

# CAPITOLUL 1.

## TEORIA BAZELOR DE DATE RELAȚIONALE

### 1.1. MODELUL RELAȚIONAL

**Modelul relațional** a fost *propus* de către IBM și a revoluționat reprezentarea datelor făcând trecerea la generația a doua de baze de date.

Modelul este *simplic*, are o solidă *fundamentare teoretică* fiind bazat pe teoria seturilor (ansamblurilor) și pe logica matematică. Pot fi reprezentate toate tipurile de structuri de date de mare complexitate, din *diferite domenii* de activitate.

Modelul relațional este definit prin: structura de date, operatorii care acționează asupra structurii și restricțiile de integritate.

1) Conceptele utilizate pentru definirea **structurii de date** sunt: domeniul, tabela (relația), atributul, tuplul, cheia și schema tabelii.

*Domeniu* este un ansamblu de valori caracterizat printr-un nume. El poate fi explicit sau implicit.

*Tabela/relația* este un subansamblu al produsului cartezian al mai multor domenii, caracterizat printr-un nume, prin care se definesc atributele ce aparțin aceleași clase de entități.

*Atributul* este coloana unei tabeli, caracterizată printr-un nume.

*Cheia* este un atribut sau un ansamblu de atribute care au rolul de a identifica un tuplu dintr-o tabelă. *Tipuri* de chei: primare/alternate, simple/comune, externe.

*Tuplul* este linia dintr-o tabelă și nu are nume. Ordinea liniilor (tupluri) și coloanelor (atribute) dintr-o tabelă nu trebuie să prezinte nici-o importanță.

*Schema tabelii* este formată din numele tabelii, urmat între paranteze rotunde de lista atributelor, iar pentru fiecare atribut se precizează domeniul asociat.

*Schema bazei de date* poate fi reprezentată printr-o diagramă de structură în care sunt puse în evidență și legăturile dintre tabele. Definirea legăturilor dintre tabele se face *logic* construind asocieri între tabele cu ajutorul unor atribute de legătură. Atributele implicate în realizarea legăturilor se găsesc fie în tabelele asociate, fie în tabele distincte construite special pentru legături. Atributul din tabela inițială se numește *cheie externă* iar cel din tabela finală este *cheie primară*. Legăturile posibile sunt *1:1*, *1:m*, *m:n*. Potențial, orice tabelă se poate lega cu orice tabelă, după orice atribute.

---

Legăturile se stabilesc la momentul descrierii datelor prin limbaje de descriere a datelor (LDD), cu ajutorul restricțiilor de integritate. Practic, se stabilesc și legături dinamice la momentul execuției.

2) **Operatorii modelului relațional** sunt operatorii din algebra relațională și operatorii din calculul relațional.

**Algebra relațională** este o colecție de operații formale aplicate asupra tabelelor (relațiilor), și a fost concepută de E.F.Codd. Operațiile sunt aplicate în *expresiile algebrice relaționale* care sunt cereri de regăsire. Acestea sunt compuse din operatorii relaționali și operanzi. *Operanzii* sunt întotdeauna tabele (una sau mai multe). *Rezultatul* evaluării unei expresii relaționale este format dintr-o singură tabelă.

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

Codd a introdus șase operatori *de bază* (reuniunea, diferența, produsul cartezian, selecția, proiecția, joncțiunea) și doi operatori *derivați* (intersecția și diviziunea). Ulterior au fost introduși și alți operatori derivați (*speciali*). În acest context, *operatorii* din algebra relațională pot fi grupați în două categorii: pe mulțimi și speciali.

*Operatori pe mulțimi* ( $R_1, R_2, R_3$  sunt relații (tabele)) *sunt*:

- *Reuniunea*.  $R_3 = R_1 \cup R_2$ , unde  $R_3$  va conține tupluri din  $R_1$  sau  $R_2$  luate o singură dată;
- *Diferența*.  $R_3 = R_1 \setminus R_2$ , unde  $R_3$  va conține tupluri din  $R_1$  care nu se regăsesc în  $R_2$ ;
- *Produsul cartezian*.  $R_3 = R_1 \times R_2$ , unde  $R_3$  va conține tupluri construite din perechi  $(x_1, x_2)$ , cu  $x_1 \in R_1$  și  $x_2 \in R_2$ ;
- *Intersecția*.  $R_3 = R_1 \cap R_2$ , unde  $R_3$  va conține tupluri care se găsesc în  $R_1$  și  $R_2$  în același timp, etc.

*Operatori relaționali speciali sunt*:

- *Selecția*. Din  $R_1$  se obține o subtabelă  $R_2$ , care va conține o submulțime din tuplurile inițiale din  $R_1$  ce satisfac un predicat (o condiție). Numărul de atribute din  $R_2$  este egal cu numărul de atribute din  $R_1$ . Numărul de tupluri din  $R_2$  este mai mic decât numărul de tupluri din  $R_1$ .
- *Proiecția*. Din  $R_1$  se obține o subtabelă  $R_2$ , care va conține o submulțime din atributele inițiale din  $R_1$  și fără tupluri duplicate. Numărul de atribute din  $R_2$  este mai mic decât numărul de atribute din  $R_1$ .
- *Joncțiunea* este o derivație a produsului cartezian, ce presupune utilizarea unui calificator care să permită compararea valorilor

---

unor atribute din R1 și R2, iar rezultatul în R3. R1 și R2 trebuie să aibă unul sau mai multe atribute comune care au valori comune.

Algebra relațională este prin definiție *neprocedurală* (descriptivă), iar calculul relațional permite o manieră de căutare *mixtă* (procedurală/neprocedurală).

**Calculul relațional** se bazează pe *calculul predicatelor* de ordinul întâi (domeniu al logicii) și a fost propus de E.F. Codd. *Predicatul* este o relație care se stabilește între anumite elemente și care poate fi confirmată sau nu. *Predicatul de ordinul 1* este o relație care are drept argumente variabile care nu sunt predicate. *Variabila* poate fi de tip tuplu (valorile sunt dintr-un tuplu al unei tabeli) sau domeniu (valorile sunt dintr-un domeniu al unei tabeli). *Cuantificatorii* (operatorii) utilizați în calculul relațional sunt: universal ( $\forall$ ) și existențial ( $\exists$ ).

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

Expresia relațională de calcul este formată din: operația de efectuat, variabile (tuplu respectiv domeniu), condiții (de comparație, de existență), formule bine definite (operanzi-constante, variabile, funcții, predicate; operatori), cuantificatori.

Pentru implementarea acestor operatori există *comenzi specifice* în limbajele de manipulare a datelor (LMD) din sistemele de gestiune a bazelor de date relaționale (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);
- transformare (SQL, SQUARE);
- grafică (QBE, QBF).

*Transformarea* oferă o putere de regăsire *echivalentă* cu cea din calculul și algebra relațională. Se bazează pe *transformarea* (mapping) unui atribut sau grup de atribute într-un atribut dorit prin intermediul unor relații. *Rezultatul* este o relație (tabelă) care se poate utiliza într-o altă transformare.

*Grafica* oferă *interactivitate* mare pentru construirea cererilor de regăsire. Utilizatorul specifică cerea *alegând* sau completând un ecran structurat grafic. Poate fi *folosit* de către toate categoriile de utilizatori în informatică.

3) **Restricțiile de integritate ale modelului relațional** sunt structurale și comportamentale.

---

*Restricțiile structurale sunt:*

- *Restricția de unicitate a cheii.* Într-o tabelă nu trebuie să existe mai multe tupluri cu aceeași valoare pentru ansamblul cheie;
- *Restricția 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);
- *Restricția entității.* Într-o tabelă, atributele din cheia primară nu trebuie să ia valoarea NULL.

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 a bazelor de date relaționale - BDR).

*Restricțiile de comportament* sunt cele care se definesc prin comportamentul datelor și țin cont de valorile din BDR:

- *Restricția de domeniu.* Domeniul corespunzător unui atribut dintr-o tabelă trebuie să se încadreze între anumite valori;
- *Restricții temporare.* Valorile anumitor atribute se compară cu niște valori temporare (rezultate din calcule etc.).

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

Restricțiile de integritate suportate de Oracle sunt:

- NOT NULL nu permite valori NULL în coloanele unei tabele;
- UNIQUE nu sunt permise valori duplicat în coloanele unei tabele;
- PRIMARY KEY nu permite valori duplicate sau NULL în coloana sau coloanele definite astfel;
- FOREIGN KEY presupune ca fiecare valoare din coloana sau setul de coloane defini astfel să aibă o valoare corespondentă identică în tabela de legătură, tabelă în care coloana corespondentă este definită cu restricția UNIQUE sau PRIMARY KEY;
- CHECK elimină valorile care nu satisfac anumite cerințe (condiții) logice.

Termenul de chei (keys) este folosit pentru definirea câtorva categorii de constrângeri și sunt: *primary key*, *unique key*, *foreign key*, *referenced key*.

Se consideră că modelul relațional are o serie de limite cum ar fi:

- Simplitatea modelului îl face dificil de aplicat pentru noile tipuri

- 
- de aplicații (multimedia, internet etc.);
  - Nu asigură o independență logică totală a datelor de aplicație;
  - Poate crește redundanța datelor.

## 1.2. BAZE DE DATE RELAȚIONALE

Bazele de date relaționale (BDR) utilizează modelul de date relațional și noțiunile aferente. BDR au o solidă fundamentare teoretică, în special prin cercetările de la IBM conduse de E.F.Codd.

**BDR** este un ansamblu organizat de *tabele* (relații) împreună cu *legăturile* dintre ele.

**Concepte** utilizate la organizarea datelor în BDR și respectiv fișiere sunt prezentate în tabelul 1.1.

### Concepte utilizate în organizarea datelor

Tabelul 1.1.

Fișiere	fișier	înregistrare	câmp	valori
BDR	tabelă(relație)	tuplu (linie)	atribut(coloană)	domeniu valori

**Avantajele** BDR față de fișiere sunt prezentate în tabelul 1.2.

### Avantajele BDR

Tabelul 1.2.

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

Atunci când dorim să realizăm o bază de date relațională trebuie să știm clar ce avem de făcut, adică să stabilim obiectivele activității noastre. În acest sens, câteva dintre cele mai importante **obiective**, le prezentăm în continuare:

- *Partiționarea* semnifică faptul că aceleași date trebuie să poată fi folosite în moduri diferite de către diferiți utilizatori;
- *Deschiderea* se referă la faptul că datele trebuie să fie ușor adaptabile la schimbările care pot apărea (actualizarea structurii, tipuri noi de date etc.);
- *Eficiența* are în vedere stocarea și prelucrarea datelor, care trebuie

---

să se facă la costuri cât mai scăzute, costuri care să fie inferioare beneficiilor obținute;

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

Bazele de date relaționale au evoluat ca un tip special de aplicații informatice, și anume cele care au organizarea datelor în memoria externă conform unui model de date specific. De aceea, în metodologia de realizare a BDR se parcurg, în cea mai mare parte, cam aceleași etape ca la realizarea unei aplicații informatice, cu o serie de aspecte specifice. Pe de altă parte, în literatura de specialitate, sunt diferite propuneri de metodologii de realizare a bazelor de date.

Ținând cont de cele două aspecte de mai sus, sunt propuse câteva activități care trebuie parcurse la realizarea unei baze de date. Aceste activități vor fi regăsite, sub aceeași denumire sau sub denumiri diferite, în majoritatea metodologiilor de realizare a bazelor de date, din literatura de specialitate.

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

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

---

Studiul situației existente se realizează prin: definirea caracteristicilor generale ale unității, identificarea activităților desfășurate, identificarea resurselor existente (informaționale, umane, energetice, echipamente, financiare etc.), identificarea necesităților de prelucrare.

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

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

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

*Rezultatul* analizei structurale este modelul static (structural) numit și diagrama entitate-asociere. Diagrama entitate-asociere (Entity-Relationship) poate fi generată cu produse software tip CASE (Computer Aided Software Engineering), ca de exemplu Oracle Designer. Pornind de la o astfel de diagramă, se pot construi, în activitatea de proiectare, schemele relațiilor (tabelelor).

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

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

*Rezultatul* analizei dinamice este modelul dinamic.

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

- Aria de cuprindere a sistemului.
- Se identifică sursele de date.

- Se identifică modul de circulație și prelucrare a datelor.
- Se identifică apoi rezultatele obținute.

*Rezultatul* analizei funcționale este modelul funcțional.

2) *Proiectarea structurii bazei de date* se face pe baza modelelor realizate în activitatea de analiză. Înainte de proiectarea bazei de date se alege tipul de sistem de gestiune a bazei de date. Alegerea SGBD-ului se face ținând cont de două aspecte: cerințele aplicației (utilizatorului) și performanțele tehnice ale SGBD-ului.

Cerințele aplicației se referă la: volumul de date estimat a fi memorat și prelucrat în BDR; complexitatea problemei de rezolvat; ponderea și frecvența operațiilor de intrare/ieșire; condițiile privind protecția datelor; operațiile necesare (încărcare/validare, actualizare, regăsire etc.); particularitățile activității pentru care se realizează baza de date.

*Performanțele tehnice* ale SGBD-ului se referă la: modelul de date pe care-l implementează; ponderea utilizării SGBD-ului pe piață și tendința; configurația de calcul minimă cerută; limbajele de programare din SGBD; facilitățile de utilizare oferite pentru diferite categorii de utilizatori; limitele SGBD-ului; optimizările realizate de SGBD; facilitățile tehnice; lucrul cu mediul distribuit și concurența de date; elementele multimedia; instrumentele CASE; interfețele de comunicare; posibilitatea de autodocumentare; instrumentele specifice oferite.

*Proiectarea BDR* se realizează prin proiectarea schemelor BDR și proiectarea modulelor funcționale specializate.

*Schemele bazei de date* sunt: conceptuală, externă și internă.

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

- Stabilirea schemei conceptuale inițiale care se deduce din modelul entitate-asociere (vezi analiza structurală). Pentru acest lucru, se transformă fiecare entitate din model într-o colecție de date (fișier), iar pentru fiecare asociere se definesc cheile aferente. Dacă rezultă colecții izolate, acestea se vor lega de alte colecții prin chei rezultând asocieri ( $1:1$ ,  $1:m$ ,  $m:n$ ).
- Ameliorarea progresivă a schemei conceptuale prin eliminarea unor anomalii (exemplu: cele cinci forme normale pentru BDR relaționale).
- Stabilirea schemei conceptuale finale trebuie să asigure un echilibru între cerințele de actualizare și performanțele de exploatare (exemplu: o formă normală superioară asigură



---

performanțe de actualizare, dar timpul de răspuns va fi mai mare).

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

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

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

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

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

**FN1.** O tabelă este în FN1 dacă toate atributele ei conțin valori elementare (nedecompozabile), adică fiecare tuplu nu trebuie să aibă date la *nivel de grup* sau *repetitiv*. Structurile de tip arborescent și rețea se transformă în tabele cu atribute elementare.

O tabelă în FN1 prezintă încă o serie de *anomalii* de actualizare datorită eventualelor dependențe funcționale incomplete.

*Fiecare* structură repetitivă generează (prin descompunere) o nouă tabelă, iar atributele la nivel de grup se înlătură, rămânând doar cele elementare.

**FN2.** O tabelă este în FN2 dacă și numai dacă este în FN1 și fiecare atribut noncheie al tablei este *dependent funcțional complet* de cheie. Un atribut B al unei table *depinde funcțional* de atributul A al aceiași table, 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 aceiași table, dacă B este dependent funcțional de întreg ansamblul A (nu numai de un atribut din ansamblu).

O tabelă în FN2 prezintă încă o serie de *anomalii* de actualizare, datorită eventualelor dependențe tranzitive.

---

*Eliminarea* dependențelor incomplete se face prin descompunerea tabelii inițiale în două tabele, ambele conținând atributul intermediar (B).

**FN3.** O tabelă este în FN3 dacă și numai dacă este în FN2 și fiecare atribut noncheie depinde în mod *netranzitiv* de cheia tabelii. Într-o tabelă T, fie A,B,C trei atribute cu A cheie. Dacă B depinde de A ( $A \rightarrow B$ ) și C depinde de B ( $B \rightarrow C$ ) atunci C depinde de A în mod *tranzitiv*. *Eliminarea* dependențelor tranzitive se face prin descompunerea tabelii inițiale în două tabele, ambele conținând atributul intermediar (B).

O tabelă în FN3 prezintă încă o serie de *anomalii* de actualizare, datorate eventualelor dependențe multivaloare.

O definiție mai riguroasă pentru FN3 a fost dată prin forma intermediară **BCNF** (Boyce Codd Normal Form): o tabelă este în BCNF dacă fiecare determinant este un candidat cheie. Determinantul este un atribut elementar sau compus față de care alte atribute sunt complet dependente funcțional.

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

**FN5.** O tabelă este în FN5 dacă și numai dacă este în FN4 și fiecare *dependență joncțiune* este generată printr-un candidat cheie al tabelii. În tabela T (A,B,C) se menține *dependența joncțiune* (AB, AC) dacă și numai dacă T menține dependența multivaloare  $A \twoheadrightarrow B$  sau  $A \twoheadrightarrow C$ .

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

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

c) *Proiectarea schemei interne* presupune stabilirea structurilor de memorare fizică a datelor și definirea căilor de acces la date. Acestea sunt specifice fie SGBD-ului (scheme de alocare), fie sistemului de operare. Proiectarea schemei interne înseamnă estimarea *spațiului fizic* pentru BDR, definirea unui model fizic de alocare (a se vedea dacă SGBD-ul permite explicit acest lucru) și definirea unor *indecși* pentru accesul direct, după

---

cheie, la date.

*Proiectarea modulelor funcționale* ține cont de concepția generală a BDR, precum și de schemele proiectate anterior. În acest sens, se proiectează fluxul informațional, modulele de încărcare și manipulare a datelor, interfețele specializate, integrarea elementelor proiectate cu organizarea și funcționarea BDR.

3) *Realizarea componentelor logice.* Componentele logice ale unei BD sunt programele de aplicație dezvoltate, în cea mai mare parte, în SGBD-ul ales. Programele se realizează conform modulelor funcționale proiectate în etapa anterioară. Componentele logice țin cont de ieșiri, intrări, prelucrări și colecțiile de date. În paralel cu dezvoltarea programelor de aplicații se întocmesc și documentațiile diferite (tehnică, de exploatare, de prezentare).

4) *Punerea în funcțiune și exploatarea.* Se testează funcțiile BDR mai întâi cu date de test, apoi cu date reale. Se încarcă datele în BDR și se efectuează procedurile de manipulare, de către beneficiar cu asistența proiectantului. Se definitivează documentațiile aplicației. Se intră în exploatare curentă de către beneficiar conform documentației.

5) *Dezvoltarea sistemului.* Imediat după darea în exploatare a BDR, în mod continuu, pot exista factori perturbatori care generează schimbări în BDR. Factorii pot fi: organizatorici, datorati progresului tehnic, rezultați din cerințele noi ale beneficiarului, din schimbarea metodologiilor etc.

### ***1.3. DEFINIREA SISTEMULUI DE GESTIUNE A BAZELOR DE DATE RELAȚIONALE (SGBDR)***

Teoria relațională, foarte bine pusă la punct într-un domeniu de cercetare distinct, a dat o fundamentare solidă realizării de SGBD-uri performante. La sfârșitul anilor 80 și apoi în anii 90 au apărut, în special o dată cu pătrunderea în masă a microcalculatoarelor, numeroase SGBDR-uri. Aceasta a însemnat o evoluție de la SGBD-urile de generația întâi (arborescente și rețea) spre cele de generația a doua (relaționale). Această evoluție s-a materializat, în principal în: oferirea de limbaje de interogare neprocedurale, îmbunătățirea integrității și securității datelor, optimizarea și simplificarea acceselor.

*Teoria relațională este un ansamblu de concepte, metode și instrumente care a dat o fundamentare riguroasă realizării de SGBDR performante.*

Paralela între conceptele utilizate în evoluția organizării datelor în memoria externă până la sistemele relaționale este prezentată în tabelul 1.3:

Tabelul 1.3

FIȘIERE	TEORIA BD	TEORIA RELAȚIONALĂ	SGBDR
Fișier	Colecție de date	Relație	Tabela
Înregistrare	Familie de caracteristici	Tuplu	Linie
Câmp	Caracteristică	Atribut	Coloană
Valoare	Domeniu de valori	Domeniu	Domeniu

### Regulile lui Codd

E.F. Codd (cercetător la IBM) a formulat 13 reguli care exprimă cerințele maxime pentru ca un SGBD să fie relațional. Regulile sunt utile pentru evoluarea performanțelor unui SGBDR. Acestea sunt:

**R<sub>0</sub>.** Gestionarea datelor la nivel de relație: limbajele utilizate trebuie să opereze cu relații (unitatea de informație).

**R<sub>1</sub>.** Reprezentarea logică a datelor: toate informațiile din BDR trebuie stocate și prelucrate ca tabele.

**R<sub>2</sub>.** Garantarea accesului la date: LMD trebuie să permită accesul la fiecare valoare atomică din BDR (tabelă, coloană, cheie).

**R<sub>3</sub>.** Valoarea NULL: trebuie să se permită declararea și prelucrarea valorii NULL ca date lipsă sau inaplicabile.

**R<sub>4</sub>.** Metadatele: informațiile despre descrierea BDR se stochează în dicționar și tratează ca tabele, la fel ca datele propriu-zise.

**R<sub>5</sub>.** Limbajele utilizate: SGBDR trebuie să permită utilizarea mai multor limbaje, dintre care cel puțin unul să permită definirea tabelelor (de bază și virtuale), definirea restricțiilor de integritate, manipularea datelor, autorizarea accesului, tratarea tranzacțiilor.

**R<sub>6</sub>.** Actualizarea tabelelor virtuale: trebuie să se permită ca tabelele virtuale să fie și efectiv actualizabile, nu numai teoretic actualizabile (exemplu atributul “valoare” dintr-o tabelă virtuală nu poate fi actualizat).

**R<sub>7</sub>.** Actualizările în baza de date: manipularea unei tabele trebuie să se facă prin operații de regăsire dar și de actualizare.

**R<sub>8</sub>.** Independența fizică a datelor: schimbarea structurii fizice a datelor (modul de reprezentare (organizare) și modul de acces) nu afectează programele.

**R<sub>9</sub>.** Independența logică a datelor: schimbarea structurii de date (logice) a tabelelor nu afectează programele.

**R<sub>10</sub>.** Restricțiile de integritate: acestea, trebuie să fie definite prin LDD și stocate în dicționarul (catalogul) BDR.

**R<sub>11</sub>.** Distribuția geografică a datelor: LMD trebuie să permită ca

---

programele de aplicație să fie aceleași atât pentru date distribuite cât și pentru date centralizate (alocarea și localizarea datelor vor fi în sarcina SGBDR-ului).

**R<sub>12</sub>.** Prelucrarea datelor la nivel de bază (scăzut): dacă SGBDR posedă un limbaj de nivel scăzut (prelucrarea datelor se face la nivel de înregistrare), acesta nu trebuie utilizat pentru a evita restricțiile de integritate.

Regulile lui Codd pot fi grupate, conform cerințelor exprimate în cinci categorii, conform tabelului 1.4.

Tabelul 1.4. Gruparea regulilor lui Codd

	R <sub>0</sub>	R <sub>1</sub>	R <sub>2</sub>	R <sub>3</sub>	R <sub>4</sub>	R <sub>5</sub>	R <sub>6</sub>	R <sub>7</sub>	R <sub>8</sub>	R <sub>9</sub>	R <sub>10</sub>	R <sub>11</sub>	R <sub>12</sub>
1.Reguli de bază (fundamentale)	da												da
2.Reguli structurale		da					da						
3.Reguli privind integritatea datelor				da							da		
4.Reguli privind manipularea datelor			da		da	da		da					
5.Reguli privind independența datelor									da	da		da	

Regulile lui Codd sunt greu de îndeplinit în totalitate de către SGBDR. Pornind de la cele 13 reguli de mai sus, au fost formulate o serie de criterii (cerințe) pe care trebuie să le îndeplinească un SGBD pentru a putea fi considerat relațional într-un anumit grad. S-a ajuns astfel, la mai multe grade de relațional pentru SGBDR: *cu interfață relațională* (toate datele se reprezintă în tabele, există operatorii de selecție, proiecție și joncțiune doar pentru interogare), *pseudorelațional* (toate datele se reprezintă în tabele, există operatorii de selecție, proiecție și joncțiune fără limitări), *minimal relațional* (este pseudorelațional și în plus, operațiile cu tabele nu fac apel la pointeri observabili de utilizatori), *complet relațional* (este minimal relațional și în plus, există operatorii de reuniune, intersecție și diferență, precum și restricțiile de integritate privind unicitatea cheii și restricția referențială).

*În concluzie, SGBDR este un sistem software complet care implementează modelul de date relațional și respectă cerințele impuse de acest model.* El este o interfață între utilizatori și baza de date.

---

## 1.4. CARACTERIZAREA SGBDR

Sistemele relaționale îndeplinesc funcțiile unui SGBD cu o serie de aspecte specifice care rezultă din definirea unui SGBDR.

*Caracterizarea SGBDR* se poate face pe două niveluri: global (sistemele relaționale sunt privite ca o categorie distinctă de SGBD) și particular (fiecare SGBDR are aspecte individuale comparativ cu altele similare).

A. Mecanismele și instrumentele care ajută la **caracterizarea globală** a SGBDR-urilor sunt: limbajele relaționale, protecția datelor, optimizarea cererilor de regăsire, utilitarele specializate.

### 1) *Limbajele relațional*

SGBDR oferă seturi de comenzi pentru descrierea și manipularea datelor. Acestea pot fi incluse într-un singur limbaj relațional (SQL, QUEL, QBE, SQUARE, ALPHA, ISBL) sau separate în LDD și LMD. În ambele situații, comenzile pentru definirea datelor sunt distincte de cele pentru manipularea datelor.

Limbajele relaționale de definire a datelor (LDD) sunt simplificate, cu puține comenzi. Descrierea datelor este memorată în BDR, sub formă de tabele, în dicționarul (metabaza) bazei de date. Facilități de descriere sunt prezente în SGBDR prin comenzi, care definesc anumite operații, la nivelurile: conceptual, logic, fizic.

Limbajele relaționale de manipulare a datelor (LMD) pot fi caracterizate după criteriile generale, funcționale și calitative.

a) *Caracterizarea generală a LMD* se face după modul de tratare a datelor, operatorii relaționali, realizatorii și utilizatorii limbajului.

*Modul de tratare a datelor.* Toate LMD relaționale realizează o tratare la nivel de *ansamblu* a datelor: unitatea de informare pentru lucru este *tabela*. Avantajele sunt date de posibilitatea gestionării automat a tuplurilor duplicate și prelucrarea paralelă a ansamblurilor.

La comunicarea unui LMD relational cu un limbaj universal, avantajele se pierd deoarece comunicarea se poate face doar tuplu cu tuplu și nu la nivel de ansamblu. Deoarece limbajele universale oferă alte avantaje legate de proceduralitate, soluția este de a integra în acestea un limbaj relațional. *Cursorul* este soluția în SGBDR pentru a face trecerea de la tratarea la nivel de ansamblu la cea la nivel de înregistrare (tuplu).

*Operatorii relaționali* implementați. SGBDR s-au dezvoltat, din punct de vedere relațional, având la bază calculul relațional orientat pe tuplu (ALPHA, QUEL), calculul relațional orientat pe domeniu (QBE), algebra relațională (ISBL), transformarea (mapping) (SQL, SQUARE). Limbajele bazate pe calculul relațional sunt neprocedurale, cele bazate pe algebra

---

relațională sunt procedurale, celelalte sunt combinații.

*Realizatorii* limbajelor relaționale s-au orientat pe domenii precise din teoria relațională. Astfel, au rezultat: limbaje relaționale standardizate internațional (exemplu SQL - ANSI), limbaje cu standard de utilizare impus de constructor (exemplu QUEL), limbaje nestandardizate (celelalte limbaje relaționale).

*Utilizatorii* limbajelor relaționale sunt mult diversificați. SGBDR oferă atât elemente procedurale (pentru specialiști) cât și neprocedurale (pentru nespecialiști).

b) *Caracterizarea funcțională a LMD* se face după facilitățile de interogare, actualizare a datelor, etc.

*Facilitățile de interogare a datelor.* Acestea sunt puternice și oferite prin comenzi pentru interogarea tabelor de bază (exemplu SELECT) și interogarea tabelor virtuale (exemplu SELECT).

*Facilitățile de actualizare a datelor.* Acestea se referă la actualizarea tabelor de bază și a tabelor virtuale prin comenzile: INSERT INTO (adaugă rânduri la sfârșitul unei tabeli); UPDATE (modifică rânduri dintr-o tabelă); DELETE FROM (șterge rânduri dintr-o tabelă). Unele SGBDR nu permit actualizarea tabelor virtuale (exemplu Foxpro), altele permit acces lucru cu o serie de restricții pentru ca operația să se propage spre tabellele de bază fără ambiguități (exemplu DB2, Oracle).

*Alte facilități funcționale.* La facilitățile relaționale de mai sus, SGBDR-urile oferă și alte facilități pe care le au toate limbajele de programare procedurale cum sunt: calculul aritmetic prin operatori specifici (+, -, \*, /, \*\*); agregarea prin funcții standard (SUM) și prin comenzi (COMPUTE OF expr); comenzi de intrare/ieșire standard (ACCEPT...PROMPT...).

c) *Caracterizarea calitativă a LMD* se face după puterea selectivă, ușurința de învățare, utilizare și eficiența limbajului.

*Puterea selectivă a LMD* relaționale este dată de posibilitatea selectării datelor după criterii (filtre) complexe (exemplu comanda SELECT).

*Ușurința de învățare și utilizare* este nuanțată în funcție de tipul LMD-ului relațional. Cele bazate pe calculul relațional sunt neprocedurale (descriptive), deci ușor de învățat și utilizat (apropiat, ca stil, de limbajul natural) (exemplu QUEL) iar cele bazate pe algebra relațională sunt procedurale (algoritmice), deci mai greu de învățat și utilizat (exemplu ISBL). Cele intermediare promovează stilul neprocedural dar acceptă și elemente de control procedural (exemplu SQL) iar cele bazate pe grafică oferă primitive grafice pentru machetarea cererilor de regăsire, deci ușor de utilizat (exemplu QBE).



---

*Eficiența utilizării* este determinată de posibilitatea optimizării cererilor de regăsire. LMD bazate pe calculul relațional lasă compilatorul să aleagă ordinea de execuție a operațiilor, deci rezultă o eficiență mare. LMD bazate pe algebra relațională au o ordine impusă pentru execuția operațiilor, deci rezultă o eficiență mică.

## 2) *Protecția datelor*

Aspectele privind protecția datelor sunt foarte importante pentru un sistem de bază de date și ele trebuie implementate de către SGBDR. Protecția bazei de date se referă la integritatea datelor (integritatea semantică, concurența la date, salvarea/restaurarea) și securitatea datelor (autorizarea accesului, viziunile, procedurile speciale, criptarea).

a) *Integritatea semantică*. Definirea restricțiilor de integritate se face, conform cerințelor modelului relațional, în LDD (exemplu CREATE TABLE, ALTER TABLE). Utilizarea restricțiilor de integritate se face cu ajutorul unor mecanisme care controlează validitatea regulilor pentru fiecare nouă stare a BD. Aceste mecanisme sunt metode de detectare a inconsistenței datelor (se verifică restricțiile de integritate) la sfârșitul tranzațiilor, care se realizează automat de SGBDR și puncte de verificare a integrității fixate de utilizator, acolo unde dorește el în program.

b) *Concurența la date (coerența)*. Unitatea de lucru pentru asigurarea coerenței datelor este *tranzația*. Aceasta este un ansamblu de comenzi tratate unitar. Tranzația se execută în totalitate sau deloc. Coerența poate fi afectată la actualizarea concurentă sau la incidente.

*Mecanismele* utilizate de SGBDR pentru asigurarea coerenței datelor (controlul accesului concurent) sunt:

- Blocarea la diferite niveluri: bază de date, tabelă, tuplu, atribut;
- Definirea unor puncte de salvare în interiorul tranzațiilor (exemplu comanda savepoint);
- Tranzații explicite (begin și end transaction) și implicite (comenzile de actualizare);
- Fișiere jurnal.

## 3) *Optimizarea regăsirii*

*Cererile de regăsire* se exprimă în SGBDR în diferite limbaje relaționale. Pentru a se obține un rezultat optim, se utilizează interfețe automate de rescriere a cererilor de regăsire, prin parcurgerea a doi pași:

- Exprimarea cererilor de regăsire sub forma unor expresii algebrice relaționale, care are la bază echivalența dintre calculul și algebra relațională.
- Aplicarea unor transformări algebrice relaționale asupra expresiilor construite în pasul anterior, pentru a se obține expresii

---

relaționale echivalente și eficiente.

*Transformarea* se poate realiza prin doua *strategii de optimizare*: generale, specifice.

Strategiile generale sunt independente de modul de memorare a datelor. Ele se bazează pe proprietățile operațiilor din algebra relațională (comutativitatea, asociativitatea, compunerea ). Astfel de strategii sunt: selecția înaintea joncțiunii, proiecția înaintea joncțiunii, selecția înaintea proiecției, combinarea selecției multiple.

Strategiile specifice țin cont de modul de memorare a datelor și ele sunt caracteristice unui SGBDR. Elementele care influențează executarea operațiilor ce intervin la o cerere de regăsire sunt: accesul direct, reguli de ordonare a expresiilor algebrice specifice unui SGBDR.

#### *4) Utilitățile specializate*

Posibilitățile de utilizare ale unui SGBDR sunt influențate de utilitățile specializate pe care le are, pentru diferitele categorii de utilizatori (în Oracle: Developer pentru dezvoltatori, Designer pentru analiști, Administration Tools și Utilities pentru administrator etc.).

B. Pentru a face o **caracterizare particulară**, un anumit SGBDR vom lua în considerare o serie de criterii de comparație. Aceste criterii se vor urmări, grupate pe anumite categorii, pentru câteva SGBDR-uri care ne interesează. După această analiză vom avea un argument serios pentru a putea alege un SGBDR în scopul dezvoltării unei aplicații cu baze de date.

Gruparea caracteristicilor particulare de comparație a SGBDR-urilor o vom face în funcție de facilitățile de descriere, manipulare, utilizare și administrare a datelor.

*Caracteristicile în funcție de facilitățile de descriere* sunt: modul de implementare a modelului relațional; conceptul de bază de date utilizat în schemă; definirea metadatelor; definirea relațiilor virtuale; actualizarea schemei relației; restricțiile de integritate ce pot fi declarate.

*Caracteristicile în funcție de facilitățile de manipulare* sunt: LMD relațional implementat; funcțiile de calcul aritmetic și funcțiile agregate; modurile de acces la date; programarea orientată-obiect; tratarea valorii NULL; optimizarea cererilor de regăsire; actualizarea relațiilor de bază și virtuale.

*Caracteristicile în funcție de facilitățile de utilizare și administrare* sunt: instrumentele de dezvoltare; instrumentele CASE; instrumentele analize statistice; software-ul pentru acces de la distanță; utilitățile de întreținere; mecanismele pentru autorizarea accesului la date.

### **1.5. EXEMPLE DE SISTEME DE GESTIUNE A BAZELOR DE DATE RELAȚIONALE**

*Oracle.* Este realizat de firma Oracle Corporation USA. Sistemul este complet relațional, robust, se bazează pe SQL standard extins. Arhitectura sistemului este client/server, permițând lucrul, cu obiecte și distribuit. Are BD Internet și modul de optimizare a regăsirii. Ultima versiune este Oracle 10g.

*DB2.* Este realizat de firma IBM. Sistemul respectă teoria relațională, este robust și se bazează pe SQL standard. Permite lucrul distribuit și are modul de optimizare a regăsirii.

*Informix.* Este realizat de firma Informix, respectă teoria relațională și permite lucru distribuit.

*Progress.* Este realizat de firma Progress Software. Are limbaj propriu (Progress 4GL) dar suportă și SQL. Rulează pe o gamă largă de calculatoare sub diferite sisteme de operare.

*SQL Server.* Este realizat de firma Microsoft. Se bazează pe SQL și rulează în arhitectura client/server.

*Ingress II.* Este realizat de firma Computer Associates. Este un SGBDR complet, implementează două limbaje relaționale (întâi QUEL și apoi SQL) și este suportat de diferite sisteme de operare (Windows, UNIX). Lucrează distribuit în arhitectura client/server, are extensie cu facilități orientate obiect și permite aplicații de tip Internet. Organizarea fizică a tabelor se face prin sistemul de operare.

*Visual FoxPro.* Este realizat de firma Microsoft. Are un limbaj procedural propriu foarte puternic, o extensie orientată obiect, programare vizuală și nucleu extins de SQL.

*Access.* Este realizat de firma Microsoft. Se bazează pe SQL, are limbajul procedural gazdă (Basic Access) și instrumente de dezvoltare.

*Paradox.* Este realizat de firma Borland. Are limbaj procedural propriu (PAL) și suportă SQL.