt dispuse în fişier una după alta fără nici o ordine prestabilită.
    - Localizarea unei înregistrări se face prin parcurgerea tuturor înregistrărilor anterioare ei (secvenţial).
    - Toate sistemele de operare şi limbajele de programare acceptă organizarea secvenţială.
  - RELATIVĂ

- Înregistrările sunt dispuse în fişier una după alta şi 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 fişier 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.
- Fişierului îi este ataşat o tabelă de indecşi care face legătura între valoarea cheii şi adresa fizică a înregistrării.
- Localizarea unei înregistrări se poate face secvențial dar şi direct prin cheie:
  - se compară cheia înregistrării căutate cu indecşii din tabela de index şi se localizează direct partea fizică a fişierului 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 şi în sisteme de fişiere.
– *Tipuri* (câteva):

➢ PARTIȚIONAREA
- Înregistrările din fişier 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 fişiere care necesită regăsiri intense multicriteriale.

➢ INVERSĂ
- Presupune existența a două fişiere: de bază şi invers.
- Fişierul de bază conține datele propriu-zise şi are organizare secvențială. El este fişierul în care se caută.
- Fişierul invers este construit din cel de bază (printr-o tehnică de inversare) şi are organizare relativă. El este fişierul prin intermediul căruia se caută.
- Spațiul ocupat necesar este cam de 3,5 ori mai mare față de cât ocupă fişierul de bază.

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

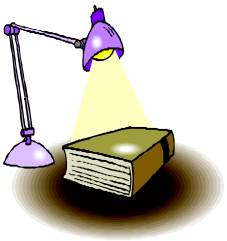
Tipuri de moduri de acces pentru fişiere:

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

- Se recomandă pentru fişierele din care sunt necesare, la o prelucrare, peste 50% din numărul total de înregistrări.
- Pentru optimizare se recomandă ordonarea fişierului.

➢ DIRECT presupune că localizarea unei înregistrări se face cu ajutorul unei chei definite de programator.
➢ DINAMIC presupune că la o singură deschidere de fişier se pot localiza, alternativ şi repetat, înregistrări în acces secvențial şi direct.

1.4. Modele de structurare a datelor în BD



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

informația —(are ca obiect)→ entitatea —(are ca proprietate)→ atributul —(are ca măsură)→ valoarea

o *Modelul de structură* se referă la descrierea tuturor atributelor unei entități în interdependență.

o Valorile atributelor se materializează prin *date*, care dau o reprezentare simbolică a informațiilor.

o *Modelul de date* este ansamblul de concepte şi 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).

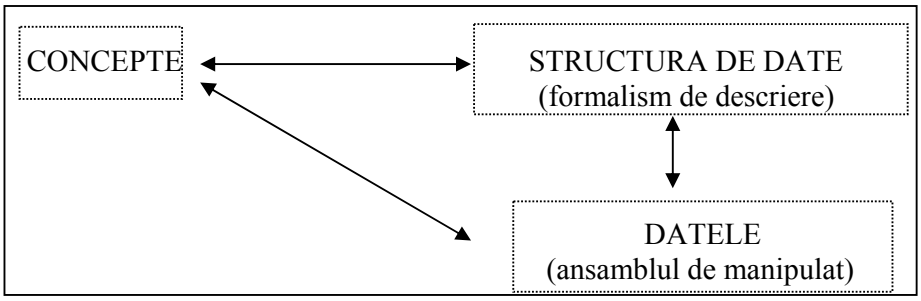


Fig. 1.1. Modelul de date

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 foloseşte pentru specificaţia iniţială a datelor şi pentru comunicarea cu utilizatorii;
  - este un mod de stabilire a unei legături între dezvoltatori şi utilizatori.
- *modelul logic* (la proiectare)
  - specifică structura bazei de date (colecţiile de date şi legăturile dintre ele);
  - se scrie într-o formă care să poată fi folosită pentru crearea bazei de date;
  - se foloseşte pentru definirea şi întreţinerea bazei de date de către SGBD şi pentru formularea cerinţelor 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 (fişiere, indecşi, discuri etc.);
  - de obicei, este generat automat, de către SGBD, pornind de la modelul logic.

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

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

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

Trebuie să se facă *distincţie clară* între structura bazei de date (schema, de exemplu: *stud(cod:N,4; nume:C,15)*) şi conţinutul ei (instanţa, de exemplu: *22 Nedelcu Anda*). Schema şi instanţele sunt stocate în baza de date, şi anume prima în dicţionar, 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 entităţilor şi a atributelor asociate;
  - definirea legăturilor (asociarea) dintre entităţi.

Asocierea poate fi de *tipul* :
  - unu la unu (1:1)
  - unu la mulţi (1:M)
  - mulţi la mulţi (M:N)
  - neprecizat explicit

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

2. Operatorii modelului (partea de manipulare) care acţionează asupra structurilor de date, dar şi asupra datelor, pentru operaţii de prelucrare (compunere, actualizare etc.).
Operatorii se implementează cu ajutorul unui LMD (Limbaj de Manipulare a Datelor) dintr-un SGBD.

3. Regulile de integritate (partea de coerenţă) sunt restricţii stabilite la descrierea datelor, care le asigură acestora menţinerea corectitudinii şi dau logica modelului.
Restricţiile se implementează cu un LDD din SGBD.

Tipuri de modele logice de date pentru BD:
- *fundamentale*: ierarhice, reţea, relaţionale, 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 entităţilor** se face prin noţiune de *tip de înregistrare* (clasă de entităţi), care este formată din *caracteristici* (câmpuri).

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

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

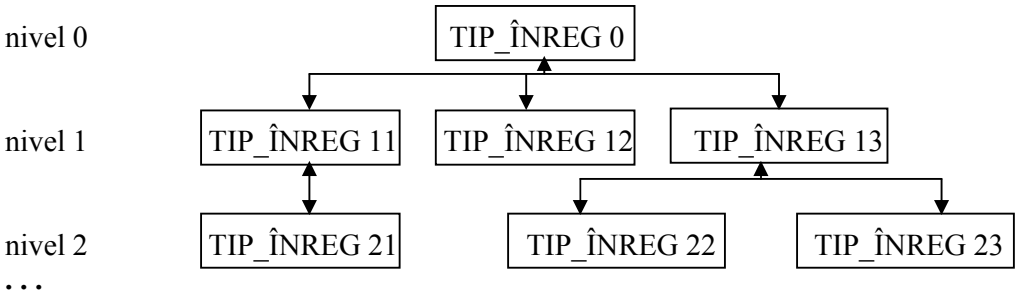


Fig. 1.2. Diagrama de structură ierarhică

Caracteristici ale structurii ierarhice (arborescente):

- Fiecare nod corespunde unui tip de înregistrare şi 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 mulţi subordonaţi.
- 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 şi unul sau mai multe noduri dependente situate pe unul sau mai multe niveluri.
  - ierarhie de tipuri de înregistrări se numeşte *tip arbore*.
  - *realizare* a unui tip arbore este formată dintr-o singură realizare a tipului de înregistrare rădăcină împreună cu o mulţime 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 *secvenţă 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 înregistrare rădăcină (secvenţă ierarhică) la o altă realizare a aceluiaşi tip 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 ştergerea unei înregistrări. Operația este greoaie şi consumatoare de resurse calculator (spațiu şi 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 şi produc tot o înregistrare.
Există şi 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 (creşte redundanța);
- dau logica şi corența modelului arborescent.

Caracterizarea generală a modelului ierarhic.

1. Modelul a fost *propus* de către IBM şi a fost primul utilizat pentru BD.
2. *Structura* modelului este simplă (graf orientat) familiară specialiştilor în informatică.
3. *Implementarea* modelului se face fizic şi 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 fişier secvențial.
4. *Aplicabilitatea* modelului se regăseşte cu succes în tehnologia construcțiilor de maşini, dar şi în alte domenii.
5. *Limitele* modelului ierarhic:
   - numărul de ierarhii posibile creşte combinatoric cu numărul înregistrărilor, în legătura părinte-copil;
   - actualizarea datelor este groaie şi consumatoare de resurse calculator;
   - nivelul logic nu este separat clar de cel fizic (exemplu: indecşii pot fi structuri logice dar şi 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* şi conduce la o structură de tip *rețea*, reprezentată sub forma unei *diagrame* (fig. 2.3.) numită şi schema (conceptuală).

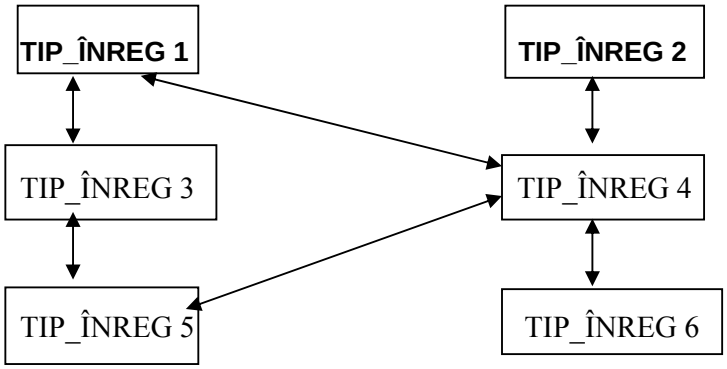


Fig. 1.3. Diagrama de structură rețea

Caracteristici ale structurii rețea :
- Un nod poate avea oricâți superiori şi 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 foloseşte conceptul de *tip set*. Un tip set este format dintr-un singur tip de *nod proprietar* şi 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 şi 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, ştergerea unor membri.
   - *Actualizarea într-o rețea*: adăugarea, modificarea, ştergerea unor seturi.
     Notă. Operațiile de actualizare sunt greoaie şi 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 aceluiaşi 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 şi nod membru în alt set.
   - Un set poate avea un singur nod proprietar.

Notă. Efectele restricțiilor de integritate sunt:

- dau logica şi 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 şi condiționează performanțele BD:
   - prin pointeri structurați în liste înlănțuite, 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 şi 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∈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 şi nu are nume.

Notă. ordinea liniilor (tupluri) şi coloanelor (atribute) dintr-o tabelă nu trebuie să prezinte nici-o importanță.

- *schema tabelei* este numele tabelei, urmat intre paranteze rotunde de lista atributelor, iar pentru fiecare atribut se precizeaza domeniul asociat.
- *cheia* este un atribut sau un ansamblu de atribute care au rolul de a identifica un tuplu dintr-o tabela. *tipuri* de chei: primare/alternate, simple/comune, externe.
- *schema relațională* este schema tabelei + cheile + restrictiile de integritate.

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

STUDENT

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

Schema tabelei : STUDENT(NUME: D1, ANSTUDIU: D2, ANNAŞTERE: D3).

Notă. Numărul de domenii este mai mic sau egal cu numărul de atribute pentru o tabelă (mai multe atribute pot lua valori din acelaşi 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 şi *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 şi un produs se poate livra către mai mulți beneficiari (M:N).

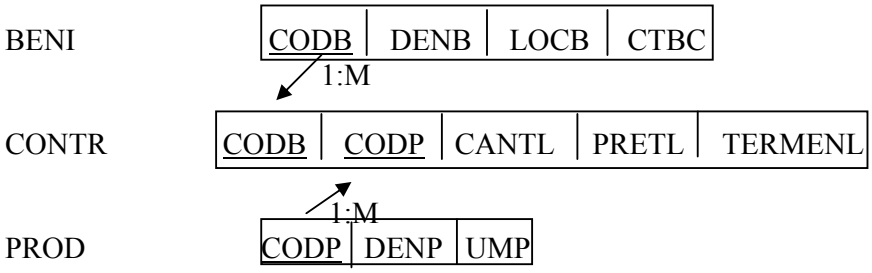


Fig. 1.4. Schema BD

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 numeşte *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 şi 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. Operanzii 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 aceeaşi 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);
- *entități*: 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 şi 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 nişte 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 şi 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) şi 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 şi 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 şi 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 şi pe comportamentul acestora, ambele încapsulate în obiect.

Conceptele utilizate sunt:

- *clasele (tipurile) de obiecte* sunt un tip abstract de date prin care se defineste structura obiectelor *(proprietatile*) si comportamentul (*metodele*) acestora.
- *obiectele* reprezinta o colectie de proprietati care se refera la aceeasi 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:
  - *incapsurarea* : descrierea obiectelor se face astfel incat nu se poate avea acces din afara obiectului la datele sale;
  - *polimorfismul* : diferite obiecte pot răspunde diferit la aceleaşi mesaje;
  - moştenirea : capacitatea unui obiect de a-şi deriva datele şi 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 moştenire;
  - 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 şi manipularea de obiecte compuse din alte obiecte. Obiectele compuse rezultate au o structură ierarhică dar nu au caracteristică de moştenire.

2. Operatorii modelului OO:

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

- *Actualizarea metodelor* : adaugare, modificare, ştergere de metode.
- *Actualizarea proprietăţilor* : adaugare, modificare, ştergere de date.
- *Actualizarea claselor* : adaugare, modificare, ştergere de clase.
- Realizarea *legăturilor între clase* : compunere, partiţionare etc..
- *Actualizarea instanţelor* : prin metode care schimbă starea internă a obiectului.

3. Restricţiile de integritate ale modelului OO:
- Orice obiect trebuie să respecte restricţiile impuse la definirea clasei din care face parte (protocol de obiect).
- Identificatorul obiectului asigură integritatea referirii la el (se atribuie şi se şterge 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.

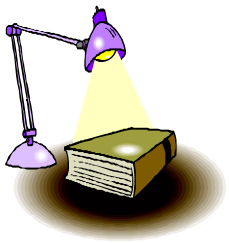
1. Modelul OO marchează trecerea la o *a treia generaţie* de BD. El aduce bazelor de date un plus de deschidere, independenţa logică a datelor de aplicaţii, reutilizarea obiectelor etc.
2. Modelul OO dă bune rezultate pentru probleme foarte *mari şi complexe* (principiile complexităţii sunt incluse în 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 specialişti. Există totuşi organisme internaţionale de standardizare – exemplu OMG (Object Management Group) – precum şi modele OO standardizate – exemplu ODMG (Object Data Management Group);
   - imaturitatea şi naturaleţea dezarmantă a tehnologiei OO;
   - elemente încă insuficient testate pentru SBDOO: controlul integrităţii, volume foarte mari, regăsirea etc.
5. *Exemple* de SGBDOO: O2 (Franţa), Jasmine (SUA) etc.
6. *Baza de date orientată obiect* este o mulţime de clase de obiecte persistente (în memoria externă), organizată coerent şi ordonată în ierarhii, partajată pentru utilizatorii concurenţi.

Note

1) O *comparaţie* între modelul relaţional şi orientat obiect vezi în Anexa 1.

2) După prezentarea de până acum a primelor două capitole, putem da o **definiţie completă şi explicativă** a noţiunii de *bază de date*, ca fiind un ansamblu de colecţii de date:
- *organizat* , pe niveluri de organizare a datelor (conceptual, logic, fizic), aşa cum reiese din arhitectura unui SBD pe niveluri;
- *coerent* , conform restricţiilor de integritate şi 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 redundanţă minimă şi controlată* , care este asigurată prin modelul de date implementat şi prin tehnicile de proiectare ale BD;
- *accesibil mai multor utilizatori în timp util* , adică mai mulţi utilizatori, concomitent, pot obţine informaţiile dorite atunci când are nevoie de ele.

1.5. Protecţia BD



- *Activitatea de protecţie a bazelor de date* este deosebit de importantă atât pentru modul de lucru pe calculatoare independente cât şi pentru modul de lucru în reţea de calculatoare.
- Asigurarea protecţiei pentru o bază de date revine atât în sarcina SGBD, prin funcţia de administrare (tot mai mult), cât şi în sarcina administratorului bazei de date.
- *Protecţia bazei de date* este un ansamblu de măsuri necesare asigurării securităţii şi integrităţii datelor.

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

Securitatea datelor semnifică interzicerea accesului la date pentru utilizatorii neautorizaţi.
Integritatea datelor înseamnă corectitudinea datelor încărcate, precum şi manipularea lor astfel încât să se respecte restricţiile de integritate ale modelului de date implementat.
În continuare, vom prezenta, pe scurt, cele două aspecte care compun activitatea de protecţie a unei baze de date. La fiecare aspect vom sintetiza activităţile aferente.

INTEGRITATEA datelor.

1. Integritatea semantică constă în prevenirea introducerii unor date incorecte în BD şi în prevenirea realizării unor prelucrări eronate. Acest lucru se asigură prin respectarea restricţiilor de integritate. Acestea pot fi *implicite* (asigurate automat de SGBD) şi *explicite* (asigurate prin proceduri incluse în programele de aplicaţie).
2. Controlul concurenţei la date constă în garantarea coerenţei (corectitudinii) şi simultaneităţii datelor în cazul prelucrării *tranzacţiilor* (unitatea logică de prelucrare) prin tehnici specifice (blocarea, interblocarea etc.).
3. Salvarea şi restaurarea. *Salvarea* este operaţia de stocare a datelor în copii de siguranţă 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 operaţia de refacere a consistenţei 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 şi controlul accesului la date constă în identificarea utilizatorilor şi restricţionarea accesului acestora, pentru diferite operaţii de prelucrare.
2. Viziunile (views) sunt partiţii logice ale BD definite pentru diferiţi utilizatori.
3. Procedurile speciale sunt rutine, oferite de SGBD, care efectuează anumite operaţii asupra datelor şi care sunt accesibile anumitori utilizatori.
4. Criptarea este operaţia de codificare a datelor în vederea stocării sau transmiterii datelor. În acest sena, se folosesc o mulţime de tehnici specifice: parole, algoritmi de criptare/decriptare, rutine speciale etc.

1.6. Rezumat

După ce s-a definit noţiunea de *organizare a datelor*, s-a prezentat *evoluţia* acesteia, pornind de la fişiere (primul mod de organizare a datelor în memoria externă) şi ajungându-se la ultimele tipuri de baze de date, cele orientate obiect. In această evoluţie, s-au urmărit câteva aspecte (caracteristici) comparative: structura de date, modul de prelucrare a datelor, redundanţa 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, persistenţă, organizare.
În continuare, se defineşte noţiunea de *fişier* şi, acesteia, i se prezintă câteva dintre cele mai importante aspecte de organizare a datelor: operaţii, 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 principalel *noţiuni* necesare: informaţia, 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, reţea, relaţional, orientat obiect. Fiecare model este prezentat în acelaşi mod: elementele componente, caracterizare generală. În acest fel se pot urmări, comparativ, cum se regăsesc noţiunile prezentate la începutul capitolului, în fiecare model în parte.
Capitolul se încheie cu o sinteză privind *protecţia bazelor de date.* Sunt luate în considerare cele două aspecte ale protecţiei: securitatea, integritatea. Fiecare dintre acestea este descompusă în principalele subactivităţi care trebuie să fie asigurate într-un sistem de bază de date, de către SGBD şi 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 tabelei
   e) proiecția, negația şi 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 – fişier

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 şi 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 şi 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.Ramakrishnan, 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 ş.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 ş. a. - "Baze de date – organizare, proiectare şi implementare ", ed. ALL, 1995

[CONN00] T.Connolly – "Database Systems", ed.Mc Graw Hill, 2000

Unitatea de învățare 2:
BAZE DE DATE – NOȚIUNI FUNDAMENTALE

Cuprins

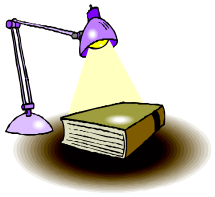
2.1. Obiective

După studiul acestei unități de învățare vei avea cunoştinte 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 şi protecția datelor; cu o redundanță minimă şi controlată, prin implementarea unui model de date şi 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 şi 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 şi în memoria externă (persistent).

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

- accesul cât mai rapid şi uşor la date;
- stocarea unui volum cât mai mare de date;
- creşterea compexității datelor;
- perfecționarea echipamentelor de culegere, stocare, transmitere şi prelucrare a datelor.

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

Sunt numeroase aspecte care priveşte trecerea de la fişiere 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 şi până astăzi, bazele de date au parcurs următoarele *generații* (determinate în principal de modelul de date implementat):

- generația I: BD arborescente şi rețea (până la sfârşitul anilor ’70);
- generația a-II-a: BD relaționale (sfârşitul anilor ’70 şi până acum);
- generația a-III-a: BD orientate obiect (sfârşitul anilor ’80 şi până acum).

Elementele componente ale unei baze de date

Am arătat faptul că bazele de date au evoluat din fişiere (vezi şi capitolul 2). Acest lucru se regăseşte şi în noțiunile utilizate în cele două moduri de organizare a datelor în memoria externă, aşa cum reiese din tabelul următor. De remarcat faptul că atât fişierele cât şi 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 ataşată o familie de caracteristici, care este formată din mai multe caracteristici, care iau valori dintr-un domeniu de valori.

În mod similar se citeşte rândul de jos: un sistem de fişiere este format din mai multe fişiere. Fiecare dintre acestea are ataşată 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 fişiere	Fişiere	Înregistrare	Câmpuri	Valori

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

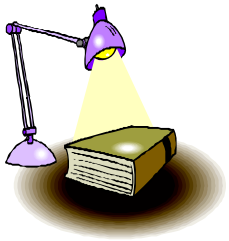
Domeniul de valori este dat de mulțimea valorilor posibile pentru o caracteristică (exemplu: culorile posibile pentru un automobil).
Caracteristica semnifică definirea şi descrierea unui anumit aspect (proprietăți) dintr-o entitate a lumii reale (exemplu: marca auto)
Familia de caracteristici este ansamblul caracteristicilor care se referă la aceeaşi 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 şi 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 şi 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 şi 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 şi protecția datelor;
- cu o redundanță minimă şi controlată, prin implementarea unui model de date şi 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 şi proiectare, care se ocupă de organizarea şi întreținerea BD.

Nivelurile de administratori de baze de date, după activitățile desfăşurate, sunt:
- administrator global realizează:
  - schema (conceptuală) a BD, pornind de la cerințele aplcației;
  - organizarea datelor la nivel logic (colaborează).
- administrator de aplicație realizează:
  - schema externă (logică) pentru o aplicație, pornind de la cerințele de prelucrare ale plicației;
  - organizarea datelor la nivel fizic (colaborare).
- administratorul bazei de date realizează:
  - schema internă (fizică) a datelor;
  - reorganizarea bazei de date;

- gestionează funcţionarea BD.

Sarcinile administratorului BD sunt structurate după activităţile de realizare a unei BD:

- la analiza şi proiectarea BD:
  - defineşte obiectivele BD;
  - colaborează la formularea cerinţelor aplicaţiei;
  - defineşte dicţionarul BD(schema, restricţiile de integritate etc.);
  - colaborează la schema externă şi la cea internă;
  - concepe protecţia datelor.
- la implementarea BD:
  - elaborează documentaţie;
  - defineşte regulile de implementare şi dare în folosinţă a BD;
  - asigură încărcarea BD din diferite surse de date.
- la exploatarea BD:
  - monitorizează accesul la date;
  - asigură protecţia datelor;
  - întreţine funcţionarea BD la parametrii proiectaţi.

Instrumente la dispoziţia administratorului BD pentru a-şi îndeplini sarcinile:

- instrumentele oferite de SGBD pentru: reorganizarea BD, refacerea BD, analize statistice, gestionarea dicţionarului de date, protecţia datelor.
- instrumente specifice create de administratori şi programatori.

Notă. Un *exemplu de administrare* a unei baze de date relaţionale se găseşte 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

Cunoaşterea concepteleor într-un domeniu de activitate este de mare importanţă pentru profesionişti. Acest lucru este valabil şi 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 apariţia primei generaţii de baze de date, ţinându-se cont de evoluţia din fişiere, ca mod de organizare a datelor în memoria externă.
Principalele *concepte prezentate* sunt: generaţii de baze de date, domeniul de valori, caracteristica, familia de caracteristici, colecţia 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 restricţiile de integritate şi protecţia datelor;
- cu o redundanţă minimă şi controlată, prin implementarea unui model de date şi 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 administartori, care sunt sarcinile administratorului BD, precum şi instrumentele necesare a fi la dispoziţia 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 redundanţă minimă şi 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) colecţia de proprietăţi
b) colecţia de date
c) familia de caracteristici
d) familia de înregistrări
e) caracteristica

2.6. Răspunsuri şi 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 ş. a. - "Baze de date – organizare, proiectare şi implementare ", ed. ALL, 1995

[VELU02] M.Velicanu, I.Lungu, M.Muntean, M.Iorga, S.Ionescu – "Oracle – platformă pentru baze de date", ed. Petrion, 2002

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

Cuprins

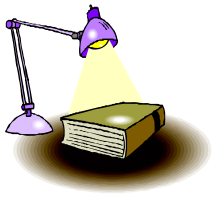
3.1. Obiective

După studiul acestei unități de învățare vei avea cunoştințe 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 şi 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 relaţionale (BDR)



- Bazele de date relaţionale (BDR) utilizează modelul de date relaţional şi noţiunile 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* (relaţii) împreună cu *legăturile* dintre ele.

Câteva dintre avantajele BDR faţă de fişiere:

CRITERIU	BDR	FIŞIERE
Independenţa datelor	logică şi fizică	fizică
Niveluri de structurare	conceptual, logic şi fizic	logic şi fizic
Deschidere şi portabilitate	mare	mică
reprezentarea şi utilizarea datelor	simplificat prin model	complicat
structura de date se păstrează	în dicţionarul BD	în programe.

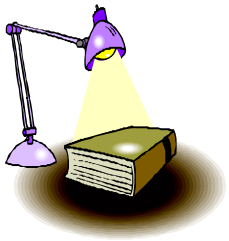
Concepte utilizate la organizarea datelor în memoria externă în BDR şi respectiv fişiere:

Organizare date în:
- fişiere → fişier → înregistrare → câmp → valori
- BDR → tabelă (relaţie) → tuplu (linie) → atribut (coloană) → domeniu valori

Condiţii minimale pentru ca un SGBD să fie relaţional:
- să implementeze modelul de date relaţionale prin LDD şi LMD;
- să implementeze cel puţin un limbaj relaţional.

Notă. Condiţiile complete ca un SGBD să fie relaţional sunt date de cele 13 reguli ale lui CODD.

3.3. Proiectarea BDR prin tehnica normalizării



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

a) **Aspecte generale privind proiectarea BD**

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

Pornind de la lumea reală se **identifică şi specifică** cerinţele aplicaţiei (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 **concepte şi instrumente** adecvate:

- *Schemele BD:*
  - Schema conceptuală specifică structura organizațională şi conținutul informațional al sistemului şi reprezintă un mod de comunicare între proiectanții şi utilizatorii BD;
  - Schema logică defineşte 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* (atribute).
- 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 şi 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 şi 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:

- *anomalia de ştergere* semnifică faptul că stergând un tuplu dintr-o tabelă, pe lângă informațiile care trebuie şterse, se pierd şi 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 şi alte informații utile (de exemplu valorile pentru cheie);
- *anomalia 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 şi prin imprimarea unor facilități sporite privind manipularea datelor.
Notă. Teoria normalizării a fost concepută inițial de către E.F.Codd, ulterior aducându-şi contribuția şi alți cercetători.
Normalizarea utilizează ca metodă d**escompunerea** (top-down) unei tabele în două sau mai multe tabele, păstrând informații (atribute) 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) şi una suplimentară (BCNF) care revizuieşte FN3.
O **bază de date este în** FNi , i=1,5 dacă toate tabelele sale sunt în FNi.
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 şi rețea *se transformă* în tabele cu atribute elemntare.
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 elemntare.

FN2
O **tabelă este în FN2** dacă şi numai dacă este în FN1 şi fiecare atribut noncheie al tabelei este *dependent funcțional complet* de cheie.
Un atribut B al unei tabele *depinde funcțional* de atributul A al aceleiaşi 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 aceleiaşi 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 tabelei inițiale în două tabele, ambele conținând atributul intermediar (B).

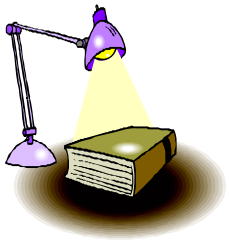
FN3
O **tabelă este în FN3** dacă şi numai dacă este în FN2 şi fiecare atribut noncheie depinde în mod *netranzitiv* de cheia tabelei.
Într-o tabelă T, fie A,B,C trei atribute cu A cheie. Dacă B depinde de A (A $\longrightarrow$ B) şi C depinde de B (B $\longrightarrow$ C) atunci C depinde de A în mod *tranzitiv*.
Eliminarea dependențelor tranzitive se face prin descompunerea tabelei 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 riguruasă 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 şi numai dacă este în FN3 şi 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 şi numai dacă mulțimea valorilor lui B ce corespunde unei perechi de date (A,C), depinde numai de o valoare a lui A şi este independentă de valorile lui C.

FN5

O **tabelă este în FN5** dacă şi numai dacă este în FN4 şi fiecare *dependență joncțiune* este generată printr-un candidat cheie al tabelei.
În tabela T (A,B,C) se menține *dependența joncțiune* (AB, AC) dacă şi 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 şi calculul relațional



- Pentru manipularea datelor conform modelului relațional se folosesc *operatorii* din algebra şi 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) şi a fost propus de E.F.Codd.
Predicatul este o relație care se stabileşte între anumite elemente şi 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 tabele) sau domeniu (valorile sunt dintr-un domeniu al unei tabele).
Cuantificatorii (operatorii) utilizați în calculul relațional sunt: universal ($\forall$) şi 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 tabelelor (relațiilor), şi 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 relaţionali şi operanzi.
Operanzii sunt întotdeauna tabele (una sau mai multe).
Rezultatul evaluării unei expresii relaţionale este format dintr-o singură tabelă.
Algebra relaţională are cel puţin puterea de regăsire a calculului relaţional. O expresie din calculul relaţional se poate transforma într-una *echivalentă* din algebra relaţională şi invers.
Operatorii din algebra relaţională pot fi grupaţi în două categorii (R1, R2, R3 sunt relaţii (tabele)):

1. *Operatori pe mulţimi*

REUNIUNEA: R3 = R1 $\cup$ R2, unde R3 va conţine tupluri din R1 sau R2 luate o singură dată.
DIFERENŢA: R3 = R1 \ R2, unde R3 va conţine tupluri din R1 care nu se regăsesc în R2
PRODUSUL CARTEZIAN: R3 = R1 $\times$ R2, unde R3 va conţine tupluri construite din perechi (x1x2), cu x1$\in$R1 şi x2$\in$R2.
INTERSECŢIA:R3 = R1 $\cap$ R2, unde R3 va conţine tupluri care se găsesc în R1 şi R2 în acelaşi timp.

2. *Operatori relaţionali speciali*

SELECŢIA: din R1 se obţine o subtabelă R2, care va conţine o submulţime din tuplurile iniţiale din R1 ce satisfac un predicat (o condiţie).
Numărul de atribute din R2 = numărul de atribute din R1
Numărul de tupluri din R2 < numărul de tupluri din R1.
PROIECŢIA: din R1 se obţine o subtabelă R2, care va conţine o submulţime din atributele iniţiale din R1 şi fără tupluri duplicate
Numărul de atribute din R2 < numărul de atribute din R1
Numărul de tupluri din R2 $\leq$ numărul de tupluri din R1.
JONCŢIUNEA: derivaţie a produsului cartezian, ce presupune utilizarea unui calificator care să permită compararea valorilor unor atribute din R1 şi R2, iar rezultatul în R3.
R1 şi R2 trebuie să aibă unul sau mai multe atribute comune care au valori comune.
Note.

1) Codd a introdus şase operatori *de bază* (reuniunea, diferenţa, produsul cartezian, selecţia, proiecţia , joncţiunea) şi doi operatori *derivaţi* (intersecţia şi diviziunea).
2) Ulterior au fost introduşi şi alţi operatori derivaţi (*speciali*).
3) Algebra relaţională este prin definiţie *neprocedurală* (descriptivă), iar calculul relaţional permite o manieră de căutare *mixtă* (procedurală/neprocedurală).

Transformarea

Oferă o putere de regăsire *echivalentă* cu cea din calculul şi algebra relaţională.
Se bazează pe *transformarea* (mapping) unui atribut sau grup de atribute într-un atribut dorit prin intermediul unor relaţii.
Rezultatul este o relaţie (tabelă) care se poate utiliza într-o altă transformare.

Grafica

Oferă *interactivitate* mare pentru constrirea 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 defineşte *noţiunea* de bază de date relaţională (BDR) ţinând cont de modelul de date relaţional. Se face o *comparaţie* a noţiunilor utilizate în BDR cu cele utlizate la lucrul cu fişiere.
Realizarea unei aplicaţii 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 aplicaţiile practice.
În finalul capitolului se prezintă o sinteză a două domenii importante din teoria relaţională: *calculul relaţional* şi *algebra relaţională*. Fiecare dintre cele două domenii este definit pe scurt, iar apoi sunt prezentaţi cei mai importanţi *operatori* aferenţi. Sunt citate exemple de limbaje relaţionale clasificate după tehnica folosită la implementare.

3.6. Teste de autoevaluare

1. O tabelă este în FN3 dacă:
   a) este în FN2 şi fiecare atribut noncheie depinde în mod netranzitiv de cheia tabelei
   b) este în FN1 şi fiecare atribut cheie depinde tranzitiv de atributele noncheie
   c) este în FN2 şi are dependenţe complete
   d) este în FN1 şi are dependenţe funcţionale incomplete
   e) este în FN2 şi are cel puţin o dependenţă funcţională completă între atributele noncheie şi cheia tabelei

2. Calculul relaţional:
   a) are drept construcţie de bază expresia de calcul tuplu sau expresia de calcul domeniu
   b) este implementat în limbajul ISBL
   c) conţine operatorul de joncţiune
   d) stă la baza limbajelor procedurale universale
   e) foloseşte noţiunea 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 şi 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

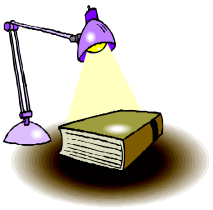
4.1. Obiective

După studiul acestei unități de învățare vei avea cunoştințe 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

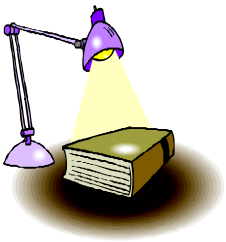


- Activitatea de realizare a unei baze de date trebuie să înceapă cu *organizarea* acesteia, adică pregătirea acțiunii. În cadrul acestei pregătiri, ținând cont de obiectivele urmărite pentru realizarea bazei de date, se va face un fel de inventar a ceea ce este un minim necesar pentru a se putea realiza baza de date. Se poate astfel, din start, hotarî dacă activitatea va demara imediat sau dacă mai sunt necesare alte acțiuni pregătitoare.

- Prezentăm, 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, fişiere 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 şi integritatea), se referă la:
   - instrumente oferite de către SGBD (autorizare acces, fişiere 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

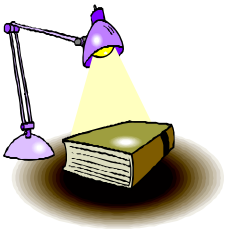


- Atunci când dorim să realizăm o bază de dată trebuie să ştim clar ce avem de făcut, adică să stabilim obiectivele activității nostre. În acest sens, câteva dintre cele mai importante *obiective*, le prezentăm în continuare. Acestea, ar trebui să constituie un set minim de obiective, de care să se țină cont la realizarea unei baze de date:

1. *Partiționarea* semnifică faptul că aceleaşi date trebuie să poată fi folosite în moduri diferite de către diferiți utilizatori.
2. *Deschiderea* se referă la faptul că datele trebuie să fie uşor adaptabile la schimbările care pot apărea (actualizarea structurii, tipuri noi de date etc.).
3. *Eficiența* are în vedere stocarea şi 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 actvitate frecventă pe bazele de date şi de aceea cererile de regăsire trebuie să poată fi adresate uşor de către toate categoriile de utilizatori, după diferite criterii.
6. *Accesul* înseamnă modul de localizare a datelor şi acest lucru trebuie să poată fi realizat prin diferite moduri de acces, rapid şi uşor.
7. *Modularizarea* presupune faptul că realizarea BD trebuie să se poată face modular pentru generalitate şi posibilitatea lucrului în echipă.
8. *Protecţia* bazei de date trebuie asigurată sub ambele aspecte: securitatea şi integritatea datelor.
9. *Redundanţa* se asigură în limite acceptabile prin implementarea unui model de date pentru baze de date şi prin utilizarea unei tehnici de proiectare a BD. Se asigură astfel, o redundanţă minimă şi controlată.
10. *Independenţa* datelor faţă de programe trebuie asigurată atât la nivel logic cât şi şi fizic.

4.4. Etape în realizarea unei BD



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

1. Analiza de sistem.

Scopul acestei activităţi este de a evidenţia cerinţele aplicaţiei şi resursele utilizate (studiul), precum şi de a evalua aceste cerinţe prin modelare (analiza).

STUDIUL situaţiei existente se realizează prin:

- definirea caracteristicilor generale ale unităţii;
- identificarea activităţilor desfăşurate;
- identificarea resurselor existente (informaţionale, umane, energetice, echipamente, financiare etc.);
- identificarea necesităţilor de prelucrare.

ANALIZA sistemului existent urmează investigării (studiului) şi utilizează informaţiile obţinute de aceasta.

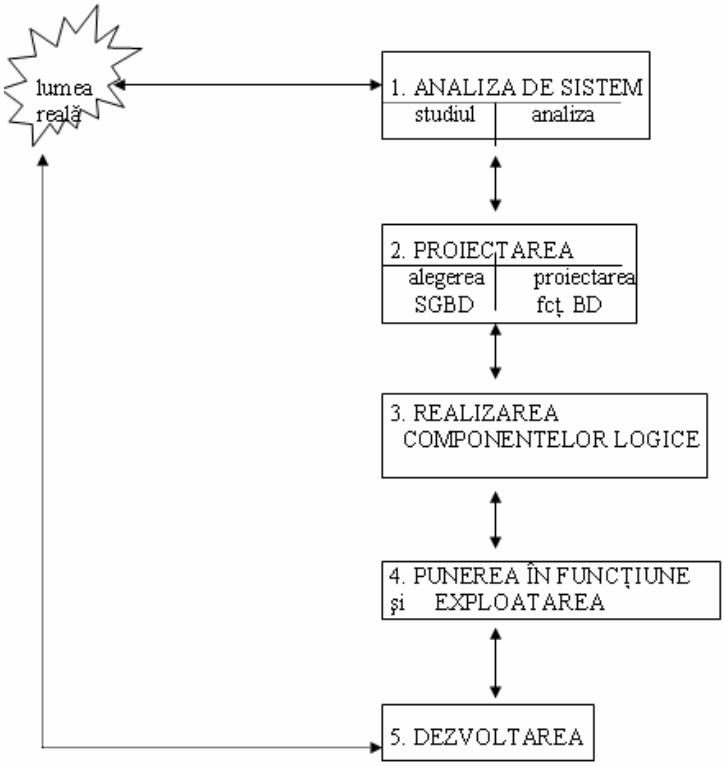


Fig. 4.1. Activităţile parcurse pentru realizarea unei baze de date

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

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

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

Rezultatul analizei structurale: modelul static (structural) numit şi 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 actvitatea de proiectare, schemele relaţiilor (tabelelor). Pentru această translatare există un set de 14 reguli [vezi RICC01] care pot fi aplicate.

b) *Analiza dinamică* evidenţiază comportamentul elementelor sistemului la anumite evenimente. Una din tehnicile utilizate este diagrama *stare-tranziţie*. 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 tranziţiilor admise între stări.
4) Construirea diagramei stare-tranziţie.

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 ieşiri. 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 şi 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 SBGD**-ului se face ținând cont de două aspecte: cerințele aplicației (utilizatorului) şi performanțele tehnice ale SGBD-ului.

Cerințele aplicației se referă la:

- volumul de date estimat a fi memorat şi prelucrat în BD;
- complexitatea problemei de rezolvat;
- ponderea şi frecvența operațiilor de intrare/ieşire;
- 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 şi 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 şi concurența de date;
    - elemente de multimedia;
    - elemente de CASE;
    - interfețe de comunicare;
    - autodocumentarea;
    - instrumente specifice oferite.

b) **Proiectarea funcțiilor BD** (vezi şi [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) şi ținând cont de modelul de date implementat de SGBD-ul ales. Se vor proiecta schemele: conceptuală, externă, internă.

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

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 (fişier), 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 şi 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 şi de cerințele utilizator. Schema externă devine operațională prin construirea unor viziuni (view) cu SGBD-ul şi acordarea drepturilor de acces. Datele într-o viziune pot proveni din una sau mai multe colecții şi nu ocupă spațiul fizic.

Proiectarea schemei interne presupune stabilirea structurilor de memorare fizică a datelor şi 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 şi definirea unui model fizic de alocare (a se vedea dacă SGBD-ul permite explicit acest lucru);
2. definirea unor *indecş*i pentru accesul direct, după cheie, la date;
   Un ghid privind activitatea de indexare [RAGE00] pentru baze de date, poate include *paşii:*
   - 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 indecşilor este dat de două aspecte: indecşii se şterg dacă operațiile care-i implică sunt rare (se reduce astfel spațiul) şi indecşii 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. Clusterele generează avantaj la regăsire şi dezavantaj la actualizare, în ceea ce priveşte timpul de răspuns.

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

- fluxul informațional;
- modulele de încărcare şi manipulare a datelor;
- interfețele specializate;
- integrarea elementelor proiectate cu organizarea şi 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 ieşiri, intrări, prelucrări şi colecțiile de date.

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

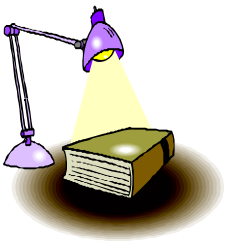
4. Punerea în funcțiune şi exploatarea

Se testează funcțiile BD mai întâi cu date de test, apoi cu date reale.
Se încarcă datele în BD şi 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 uşor şi "din mers".
Factorii pot fi: organizatorici, datorați progresului tehnic, rezultați din cerințele noi ale beneficiarului, din schimbarea metodologiilor etc.

4.5. Utilitatea metodologiei UML în proiectarea bazelor de date



- Complexitatea deosebita a sistemelor actuale a determinat abordarea pe scara larga a principiilor orientarii obiectuale nu doar în ceea ce priveste programarea ci si în proiectarea lor, lucru care a avut consecinte si asupra proiectarii bazelor de date. În general modelarea orientata obiect (gen UML) este privita ca apanajul doar al proiectarii (sau dezvoltarii) aplicatiilor orientate obiect.

Aceasta abordare se dovedeste însa **limitata** având în vedere doua aspecte :
1. *Pe de o parte proiectarea aplicatiilor (orientate obiect) cu baze de date separata de proiectarea bazei de date va produce un* ***nivel suplimentar de mapare*** *a entitatilor care figureaza în cele doua arii* ***la nivel conceptual***. Proiectarea unui sistem integrat din care sa rezulte concertat ce obiecte vor fi stocate persistent în baza de date, ce obiecte vor face parte din package -urile sau modulele aplicatiilor rezidente sub forma componentelor si modul cum vor interactiona între ele se poate dovedi o alternativa mai buna. Singurul substrat suplimentar (care poate fi modelat si el prin stereotipuri UML) ar fi cel al maparii entitatilor conceptuale în structurile oferite de modelul logic al bazei de date pentru stocare si regasire.

2. *Pe de alta parte modelele logice ale bazelor de date relationale comerciale propun astazi o serie de concepte noi cum sunt procedurile stocate si declansatoarele care nu fac parte din modele relationale traditionale si de obicei nu sunt luate în considerare în faza proiectarii bazei de date* . Valorificarea direct în proiectarea conceptuala a acestora se poate face folosind concepte orientate obiect si mecanismele de extensibilitate ale unui limbaj de modelare cum este UML.
Motivatiile pentru care metodologia UML ar fi potrivita si pentru specificarea modelelor implicate în proiectarea bazelor de date (fata metodele traditionale) ar putea fi rezumate astfel:

- modelul structural al UML încorporeaza toate conceptele modelului entitate –relatie (cel mai popular model pentru proiectarea conceptuala a bazelor de date) plus conceptele specifice modelelor semantice (generalizare, agregare) si abordarii orientate obiect (mostenire, polimorfism, supraîncarcare). Limbajul UML poseda

mecanismele de extensibilitate (restrictii, stereotipuri, note, etichete) care fac posibila specializarea modelului în functie de aspectele particulare ale sistemelor cu baze de date (flexibilitate sporita în modelare);

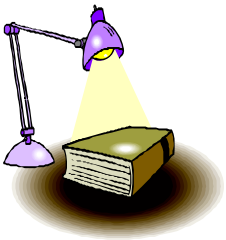
- conceptele limbajului UML propun un vocabular comun între modelele rezultate din proiectarea sistemului ca ansamblu si cele din proiectarea specifica bazei de date;
- metodologia UML (procesul unificat) favorizeaza munca în echipa: scopul folosirii UML pentru modelarea proceselor afacerii, dezvoltarii aplicatiilor si modelarii bazei de date este integrarea echipelor de dezvoltare pentru a împiedica construirea unei arhitecturi fara implicarea corespunzatoare în acest proces a tuturor echipelor care concura la realizarea sistemului în ansamblu;
- integrarea instrumentelor de dezvoltare din diferitele arii de interes din care rezulta arhitectura sistemului pe baza unei platforme conceptuale comune. Instrumentele CASE bazate pe metodologii obiectuale ofereau initial suport pentru proiectarea generala a sistemului si mai putin pentru proiectarea de detaliu. Cu alte cuvinte arhitectura sistemului era sprijinita prin instrumente *upper CASE* destul de slab integrate cu cele *lower CASE*.

Adoptarea UML ca limbaj de modelare a produs în acest sens doua mutatii:

- aparitia de noi instrumente CASE care ofera suport pentru întreg ciclu de dezvoltare pâna la obtinerea unei versiuni complete a sistemului (gen Rational Rose);
- integrarea de noi facilitati dedicate modelarii în mediile de dezvoltare integrate comerciale (gen suitele de dezvoltare de la Borland, Microsoft sau Oracle) cu posibilitati de proiectare vizuala din care sa rezulte componentele continând codul sursa sau scripturile pentru generarea schemelor bazelor de date.

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

4.6. Proiectarea bazelor de date relaţionale – o extensie UML



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

Diagrama claselor din UML permite modelarea unor aspecte specifice proiectării bazelor de date relaţionale, dar nu acoperă în întregime această problematică, de notat faptul că nu prevede noţiunea de atribute cheie care sunt mecanismul principal de relaţionare a tabelelor. Pentru a surprinde mai bine aceste aspecte este bine să se apeleze la diagrama entitate asociere (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 relaţională sau orientată obiect, prin clase reprezentându-se tabelele, iar prin atribute coloanele acestora. Dacă se alege pentru implementare un mediu relaţional, atunci diagrama claselor poate fi uşor translatată într-o diagramă logică entitate asociere. Claselor persistente şi atributelor acestora le corespund entităţile logice şi atributele lor, iar relaţiilor dintre clase le corespund relaţii între entităţi.

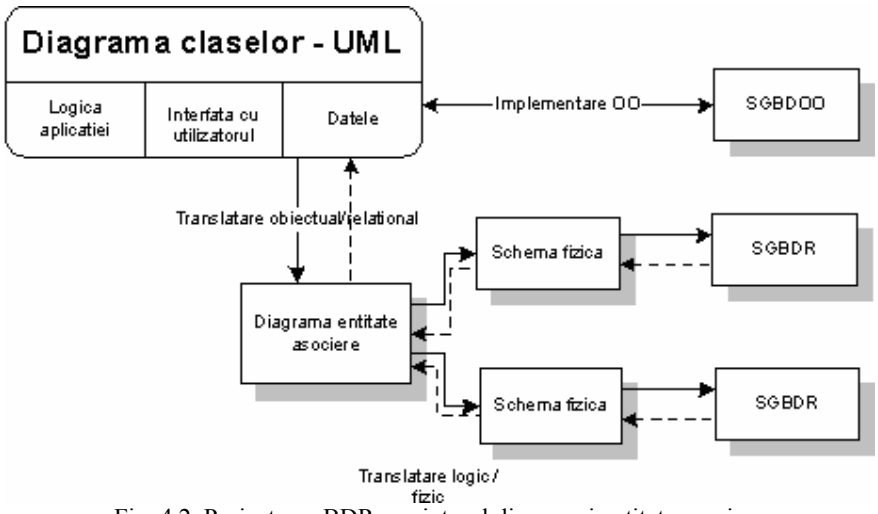
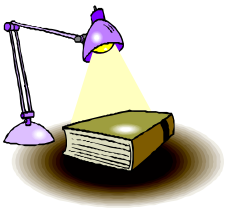


Fig. 4.2. Proiectarea BDR cu ajutorul diagramei entitate asociere

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

4.7. Eficienţa bazelor de date



- *Efectele* directe şi indirecte ale bazelor de date conduc spre eficienţă, mai ales pentru aplicaţiile mari şi complexe, acolo unde alte produse informatice nu fac faţă.

Efectele utilizării BD conduc spre **avantajele** oferite de acestea:
- elimină redundanţele necontrolate;
- creşte productivitatea activităţii informatice;
- oferă suport informaţional optim pentru conducere şi execuţie;
- permite interogarea datelor de către toate categoriile de utilizatori;
- conduce spre aplicaţii deschise.

Efortul depus pentru realizarea şi întreţinerea unei BD este considerabil şi de aceea investiţia necesară trebuie fundamentată pentru a putea fi recuperată.
Fundamentarea investiţiei pentru o aplicaţie cu BD se face prin studiul de fezabilitate (analiza preliminară) care va conţine un *studiu al costurilor* şi o *evaluare a efortului* de realizare şi întreţinere.
Etape care trebuie parcurse pentru stabilirea eficienţei BD:
1. *Studiul costurilor iniţiale* (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, informaţionale etc.);
   - costul consultanţei tehnice.
2. *Evaluarea costurilor* de funcţionare a BD
   - costul de punere în funcţiune;

- costul de exploatare;
- costul de întreținere şi dezvoltare.

3. *Calcule de rentabilitate*
   - utilizarea unor metode specifice (exemplu: Analiza Cost Beneficiu)
   - utilizarea unor instrumente hardware şi software specializate pentru BD (cresc performanța, scad portabilitatea).

4.8. Rezumat

Activitatea de realizare a unei baze de date este complexă şi 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 şi după aceea, trebuie avută în vedere *eficiența* acestei activități. Acest lucru este important pentru că investiția pentru realizarea unei baze de date este mare şi ea se recuperează în timp. Eficența 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) aceleaşi date nu pot fi utilizate în moduri diferite
   b) creşterea 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 şi 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 ş.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.Ramakrishnan, 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 ş.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

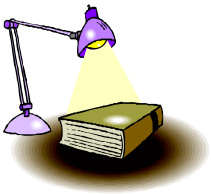
5.1. Obiective

După studiul acestei unități de învățare vei avea cunoştințe 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 şi principalele concepte utilizate la realizarea unei astfel de baze de date;*
- *proiectarea unei BDOO, conform unei anumite metodologii, care precizează nişte paşi care trebuie parcurşi;*
- *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 şi complexitatea ridicată a datelor, au determinat *evoluția* spre sistemele orientate obiect.

- *Dezvoltarea* BDOO (generația a treia de BD) a fost favorizată (începand 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 şi instrumentelor utilizate pentru construcția sistemelor cu obiecte.
- *Aplicarea* tehnologiei orientată obiect la SBD a însemnat *avantaje* privind îmbunătățirea structurii de date (modelul orientat obiect) şi 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 şi distribuirea* sunt asigurate atât între date, cât şi între programe.
3. *Structura de date* este simplificată foarte mult, de unde rezultă uşurința în utilizare şi portabilitatea ridicată a sistemelor rezultate.
4. Lucrul cu obiecte ne apropie firesc de *lumea reală*, în care se gasesc obiecte, care au proprietați, 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

Ramân 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 liniara de definiri şi apeluri de obiecte şi clase de obiecte.
BDOO este o mulțime de clase de obiecte persistente (în memoria externă), organizată coerent şi ordonată în ierarhii, partajată pentru utilizatorii concurenți.
SGBDOO este suportul software complet care asigură descrierea (structura obiectului), manipularea şi protecția datelor organizate dupa modelul OO.

Sinteza conceptelor din BDOO:

Lumea reală (entități fizice) —se reprezintă prin→ Clase de obiecte —au drept instanțe→ Obiectele —conțin→ Proprietăți (variabile de instanță) → Domeniu de valori şi stări

Un obiect se identifică prin:

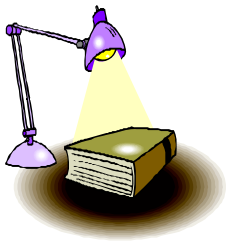
- nume
- identificator
- metode
- mesaje
- implementare (privată)
- interfață (publică)
- caracteristici fundamentale

Paralelă între noțiunile utilizate în sistemele relaționale şi orientate obiect:

Sistem orientat	Ierarhia de	Clasă de	Obiect	Variabilă de	Identificator

obiect	clase	obiecte		instanță	
Sistem relațional	Schema BD	Tabelă	Tuplu	Atribut	Cheie

5.3. Elemente de proiectare a BDOO



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

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

a)Analiza

- *Rolul* activității de analiză este de a preciza universul de discurs (se identifică entitățile fizice şi operațiile necesare).
- *Paşii* parcurşi 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 dupa proprietăți şi comportament obtinandu-se clasele de obiecte;
  - se identifică secvențele de mesaje către obiecte (legăturile dintre obiecte si clase de obiecte);
  - se determină secvențele de activități şi ciclu de viață al obiectelor (modelarea).
- *Parcurgerea* paşilor de mai sus se face prin reluari succesive a unor paşi 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 şi *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 moştenire.

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*) şi se descrie prin: tip, lungime, restricții.

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

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

Metoda diagramei dă o *imagine grafică* a structurii obiectelor şi a legaturilor dintre ele.

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

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

-definiții ale obiectelor ;
-enumerarea obiectelor şi a proprietăților acestora ;
-enumerarea domeniilor şi 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 singura valoare (domeniu cu o valoare) şi au proprietăți care nu sunt obiecte
- *compozite* sunt cele care au cel puțin o proprietate cu valori multiple şi 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 legatura între alte obiecte (apar ca proprietăți ale obiectelor care le leagă);
- *agregate* sunt obiectele care reprezintă grupuri de entități şi conțin proprietăți de grup ce se transmit tuturor instanțelor subclaselor (moştenirea).

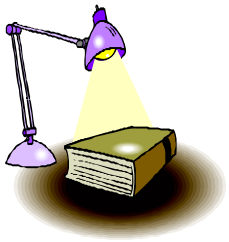
c)Elaborarea programelor

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

Elaborarea programelor se desfăşoară pe baza specificațiilor obtinute prin activitatea de proiectare a BDOO.
Programele sunt secvențe liniare de descrieri de obiecte şi de apeluri de obiecte (mesaje).
În SBDOO se foloseşte intens *reutilizarea* programelor.

d) Implementarea şi exploatarea
Implementarea şi 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 şi 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 şi apoi o scurtă explicație la fiecare.

1. *Gestiunea mediului* limbajului de programare: trebuie asigurată tratarea conceptelor modelului orientat obiect prin limbaje de programare corespunzatoare.
2. *Reprezentarea obiectelor*: modelul utilizat poate fi simplu (obiecte mici şi puține) sau complex (obiecte mari şi multe — structurate).
3. *Gestiunea persistenței* : se descriu şi manipuleaza date permanente (persistente), în acelaşi mod cu cele temporare. Persistența se asigura 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 şi 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 permita lucrul distribuit în rețea şi să se separe prelucrarea (consumă memorie interna) de stocarea (consumă memorie externa) obiectelor.
7. *Gestiunea versiunilor*: trebuie păstrate stările succesive ale obiectelor şi evoluția lor în timp.
8. *Gestiunea obiectivelor* traditionale ale unei BD.

5.5. Rezumat

Se defineşte noţiunea de baze de date orientate obiect (BDOO) în contextul trecerii de la sistemele relaţionale la a treia generaţie de baze de date.
Sunt prezentate câteva caracteristici specifice unei BDOO şi principalele concepte utilizate la realizarea unei astfel de baze de date.
Proiectarea unei BDOO se face conform unei anumite metodologii, care precizează nişte paşi care trebuie parcurşi. Sinteza acestori paşi ne conduce spre o serie de activităţi care se regăsesc în procesul de realizare a unei BDOO: analiza, proiectarea, elaborarea programelor, implementarea şi exploatarea. Fiecare dintre aceste activităţi este prezentată pe scurt, precizându-se rolul ei, o descriere, precum şi eventualele metode utilizate.
În finalul capitolului sunt prezentate câteva dintre tehnicile utilizate pentru implementarea într-o BDOO, a principalelor elemente care dau facilităţile 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) independenţa datelor faţă de programe este parţială
   d) se tratează în mod unitar datele, programele şi comunicaţia
   e) asigură comunicaţia atât între date cât şi între programe

6. La proiectarea unei BDOO se pot utiliza metodele:
   a) diagrama obiectelor
   b) specificaţiile obiectelor
   c) normalizarea
   d) diagramele de dependenţă
   e) distribuirea obiectelor

7. Tehnici de implementare a BDOO sunt:
   a) portabilitatea, partiţionarea
   b) gestiunea mediului, reprezentarea obiectelor
   c) gestiunea persistenţei, gestiunea memoriei
   d) extensibilitatea, distribuirea majoră
   e) flexibilitatea, portabilitatea

5.7. Răspunsuri şi 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