CUPRINS

PREFAŢĂ

Bazele de date reprezintă o aplicaţie fundamentală din domeniile programării şi al reţelelor de calculatoare. Potenţialul de utilizare a bazelor de date este uriaş, iar popularitatea lor, la ora actuală, nu poate fi contestată.

Proiectarea bazei de date reprezintă o etapă esenţială şi critică pentru succesul oricărei aplicaţii cu baze de date, cu accesare locală sau online. Orice eroare sau omisiune din etapa de proiectare va determina erori în interogarea şi actualizarea bazei de date, implementată ca aplicaţie, într-un limbaj de programare specific. Eventualele erori sesizate de utilizatori pot fi cu greu corectate după ce programarea aplicaţiei s-a încheiat. Programatorul de baze de date nu este interesat de specificul bazei de date. Doar proiectantul şi beneficiarul acesteia pot să sesizeze şi să remedieze erorile, în faza de proiectare a bazei de date, prin consultare reciprocă şi repetată.

Această carte este dedicată proiectării sistematice a bazelor de date relaţionale şi prezentării tuturor aspectelor şi a problemelor care pot să apară pe durata acestui proces. Aşa cum spunea Codd, nu orice bază de date tabelară este şi relaţională. Multe condiţii trebuie îndeplinite astfel încât baza de date proiectată să fie relaţională şi optimizată. Eventualele erori de operare în baza de date pe care le pot face utilizatorii mai puţin sau deloc specializaţi în acest domeniu, trebuie anticipate şi prevenite, prin măsuri specifice de proiectare, care trebuie transmise programatorilor în documentaţia bazei de date.

Cartea se adresează studenţilor şi tuturor celor interesaţi de proiectarea bazelor de date, fiind un bun instrument de studiu, prin modul de prezentare a noţiunilor, prin exemplele date, precum şi prin aplicaţiile şi testele propuse spre rezolvare, pentru verificarea cunoştinţelor.

Conf. dr.ing. Luminiţa SCRIPCARIU

Universitatea Tehnică „Gheorghe Asachi" din Iaşi

CAPITOLUL I

INTRODUCERE ÎN TEORIA BAZELOR DE DATE

DIN CUPRINS:

I.1 DEFINIŢII ŞI APLICATIVITATE

Câţi dintre noi au folosit o bază de date sau mai precis câţi dintre noi au beneficiat de serviciile unui server de baze de date? În prezent, aproape în toate activităţile se accesează o bază de date, pentru gestionarea persoanelor, serviciilor sau a obiectelor cu care se lucrează. Orice aplicaţie online are „în spate" o bază de date. Dacă folosiţi un program de mesagerie, primul pas este acela de autentificare. Numele de utilizator şi parola de acces pe care le introduceţi sunt comparate cu datele dumneavoastră personale stocate în baza de date de pe serverul de autentificare. Dacă aţi făcut vreodată cumpărături online, datele dumneavoastră ca şi client există deja în baza de date a magazinului online. De asemenea, orice activitate administrativă sau financiară desfăşurată de diverse instituţii (serviciul de evidenţă a populaţiei, direcţiile de taxe şi impozite, agenţiile de asigurare, aplicaţiile online de plată a facturilor etc.) utilizează baze de date care permit organizarea eficientă a informaţiilor şi accesul rapid, sigur, local şi de la distanţă la acestea.

Baza de date (BD, *Database - DB*) constituie o aplicaţie fundamentală în toate domeniile de activitate, civile sau de apărare: financiar, administrativ, educaţional, informaţional, de comunicaţii şi, nu în ultimul rând, cel al calculatoarelor.

Sistemele de baze de date, ca aplicaţii software, reprezintă poate cea mai importantă realizare din domeniul ingineriei programării pe calculator.

*Prin **bază de date** se înţelege orice colecţie partajată de date, între care există relaţii logice, care conţine descrierea datelor şi este proiectată pentru a satisface necesităţile informaţionale ale unei organizaţii sau ale unui grup de utilizatori.*

Iniţial a apărut necesitatea computerizării sistemului de îndosariere. O primă soluţie a acestei probleme a constituit-o sistemul bazat pe fişiere, cu mai multe programe de aplicaţie care oferă diverse servicii utilizatorilor, printre care şi generarea de rapoarte. Deşi foarte folosit, acest sistem se dovedeşte a fi extrem de redundant şi greu de actualizat prin dublarea datelor şi complexitatea relativ mare. În acest context, bazele de date au oferit o modalitate eficientă de tratare distribuită a datelor, într-o resursă comună partajată în care se include şi descrierea acestora în aşa-numitul **catalog de sistem.**

În prezent, orice activitate de gestiune din orice domeniu de activitate implică folosirea unei baze de date, cu acces local sau de la distanţă, cu caracter public sau privat. Orice BD este gândită ca o resursă unică, utilizată simultan de mai mulţi utilizatori.
Complexitatea şi dimensiunea BD evoluează rapid, determinând reproiectarea algoritmilor de stocare şi acces al fişierelor şi chiar schimbarea unor principii de administrare a acestora.
Avantajele utilizării bazelor de date sunt numeroase. BD asigură independenţa program-date, sunt scalabile, flexibile, portabile şi permit controlul accesului şi al manipulării datelor.
BD stochează datele, eliminând redundanţele, şi, prin prelucrarea acestora, furnizează **informaţii**.
În funcţie de aplicaţia pe care o deserveşte, BD pot fi centralizate sau distribuite în reţea, iar modalităţile de gestionare diferă de la caz la caz. Sistemul de programe software care permite crearea, administrarea şi accesarea bazei de date este numit **sistem de gestiune a bazei de date (SGBD).**

Sistemul de gestiune a bazelor de date *(SGBD, Database Management System - DBMS) administrează BD şi controlează accesul la BD.*

De exemplu, pentru BD cu un volum mic sau mediu de date se pot folosi ca SGBD programele MS Access, Visual Fox, Dbase, FoxPro iar în cazul BD care stochează un volum mare de date sunt recomandate programele produse de compania Oracle.

Aplicaţiile cu baze de date facilitează accesul la informaţii şi reduc timpul de efectuare a anumitor operaţiuni fiind utilizate în prezent de magazinele online pentru comerţ electronic, de bănci pentru tranzacţii electronice de la distanţă, în universităţi, şcoli, biblioteci şi în sistemele administrative pentru gestionarea informaţiilor şi nu numai. Este dificil de cuprins în câteva cuvinte aplicaţiile care folosesc baze de date.
Din anii ’90, de când Internetul şi World Wide Web-ul sunt folosite pentru diverse aplicaţii online (*e-commerce, e-learning, e-banking* şi altele), bazele de date au devenit esenţiale pentru gestionarea şi prelucrarea unui volum tot mai mare de date. De asemenea, securitatea bazelor de date în ceea ce priveşte accesul la acestea dar şi asigurarea confidenţialităţii informaţiilor cu caracter privat constituie un element-cheie în dezvoltarea sistemelor de baze de date.

Securitatea *unei aplicaţii de baze de date are în vedere stabilirea drepturilor de acces la BD, a drepturilor de citire, scriere, modificare sau ştergere a obiectelor şi datelor din baza de date şi chiar a întregii baze de date.*

I.2 CATEGORII DE PERSONAL

Diferitele categorii de personal implicate în aplicaţiile de BD necesită o anumită pregătire precum şi diferite competenţe şi abilităţi, fiind necesară pregătirea de personal specializat în acest domeniu.

Categoriile de persoane implicate în mediul BD sunt:

- *proiectanţi*
- *programatori*
- *administratori de sistem*
- *administratori de baze de date.*

Proiectanţii bazei de date se ocupă de culegerea informaţiilor care descriu cerinţele şi aspectele specifice ale aplicaţiei dorite de client, sistematizarea acestora şi transpunerea lor în noţiuni de BD (entităţi, atribute, relaţii). În diferitele faze ale proiectării se optimizează modelul BD şi periodic au loc consultări între proiectanţi şi reprezentanţi ai categoriilor specifice de personal ale clientului (manageri, personal tehnic, administrativ, financiar-contabil etc.) pentru a se identifica toate cerinţele la care trebuie să răspundă aplicaţia finală de BD. Odată trecută aplicaţia în faza de programare, este dificil sau chiar imposibil să se rezolve noi cerinţe.

Trebuie ştiut că cele mai multe probleme (*bug-uri*) ale sistemelor de BD referitoare la actualizarea datelor şi chiar de securitate a BD dar nu numai, pot fi rezolvate doar în faza de proiectare şi mai puţin în faza programării.

Programatorii de BD de cele mai multe ori nu sunt familiarizaţi cu noţiunile specifice proiectului, sarcina lor fiind doar aceea de a-l transpune pe acesta în limbaj de programare, ca aplicaţie software performantă, cu o interfaţă de utilizator eficientă.

Administratorul de sistem instalează, configurează, securizează şi monitorizează funcţionarea aplicaţiei de BD.

Administratorul bazei de date (DBA – Database Administrator) se ocupă de popularea cu date a BD, precum şi de gestionarea şi „întreţinerea" datelor (arhivarea datelor vechi, ştergerea datelor perimate, urmărirea coerenţei BD etc.)
O categorie aparte o constituie **utilizatorii** BD care trebuie să fie instruiţi referitor la modul de utilizare a BD, la facilităţile şi informaţiile pe care aceasta le oferă.
Scopul întregii activităţi de proiectare, programare şi administrare a BD constă în punerea la dispoziţia utilizatorului final a unor BD permanent actualizate, coerente şi uşor accesibile.

I.3 NOŢIUNI SPECIFICE BAZELOR DE DATE

BD operează în termeni de entitate, atribut, relaţie.

*O **entitate** este un obiect distinct inclus în BD asociat unei persoane, firme, unui loc, document sau concept etc.*

Fiecărei categorii de obiecte descrise în BD i se asociază un tip de entitate cu un nume sugestiv. De exemplu, studenţii dintr-o facultate sunt descrişi de **tipul de entitate** *STUDENŢI,* al **entităţii** *STUDENT*. Fiecare student, prin numele şi prenumele său, este reprezentat ca o entitate distinctă din BD.
Fiecare entitate este descrisă printr-un set de **atribute**. De exemplu, entitatea *STUDENT* poate avea mai multe atribute: nume, prenume, cod numeric personal (CNP), număr matricol, specializare şi altele.

*Un **atribut** este o proprietate care descrie un anumit aspect al unui tip de entitate.*

Atributele pot lua valori unice sau multiple, pot fi impuse sau opţionale, pot avea caracter volatil sau nu, şi de asemenea pot juca un anumit rol în descrierea unui tip de entitate, numit **cheie**.

***Cheia** este un atribut sau un grup de atribute care identifică în mod unic o entitate.*

Pentru un tip de entitate, pot fi mai multe atribute de tip cheie, dar un singur atribut sau set de atribute este folosit la un anumit moment pentru identificarea unică a entităţilor înregistrate într-un tabel din BD şi acela se numeşte **cheie primară** (PK – *Primary Key*).
Celelalte chei sunt denumite **chei alternative** sau **chei candidat**.
Proiectarea unei baze de date începe cu identificarea entităţilor şi a atributelor care vor fi folosite pentru modelarea datelor. Proiectarea este în general făcută de o echipă cu mai multe persoane care trebuie să pună „cap la cap", într-o aplicaţie unică, ceea ce a realizat fiecare în parte. Este posibil ca pentru acelaşi obiect sau atribut să se folosească denumiri diferite. De exemplu, pentru entitatea *student,* unul dintre proiectanţi poate folosi atributul *număr matricol,* altul *cod_student,* altul *id_student,* toate acestea având de fapt acelaşi rol, acela de identificare fără echivoc a fiecărui student în parte. Este necesar ca atunci când se reunesc entităţile şi atributele create de toţi membrii echipei de proiectare, să se definească în mod unic toate elementele din BD şi descrierea lor să apară în clar în **dicţionarul de sistem**. De asemenea, este utilă stabilirea unor convenţii de lucru, care să permită proiectarea într-un mod unitar a BD de către toţi membrii echipei.
Între diferitele tipuri de entităţi, se stabilesc **relaţii** care descriu interacţiunile dintre obiectele reprezentate în BD.

*O **relaţie** reprezintă o asociaţie între mai multe entităţi.*

De exemplu, studenţii de la o anumită specializare şi dintr-un anumit an de studiu, studiază un anumite discipline. Rezultă astfel că există o relaţie între tipul de entitate *STUDENŢI* şi cel numit *DISCIPLINE*.
BD poate fi modelată grafic sub forma unei **diagrame Entitate-Relaţie (**ERD – *Entity - Relation Diagram*) în care sunt reprezentate entităţile, relaţiile şi atributele, sub forma unui graf neorientat.
Prin convenţie, entităţile sunt reprezentate sub formă de dreptunghiuri, relaţiile cu linii şi romburi iar atributele ca ovale în jurul entităţii descrise (figura I.1).
Se folosesc şi alte convenţii de reprezentare a diagramei ER. De exemplu, atributele pot fi enumerate unul sub celălalt, în dreptunghiul aferent entităţii pe care o descriu (figura I.2). Prin convenţie, cheia primară se scrie pe prima poziţie şi se subliniază.

Orice BD este creată cu scopul de a manipula datele prin operații specifice (introducere, extragere, ştergere, modificare) şi de a extrage informațiile eficient, folosind un **limbaj de manipulare a datelor** (DML – *Data Manipulation Language*).

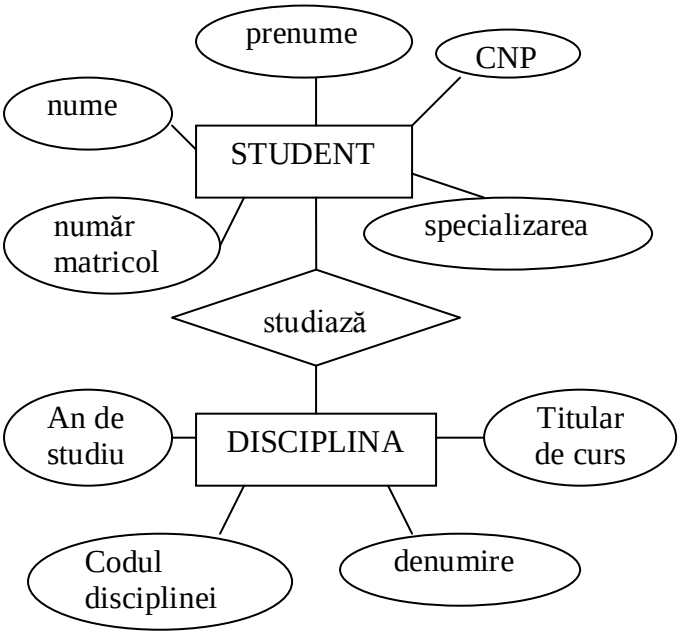

Figura I.1 Diagramă Entitate - Relaţie

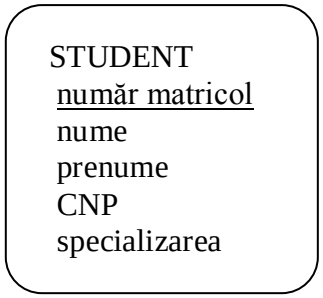

Figura I.2 Exemplu de reprezentare grafică a unui tip de entitate,
cu atribute specifice

Crearea structurii în care sunt stocate datele se realizează cu ajutorul unui **limbaj de definire a datelor** (DDL – *Data Definition Language*). Acesta permite denumirea entităţilor BD şi a relaţiilor logice dintre acestea în **schema bazei de date**, cu specificarea tipurilor şi structurilor de date, precum şi a modurilor de vizualizare personalizate**.**

Acea parte a unui limbaj DML utilizată pentru regăsirea datelor în BD se numeşte **limbaj de interogare** (*Query Language*).
DDL şi DML sunt considerate sublimbaje de date care pot fi încorporate într-un limbaj de nivel înalt denumit şi limbaj gazdă.
***SQL** – Structured Query Language – este limbajul specific bazelor de date.*

Prima aplicaţie de baze de date a fost creată de compania Oracle în anii ’80, iar limbajul SQL (*Structured Query Language*) a devenit limbajul de programare standard pentru aplicaţiile cu baze de date.
SQL este un limbaj de interogare a BD, neprocedural, care constituie limbajul standard pentru SGBD relaţionale. SQL include comenzi de definire a structurii BD, de manipulare a datelor precum şi de securizare a accesului la BD prin stabilirea şi/sau revocarea drepturilor utilizatorilor.
Un alt limbaj de manipulare a datelor din BD este limbajul QBE (*Query by Example*).
SQL şi QBE sunt limbaje din a patra generaţie (4GL – *Fourth Generation Language*) care definesc **ce** trebuie făcut şi nu **cum** se procedează. Limbajele din generaţia a treia sunt procedurale şi complicate sintactic.
Limbajele 4GL sunt de mai multe tipuri:

- **limbaje de prezentare** (de interogare, generatoare de rapoarte, generatoare grafice etc.)
- **limbaje de specialitate** (de exemplu, pentru calcul tabelar)
- **generatoare de aplicaţii** (care construiesc aplicaţii folosind datele din BD)
- **limbaje de nivel foarte înalt** (care generează codul-sursă al aplicaţiei).

Rezolvarea tranzacţiilor concurente, în funcţie de cerinţele clientului, reprezintă un alt aspect critic al aplicaţiei cu BD.

***Tranzacţia** reprezintă o operaţie de manipulare a datelor din BD .*

Controlul tranzacţiilor este avut în vedere în faza de programare, pe baza documentaţiei BD care însoţeşte proiectul BD.
SGBD are rolul să menţină coerenţa BD, chiar şi în cazul efectuării unor tranzacţii concurente.

De exemplu, în timp ce un client lansează comanda de achiziţie a unui produs de la un magazin online, administratorul de date măreşte cu 5 % preţurile produselor. Întrebarea este dacă în acel moment, clientul cumpără produsul cu preţul iniţial afişat sau cu preţul mărit. Rezolvarea acestor tranzacţii simultane concurente se face de către SGBD pe baza politicii firmei. Doar aceasta poate stabili principiile de soluţionare a tranzacţiilor concurente. Programatorul BD aplică aceste principii. Dacă nu se au în vedere cazurile critice, este foarte posibil să apară erori în utilizarea BD (erori de afişare, pierderea unor sume de bani, încălcarea prevederilor legale şi altele).

Utilizatorii BD sunt în general persoane cu interese diferite referitoare la informaţiile care pot fi extrase din BD. De aceea, este necesar să se creeze aşa-numite „**vederi**" din BD, în mod distinct pentru fiecare categorie de utilizatori în parte.

Vederea externă *reprezintă forma de vizualizare a anumitor informaţii incluse în BD care accesează anumite atribute ale uneia sau ale mai multor entităţi, prezentate în formatul dorit de utilizor.*

De exemplu, departamentul tehnic al unei firme poate solicita ca aplicaţia de BD să genereze rapoarte cu parametrii tehnici ai echipamentelor folosite, fără a se specifica şi costurile acestora. În acelaşi timp, cei de la departamentul financiar-contabil sunt interesaţi de preţurile de achiziţie şi de vânzare ale echipamentelor, nu şi de detaliile tehnice.
Ordinea atributelor unei entităţi sau a coloanelor dintr-un tabel din BD nu este importantă, deoarece extragerea informaţiilor şi crearea vederilor externe se va face independent de aceasta.
De exemplu, pe baza tabelului asociat entităţii *STUDENT,* vom putea afişa lista ordonată alfabetic a studenţilor, într-un tabel în care să cuprindă coloanele: numele şi prenumele, CNP, numărul matricol. Nu este obligatoriu ca o vedere externă să afişeze toate atributele unei entităţii. În plus, o vedere externă poate fi creată pe baza mai multor tabele din BD. Nici ordinea înregistrărilor dintr-un tabel nu are nici un rol, pentru că la crearea vederilor externe se vor putea ordona informaţiile după preferinţele beneficiarilor (alfabetic, crescător, după dată etc.).
SGBD permite personalizarea BD pentru a satisface cerinţele utilizatorilor, simultan cu realizarea independenţei program-date.

În prezent, se dezvoltă SGBD multiutilizator pentru BD cu capacităţi mari de stocare pentru aplicaţii grafice, video, multimedia în general, cu interfeţe grafice de utilizator (GUI – *Graphic User Interface).*
De asemenea, SGBD are rolul de a asigurara securitatea BD şi confidenţialitea informaţiilor cu caracter restricţionat care sunt stocate în BD respectivă.

I.4 MODELAREA DATELOR

Modelarea datelor reprezintă etapa de început a proiectării unei baze de date. Operaţia de modelare include stabilirea entităţilor şi a atributelor acestora precum şi a relaţiilor dintre ele.

Modelul de date conceptual este creat independent de limbajul de programare în care se va dezvolta aplicaţia de baze de date.

Modelarea datelor începe cu analiza cerinţelor beneficiarilor bazei de date, aşa-numitele **reguli de afaceri** (*Business rules*). Beneficiarii pot fi persoane cu preferinţe cunoscute, cum ar fi angajaţii unei companii, sau total necunoscute, ca de exemplu clienţii unui magazin online, ale căror preferinţe pot fi doar estimate prin sondaje făcute pe eşantioane reprezentative din rândul potenţialilor utilizatori ai aplicaţiei.
De exemplu, dacă se doreşte crearea unei baze de date pentru o facultate, beneficiarii acesteia vor fi studenţii, profesorii şi personalul auxiliar. Studenţii vor dori să extragă din baza de date informaţii referitoare la notele obţinute, orar, planificarea examenelor, taxe achitate etc. Profesorii vor dori de asemenea să îşi consulte online orarul, să scrie în baza de date notele obţinute de studenţi sau să calculeze medii ale grupelor. Secretara va înscrie în baza de date informaţiile cu caracter personal ale studenţilor, cuantumul burselor, taxele achitate de aceştia şi va dori să extragă rapoarte privind situaţia şcolară a fiecărui student, media notelor obţinute de aceştia în sesiune, clasamente ale studenţilor ordonate descrescător după medie şi număr de credite acumulat, fără a putea scrie sau modifica notele studenţilor.
De multe ori, persoanele care culeg aceste preferinţe de la beneficiari, preferă să le exprime sub forma unor întrebări.
Pentru exemplul BD a unei facultăţi, utilizatorii au adresat următoarele întrebări:
STUDENŢII: Ce orar am? Ce notă am obţinut la un examen? Ce bursă primesc?

PROFESORII: În care sală se ţine cursul de la disciplina pe care o predau? Care este lista studenţilor dintr-o grupă? Unde scriu notele obţinute de studenţi la examen în BD? Ce medie are o grupă la examen? Care sunt studenţii restanţieri?

SECRETARELE: Care este adresa de domiciliu a studentului X? La ce număr de telefon poate fi găsit un student? Ce vârstă are un student?

Observăm din acest exemplu că sunt mai multe categorii de beneficiari sau utilizatori ai bazei de date respective, cu cerinţe diverse şi drepturi diferite.

Identificarea entităţilor din BD şi a atributelor lor se va face astfel încât BD să poată oferi răspuns la toate întrebările utilizatorilor.

Se adoptă o anumită strategie de definire a entităţilor şi atributelor acestora, precum şi de creare a tabelelor asociate lor, luând în considerare atât vederile externe care vor fi create, cât şi cerinţele de securitate ale aplicaţiei.

În dezvoltarea aplicaţiei respective se vor avea în vedere aspectele de securitate, încă din faza modelării datelor.

Cerinţele informaţionale ale organizaţiei care solicită realizarea unei bazei de date trebuie să fie corect şi complet reflectate de modelul ER creat pentru acea aplicaţie.

Figura I.3 prezintă o parte din diagrama ER asociată modelului conceptual creat pentru baza de date a unei facultăţi.

În acest caz, s-au avut în vedere cerinţele personalului de la secretariatul facultăţii de a putea citi din baza de date informaţii specifice despre fiecare student, precum şi de a cunoaşte componenţa fiecărei grupe şi împărţirea lor pe specializări.

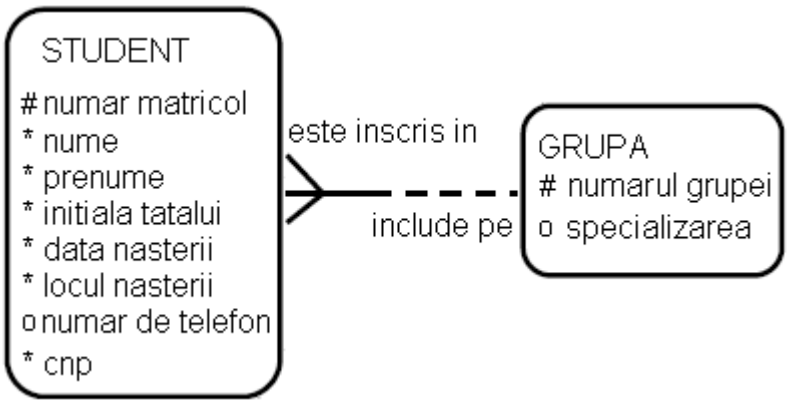

Figura I.3 Diagramă ER parţială - exemplu

Este esenţială analiza în detaliu a modelului creat, pentru eliminarea tuturor redundanţelor şi a elementelor care pot crea ambiguităţi şi erori în prelucrarea sau acualizarea datelor.

Normalizarea *este procesul de optimizare a modelului de date conceptual, prin care se elimină deficienţele modelului creat (atribute cu valori multiple, dependenţe funcţionale parţiale ş.a.).*

Primul model pentru baze de date a fost creat de E.F. Codd în anii 1970-1972, pentru bazele de date relaţionale.
O bază de date relaţională depozitează datele în tabele, între care se stabilesc anumite conexiuni.

În 1976, P. Chen propune un model avansat pentru baze de date şi anume modelul entitate-relaţie (modelul ER – *Entity-Relation model*), care separă nivelul fizic de cel logic, model care va deveni ulterior referinţa în domeniul proiectării bazelor de date.

I.5 MODELAREA FIZICĂ

Realizarea propriu-zisă a BD (*Database Design*) constă în crearea structurii acesteia, folosind **schema BD**.
Structura BD este compusă din tabele, cel puţin unul pentru fiecare tip de entitate. Pentru tabel se foloseşte termenul echivalent de **relaţie i**ar baza de date este numită **relaţională.** Însă, nu este suficient ca structura BD să fie tabelară pentru ca BD să fie considerată relaţională. O BD relaţională (BDR) trebuie să verifice regulile lui Codd, care vor fi prezentate într-un paragraf ulterior.

Un tabel din BD se asociază unui tip de entitate.

Este indicat ca numele tabelului să sugereze tipul de entitate şi mai precis să corespundă formei de plural a denumirii entităţii. De exemplu, se va crea tabelul cu denumirea *studenţi* pentru entitatea de tip *STUDENT*.

O coloană din tabel corespunde unui atribut al tipului de entitate.

Numele coloanelor dintr-un tabel pot să difere în structura internă a BD faţă de ceea ce se afişează în vederile externe din BD. De obicei, în structura internă a BD se foloseşte o formă abreviată a denumirii atributului pentru a defini o coloană din tabel, respectând restricţiile limbajului folosit. De exemplu, atributului *număr matricol* din diagrama ER îi poate corespunde coloana cu numele *nrmatr* din tabelul *studenţi.*
Tabelul *studenţi* poate fi descris prin lista atributelor entităţii *student*:
studenti (nrmatr, nume, prenume, cnp, specializare)
În vederile externe nu se folosesc forme prescurtate ci denumiri detaliate ale atributelor entităţilor, pe înţelesul utilizatorilor, numite şi **etichete** (*label*). De exemplu, într-un formular de introducere a datelor personale, poate să apară câmpul-text cu denumirea „Telefon mobil”, care să corespundă unei coloane din tabelul din BD cu denumirea *mobil.*
În orice tabel din BD, se introduce câte o linie pentru fiecare entitate nou înregistrată. O linie din tabel se mai numeşte **înregistrare** sau un **n-tuplu**.

O linie din tabel reprezintă un set de valori ale atributelor unei entităţi.

La intersecţia unei linii cu o coloană din tabel, se găseşte un **câmp** sau o **celulă** în care este înscrisă valoarea atributului definit pe acea coloană, corespunzătoare înregistrării din linia respectivă.
Valoarea unui atribut dintr-o celulă poate să lipsească, situaţie în care spunem că atributul poate fi *NULL.*

Prin NULL se înţelege lipsa valorii unui atribut şi nu valoarea zero.

În cazul în care nu se admite lipsa valorii unui atribut, trebuie să se specifice în linia de comandă opţiunea *NOT NULL* pentru acea coloană.
De exemplu, admitem că pentru un student din anul I, câmpul *specializare* rămâne necompletat, în timp ce din câmpurile *nume, prenume, nrmatr* valorile nu pot să lipsească.
De aceea, în formularele de introducere a datelor în BD, unele câmpuri sunt marcate ca fiind obligatorii iar altele ca opţionale. În plus, dacă toate câmpurile obligatorii nu sunt completate, atunci formularul nu poate fi trimis (*submit*) şi apare un mesaj de eroare care îl înştiinţează pe utilizator ce mai trebuie scris. Dacă aplicaţia nu este corect realizată şi utilizatorul trimite un set de informaţii incomplet, atunci în BD nu se poate face înregistrarea datelor dacă lipseşte

valoarea unui atribut pentru care s-a menţionat opţiunea *NOT NULL*. Ca urmare, datele înscrise în formular vor fi pierdute.
Dacă nu este permis ca valorile unui atribut să se repete, atunci trebuie specificată opţiunea *UNIQUE* pentru a evita unele erori la introducerea datelor în BD. De exemplu, valorile atributului *cnp* sunt unice.

Atributele care joacă rolul de chei ale entităţilor trebuie să aibă valori unice.

În BD, datele sunt integrate împreună cu o descriere a lor. Definirea coloanelor din tabelele BD include şi specificarea tipului de date folosit pe fiecare coloană. De exemplu, coloanele *nume* şi *prenume* vor conţine date de tip caracter, în timp ce numărul matricol va fi exprimat prin date de tip numeric. Dimensiunile câmpurilor sunt de asemenea specificate acolo unde este cazul.
Trebuie menţionată şi calitatea de cheie primară a unui atribut sau set de atribute, precum şi eventualele referinţe care se fac între tabelele BD. Un atribut care este cheie într-un tabel dar apare şi în alt tabel, pentru a face legătura dintre cele două tabele, se numeşte **cheie străină** (FK - *Foreign Key*).
Este esenţial să se lucreze cu o dublare minimă a datelor, în scopul reducerii redundanţei şi al menţinerii coerenţei lor. În principiu, orice informaţie trebuie să apară într-un singur loc din BD. De exemplu, nu înscriem în locuri diferite din BD gradul didactic al unei persoane, pentru ca la o eventuală promovare a acesteia, să nu fie necesară modificarea lui în mai multe locuri. Dacă totuşi îl scriem de mai multe ori, există riscul să se omită unele valori şi avem în acest caz de a face cu o **eroare de actualizare** cauzată de dublarea datelor şi de redundanţa BD, concretizată în pierderea coerenţei BD.
Descrierea datelor, prin tipul şi dimensiunea datelor, opţiuni şi alte specificaţii, reprezintă **metadatele**, care sunt stocate în catalogul de sistem.
Limbajul de definire a datelor (DDL) este un limbaj descriptiv cu comenzi specifice, utilizat pentru denumirea entităţilor BD şi a relaţiilor logice dintre acestea, pentru specificarea tipurilor şi a structurilor de date, precum şi a modurilor de vizualizare personalizate**.** Prin compilarea instrucţiunilor DDL rezultă setul de tabele al BD şi astfel se creează structura BD şi **catalogul de sistem**.
Urmează ca această structură să fie populată cu date. Conţinutul BD de la un anumit moment constituie o **instanţă** a BD. Prin manipularea datelor, rezultă noi instanţe ale acesteia.

I.6 ARHITECTURA BAZELOR DE DATE

Iniţial grupul DBTG (*Data Base Task Group*) a propus o arhitectură a sistemelor de BD cu două nivele:

- **schema** BD - reprezintă nivelul inferior, de implementare şi întreţinere
- **subschema** BD – este nivelul superior, folosit pentru realizarea vederilor utilizatorilor.

Această arhitectură are dezavantajul că nu permite generarea vederilor externe independent de schema internă a BD. Orice modificare a acesteia din urmă conduce la schimbarea vederii externe.

Personalizarea acestor vederi în sistemele multiutilizator este posibilă prin introducerea unui al treilea nivel, intermediar, care să separe detaliile de implementare de cele impuse la vizualizare. De aceea, institutul ANSI (*American National Standards Institute*) şi comitetul SPARC (*Standards Planning and Requirements Committee*) au propus arhitectura cu trei nivele ANSI/X3/SPARC sau ANSI-SPARC, care include:

- nivelul intern
- nivelul conceptual
- nivelul extern.

În figura I.4 sunt reprezentate cele trei nivele ale arhitecturii ANSI-SPARC pentru BD, cu toate elementele componente.

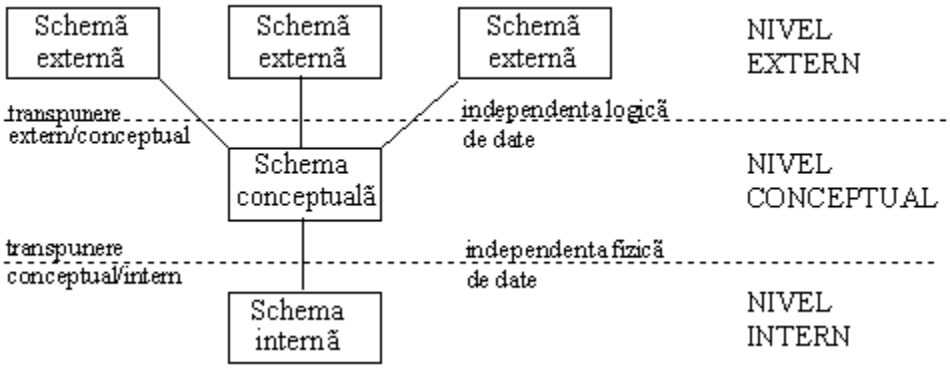

Figura I.4 Arhitectura ANSI-SPARC a unei BD

Nivelul extern este format din vederile utilizatorilor, fiecare incluzând anumite entităţi, relaţii şi atribute, eventual cu reprezentări diferite ale aceloraşi date, cu combinaţii ale acestora sau cu atribute derivate, pe baza unor **scheme externe**. Pe acest nivel, se lucrează cu

modele externe bazate pe înregistrări şi generate în funcţie de diferitele cerinţe ale beneficiarilor bazei de date:

- **modelul de date relaţional**, bazat pe conceptul de relaţii matematice, reprezintă datele din BD şi relaţiile dintre ele sub formă de tabele;
- **modelul de date în reţea** reprezintă datele ca o colecţie de înregistrări (noduri) iar relaţiile dintre acestea prin direcţiile sau muchiile unui graf.
- **modelul de date ierarhic**, similar modelului în reţea, foloseşte conceptul de nod-părinte şi permite unui nod din graf să posede numai un singur părinte, realizând o structură arborescentă.

Nivelul conceptual reprezintă o vedere generală a BD şi descrie datele şi relaţiile care sunt stocate în BD, într-o structură logică denumită şi **schemă conceptuală.** Aceasta nu depinde nici de modul de implementare a BD, nici de cerinţele de vizualizare ale utilizatorilor. Schema conceptuală cuprinde toate entităţile, atributele, relaţiile şi constrângerile datelor, informaţiile semantice despre date, normele de securitate şi regulile de integritate. Pe acest nivel se realizează modelul conceptual, bazat pe obiecte. Cele mai utilizate **modele conceptuale** sunt:

- **modelul Entitate-Relaţie** (ER)
- **modelul orientat pe obiecte**, care ia în considerare şi comportamentul entităţii, pe lângă atributele care descriu starea ei.

Procesul de modelare a datelor independent de modul de implementare, de SGBD ţintă, de programele de aplicaţie, limbaje de programare sau aspecte fizice, reprezintă ***modelarea conceptuală a BD.***

Nivelul intern reprezintă implementarea fizică a BD, pe baza unei **scheme interne** cuprinzând structurile de date şi de organizare a tabelelor şi fişierelor asociate. Acest nivel este responsabil de alocarea spaţiului de stocare a datelor şi a indexurilor, de descrierea şi plasarea înregistrărilor în spaţiul alocat, de codarea şi compresia datelor.
Pe nivelul intern se folosesc **modele fizice** ale BD care descriu modul de stocare a datelor în memoria calculatorului, structurile, ordinea şi căile de accesare ale înregistrărilor.
Nivelul intern interacţionează direct cu nivelul inferior, cel fizic, gestionat de sistemul de operare.

Schema internă este legată de cea conceptuală printr-o interfaţă de transpunere conceptual/intern care transformă ca format cererea procesului-client şi răspunsul procesului-server între cele două nivele.
Schemele externe sunt deduse din cea conceptuală prin transpunerea extern/conceptual.
Imunitatea schemelor externe faţă de modificările efectuate în schema conceptuală reprezintă **independenţa logică de date** a BD.
Imunitatea schemei conceptuale faţă de schimbările intervenite în schema internă este denumită **independenţă fizică de date** a BD**.**

I.7 REZUMATUL CAPITOLULUI

Baza de date (BD) reprezintă o resursă unică de informaţii, computerizată, stocată pe un server sau distribuită pe mai multe servere, partajată între mai mulţi utilizatori, accesibilă local sau de la distanţă.
Proiectarea unei BD începe cu procesul de modelare a datelor, pe baza regulilor de afaceri, exprimate de beneficiari în mod descriptiv sau sub forma unor întrebări.
Primul pas este acela de identificare a entităţilor, atributelor şi a relaţiilor care descriu baza de date. Urmează realizarea diagramei entitate-relaţie într-o primă formă. Printr-o analiză complexă şi sistematică, se reorganizează modelul astfel încât să nu existe elemente redundante, atribute cu valori multiple, derivate sau volatile, astfel încât BD să poată răspunde la toate întrebările exprimate de utilizatori şi să se poată securiza accesul la informaţiile cu caracter critic. Entităţile se asociază cu tabelele din BD, atributele cu coloanele din tabele iar înregistrările care se fac în BD devin linii ale tabelelor.
Arhitectura ANSI-SPARC cu trei nivele a BD permite realizarea independentă a structurii BD faţă de vederile externe create pentru diferitele categorii de utilizatori.

I.8 TERMENI SPECIFICI

bază de date (BD)
date
informaţii
catalog de sistem

dicţionar de sistem
sistem de gestiune a bazei de date (SGBD)
tip de entitate
entitate
atribut
relaţie
diagrama Entitate-Relaţie
cheie primară
cheie alternativă
cheie candidat
cheie străină
model de date
model conceptual
reguli de afaceri
normalizare
limbaj de manipulare a datelor
limbaj de definire a datelor
limbaj de interogare
schema bazei de date
subschema bazei de date
tabel
tranzacţie
vedere externă
baze de date relaţionale
NULL
NOT NULL
UNIQUE

I.9 APLICAŢIE PROPUSĂ

Realizaţi modelul conceptual şi diagrama entitate-relaţie pentru o BD folosind scenariul următor:

Se doreşte o BD pentru evidenţa produselor, clienţilor şi a angajaţilor unui magazin de electrocasnice, organizat în mai multe departamente: vânzări, achiziţii, personal, financiar, transport.

BD va fi folosită pentru gestiunea produselor (denumire, categorie, preţ de achiziţie, preţ de vînzare, formă de prezentare), a achiziţiilor (date, cantităţi, valori) şi a stocurilor. Departamentul de achiziţii doreşte să cunoască situaţia stocurilor pentru a planifica din timp achiziţiile de la furnizori. Se doreşte să se dispună de evidenţa clienţilor (date personale: nume, prenume, cnp, adresă, localitate de domiciliu, telefon). Departamentul Personal solicită ca BD să gestioneze informaţiile despre angajaţi (cod_angajat, nume, prenume, cnp, funcţie, departament, salariu, telefon).

Departamentul de Transport doreşte să dispună de evidenţa comenzilor de transport a produselor, făcute de clienţi la achiziţionarea acestora, pentru planificarea lor pe zone. Managerul magazinului doreşte să cunoască activitatea de vânzare a fiecărui vânzător pentru a recompensa pe cei mai performanţi angajaţi. Managerul este interesat de situaţia încasărilor şi a profitului obţinut în diverse perioade de timp (zi, lună, an).

I.10 TEST-GRILĂ

1. Cum se defineşte o bază de date?

- ❑ Colecţie de date
- ❑ Colecţie partajată de date
- ❑ Colecţie partajată de date, între care există relaţii logice
- ❑ Colecţie partajată de date, între care există relaţii logice, împreună cu descrierea datelor

2. Într-un tabel din BD, un atribut al unei entităţi defineşte:

- ❑ o coloană
- ❑ un câmp
- ❑ o linie
- ❑ un tabel

3. Null-ul reprezintă:

- ❑ Caracterul SPACE
- ❑ Valoarea ZERO
- ❑ Un şir de caractere zero
- ❑ Lipsa valorii dintr-un câmp

4. Popularea BD cu date este realizată de către:
- ❑ proiectanţi
- ❑ programatori
- ❑ administratorul bazei de date
- ❑ administratorul de sistem

5. Controlul accesului la baza de date se face prin:
- ❑ catalogul de sistem
- ❑ dicţionarul de date
- ❑ sistemul de gestiune al BD
- ❑ schema BD

6. Identificarea înregistrărilor dintr-un tabel din BD se face printr-un atribut:
- ❑ cheie
- ❑ derivat
- ❑ opţional
- ❑ volatil

7. Manipularea datelor din BD se face prin:
- ❑ autentificare
- ❑ citire
- ❑ scriere
- ❑ ştergere

8. O entitate a unei BD se asociază cu:
- ❑ un tabel
- ❑ o coloană din tabel
- ❑ o linie din tabel
- ❑ o celulă din tabel

9. Descrierea datelor se face prin:
- ❑ atribute
- ❑ înregistrări
- ❑ specificarea tipului de date
- ❑ metadate

10. Modelul ANSI-SPARC pentru BD are:
- ❑ două nivele
- ❑ trei nivele
- ❑ patru nivele
- ❑ şapte nivele

CAPITOLUL II

ELEMENTELE SPECIFICE BAZELOR DE DATE

DIN CUPRINS:

II.1 ENTITATE

O bază de date poate fi descrisă în termenii fundamentali de entitate, instanţă, atribut, valoare, identificator, relaţie şi tranzacţie.
Orice bază de date foloseşte entităţi pentru a clasifica „obiectele" pe care le gestionează.

Prin definiţie, ***entitatea*** *este un obiect distinct inclus în BD, asociat unei persoane, firme, unui loc, document sau concept etc.*

Atunci când se doreşte folosirea unei baze de date pentru organizarea unei activităţi, trebuie puse în evidenţă elementele esenţiale ale acesteia şi clasificate, folosind entităţi. De exemplu, pentru baza de date a unei facultăţi este nevoie să folosim entităţile *STUDENT, PROFESOR, DISCIPLINĂ, SALĂ DE CURS* şi altele.
Putem asocia entitatea unui grup de elemente de acelaşi fel, care pot fi înşiruite într-o listă. Numele entităţii este întotdeauna un substantiv, care semnifică persoane, obiecte, evenimente. Entităţile au instanţe, adică valori particulare, care sunt asociate cu o singură apariţie a entităţii respective în activitatea curentă.
Studentul POPESCU ION va fi înregistrat în această bază de date ca o instanţă a entităţii *STUDENT*.
Studenta IONESCU ANA va fi înregistrată ca fiind o altă instanţă a entităţii *STUDENT*.
Dacă se asociază un tabel entităţii *STUDENT*, atunci în acest tabel vor apărea două linii corespunzătoare celor doi studenţi amintiţi.

Exerciţiu propus: Daţi exemplu de o entitate şi de câteva instanţe ale ei.

ATENŢIE! Unul şi acelaşi substantiv poate desemna într-o bază de date o entitate, iar în altă bază de date o instanţă a unei entităţi.
De exemplu, în baza de date cu animale, în entitatea *ANIMAL* poate să apară instanţa *pisică,* alături de altele precum *câine, cal, tigru.* În altă bază de date care descrie rasele de animale, fiecare dintre acestea va constitui o entitate aparte, cu un set de înregistrări cu rasele specifice. De exemplu, în al doilea caz vom include entitatea *PISICĂ,* cu rasele *siameză, persană, albastru de Rusia* etc.

Dependenţa existenţei unei entităţi de o altă entitate se numeşte **constrângere de participare**.

În general, acest tip de constrângere este dictat de regulile de afaceri ale activităţii descrise în BD. De exemplu, dacă divizăm o entitate în mai multe entităţi pentru a separa informaţiile cu caracter privat de cele cu caracter public, atunci la ştergerea entităţii-părinte va dispărea şi entitatea nou-creată din aceasta. Este un caz în care apare o constrângere de participare.

Entităţile pot fi clasificate după mai multe criterii.

Entitatea poate avea un caracter concret, precum o persoană, un animal, o maşină, caz în care spunem că avem de a face cu o entitate **tangibilă**, sau poate avea un caracter abstract, precum nivelul de pregătire al unui student sau gradul de dificultate al unui test, situaţie în care considerăm ca fiind entitatea **intangibilă**.

În cadrul aceleiaşi baze de date, putem lucra cu entităţi mai importante, specifice şi, în acelaşi timp, esenţiale în activitatea pe care o descrie BD, precum studenţii unei facultăţi, numite **entităţi tari** sau putem folosi şi entităţi cu caracter general, precum localităţile de domiciliu sau ţările din care provin studenţii, pe care le folosim ca să creem nişte liste de căutare pentru completarea uşoară şi fără erori a unor formulare de înscriere în BD şi pe care le vom numi **entităţi slabe**.

Fiecare entitate este asociată cu un tabel din structura internă a BD, pe care îl numim **relaţie** în cazul BD relaţionale.

II.2 ATRIBUT

O entitate este caracterizată printr-un set de **atribute** care permite identificarea, clasificarea sau cuantificarea instanţelor sale.

Numărul de atribute ale unui tabel sau relaţii reprezintă ***gradul relaţiei.***

Un atribut are o singură valoare pentru fiecare instanţă, la un anumit moment de timp. O valoare a unui atribut se poate schimba în timp, prin modificarea sau actualizarea datelor.

De exemplu, atributul *anul naşterii* al studentului POPESCU ION are valoarea ’1990’, în timp ce adresa de domiciliu a acestuia, exprimată ca şir de caractere ’Iaşi, bd. Independenţei, nr. 10, bl. T, et.3, ap.12’, este considerată tot o valoare unică. În acest caz, putem selecta

studenţii înscrişi în BD şi născuţi într-un anumit an, dar nu putem face selecţia după localitatea de domiciliu, pentru că localitatea apare doar ca o porţiune dintr-un şir şi nu ca un atribut independent. Este important să se stabilească de la început la ce întrebări trebuie să răspundă BD pentru a alege corect toate atributele necesare penrtu descrierea unei entităţi.
Atributele pot fi de două tipuri:

- **volatile**, cu valori care se schimbă automat în timp, precum vârsta studentului, sau în timpul activităţii, de exemplu stocul de produse al unui magazin.
- **nevolatile**, cu valoare fixă, cum este anul de naştere al studentului.

Chiar şi unele atribute care se schimbă foarte rar şi atunci numai prin modificare directă, dictată de administrator, vor fi considerate nevolatile. Este cazul, de exemplu, al atributului *număr de telefon*. Dacă o persoană îşi schimbă numărul de telefon, poate solicita administratorului BD să actualizeze „valoarea" câmpului respectiv. Avem de a face cu un atribut nevolatil, care se schimbă în anumite condiţii şi nu în mod automat.

Recomandare: Folosiţi atribute nevolatile şi evitaţi, dacă este posibil, atributele volatile.

Atributele volatile pot fi deduse din cele nevolatile şi, în general, sunt folosite la afişarea unor informaţii specifice, precum vârsta unei persoane.
Valoarea unui atribut poate fi un şir de caractere, un număr, o dată calendaristică, o valoare de timp, o imagine, un fişier audio etc. Toate acestea reprezintă **tipul** sau **formatul datelor**.
Fiecare atribut este descris prin tipul de date folosit pentru exprimarea valorilor lui. Metadatele includ tipul datelor pentru fiecare atribut din catalogul de sistem.

Valoarea unui atribut poate fi:

- **obligatorie** (opţiunea *NOT NULL,* notată uneori cu prefixul „asterisc" *)
- **opţională** (opţiunea *NULL,* notată cu un „cerculeţ" o)

De exemplu, *numele*, *prenumele* şi adresa unui client al unui magazin online sunt atribute cu valori obligatorii pentru a putea expedia marfa comandată unui magazin online. Însă, *numărul de telefon fix* este un atribut cu valoare opţională, deoarece clientul poate să indice doar un număr de telefon mobil.

Valoarea unui atribut care nu este cunoscută la un anumit moment sau nu este aplicabilă reprezintă un ***null.***

O altă caracterizare extrem de importantă a atributelor este aceea dată de unicitatea valorii acestuia. Acele atribute folosite pentru identificarea unică a înregistrărilor dintr-un tabel din BD trebuie să aibă o **valoare unică** (opţiunea *UNIQUE*) şi nu **valori repetate**. De exemplu, codul numeric personal este unic dar anul naşterii unei persoane poate avea aceeaşi valoare în mai multe înregistrări.

Exerciţiu propus: Alegeţi o entitate dintr-o BD şi enumeraţi câteva atribute ale ei. Specificaţi pentru fiecare atribut caracterul pe care îl are: obligatoriu/opţional, volatil/nevolatil, unic/repetitiv. Faceţi câteva înregistrări în tabelul asociat, cu valori specifice.

Dacă o persoană, furnizează administratorului BD mai multe numere de telefon, atunci am fi tentaţi să le înscriem pe toate în acelaşi câmp, eventual separate prin virgule. Spunem că avem un atribut cu valori multiple, care îngreunează procesul de căutare şi de sortare a datelor din BD. Nu este indicat să folosiţi atribute cu valori multiple!

Recomandare: Definiţi mai multe atribute similare atunci când estimaţi posibilitatea apariţiei de valori multiple.

Înregistrările dintr-un tabel din BD sunt diferenţiate printr-un identificator unic (UID – *Unique IDentifier*) care poate fi un atribut sau un grup de atribute ale entităţii asociate tabelului, numit **cheie** sau **supercheie.**

O supercheie pentru care nici un subset de atribute nu este o supercheie se numeşte **cheie candidat.**

Se pot găsi mai multe chei candidat pentru aceeaşi relaţie, dar una singură este aleasă, la un anumit moment, ca UID şi aceea este numită **cheie primară** (PK – *Primary Key*). Cheia primară se scrie subliniat în lista atributelor entităţii, realizată în limbaj descriptiv, sau precedată de caracterul „diez” #. Celelalte chei se numesc **chei alternative.** În mod firesc, setul tuturor atributelor unei entităţi, constituie un UID şi o cheie. O cheie formată din mai multe atribute se numeşte **cheie compusă.**

Uzual, se aleg însă acele chei cu cât mai puţine atribute componente, de preferat unul singur, cu redundanţă nulă. Se obişnuieşte chiar să se definească un atribut abstract de tip identificator, cu autoincrementare la fiecare nouă înregistrare, asemenea numerelor de ordine

folosite în tabele, care să fie cheia primară a entităţii descrise şi să fie formată dintr-un singur atribut.

Exerciţiu propus: Deduceţi din setul de atribute al entităţii STUDENT cheile posibile şi alegeţi cheia primară.

Atributul este asociat cu o coloană a unui tabel, având o denumire proprie, distinctă. Ordinea atributelor sau a coloanelor din tabel nu este importantă.

Domeniul este mulţimea valorilor pe care le poate lua un atribut. Când vorbim de valori, nu trebuie să ne gândim numai la valori numerice. De exemplu, putem avea valori de tip „şir de caractere" sau de tip „dată-timp" şi să impunem anumite restricţii asupra acestora. Dacă înscriem codul numeric personal al unei persoane, este util să restricţionăm lungimea şirului şi tipul caracterelor, adică să fie introduse exact 13 caractere şi toate de tip „cifră", de la 0 la 9.

Faptul că oricărui atribut i se dau valori dintr-un anumit domeniu reprezintă o **constrângere de domeniu.**

Alte reguli impuse de utilizatorii sau de administratorii unei BD se numesc **constrângeri de întreprindere.**

De exemplu, dacă regulile afacerii specifică moneda în care se exprimă preţurile produselor unui magazin spunem că lucrăm cu o constrângere de întreprindere.

II.3 RELAŢIE

Între entităţile folosite pentru modelarea unei baze de date, se stabilesc o serie de relaţii.

De exemplu, profesorul pune note studenţilor. Studenţii de la o specializare frecventează orele de curs de la anumite discipline. Un profesor este titular la una sau mai multe discipline.

Toate aceste relaţii pe care le identificăm în scenariul bazei de date a unei facultăţi, pot fi descrise prin anumite atribute de legătură. De exemplu, relaţia *EXAMEN – STUDENT* va fi descrisă de atributul *nrmatr*, în relaţia *EXAMEN - DISCIPLINĂ* intervine atributul *cod_disciplină* şi aşa mai departe.

Relaţia dintre două entităţi, privită într-un sens, poate avea fie caracter **sigur,** fie **opţional**. De exemplu, la o disciplină sigur se dă măcar un examen dar este opţional studiul unei discipline de către un student, aceasta depinzând de specializarea la care acesta este înscris. Prin

convenţie, relaţiile sigure sunt reprezentate grafic, cu o linie continuă, în timp ce relaţiile opţionale se reprezintă cu o linie întreruptă (Figura II.1).
Anumite atribute se vor repeta atunci când se implementează baza de date, de exemplu în limbaj SQL, tocmai pentru a evidenţia relaţiile dintre entităţi.
Dacă atributul de legătură este cheie într-un tabel, atunci, în tabelul în care este folosit pentru relaţionare, el se numeşte **cheie străină** (FK – *Foreign Key*). De exemplu, atributul care exprimă numărul grupei, *nr_grupa*, care este cheie primară a entităţii *GRUPA*, apare ca şi cheie străină în setul de atribute al entităţii *STUDENT*.

Setul de atribute care constituie o cheie candidat a altei relaţii se numeşte ***cheie străină.***

Avem de a face aici cu o dublare aparentă a datelor dar prin referinţa care se face între tabele, datele se introduc o singură dată, în tabelul principal, urmând să fie înscrise şi actualizate automat în tabelul relaţionat cu primul.
Nici un atribut dintr-o cheie primară a unei relaţii de bază nu poate fi **null** (lipsă valoare), aceasta reprezentând **regula** sau **constrângerea de integritate a entităţilor.**
Valoarea unei chei străine trebuie să fie egală cu valoarea unei chei candidat din relaţia sa de bază sau un null. Această condiţie reprezintă **regula de integritate referenţială.**
Se observă că este greu de descris în cuvinte scenariul pe care se va construi baza de date şi, de aceea, se preferă să se reprezinte grafic entităţile, atributele şi relaţiile dintre acestea, sub forma unei **diagrame Entitate – Relaţie**.

Exerciţiu propus: Exprimaţi în cuvinte relaţiile dintre entităţile folosite în baza de date a unui magazin (cea propusă în paragraful I.9) şi identificaţi atributele de legătură.

II.4 DIAGRAMA ENTITATE - RELAŢIE

Diagrama Entitate-Relaţie (ERD – *Entity - Relation Diagram*) modelează grafic BD. În această diagramă sunt reprezentate entităţile, relaţiile şi, eventual, atributele, sub forma unui graf neorientat.

De exemplu, în figura II.1 este reprezentată succint (fără atribute) diagrama ER a BD a unei facultăţi.

Se observă că diagrama ER nu depinde de modul de implementare a BD ca aplicaţie finală (*implementation-free*). Nu am spus nimic despre cum se va implementa BD pentru că, în faza modelării conceptuale, acest lucru nu contează şi pe baza diagramei ER se poate face implementarea BD în mai multe moduri. Nici tipul de BD nu contează în etapa modelării

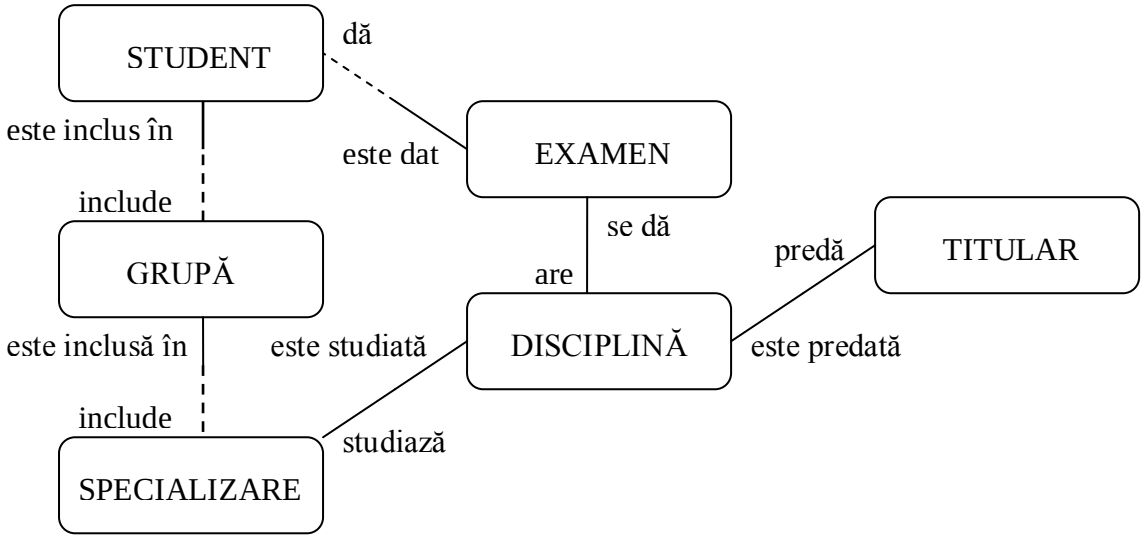

Fig. II.1 Diagramă-Entitate – Relaţie

conceptuale. Aceasta poate fi realizată pe principiul BD relaţionale, sau a BD ierarhice sau de tip reţea, inclusiv în varianta necomputerizată, cu date înregistrate pe suport de hârtie şi îndosariate. Organizarea datelor se face indiferent de tipul BD, prin modelarea conceptuală.

Diagrama ER trebuie reprezentată şi analizată cât mai minuţios şi sistematic, pentru a fi siguri că prin entităţile, atributele şi relaţiile definite, diagrama răspunde la toate întrebările puse în faza de descriere a BD prin regulile de afaceri, exprimate în documentaţia BD.

De exemplu, diagrama din figura II.1 poate să răspundă la diverse întrebări: Care dintre studenţi participă la un examen? Cu ce profesor se dă un examen? Câţi studenţi sunt într-o anumită grupă?

Sunt foarte importante şi atributele fiecărei entităţi pentru că lipsa unor atribute poate să facă imposibil răspunsul la anumite întrebări ale beneficiarilor BD. De exemplu, dacă în exemplul de mai sus, entitatea *EXAMEN* nu are un atribut *sală* şi atribute de tip dată şi timp, atunci din interogarea BD nu vom putea şti în ce sală, în ce zi şi la ce oră este planificat un examen.

Exerciţiu propus: Completaţi diagrama ER din figura II.1 cu atributele asociate entităţilor şi eventual cu noi entităţi, care sunt necesare pentru ca BD să răspundă la anumite întrebări. Scrieţi pe fiecare linie linie de legătură dintre entităţile relaţionate, care este atributul prin care se exprimă acea relaţie.

Diagrama ER este realizată cu scopul de a cuprinde într-o structură simplă şi bine organizată toate datele care sunt necesare organizaţiei beneficiare.

Regulă: Informaţiile trebuie să apară într-un singur loc din BD, adică să nu se repete.

De exemplu, nu vom scrie în tabelul disciplinelor informaţii specifice cadrului didactic, precum gradul didactic, deoarece aceasta ar însemna să repetăm aceeaşi informaţie pe mai multe linii din tabel, dacă acesta predă mai multe discipline. Când titularul este promovat, informaţiile despre gradul didactic ar trebui reactualizate pe toate liniile disciplinelor predate de el şi există riscul de a nu face toate modificările necesare (erori ale operatorului uman). De aceea, am definit o entitate separată pentru titularii de discipline, cu atribute specifice, pe care o relaţionăm cu entitatea *DISCIPLINĂ* doar prin identificatorul unic al titularului.
Informaţiile care pot fi deduse sau derivate din altele, nu vor fi şi ele incluse în modelul BD prin alte atribute. De exemplu, dacă pentru o entitate *ANGAJAT* se foloseşte atributul *data_angajării*, atunci vechimea angajatului poate fi dedusă din data curentă şi data angajării şi nu trebuie să apară ca un atribut separat. În cazul atributelor derivabile, se păstrează acel atribut care nu are caracter volatil.

II.5 CARDINALITATEA UNEI RELAŢII

Descrierea unei relaţii dintre entităţi, se face şi prin numărul de participanţi la o relaţie. De exemplu, într-o relaţie *DISCIPLINĂ – TITULAR*, un singur titular predă într-un an o disciplină, la o anumită specializare. Totuşi un titular poate să predea mai multe discipline în acelaşi an universitar. Vorbim în acest caz de o relaţie „**unul-la-mulţi**”, notată simplu 1:M.
Numărul de participanţi la o relaţie, exprimat în ambele sensuri, reprezintă **raportul de cardinalitate** al relaţiei.
Raportul de cardinalitate este impus prin regulile de afaceri ale activităţii decrise în BD.

De exemplu, este posibil ca relaţia *DISCIPLINĂ – TITULAR* să aibă un alt raport de cardinalitate, de tipul „**mulţi-la-mulţi**" (M:M), dacă regulile facultăţii permit ca aceeaşi disciplină să poată fi predată de mai mulţi titulari iar studenţii să poată alege între aceştia.
Există însă şi relaţii restrictive, de tipul „**unul-la-unul**" (1:1), în care intervine câte un singur participant din partea fiecărei entităţi implicate într-o relaţie.
Un astfel de raport 1:1 poate să apară în mod firesc sau să fie restricţionat de regulile activităţii. De exemplu, dacă se consideră baza de date a unui cabinet medical, relaţia *PACIENT – FIŞĂ MEDICALĂ* are un raport de cardinalitate 1:1 pe care îl interpretăm astfel: orice pacient are o singură fişă medicală la acel cabinet, iar o fişă medicală corespunde unui singur pacient.
In diagrama ER, raportul de cardinalitate poate fi pur şi simplu scris pe fiecare linie prin care se reprezintă o relaţie dintre entităţi sau poate fi reprezentat grafic printr-o linie simplă în cazul unui participant sau printr-un mănunchi de trei linii, pentru mai mulţi.
În Figura II.2, este reprezentată doar o parte a diagramei ER pentru BD a unei facultăţi. Sunt figurate două relaţii, una de tipul 1:M şi cealaltă de tipul M:M. Interpretarea lor este următoarea: Un student este inclus într-o singură grupă, dar o grupă include mai mulţi studenţi şi un student dă mai multe examene iar un examen este dat de mai mulţi studenţi.

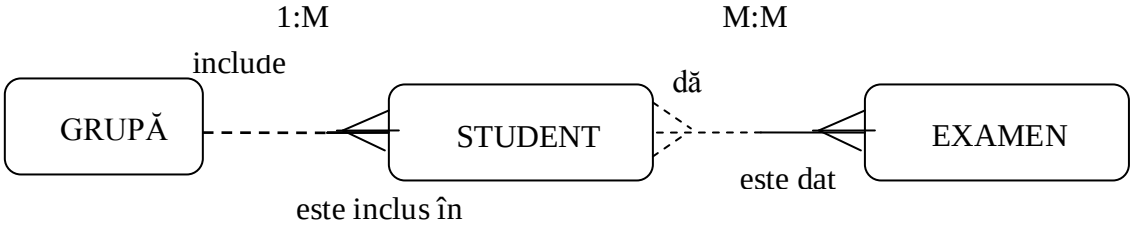

Figura II.2 Reprezentarea grafică a cardinalităţii unor relaţii

Restricţiile impuse asupra raportului de cardinalitate al unei relaţii se numesc **constrângeri de cardinalitate**.

Exerciţiu propus: Completaţi diagrama ER din figura II.1 cu valorile rapoartelor de cardinalitate corespunzătoare relaţiilor dintre entităţi.

Atributul care exprimă o relaţie 1:1 poate fi ales din setul de chei al oricărei din cele două entităţi implicate în relaţie.

În cazul relaţiilor 1:M, atributul de legătură trebuie ales numai de la entitatea cu participare singulară la relaţie. Dacă s-ar alege pentru exprimarea relaţiei un atribut cheie de la entitatea cu participare multiplă, atunci în tabelul celeilalte entităţi vor apărea valori multiple pentru acel atribut.

În relaţiile M:M, indiferent din care parte a relaţiei se alege atributul de relaţionare, acesta va avea valori multiple.

În general, nu se admit atribute cu valori multiple şi de aceea nici relaţiile M:M nu sunt acceptate. Relaţiile M:M trebuie eliminate din diagrama ER prin definirea de noi entităţi.

II.6 VEDERE

O relaţie (tabel) produsă la cererea unui client pe baza relaţiilor existente în BD se numeşte **vedere** (view).

Putem considera vederea externă din BD ca fiind o **relaţie virtuală**, adică se generează un tabel virtual, afişat în vederea externă, pe baza datelor existente în tabelele reale din BD. Acel tabel din vederea externă nu există de fapt în BD.

Pentru a diferenţia tabelele din BD, în care stocăm datele, de cele create în vederile externe, vom folosi termenul de **relaţie de bază** pentru tabelele interne.

O relaţie cu o anumită denumire, corespunzătoare unei entităţi din schema conceptuală a BD, ale cărei tupluri sunt stocate fizic în BD, se numeşte **relaţie de bază.**

Trebuie să se facă distincţie între datele stocate intern, în tabelele din BD, şi informaţiile care sunt afişate în vederile externe.

Conform arhitecturii ANSI-SPARC, nivelul extern este complet separat de cel intern, şi de aceea structura internă a unei BD nu poate fi dedusă din vederile externe. De aceea spunem că vederile se generează în mod transparent. Informaţiile care se afişează pot fi toate citite sau derivate din diferite date, stocate în unul sau mai multe tabele.

De exemplu, dacă se doreşte afişarea notelor obţinute la examene de studenţii unei grupe la toate disciplinele dintr-un semestru, atunci se vor citi datele din relaţiile de bază corespunzătoare entităţilor *STUDENT, GRUPĂ, EXAMEN, DISCIPLINĂ,* relaţionate prin atribute abstracte de tip *nrmatr*, *nr_grupă, cod_disciplină,* urmând ca afişarea listei să se realizeze cu etichete sugestive de genul *Numele şi prenumele, Grupa, Denumirea disciplinei, Notă.*

Exerciţiu propus: Reprezentaţi capul de tabel al unei vederi externe prin care să se afişeze lista disiciplinelor şi a titularilor care predau la o specializare, generată pe baza diagramei ER din figura II.1, cu denumiri de coloană cât mai sugestive. Din ce relaţii de bază se culeg datele pentru această vedere externă? Care sunt atributele folosite?

Se pot adopta diferite formate de afişare a rezultatelor interogărilor, în funcţie de specificul aplicaţiei şi de preferinţele utilizatorilor bazei de date.

Prin utilizarea vederilor se asigură securitatea BD şi se personalizează vederile fiecărui utilizator.

Vederile sunt dinamice şi orice modificare în relaţiile de bază se reflectă în vederile externe.

Succesiunea evenimentelor, în cazul realizării unor modificări ale datelor, este dictată de regulile de afaceri care descriu aplicaţia.

De exemplu, se poate solicita modificarea instantanee a valorii afişate dacă se reactualizează datele din BD, sau se poate impune menţinerea vechilor valori în vederile externe şi doar la o nouă accesare a aplicaţiei să se actualizeze şi vederile externe.

De exemplu, la o bursă de valori utilizatorii vor să ştie imediat dacă se schimbă valorile acţiunilor, în timp ce pentru BD a unei biblioteci care primeşte volume noi, este posibil să nu se dorească afişarea instantanee a noilor titluri, până nu se definitivează lista acestora.

De aceea spunem că nu toate vederile pot fi reactualizate.

Există vederi externe prin intermediul cărora se pot manipula datele din BD. De exemplu, o vedere externă care conţine un formular de înscriere în BD cu clienţii unui magazin, permite introducerea datelor în relaţiile de bază ale BD. Dacă vederea este reactualizată, de exemplu clientul îşi schimbă domiciliul sau numărul de telefon, atunci şi relaţia de bază prin care a fost generată, trebuie reactualizată.

Observaţii:

- Nu sunt permise reactualizările prin intermediul vederilor care implică relaţii de bază multiple sau operaţii de acumulare sau de grupare a atributelor.
- Este permisă modificarea structurii datelor sau a modalităţilor de stocare a lor fără a afecta vederile externe (tabelele virtuale).
- Eliminarea anumitor elemente (câmpuri, atribute etc) din schema BD poate deranja vederile care le utilizează la momentul respectiv. Acest neajuns este evitat prin tratarea corespunzătoare a tranzacţiilor în BD.

II.7 REGULILE LUI CODD

În prezent, cele mai utilizate sunt bazele de date relaţionale (BDR). Accesul la acestea şi gestionarea lor se face cu un sistem de gestiune a bazelor de date relaţionale (SGBDR, în engleză RDBMS – *Relational Database Management System*).
Codd a enunţat o regulă fundamentală şi 12 reguli specifice pentru SGBDR:

Regula fundamentală
Orice sistem care pretinde sau i se face reclamă de a fi un SGBDR trebuie să fie capabil să gestioneaze în întregime bazele de date prin capacităţile sale de relaţionare.

Când vorbim de relaţionare, ne referim la relaţiile dintre atributele unei entităţi, în cadrul unui singur tabel, dar şi la relaţiile care se stabilesc între mai multe entităţi, prin atributele de legătură. Capacitatea de a relaţiona informaţiile se referă la o funcţionare perfectă, fără erori, a SGBD. Dacă datele sunt dublate în BD şi pot să apară erori de acualizare, considerăm că acel SGBD nu gestionează corect datele. Dacă nu există anumite legături între entităţi, atunci este posibil ca unele raportări să fie eronate, prin urmare avem de a face cu un SGBD imperfect.

Cele douăsprezece reguli enunţate de Codd pentru BDR sunt următoarele:

1. **Reprezentarea informaţiilor**

La nivelul logic, toate informaţiile dintr-o BD relaţională sunt reprezentate explicit într-un singur mod – prin valorile din tabele (relaţii de bază).

2. **Accesul garantat**

Se garantează faptul că orice element (dată) dintr-o BDR este accesibil din punct de vedere logic prin apelarea la o combinaţie de nume de tabel, valoare a cheii primare şi nume de coloană.

3. **Tratarea sistematică a valorilor null**

Valorile null sunt acceptate pentru a reprezenta informaţiile lipsă şi pe cele care nu pot fi aplicate în mod sistematic, indiferent de tipul de date.

4. **Catalog dinamic online, bazat pe modelul relaţional**

Descrierea BD este reprezentată la nivel logic în acelaşi mod ca şi datele obişnuite, astfel încât utilizatorii autorizaţi pot folosi pentru interogarea acesteia acelaşi limbaj relaţional aplicat datelor curente.

5. **Sublimbaje de date cuprinzătoare**

Un sistem relaţional poate accepta mai multe limbaje şi diverse moduri de utilizare a terminalelor. Totuşi trebuie să existe cel puţin un limbaj ale cărui instrucţiuni să poată exprima următoarele:

- *definirea datelor*
- *definirea vederilor*
- *manipularea datelor*
- *constrângerile de integritate*
- *autorizarea*
- *limitele tranzacţiilor (început, efectuare şi rulare înapoi).*

6. **Reactualizarea vederilor**

Toate vederile care sunt teoretic reactualizabile, pot fi reactualizate de către sistem.

7. **Operaţii de inserare, reactualizare şi ştergere de nivel înalt**

Capacitatea de tratare a unei relaţii de bază sau a unei relaţii derivate (vedere) ca pe un singur operand se aplică nu numai regăsirii datelor în BD, ci şi inserării, reactualizării şi ştergerii acestora.

8. **Independenţa fizică de date**

Programele de aplicaţii şi activităţile de la terminale rămân logic intacte ori de câte ori sunt făcute modificări, fie în metodele de stocare, fie în metodele de acces la date.

9. Independenţa logică de date

Programele de aplicaţii şi activităţile de la terminale rămân logic intacte ori de câte ori sunt făcute modificări în tabelele de bază cu păstrarea informaţiilor sau deteriorarea acestora.

10. **Independenţa de integritate**

Constrângerile de integritate specifice unei anumite BDR trebuie să poată fi definite în sublimbajul relaţional de date şi stocate în catalogul de sistem, nu în programele de aplicaţii.

11. **Independenţa de distribuţie**

Sublimbajul de manipulare a datelor dintr-un SGBDR trebuie să permită programelor de aplicaţiişi interogărilor să rămână aceleaşi din punct de vedere logic dacă şi ori de câte ori datele sunt centralizate sau distribuite fizic.

12. **Regula de nonsubversiune**

Dacă un sistem relaţional are un limbaj de nivel jos (câte-o-înregistrare-o-dată), atunci acel nivel nu poate fi folosit pentru a submina sau a ocoli regulile de integritate şi constrângerile exprimate în limbajul relaţional de nivel mai înalt (mai-multe-înregistrări-deodată).

De-a lungul timpului regulile lui Codd au fost stârnit multe controverse, dar se pare că ele sunt utile pentru analiza caracterului relaţional al oricărui SGBD. Un SGBD care nu le îndeplineşte este cu siguranţă nerelaţional.

II.8 TRANZACŢIA

Definiţie: Tranzacţia este definită ca o secvenţă de operaţii care se execută ca "o singură funcţie logică", asupra unei BD partajate între mai mulţi utilizatori.

Operaţiile realizate într-o tranzacţie, ca un tot unitar, pot să fie de citire, scriere, ştergere, creare şi se pot efectua asupra datelor dar şi asupra structurii de date. Este important ca să se efectueze integral şi corect succesiunea de operaţii propusă, nu parţial. În cazul în care o tranzacţie este incompletă, din diferite cauze (defecţiune tehnică, eroare de program etc.), BD trebuie să revină la starea iniţială, pe care o avea înainte de iniţierea tranzacţiei (Figura II.3).
Spunem că BD trebuie să fie permanent într-o stare stabilă, coerentă sau consistentă.
Tranzacţia efectuează transformări consistente asupra stării sistemului, menţinând consistenţa acestuia.

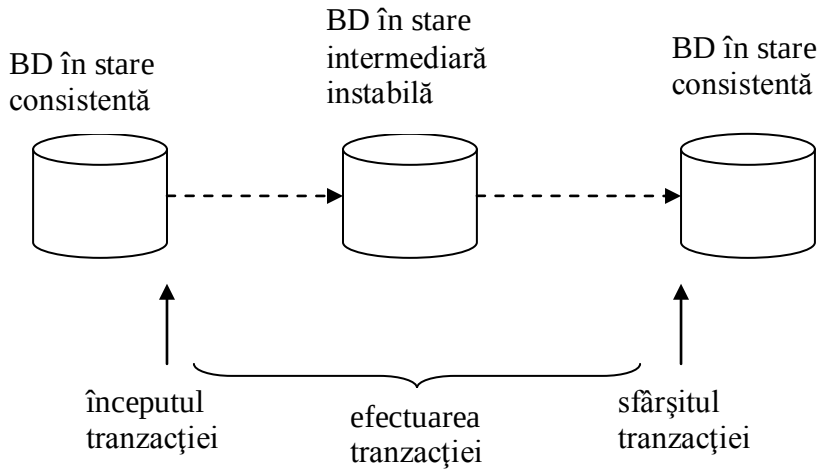

Fig.II.3 Etapele unei tranzacţii

Se definesc trei tipuri de tranzacţii:

- **de regăsire** a datelor în BD, pentru afişarea lor sau crearea unui raport (de tip tabel, diagramă etc.)
- **de reactualizare** a datelor, prin inserarea de noi date, ştergerea sau modificarea celor existente.
- **mixte**, de regăsire şi reactualizare.

Tranzacţiile se proiectează odată cu BD şi trebuie specificat comportamentul SGBD în cazul fiecăreia, prin opţiuni de derulare a tranzacţiilor, exprimate apoi, în faza de implementare a BD, în limbajul de programare adoptat (în particular SQL).

De exemplu, este importantă ordinea în care se vor efectua comenzile de mărire de salariu şi de acordare a unui premiu unui angajat. Să presupunem că la un salariu de 3000 RON, se acordă, la aceeaşi dată, o majorare de salariu de 20 % şi o primă de 10 %. Dacă se face mai întâi calculul primei, atunci aceasta va fi de 300 RON. Dacă mai întâi se majorează salariul la 3600 RON şi apoi se acordă prima, valoarea ei va fi de 360 RON.

Ordinea în care se fac aceste operaţii contează şi ea este stabilită prin regulile de afaceri care descriu activitatea firmei în cauză.

II.9 REZUMATUL CAPITOLULUI

Modelarea conceptuală a unei BD se face în termenii specifici de entitate, instanţă, atribut, valoare, identificator, relaţie şi tranzacţie.

Modul de alegere a entităţilor şi atributelor, stabilirea cheilor şi a relaţiilor dintre entităţi, precum şi constrângerile care se impun (de domeniu, de participare, de integritate, de cardinalitate) depind de scenariul sau regulile de afaceri care descriu aplicaţia bazei de date.

Tranzacţiile trebuie şi ele descrise în acest scenariu pentru ca proiectanţii şi programatorii să poată modela şi implementa corect BD.

Unei entităţi i se asociază un tabel, unei instanţe a entităţii îi corespunde o linie sau o înregistrare din tabel iar atributele apar ca şi coloane.

Fiecare atribut are pentru o instanţă o valoare unică. Aşa-zisele valori multiple trebuie evitate, deoarece nu permit sortarea corectă a datelor.

Cardinalitatea relaţiei, exprimată prin numărul participanţilor la o relaţie dintre două entităţi, poate fi 1:1, 1:M sau M:M. Relaţiile M:M trebuie eliminate din diagrama ER.

II.10 TERMENI SPECIFICI

Entitate
Instanţă
Entitate tare
Entitate slabă
Entitate tangibilă
Entitate intangibilă
Atribut
Atribut volatil
Atribut nevolatil
Atribut derivat
Valoare
Valoare obligatorie
Valoare opţională
Valoare unică

Valori multiple
Tipul datelor
Identificator unic (UID)
Cheie primară (PK)
Cheie candidat
Cheie alternativă
Cheie compusă
Cheie străină (FK)
Diagrama Entitate – Relaţie
Raportul de cardinalitate
1:1
1:M
M:M
Constrângere de domeniu
Constrângere de integritate
Constrângere de cardinalitate
Constrângere de participare
Regula de integritate referenţială
Regulile lui Codd
Relaţie de bază
Relaţie virtuală
Vedere
Tranzacţii de regăsire
Tranzacţii de reactualizare
Tranzacţii mixte

II.11 APLICAŢIE PROPUSĂ

Considerăm scenariul BD a unui magazin online cu un anumit specific. Stabiliţi întrebările la care trebuie să răspundă aceasta din diverse perspective de utilizare (client, agent de vânzări, manager). Reprezentaţi diagrama ER, indicând raportul de cardinalitate al fiecărei relaţii şi

atributele de legătură. Verificaţi dacă modelul ER astfel conceput răspunde întrebărilor beneficiarilor. Formulaţi modul de desfăşurare a unor tranzacţii şi impuneţi constrângerile necesare pentru funcţionarea corectă a SGBD.

II.12 TEST-GRILĂ

1. Ce elemente sunt reprezentate în diagrama ER?

- ❑ atribute
- ❑ entităţi
- ❑ tabele
- ❑ tranzacţii

2. O coloană dintr-un tabel din BD se asociază cu:

- ❑ un atribut
- ❑ o entitate
- ❑ o înregistrare
- ❑ o relaţie

3. Restricţiile impuse valorilor pe care le poate lua un atribut, reprezintă:

- ❑ Constrângeri de cardinalitate
- ❑ Constrângeri de domeniu
- ❑ Constrângeri de integritate
- ❑ Constrângeri de participare

4. În BD a unui cabinet medical, pentru entitatea *PACIENT* se folosesc atributele de mai jos. Care dintre ele are caracter volatil?

- ❑ nume
- ❑ prenume
- ❑ vârsta
- ❑ ocupaţia

5. Respectarea constrângerii de integritate se face prin opţiunea:

- ❑ *DEFAULT*
- ❑ *NULL*
- ❑ *NOT NULL*
- ❑ *UNIQUE*

6. Este adevărat că atributul de legătură dintre două entităţi:
- ❑ este cheia primară a unei entităţi
- ❑ este cheie candidat a unei entităţi
- ❑ este cheie străină a unei entităţi
- ❑ poate avea valori multiple

7. O entitate are o cheie primară, două atribute de tip *NOT NULL* şi trei atribute opţionale. Care este gradul relaţiei asociate entităţii?
- ❑ 1
- ❑ 2
- ❑ 3
- ❑ 6

8. Între două entităţi ale unui magazin, *CUMPĂRĂTOR* şi *PRODUS*, ce fel de relaţie se stabileşte?
- ❑ 1:1
- ❑ 1:M
- ❑ M:M

9. Din setul de atribute al entităţii *PERSOANĂ* (*nume, prenume, data_naşterii, adresă, localitate, judeţ, serie_carte_de_identitate, număr_carte_de_identitate, cnp*), care sunt cheile candidat care se pot defini?

.....
.....
.....
.....

10. Un set de operaţii care trebuie efectuate într-o BD, toate în bloc, se numeşte:
- ❑ cheie
- ❑ instanţă
- ❑ tranzacţie
- ❑ vedere

CAPITOLUL III

DIAGRAMA ENTITATE-RELAŢIE

DIN CUPRINS:

III.1 ALEGEREA SETULUI DE ENTITĂŢI ŞI ATRIBUTE

Modelarea unei BD începe cu identificarea entităţilor, atributelor şi a relaţiilor dintre entităţi. Reprezentarea lor grafică se face sub forma diagramei Entitate-Relaţie (ER), respectând anumite convenţii.

Modelul Entitate-Relaţie (ER) reprezintă principala tehnică de proiectare conceptuală a BD. Se mai foloseşte şi modelul orientat pe obiecte, care extinde definiţia entităţii prin luarea în considerare şi a comportamentului acesteia, pe lângă atributele care descriu starea ei.

Modelul ER sau schema conceptuală a bazei de date se exprimă în limbaj descriptiv printr-o serie de relaţii, specificate prin numele lor (numele entităţii), urmat de atributele fiecăreia între paranteze, cheia primară fiind subliniată.

De exemplu, o bază de date a unui magazin cuprinde date despre clienţi, angajaţi şi produsele oferite:

clienţi *(cod_client, nume, prenume, adresă, telefon, cnp, adresă_de_email)*

angajaţi *(id_angajat, nume, prenume, funcţie, salariu, data de naştere, cnp, adresa, telefon)*

produse *(cod_produs, denumire, categorie, preţ de vânzare, furnizor).*

Alegerea identificatorului unic (UID) şi a cheii primare ai unei entităţi nu este o chestiune foarte simplă. UID poate fi ceva foarte natural precum numele unei străzi sau ceva artificial, creat special pentru a face diferenţa între mai multe instanţe (codul poştal). Se poate folosi ca UID un singur atribut sau un set de mai multe atribute. Clasificăm astfel identificatorii unici ai entităţilor în **simpli** şi **compuşi.** Dintre toţi UID posibili ai unei entităţi, trebuie ales cel mai potrivit UID ca şi cheie primară, pe care îl numim **UID primar**. Ceilalţi UID se numesc **identificatori unici secundari** sau **chei candidat**. De exemplu, pentru entitatea *CLIENT* descrisă mai sus, se pot folosi mai mulţi UID: *cod_client, cnp, adresă_de_email* care sunt fiecare în parte identificatori simpli, sau se poate folosi un UID compus (*nume, prenume, adresă*). Alegerea cheii primare dintre toţi aceşti UID rămâne la latitudinea proiectantului care îl va alege pe cel mai potrivit din perspectiva clientului precum şi din a cea a aplicaţiei.

Nu sunt deocamdată exprimate relaţiile între entităţi, dar formulând eventuale interogări ale BD vom putea stabili anumite legături.

De exemplu, să presupunem că managerul magazinului doreşte să interogheze BD pentru aflarea încasărilor dintr-o lună. Dacă nu există o entitate asociată vânzărilor, cu un atribut

referitor la preţul de vânzare şi la numărul de bucăţi vândute, atunci efectuarea acelei interogări este imposibilă. Dacă acelaşi manager vrea să afle profitul magazinului pe o lună dar BD nu oferă informaţii privind cheltuielile magazinului (salariale, administrative sau de transport) şi nici preţul de achiziţie al fiecărui produs, ci numai preţul de vânzare, este din nou imposibil ca BD să ofere informaţia cerută.
Pe de altă parte dacă acea interogare nu este dorită şi BD serveşte numai la gestiunea produselor, atunci este posibil să încărcăm excesiv modelul BD cu entităţi şi atribute care nu servesc unor interogări făcute de utilizatori.
În acest sens, este necesară analiza setului de entităţi definite, precum şi a atributelor folosite, pentru a fi siguri că BD răspunde la toate interogările probabile.

Exerciţiu propus: Pentru BD descrisă în modelul anterior, stabiliţi noi entităţi şi atribute necesare realizării interogărilor referitoare la vânzări şi profit, descrise anterior. Transcrieţi în limbaj descriptiv noul set entităţi-atribute.

Relaţionarea entităţilor seamănă cu un joc de puzzle. De obicei avem multe entităţi, cu multe atribute şi chiar reprezentarea grafică a diagramei poate fi dificilă şi greu de urmărit.
Diagrama ER trebuie analizată cu atenţie pentru a nu avea relaţii lipsă. Optimizarea diagramei implică şi mai multe aspecte, de aplicare a unor reguli de normalizare, care permit eliminarea riscurilor de apariţie a unor erori sau omisiuni, la folosirea propriu-zisă a bazei de date. Acestea sunt de regulă greu observabile în procesul de proiectare a BD şi odată implementată aplicaţia cu BD, ele nu mai pot fi corectate.
De aceea este util să abordăm sistematic procesul de reprezentare, analiză şi de prelucrare a diagramei ER.

III.2 CONCEPTELE MODELULUI ENTITATE-RELAŢIE

Modelul conceptual Entitate-Relaţie (ER), dezvoltat iniţial de Chen în 1976, descrie structura BD şi tranzacţiile de regăsire şi reactualizare a datelor. Modelul ER se realizează independent de SGBD şi de platforma hardware pe care se va rula aplicaţia BD.
Să ne reaminitim că modelul ER include următoarele concepte:

Tip de entitate – obiect (fizic) sau concept (abstract) inclus în BD, cu existenţă independentă.
Entitate – instanţă a unui tip de entitate, unic identificabilă.
Tip de entitate slabă – tip de entitate a cărei existenţă depinde de alte tipuri de entităţi.
Tip de entitate tare – tip de entitate a cărei existenţă nu depinde de alte tipuri de entităţi.
Tip de relaţie – asociere între tipuri de entităţi.

Prin convenţie, fiecărei entităţi îi corespunde în BD un tabel sau o aşa-numită relaţie, pe care o denumim prin forma de plural a numelui entităţii în cauză, uzual scrisă cu litere mici.

De exemplu, entităţii *STUDENT* îi va corespunde în BD tabelul *studenţi*.

Relaţie – o instanţă a unui tip de relaţie.
Gradul unei relaţii – numărul de entităţi implicate în acea relaţie (unară, binară, ternară etc)
Relaţie recursivă – relaţie în care o entitate participă de mai multe ori, cu diferite roluri.
Raportul de cardinalitate – descrie numărul de entităţi implicate de fiecare parte a unei relaţii (1:1, unu-la-mulţi 1:M, mulţi-la-mulţi M:N).
Regulă de afaceri – regula pe baza căreia se stabileşte un raport de cardinalitate.
Atributul entităţii – proprietatea unui tip de entitate**.**
Atributul relaţiei – proprietate a unei relaţii.
Atribut simplu – atribut cu o singură componentă**.**
Atribut compus – atribut cu mai multe componente, cu existenţe independente.
Domeniul atributului – mulţimea valorilor pe care le poate lua un atribut.
Atribut cu o singură valoare – atribut care conţine o singură valoare pentru o entitate.
Atribut cu valori multiple – atribut care conţine mai multe valori în acelaşi câmp.
Atribut derivat – atribut dedus din valorile unui alt atribut sau set de atribute.
Cheie – articol de date care permite identificarea în mod unic instanţa unui tip de entitate.
Cheie candidat – atribut sau set de atribute care identifică în mod unic instanţa unui tip de entitate.
Cheie primară – cheia candidat aleasă pentru identificarea instanţelor unei entităţi.
Cheie alternativă – cheie candidat neutilizată ca şi cheie primară.
Cheie simplă – cheie candidat formată dintr-un singur atribut.
Cheie compusă – cheie candidat formată din mai multe atribute.

Asupra entităţilor şi relaţiilor din modelul ER se pot defini **constrângeri**:

- **de cardinalitate**, exprimate prin raportul de cardinalitate şi impuse de regulile de afaceri ale firmei.
- **de participare**, exprimând dependenţa existenţei unei entităţi de o altă entitate.
- **de integritate,** prin care se impune existenţa unei valori într-un anumit câmp aparţinând cheii primare.
- **referenţiale,** impuse asupra relaţiilor dintre entităţi sau tabele, prin care cheia străină ia aceleaşi valori ca şi cheia primară din tabelul de referinţă sau este NULL.

Participarea unei entităţi într-o relaţie poate fi:

- totală sau obligatorie (participarea la acea relaţie este garantată şi se exprimă prin termenii „trebuie să ...” sau „sigur ...”);
- parţială sau opţională (participarea la acea relaţie este posibilă, dar nu sigură, şi poate fi exprimată prin termenii „poate să ...” sau „este posibil ca ...”).

III.3 REPREZENTAREA GRAFICĂ A DIAGRAMEI ENTITATE-RELAŢIE

Modelul ER poate fi reprezentat grafic sub forma unei diagrame ER în care se aplică regulile următoare, ale aşa-numitei convenţii „în stea” (Figura III.1):

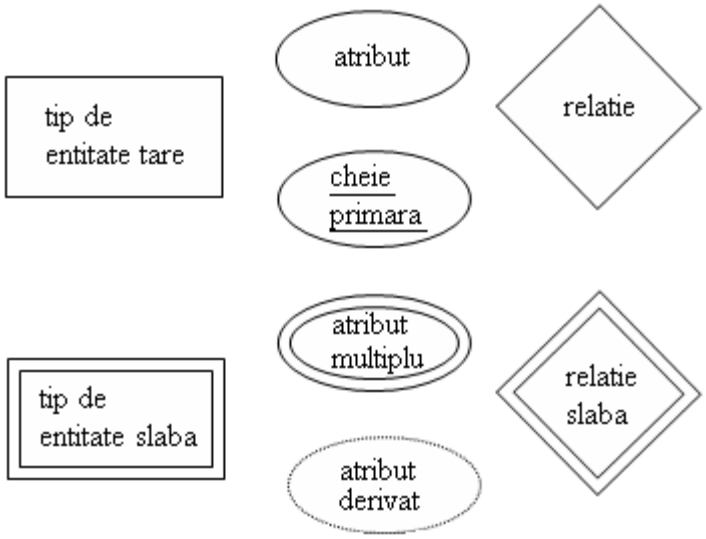

Fig. III.1 Reprezentarea grafică a elementelor modelului ER

- Entităţile sunt reprezentate ca dreptunghiuri etichetate cu numele acestora, cu chenar simplu pentru entităţile tari şi dublu pentru cele slabe.
- Atributele sunt reprezentate schematic sub forma unor elipse etichetate cu numele acestora şi legate de entitatea pe care o descriu prin segmente de dreaptă, sub forma unor raze. Denumirea cheii primare se subliniază. Conturul elipsei este continuu dacă reprezintă atribute propriu-zise şi discontinuu pentru atributele derivate. Conturul elipsei este reprezentat printr-o linie dublă în cazul atributelor cu valori multiple.
- Relaţiile sunt reprezentate sub formă de romburi, etichetate cu numele relaţiei, cu chenar simplu între entităţi tari şi cu chenar dublu când în relaţie este implicată o entitate slabă.
- Rapoartele de cardinalitate se reprezintă ca etichete ale liniilor ce unesc entităţile implicate într-o relaţie.
- Liniile de legătură între simbolurile grafice ale entităţilor şi relaţiilor sunt simple dacă participarea este parţială şi duble când participarea entităţii la acea relaţie este totală.
- Liniile de legătură pot fi etichetate şi cu valorile minimă şi maximă ale numărului de entităţi implicate într-o relaţie, scrise între paranteze.

Se observă din figura III.2 că, dacă se foloseşte un număr mare de atribute, reprezentarea grafică este greoaie şi dificil de urmărit.

De aceea, se preferă convenţia de reprezentare „în dreptunghi”, în care atributele unei entităţi sunt înscrise toate în dreptunghiul asociat acesteia şi se au în vedere următoarele reguli:

- O entitate se reprezintă printr-un dreptunghi (*softbox*), în care se scrie numele entităţii, la singular, cu toate literele mari.
- Aributele entităţii se scriu unul sub celălalt în acelaşi dreptunghi, sub numele entităţii, cu litere mici.
- Cheia primară se scrie prima în lista de atribute şi se marchează cu un „diez” (#).
- Un atribut obligatoriu se marchează cu un „asterisc” (*).
- Un atribut cu valoare opţională se marchează cu un „cerculeţ” (o).
- Relaţia dintre două entităţi se reprezintă printr-o linie, etichetată cu numele relaţiei.
- Linia de legătură dintre entităţi este continuă dacă participarea entităţii la relaţie este totală sau obligatorie.
- Linia de legătură dintre entităţi este întreruptă la capătul la care participarea entităţii este opţională.

- Relaţia se reprezintă cu un capăt ramificat, în trei („talpa gâştei”), dacă o entitate are o participare multiplă la acea relaţie.

Exerciţiu propus: Redesenaţi diagrama din figura III.2, folosind convenţia „în dreptunghi”.

Dacă mai multe persoane folosesc acelaşi set de entităţi şi atribute pentru a desena diagrama ER, este posibil să se obţină diferite reprezentări, în funcţie de amplasarea în plan a dreptunghiurilor prin care reprezentăm entităţile. De aceea, s-a stabilit **regula de orientare** a diagramei ER care impune sensul în care trebuie reprezentată şi „citită” aceasta: **de la stânga la dreapta** şi **de sus în jos**. Putem interpreta această regulă astfel: colţul de pornire este cel din stânga-sus, iar cel de finalizare a diagramei este cel din dreapta-jos (Figura III.3).

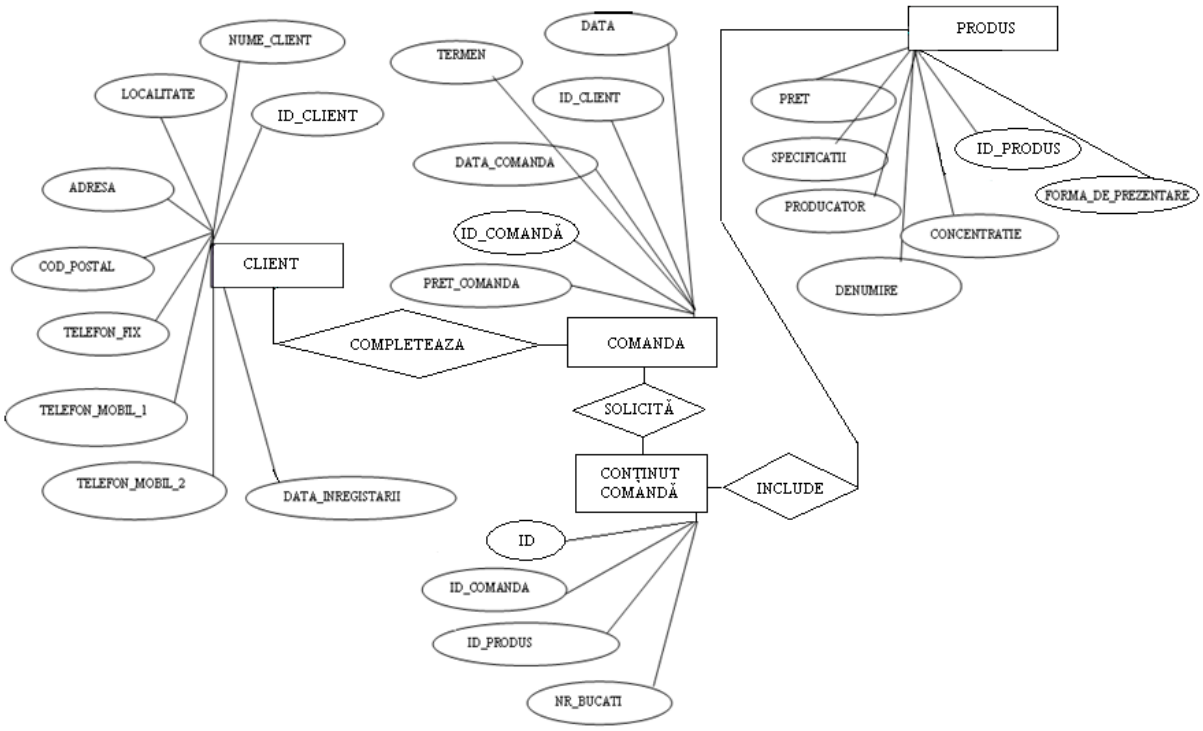

Figura III.2 Exemplu de diagramă ER, reprezentată prin convenţia „în stea”

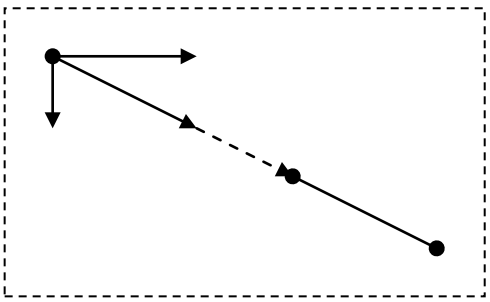

Figura III.3 Sensul de reprezentare şi de „citire” a diagramei ER

Reprezentarea grafică a modelului ER ca diagramă ER, respectând anumite convenţii prestabilite, constituie un mod universal de comunicare, pe care nu ni-l oferă descrierea în cuvinte a aplicaţiei cu BD.

III.4 CAPCANE DE CONECTARE

În proiectarea modelului de date conceptual, pot să apară anumite ambiguităţi de interpretare a relaţiilor care să conducă la imposibilitatea soluţionării unor interogări asupra BD. Aceste probleme ale modelului ER se numesc capcane de conectare şi sunt de două tipuri:

Capcanele în „evantai” apar atunci când aceeaşi entitate este implicată în mai multe relaţii de tip 1:M şi căile dintre entităţi devin ambigue. De exemplu, din diagrama ER reprezentată în figura III.4 nu putem deduce exact care proprietăţi sunt administrate de un anumit membru de personal. Prin modificarea ordinii de reprezentare a entităţilor implicate în aceste relaţii se obţine o structură de tip „arbore” prin care se elimină ambiguitatea diagramei (Figura III.5).

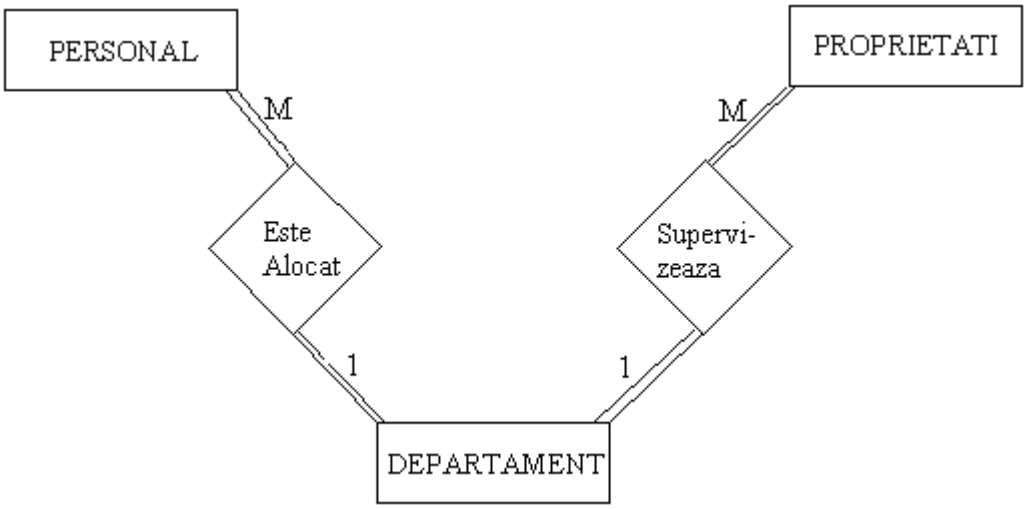

Figura III.4 Model ER cu capcană în evantai

Capcanele de întrerupere sunt cauzate de absenţa reprezentării unor relaţii în modelul ER. De exemplu, în modelul din figura III.5 lipseşte relaţia directă dintre entităţile Departament şi Proprietăţi, necesară identificării departamentului care se ocupă de o anumită proprietate. Introducând-o (Figura III.6), se creează o anumită redundanţă, preferabilă în unele situaţii.

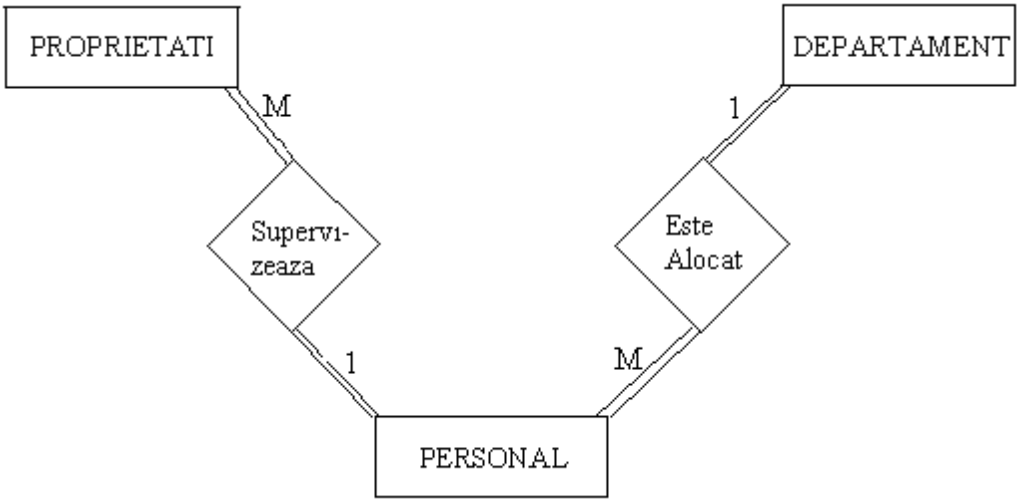

Figura III.5 Model ER modificat pentru eliminarea capcanei în evantai

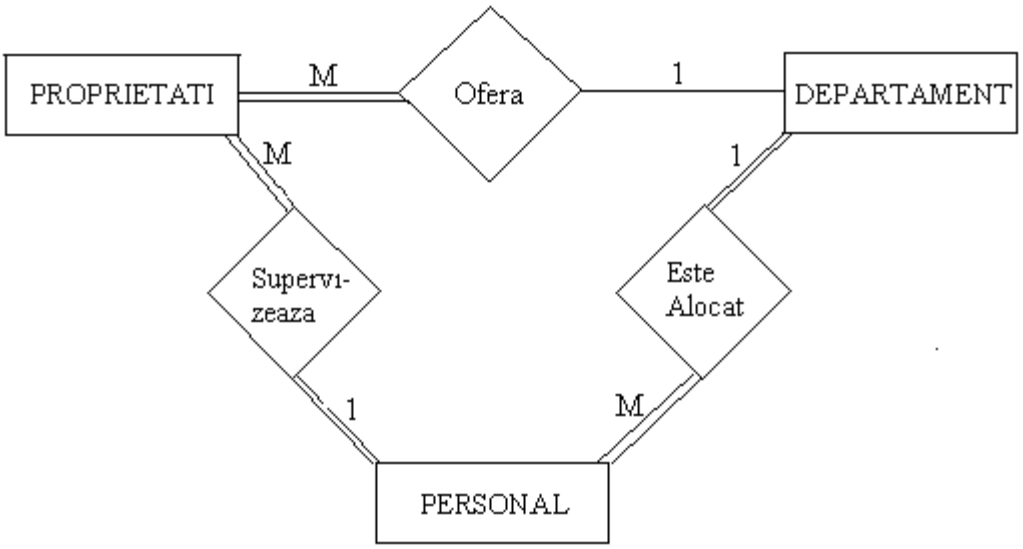

Figura III.6 Model ER modificat pentru eliminarea capcanei de întrerupere

III.5 INTERPRETAREA DIAGRAMEI ENTITATE-RELAŢIE

Revenind diagrama ER, pentru a o urmări ca şi continuitate, este util să exprimăm în cuvinte, cursiv, relaţiile dintre entităţi, pentra a fi siguri că sunt logice, asemănător modului în care se exprimă relaţiile de rudenie, de colegialitate sau de prietenie dintre mai multe persoane. De exemplu, „eu sunt prietena Cristinei, sora lui Radu, colegul tău de grupă”.

Este util să exprimăm adecvat **opţionalitatea** relaţiei (sigură sau probabilă) şi **cardinalitatea** ei (mai multe entităţi sau doar una singură intră în relaţie), la ambele capete ale unei conexiuni.

Pentru fiecare relaţie din diagrama ER, vom formula câte două fraze de forma:

*„**Fiecare** ENTITATE de tip 1 – **sigur** sau **poate** - se relaţionează cu – **una** sau **mai multe** ENTITĂŢI de tip 2”.*

Să nu uităm că relaţia trebuie interpretată în ambele sensuri, prima oră în sensul de la stânga la dreapta sau de sus în jos.

Să descompunem fraza de mai sus în elementele sale componente:

1. *Fiecare*
2. *ENTITATE de tip 1*
3. *sigur sau poate* (OPŢIONALITATEA)
4. *se relaţionează cu* (NUMELE RELAŢIEI)
5. *una sau mai multe* (CARDINALITATEA)
6. *ENTITĂŢI de tip 2.*

Opţionalitatea şi cardinalitatea unei conexiuni dintre entităţi pot să difere de la un capăt la altul.

Să „citim”, de exemplu, diagrama din figura III.5.

„Fiecare proprietate este sigur supervizată de un singur membru al personalului”.

„Fiecare membru din personal poate să supervizeze mai multe proprietăţi”.

„Fiecare membru al personalului sigur este alocat unui singur departament”.

„Fiecare departament sigur include mai mulţi membri”.

În general, ceea ce „citim” din diagrama ER, regăsim în descrierea sau scenariul aplicaţiei, adică în regulile de afaceri ale acesteia. De aceea, este util să se compare aceste reguli, cu relaţiile „citite” din diagrama ER, pentru a identifica eventualele relaţii care lipsesc din diagramă sau pe cele care nu corespund regulilor afacerii. De asemenea, pot să existe neclarităţi în interpretarea diagramei şi în astfel de cazuri se pot cere lămuriri reprezentanţilor beneficiarului, pentru a formula regulile în cauză şi a elimina ambiguităţile.

Exerciţiu propus: Interpretaţi diagrama ER din figura III.2, exprimând de fiecare dată opţionalitatea şi cardinalitatea relaţiei.

Este posibil ca o relaţie să se stabilească „în buclă”, de la o entitate la ea însăşi. Este cazul care se referă la o asociere între membrii aceluiaşi tip de entitate. De exemplu, ne interesează să ierarhizăm angajaţii dintr-un departament, pentru a şti fiecare angajat sau membru din departament, cui îi este subordonat. Pentru aceasta, se poate crea o buclă pe entitatea *PERSONAL*, prin atributul *şef*.

Interpretarea diagramei ER face posibilă verificarea modelului ER creat pentru o aplicaţie de BD. Se pot pune noi întrebări beneficiarului, pentru a stabili atât relaţiile-lipsă, cât şi opţionalitatea şi cardinalitatea relaţiilor, aşa cum sunt ele văzute de beneficiar, de specialistul în domeniul aplicaţiei şi nu de proiectanţii BD. Proiectantul BD nu trebuie să impună el câţi participanţi sunt la o relaţie şi nici dacă această participare este sigură sau opţională. Regulile de opţionalitate şi de cardinalitate sunt impuse de beneficiar.

Interpretarea diagramei ER pe care o face proiectantul BD urmează să fie citită de beneficiar, pentru ca acesta să îşi exprime acordul sau dezacordul cu modul în care au fost înţelese aspectele activităţii sale de către proiectant.

De exemplu, dacă modelaţi datele pentru o BD dintr-o uzină chimică, fără a avea cunoştinţe de specialitate în acel domeniu, este util ca modelul sau modelele pe care le faceţi să fie de fiecare dată verificate de un specialist, pentru a nu avea erori de interpretare.

III.6 MATRICEA RELAŢIILOR

O diagramă ER în prima sa formă, neprelucrată, conţine o serie de relaţii pe care proiectantul le-a identificat pe baza întrebărilor posibile ale utilizatorilor.

Este util să stabilim toate interogările posibile ale BD înainte de a trece la optimizarea diagramei entitate-relaţie.

Este posibil să nu fie identificate toate entităţile sau atributele necesare pentru realizarea unor interogări.

Exerciţiu propus: Pentru BD a unui cabinet medical, stabiliţi setul de entităţi şi atribute pe care le consideraţi necesare a le folosi la câteva interogări specifice şi faceţi asocierile dintre entităţi, atribute şi interogări. După aceasta, formulaţi o nouă interogare şi vedeţi dacă modelul BD creat răspunde şi la aceasta. Dacă nu, identificaţi elementele lipsă.

Odată stabilit setul de entităţi şi atribute, trecem la analiza relaţiilor dintre entităţi.

Care sunt informaţiile pe care BD vrem să le furnizeze? Aceasta este întrebarea de la care plecăm atunci când facem analiza relaţiilor. Ce elemente lipsesc din modelul de date construit? Acestea pot să treacă uşor neobservate dacă nu există o comunicare permanentă între proiectanţi şi reprezentanţii beneficiarilor BD.

Util este să reprezentăm diagrama ER, fără atribute, cu relaţiile identificate iniţial, mai ales dacă numărul de entităţi şi atribute cu care lucrăm este foarte mare. Analiza relaţiilor dintre entităţi o putem face urmărind pas-cu-pas interogările la care vrem să răspundă BD, sau în mod sistematic, folosind **matricea relaţiilor**.

Matricea relaţiilor permite identificarea în mod sistematic a tuturor relaţiilor dintre entităţi, dacă o citim pe linii şi pe coloane, inclusiv a acelor relaţii „în buclă”.

Să urmărim ca exemplu modelul de date pentru un cabinet medical (Figura III.7).

	MEDIC	PACIENT	FIŞĂ MEDICALĂ
MEDIC	—	tratează	completează
PACIENT	este tratat de	—	are
FIŞĂ MEDICALĂ	este completată de	corespunde	—

Figura III.7 Matricea relaţiilor pentru un model de date cu trei entităţi

Matricea se „citeşte” pe orizontală.

De exemplu: „*Fiecare MEDIC (sigur) tratează (mai mulţi) PACIENŢI*”.

Matricea relaţiilor nu pune în evidenţă opţionalitatea şi cardinalitatea relaţiilor. Dar puteţi observa cât de uşor se identifică eventualele relaţii care lipsesc din modelul de date, folosind această matrice. Toate celulele trebuie analizate, linie cu linie. Nici o celulă nu trebuie să rămână necompletată. Nu este obligatoriu să existe relaţii între oricare două entităţi, dar lipsa unei astfel de relaţii trebuie analizată şi stabilită pe baza regulilor de afaceri care descriu aplicaţia.

După stabilirea tuturor relaţiilor dintre entităţi în matricea relaţiilor, se redesenează diagrama ER completă.

Exerciţiu propus: Completaţi matricea relaţiilor pentru BD a unei facultăţi, care cuprinde entităţile STUDENT, GRUPĂ, AN DE STUDIU, PROFESOR, DISCIPLINĂ, SALĂ, ORĂ, EXAMEN, TIP DE ACTIVITATE.

III.7 NORMALIZAREA

Modelul de date creat pentru o aplicaţie de BD trebuie optimizat pentru a elimina anumite aspecte nedorite, precum redundanţa datelor sau unele dependenţe parţiale dintre atribute, astfel încât să nu mai apară erori în procesele de selecţie sau de actualizare a datelor şi, în general, la utilizarea BD. Acest proces de optimizare a modelului BD poartă numele de **normalizare.**

Normalizarea *este un procedeu de identificare a setului optim de relaţii, bazat pe dependenţele funcţionale dintre atribute, care are ca scop crearea unui model logic de date valid, neredundant, care să elimine riscurile de apariţie a anomaliilor de reactualizare a datelor în baza de date.*

Normalizarea precede etapa de implementare a BD deoarece aceasta implică unele modificări de structură în modelul creat (schimbări de entităţi, atribute sau relaţii).

Forma nenormalizată (UNF – *Unnormalized Form***)** a unui tabel din BD include date repetitive. Un astfel de tabel nu este considerat „relaţie” în BD sau BD respectivă nu poate fi considerată ca fiind relaţională.

Un exemplu de dublare a datelor este acela în care stocaţi un număr de telefon în mai multe agende (pe mai multe cartele SIM, eventual şi în format scris, pe hârtie). Dacă numărul respectiv trebuie modificat, atunci trebuie să se aibă în vedere toate locaţiile unde apare acel număr sau, la o dată ulterioară, nu veţi mai şti care este numărul corect sau veţi apela numărul care nu mai este valabil. În acest caz este vorba de o eroare de actualizare şi de o inconsistenţă a informaţiilor. Prin normalizarea BD, se elimină aceste probleme, scopul normalizării fiind acela de a ne asigura că datele sunt stocate fiecare numai într-un singur loc şi anume în cel mai bun loc din BD.

Erorile de actualizare se clasifică în trei categorii:

- **erori de inserare** de noi date sau înregistrări cauzate de incoerenţa unor relaţii din schema relaţională. De exemplu, dacă se defineşte o relaţie (tabel) *personal_departament*, în care sunt incluşi toţi membrii de personal din agenţie şi fiecare departament apare de mai multe ori în tabel, cu toate atributele sale, atunci pe rândurile corespunzătoare reprezentanţilor săi, se produce o dublare a datelor referitoare la departamente (denumiri, adrese, numere de telefon). În cazul în care se doreşte înfiinţarea unui nou departament care iniţial nu are angajaţi, este necesară introducerea de null-uri în cheia primară a relaţiei ceea ce nu este admis. Dacă se angajează un nou membru de personal trebuie să introducem corect datele referitoare la departamentul în care va fi repartizat, aşa cum apar ele şi în celelalte înregistrări ale personalului aferent aceluiaşi departament. Reprezentarea distinctă a tipurilor de entităţi în două relaţii *personal* şi *departament* elimină această dublare a datelor şi anomaliile de inserare. Fiecare membru de personal este asociat numai cu numărul de identificare al departamentului, atributele acestuia fiind incluse în alt tabel.
- **erori de modificare** sunt cauzate de dublarea datelor în tabele. De exemplu, modificarea unui atribut al unui departament, impune schimbarea valorii datelor din mai multe rânduri. Dacă nu se reactualizează toate înregistrările corespunzătoare, se vor genera rapoarte incorecte, cu aşa-numite **date depreciate**.
- **erori de ştergere** determină pierderea unor date din BD. De exemplu, dacă în relaţia *personal_departament* se elimină singurul membru al unui departament, se şterg şi datele referitoare la acest departament.

Pentru a evita erorile de actualizare este necesară descompunerea unor relaţii în mai multe relaţii, cu seturi parţiale din atributele iniţiale, astfel încât să nu mai existe date dublate în nici o relaţie.

Normalizarea se realizează prin analiza şi testarea relaţiilor, pe baza cheilor primare şi a celor candidat, şi aplicarea anumitor reguli de normalizare.

Procesul de normalizare se realizează în mai multe etape, numite **forme de normalizare:**

- **Prima formă de normalizare** (*First Normalized Form – 1NF*) are ca scop eliminarea atributelor cu valori multiple şi se obţine atunci când, la intersecţia fiecărei linii cu fiecare coloană din orice relaţie a schemei BD, apare numai **o singură valoare**.

  De exemplu, să presupunem că într-un tabel din BD se scriu disciplinele la care se ţin ore într-o anumită sală. Firește că pentru un anumit amfiteatru sau un laborator

multidisciplinar, atributul *disciplină* va avea valori multiple. Entitatea nu este în prima formă de normalizare (figura III.8).

Trecerea unei entități în 1NF se face prin definirea unei noi entități corespunzătoare atributului cu valori multiple, eliminarea atributului respectiv şi crearea unei relații 1:M cu vechea entitate.

Exercițiu propus: Analizați entitățile următoare:
GRUPĂ (cod_grupă, număr_grupă, an_de_studiu, student)
MEDIC (id_medic, nume, prenume, cod_parafă, pacient)
PROPRIETATE (id proprietate, tip, suprafață, preț, vizitator)
Care dintre ele nu este în 1NF? Convertiți-le la 1NF, dacă este cazul.

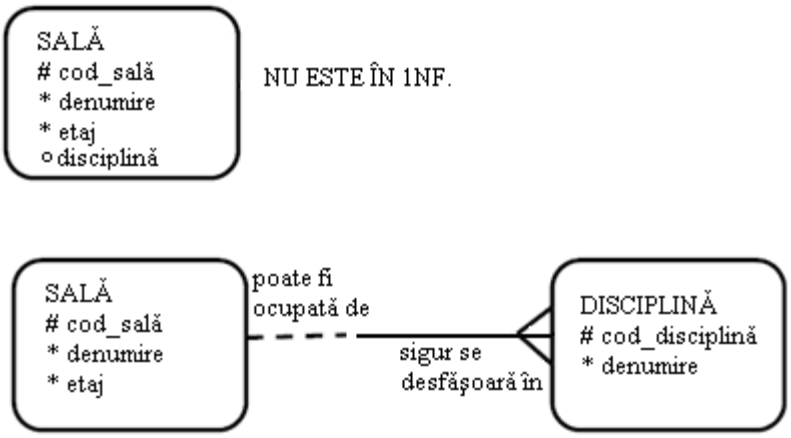

Fig. III. 8 Trecerea în prima formă de normalizare

- **A doua formă normalizată** (*Second Normalized Form - 2NF*) se aplică relațiilor aflate în 1NF, care au chei compuse, şi constă în faptul că fiecare atribut care nu aparține cheii primare este total dependent funcțional de aceasta.

  Dependența funcțională *dintre atribute semnifică faptul că unul sau mai multe atribute sunt determinate în mod unic de un alt atribut sau set de atribute, care constituie* ***determinantul***.

  De exemplu, toate atributele unei relații sunt dependente funcțional de cheia primară, cu excepția atributelor care o formează.
  Dependența funcțională totală apare atunci când atributul dependent nu depinde de un subset de atribute al unei chei şi eliminarea oricărei componente din determinant

conduce la dispariţia dependenţei. În caz contrar, se consideră că este o **dependenţă funcţională parţială**.

Normalizarea în a doua formă (2NF) se face prin identificarea şi eliminarea tuturor dependenţelor funcţionale parţiale de cheia primară sau de cheile candidat.

O relaţie cu cheie primară singulară (cu un singur atribut) este automat în 2NF.
Pentru o relaţie cu cheie primară compusă, trecerea în 2NF se face prin eliminarea dependenţelor funcţionale parţiale. Pentru aceasta, fiecare atribut dependent funcţional parţial este copiat într-un nou tabel, împreună cu o copie a determinantului.
De exemplu, în BD a unei companii telefonice, sunt incluse entităţile *ABONAT* şi *JUDEŢ*. Pentru apelare, trebuie combinat numărul abonatului cu prefixul de judeţ, pentru că acelaşi număr de telefon poate să apară la mai mulţi abonaţi, din judeţe diferite. În figura III.9, este reprezentată barat relaţia dintre cele două entităţi.

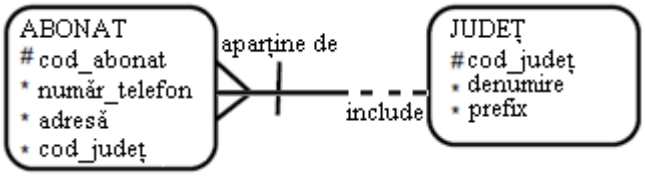

Figura III.9 Relaţie barată între două entităţi

Dacă s-ar folosi o singură entitate cu cheia primară compusă (*cod_abonat, cod_judeţ*) şi atributele specifice judeţului (*denumire, prefix*) ar fi înscrise toate în entitatea *ABONAT*, atunci ar exista o dublare a datelor iar la schimbarea prefixelor de judeţ, ar trebui făcute modificări identice pe foarte multe linii din tabel. Într-o astfel de reprezentare, atributele de judeţ depind parţial de cheia primară prin atributul *cod_judeţ*. Separarea entităţilor şi reprezentarea din figura de mai sus, rezolvă această problemă şi trece BD în forma a doua de mormalizare.
Să considerăm un alt exemplu. În modelul de date al unei agenţii imobiliare, într-o relaţie *clienţi_proprietăţi* care are ca UID, cheia primară compusă (*cod_client, cod_proprietate*), mai multe atribute (*nume_client, telefon, adresă)* depind parţial de cheia primară deoarece depind numai de atributul *cod_client*, dar nu şi de atributul *cod_proprietate*. Dacă un client, care oferă spre închiriere mai multe proprietăţi, îşi

schimbă numărul de telefon sau adresa de contact, atunci trebuie modificată valoarea din mai multe înregistrări din tabel şi este foarte posibil să se omită unele linii. Deoarece în tabel apar valori repetitive, redundante, există riscul apariţiei unor erori de actualizare. De aceea este necesară trecerea în forma a doua de normalizare care asigură absenţa valorilor repetitive din tabelele bazei de date. În exemplul dat, pentru normalizarea în 2NF, se defineşte o nouă relaţie *clienţi* în care se includ toate atributele clienţilor împreună cu determinantul *cod_client* ca şi cheie primară, iar în relaţia iniţială *clienţi_proprietăţi* atributul *cod_client* rămâne ca atribut simplu, cu rol de cheie străină, care nu mai face parte din cheia primară a acestei relaţii.

Exerciţiu propus: Analizaţi şi treceţi în 2NF următoarea entitate, din BD a unui magazin:
FURNIZORI_PRODUSE (cod_furnizor, cod_produs, denumire_furnizor, localitate, judeţ, persoană_de_contact, telefon, denumire_produs, categorie, preţ_de_achiziţie)
Reprezentaţi grafic diagrama parţială entitate – relaţie obţinută.

- **A treia formă normalizată** (*Third Normalized Form - 3NF*) se obţine prin eliminarea aşa-numitelor *dependenţe tranzitive* dintr-un model aflat în 1NF şi 2NF. Dar mai întâi, ce semnifică dependenţa tranzitivă?

  *Prin definiţie, atunci când un atribut A determină atributul B, iar B determină atributul C, apare **dependenţa tranzitivă** a atributului C de atributul determinant A, prin intermediul atributului B (Figura III.10).*

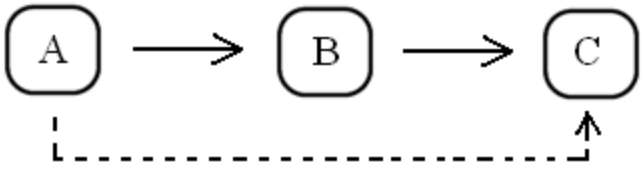

Figura III.10 Dependenţa tranzitivă a lui C de A

Încălcarea formei a treia de normalizare apare atunci când un atribut din fara cheii primare (atribut non-UID) depinde de un alt atribut non-UID.
De exemplu, dacă BD a unui cabinet medical include în aceeaşi entitate şi în acelaşi tabel *pacienţi* toate datele despre pacienţi, inclusiv datele din fişele lor medicale, a

celor despre consultaţiile făcute şi despre asistentele medicale care asistă la consultaţii sau realizează tratamente, atunci există sigur o dependenţă tranzitivă între atributul cheie primară *cod_pacient* (A), cel de identificare a medicului *cod_medic* (B) unic determinat de A şi *nume_medic* (C), determinat de B. Dacă se schimbă numele unuia dintre medici, atunci foarte multe înregistrări trebuie modificate în BD.
Dependenţele funcţionale tranzitive dintre atributele unei relaţii pot cauza erori de actualizare.

Exerciţiu propus: Identificaţi alte posibile dependenţe tranzitive în acest model.

O relaţie aflată în 1NF şi 2NF se găseşte şi în 3NF dacă nici un atribut din afara cheii primare nu este dependent tranzitiv de cheia primară, prin intermediul altui atribut care nu intră în componenţa cheii primare. Altfel spus, nici un atribut non-UID nu poate avea propriile sale atribute deoarece el devine determinantul acestora şi creează dependenţe tranzitive în afara cheii primare.

Pentru obţinerea formei a treia de normalizare (3NF), se identifică eventualele dependenţe tranzitive şi orice atribut cu dependenţă tranzitivă se plasează într-o nouă relaţie, adică un nou tabel, împreună cu copia determinantului (determinanţilor).

Exerciţiu propus: Rezolvaţi dependenţele tranzitive din exemplul anterior.

- **forma normală Boyce-Codd (BCNF)** elimină dintr-o relaţie **dependenţele tranzitive faţă de cheile candidat** şi se obţine după crearea formei 3NF, atunci când toţi determinanţii sunt chei candidat diferite de cheia primară. Acesta este un aspect mai complex şi mai greu de observat.
  Pentru o relaţie cu o singură cheie candidat, care implicit este şi cheie primară, formele 3NF şi BCNF sunt echivalente.
  Dacă o entitate are în afara cheii primare, una sau mai multe chei candidat, atunci, pentru a obţine forma Boyce-Codd, trebuie căutate şi eliminate eventualele dependenţe tranzitive faţă de acestea.
  De exemplu, într-o relaţie *VIZITARE_PROPRIETĂŢI* sunt înregistrate vizitele pe care le fac, la diferite proprietăţi, clienţii unei agenţii imobiliare, în vederea închirierii. Ca

regulă, agentul prezintă proprietatea numai unui singur client la un anumit moment. Cheia primară este compusă (*cod_proprietate, cod_client*). În relaţie, apar mai multe dependenţe funcţionale care sunt admise de 3NF, dacă determinantul lor este cheie candidat. Atributele *data_vizitei, ora_vizitei, cod_agent, nr_înmatriculare, marca_auto, nr_locuri, comentarii* sunt determinate funcţional total de cheia primară, deci relaţia respectă regula 2NF. Dar atributele *marca_auto, nr_locuri* depind de atributul *nr_înmatriculare,* dependent funcţional de setul de atribute (*data_vizitei, ora_vizitei, cod_agent*) care constituie o cheie candidat. Prin urmare, există o dependenţă tranzitivă de cheia candidat. Relaţia este în 3NF, nu şi în BCNF. Pentru a trece la forma BCNF trebuie eliminată dependenţa tranzitivă de cheia candidat şi, în acest scop, este necesară crearea unei noi relaţii *AUTOVEHICULE*, cu atributele specifice (*nr_înmatriculare, marca_auto, nr_locuri*) şi se elimină atributele dependente tranzitiv de cheia candidat din tabelul iniţial *VIZITARE_PROPRIETĂŢI.*

Pentru a sistematiza procesul de analiză a dependenţelor tranzitive, este util să se realizeze o matrice cu toate atributele entităţii analizate, scrise atât pe linii, cât şi pe coloane (asemenea matricii relaţiilor) şi să se marcheze dependenţele dintre ele. Pe aceasta, o vom numi **matricea dependenţelor.** Această matrice este utilă şi la normalizarea în 2NF, precum şi în 3NF şi în BCNF.

Prin analiza tuturor dependenţelor din relaţie, se stabileşte dacă aceasta este sau nu în forma de normalizare dorită.

Exerciţiu propus: Identificaţi dependenţele tranzitive din tabelul „pacienţi", din exemplul BD a unui cabinet medical, folosind matricea dependenţelor. Treceţi tabelul în forma BCNF.

- **A patra formă normalizată (4NF)** se obţine din forma BCNF prin eliminarea **dependenţelor funcţionale multivalorice (MVD) netriviale** (nu determină integral tabelul), care apar în procesul de generare a primei forme de normalizare din cauza relaţiilor de tip 1:M independente dintre tipurile de entităţi. De exemplu, există astfel de relaţii 1:M între entităţile *DEPARTAMENT* şi *PERSONAL* respectiv între *DEPARTAMENT* şi *PROPRIETĂŢI* (Figura III.5). Într-o relaţie *DEPARTAMENT_PERSONAL_PROPRIETĂŢI* există o singură cheie candidat (*cod_departament, cod_agent, cod_proprietate*) deci relaţia este în forma BCNF.

Pentru a evita apariţia atributelor cu valori multiple, pentru acelaşi departament se combină în mai multe rânduri numele agenţilor cu atributele proprietăţilor pe care le gestionează. Deci numele unui agent poate corespunde la mai multe proprietăţi iar combinaţia (*cod_departament, cod_agent*) apare redundant, de mai multe ori. Dacă un agent se mută de la un deparetament la altul, atunci trebuie făcute reactualizări în mai multe linii de tabel. Pentru eliminarea acestei dependenţe multivalorice, se descompune relaţia iniţială în două relaţii, de exemplu, *DEPARTAMENT_PERSONAL* şi *DEPARTAMENT_ PROPRIETĂŢI*.

- **A cincea formă normalizată (5NF)** se obţine din 4NF prin descompunerea unei relaţii în două sau mai multe relaţii independente care elimină **dependenţele de tip uniune fără pierderi** şi care garantează faptul că prin uniunea naturală a mai multor tabele rezultate din descompunerea altuia nu se generează date sau corespondenţe false. De obicei, raportările din BD se fac prin reunirea datelor din mai multe tabele.
  Între acestea trebuie stabilite clar relaţiile şi atributele de legătură (cheile străine), astfel încât selecţia valorilor să se realizeze corect.
  De exemplu, dacă BD a unei librării, conţine tabelele *EDITURI, CĂRŢI, FURNIZORI,* fără a se face relaţiile corecte dintre acestea, prin atributele *cod_editură, cod_furnizor,* cu referinţe între cele trei tabele, atunci o interogare prin care se reunesc informaţiile din tabelele *EDITURI, FURNIZORI* prin care se caută lista furnizorilor unei edituri, va genera toate combinaţiile posibile dintre edituri şi furnizori.

Observaţie: De regulă, se realizează normalizarea modelului BD până la forma BCNF, fiind relativ dificile regulile pentru 4NF şi 5NF. Totuşi, dacă la testarea modelului apar erori, este bine să se aplice şi aceste forme de normalizare.

III.8 REZUMATUL CAPITOLULUI

Alegerea entităţilor, a atributelor şi a relaţiilor dintre entităţi, pe care le luăm considerare în modelul de date ER al unei aplicaţii de BD, necesită o bună documentare, o comunicare permanentă cu beneficiarii aplicaţiei precum şi mai multe etape de normalizare, pentru a răspunde tuturor cerinţelor de interogare ale clienţilor precum şi pentru a evita erorile de actualizare care pot să apară în faza de utilizare a BD. Respectarea constrângerilor aplicaţiei (de opţionalitate, cardinalitate, referenţială şi de participare) precum şi a regulilor de

normalizare, uzual până în forma Boyce-Codd, asigură performaţele modelului bazei de date. Reprezentarea grafică a modelului ER sub forma diagramei ER oferă posibilitatea analizei rapide a modelului. Matricea relaţiilor reprezintă, de asemenea, un mod eficient de reprezentare şi identificare a relaţiilor dintre entităţi în vederea completării modelului conceptual al BD.

III.9 TERMENI SPECIFICI

Modelul Entitate-Relaţie (ER)
Identificator unic (UID)
UID primar
UID secundar
UID simplu
UID compus
Diagrama ER
Reprezentare „în stea”
Reprezentare „în dreptunghi”
Regula de opţionalitate
Regula de cardinalitate
Regula de orientare
Capcane de conectare
Matricea relaţiilor
Normalizare
Forma nenormalizată (UNF)
Prima formă de normalizare (1NF)
Valori multiple
A doua formă de normalizare (2NF)
Dependenţă funcţională
A treia formă de normalizare (3NF)
Dependenţă tranzitivă
Forma de normalizare Boyce-Codd (BCNF)
A patra formă de normalizare (4NF)
A cincea formă de normalizare (5NF)

III.10 TEST - GRILĂ

În BD a unei farmacii, se aleg entităţile *CLIENT, PRODUS, PRODUCĂTOR, FURNIZOR, COMANDĂ.*

1. Care dintre următoarele relaţii este de tip 1:M?
- ❑ *client - produs*
- ❑ *comandă - furnizor*
- ❑ *produs – furnizor*
- ❑ *produs - producător*

2. Care dintre atributele entităţii *PRODUS* poate avea valori multiple?
- ❑ *cod_produs*
- ❑ *denumire*
- ❑ *preţ de achiziţie*
- ❑ *producător*

3. Care dintre atributele entităţii *PRODUS* este artificial?
- ❑ *cod_produs*
- ❑ *denumire*
- ❑ *preţ de achiziţie*
- ❑ *producător*

4. Care dintre următoarele relaţii este opţională?
- ❑ Un *CLIENT* cumpără unul sau mai multe *PRODUSE*.
- ❑ Un *FURNIZOR* primeşte una sau mai multe *COMENZI*.
- ❑ Un *PRODUS* este furnizat de unul sau mai mulţi *FURNIZORI*.
- ❑ Un *PRODUS* este înscris pe una sau mai multe *COMENZI*.

5. În tabelul *LISTA_PRODUSELOR* se includ atributele *cod_produs, denumire_produs, cod_producător, cod_furnizor, preţ_achiziţie, preţ_vânzare*. În ce formă de normalizare este tabelul?
- ❑ 1NF
- ❑ 2NF
- ❑ 3NF
- ❑ BCNF

CAPITOLUL IV

ASPECTE PARTICULARE ALE MODELĂRII DATELOR

DIN CUPRINS:

IV.1 RELAŢII REDUNDANTE

Modelul ER al unei BD include mai multe entităţi, cu atributele proprii, şi relaţiile dintre acestea.
Relaţiile dintre entităţi se analizează după criteriile de opţionalitate (sigură sau posibilă) şi de cardinalitate (1:1, 1:M, M:M).
De exemplu, BD a unui magazin cuprinde date despre clienţi, angajaţi şi produse:

clienţi *(cod_client, nume, prenume, adresă, telefon, cnp, adresă_de_email)*
angajaţi *(id_angajat, nume, prenume, funcţie, salariu, data de naştere, cnp, adresa, telefon)*
produse *(cod_produs, denumire, categorie, preţ de vânzare, furnizor).*

Între aceste trei entităţi, se stabilesc mai multe relaţii pe care le putem descrie astfel:

1. *„Un client sigur este deservit de un angajat”* şi *„un angajat poate să deservească unul sau mai mulţi clienţi”.*
2. *„Un angajat poate să vîndă unul sau mai multe produse”* şi *„un produs poate fi vândut de unul sau mai mulţi angajaţi”.*
3. *„Un produs poate fi comandat de unul sau mai mulţi clienţi”* şi *„un client cumpără unul sau mai multe produse”.*
4. *„Un angajat este sigur condus de un manager”* şi *„un manager sigur are în subordine unul sau mai mulţi angajaţi”.*

Relaţia (1) este de tip 1:M între două entităţi distincte.
Relaţia (4) este de tip 1:M, dar este o relaţie „în buclă”, de la entitatea *ANGAJAT* la ea însăşi.
Relaţiile (2) şi (3) sunt de tip M:M.

Exerciţiu propus: Pentru BD descrisă în modelul de mai sus, reprezentaţi diagrama ER, folosind convenţia „în dreptunghi” şi punând în evidenţă opţionalitatea şi cardinalitatea fiecărei relaţii.

Se observă în diagrama ER crearea unui triunghi, prin relaţionarea celor trei entităţi. Putem afirma în acest caz că avem **relaţii redundante**, în sensul că una din cele trei relaţii din triunghi poate fi înlocuită cu celelalte două. De exemplu, un angajat deserveşte unul sau mai mulţi clienţi (relaţia 1) care cumpără unul sau mai multe produse (relaţia 3), prin urmare din aceste două relaţii ştim care sunt produsele vândute de un angajat (relaţia 2). Nu este nevoie să folosim toate cele trei relaţii dintre entităţi, ci numai două. Problema care apare se referă la modul în care alegem cele două relaţii pe care le păstrăm în diagramă şi pe care o eliminăm.
În cazul în care analizăm relaţiile dintre entităţi, luăm în considerare următoarele **reguli**:
R1. Nu se admit relaţii M:M deoarece cresc riscul erorilor de actualizare a datelor din BD.
R2. Nu se admit relaţii 1:M care creează capcane „în evantai".
R3. Se păstrează cu prioritate, relaţiile de tip 1:1.
R4. Se pot păstra unele relaţii redundante dacă prin aceasta se elimină riscul apariţiei unei capcane de întrerupere.
În baza acestor reguli, în exemplul BD a unui magazin, una dintre relaţiile (2) şi (3) trebuie eliminată iar cealaltă care este tot de tip M:M tebuie rezolvată. De exemplu, eliminăm în prima fază relaţia (2), dintre entităţile *ANGAJAT* şi *PRODUS*, care este redundantă şi de tip M:M, deoarece ea poate fi înlocuită de celelalte două relaţii (1) şi (3).
Rămâne să rezolvăm relaţia M:M, în modul prezentat în paragraful următor.

Exerciţiu propus: Pentru BD a unei facultăţi, în care se consideră entităţile STUDENT, PROFESOR, DISCIPLINĂ, reprezentaţi diagrama ER, folosind convenţia „în dreptunghi", cu toate relaţiile posibile dintre entităţi. Observaţi apariţia unor relaţii redundante? Dacă da, decideţi care dintre relaţii trebuie eliminate şi care trebuie păstrate.

IV.2 REZOLVAREA RELAŢIILOR M:M

Relaţiile „mulţi-la-mulţi" (M:M) sunt des întâlnite în primele etape ale modelării datelor. Practic, o astfel de relaţie se traduce prin apariţia aceleiaşi valori pe mai multe linii dintr-un tabel, ceea ce ar crea probleme la reactualizarea datelor. De exemplu, un produs este comandat de mai mulţi clienţi, prin urmare în tabelul *PRODUSE* pe mai multe linii apare acelaşi produs, cu câte un alt client cumpărător specificat pe fiecare linie (pentru a nu avea

valori multiple în câmpul *client*), şi în situaţia modificării codului produsului, este nevoie să se facă actualizarea codului în toate liniile respective.

Este necesar să se înloicuiască relaţiile M:M cu relaţii 1:1 sau 1:M. **Modelul ER final nu trebuie să conţină relaţii M:M**.

Rezolvarea unei relaţii M:M dintre două entităţi se face prin intersectarea mulţimilor atributelor lor şi definirea unei noi entităţi **de intersecţie**, cu atributele rezultate din operaţia de intersecţie.

În exemplul anterior, se intersectează entitatea *CLIENT* cu entitatea *PRODUS* şi denumim entitatea nou rezultată *COMANDĂ*. Este evident faptul că un client lansează una sau mai multe comenzi, dar o comandă este asociată unui singur client, deci între cele două entităţi apare o relaţie 1:M. De asemenea, o comandă include un singur produs, dar un produs apare în una sau mai multe comenzi şi din nou rezultă o relaţie 1:M (a nu se confunda „comanda" unui produs, lansată prin butonul „CUMPĂRĂ" asociat produsului, cu „coşul de cumpărături" al clientului, în care se înscriu toate comenzile făcute de client la o vizitare a site-ului magazinului online). Entitatea *COMANDĂ* va avea atributele rezultate din intersecţia entităţilor (*cod_client, cod_produs*) precum şi un atribut propriu cu rol de UID, *cod_comandă*. Se poate folosi şi o cheie primară compusă din cele două coduri de client şi de produs, ceea ce ar crea aşa-numitele relaţii barate dintre entitatea de intersecţie şi entităţile iniţiale. Pe lângă acestea se vor folosi şi alte atribute (*număr_bucăţi, data_lansării_comenzii, data_livrării_produsului*). Astfel, relaţia M:M se înlocuieşte cu două relaţii 1:M, fără a apărea o capcană „în evantai".

Exerciţiu propus: Rezolvaţi relaţia M:M dintre entităţile STUDENT şi DISCIPLINĂ, din BD a unei facultăţi. Reprezentaţi diagrama ER modificată.

De obicei, entitatea de intersecţie are un atribut opţional de stare, care exprimă o etapă a unui proces, identificabilă în mod unic.

De exemplu, o agenţie de turism organizează mai multe excursii, cu mai mulţi prestatori de servicii (de cazare, de transport local, de transport internaţional etc.) astfel încât relaţia dintre excursii şi prestatorii de servicii este evident de tip M:M. Dar pentru o anumită destinaţie şi la o anumită dată, participarea celor două entităţi la intersecţie este singulară, adică un singur transportator internaţional deserveşte grupul de excursionişti, cazarea unui turist, într-o

noapte se face la un singur hotel etc. Intersecţia respectivă devine un eveniment clar, identificat unic prin dată, timp, locaţie.

De multe ori entitatea de intersecţie are un caracter artificial, cum ar fi un contract de prestări servicii, dar rolul ei este clar în rezolvarea unei relaţii M:M tocmai prin participarea singulară a cel puţin unei entităţi în fiecare din relaţiile nou-create. Relaţia M:M este înlocuită cu relaţii 1:1 sau cel mult 1:M.

IV.3 RELAŢII IERARHICE. RELAŢII RECURSIVE

De multe ori, avem de a face cu ierarhii de persoane sau de instituţii, în funcţie de responsabilităţile acestora sau de modul de subordonare dintre ele. Se pot stabili astfel ierarhii de entităţi, prin care se realizează o schemă mai clară a BD.

Un bun exemplu, este acela al localizării camerelor din căminele dintr-un campus studenţesc, prin numărul de cămin, prin etaj şi numărul camerei. Aceasta este o ierarhie pe trei nivele (figura IV.1).

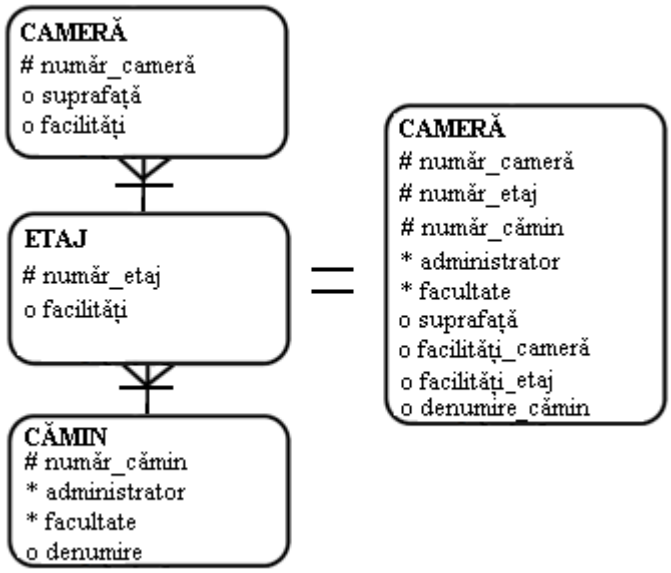

Figura IV.1 Ierarhizarea entităţilor

Se observă că identificarea unei camere din campus, nu se face doar prin numărul camerei, ci prin toate cele trei chei primare: *număr_cămin, număr_etaj, număr_cameră*, ceea ce se reprezintă prin **relaţii barate** între entităţile din ierarhie.

Relaţia este barată la capătul corespunzător entităţii determinate. De exemplu, fiecare cameră a unui hotel poate fi identificată doar dacă se ştie pe ce etaj se află. Entitatea *CAMERĂ* este determinată de entitatea *ETAJ*. Relaţia dintre ele este barată la capătul dinspre entitatea *CAMERĂ*.

Bineînţeles că se poate defini o singură entitate cu toate atributele entităţilor din ierarhie dar este foarte posibil ca aceasta să nu respecte formele de normalizare a BD (apar de exemplu, dependenţe funcţionale parţiale de cheia primară, care se rezolvă tot prin definirea mai multor entităţi normalizând entitatea respectivă).

Folosirea relaţiilor ierarhice este recomandată în toate cazurile în care se modelează anumite ierarhii ale instanţelor unei entităţi.

Exerciţiu propus: Reprezentaţi ierarhic, ca diagramă ER, pe ani de studiu, grupele de studenţi ale facultăţilor din cadrul unei universităţi.

Recursivitatea relaţiilor apare în multe cazuri în care entităţile sunt modelate ierarhic, îndeosebi atunci când entităţile în cauză reprezintă persoane.

De exemplu, în ierarhia militară, gradele militare se subordonează unele altora, până la cel mai înalt grad. Reprezentarea în BD a persoanelor din cadrul unei unităţi militare poate fi făcută fie cu mai multe entităţi ierarhizate, fie printr-o singură entitate cu o relaţie în buclă, numită **relaţie recursivă.**

Relaţia recursivă *apare ca relaţie „în buclă”, de la o entitate la ea însăşi, şi reprezintă o subordonare a tuturor instanţelor entităţii faţă de una dintre instanţele ei.*

De exemplu, dacă printre atributele entităţii *CADRU_MILITAR* apare atributul *comandant,* atunci un membru al personalului unităţii are această calitate de comandat şi, în acelaşi timp, este şi cadru militar, prin urmare apare în lista respectivă ca instanţă a entităţii, alături de subordonaţii săi (Figura IV.2).

Tabelul asociat entităţii din figura IV.2, cu relaţie recursivă, este mai greu de urmărit decât dacă s-ar folosi o reprezentare ierarhică. Adică, un soldat este subordonat comandantului de pluton, acesta la rândul său este subordonat comandantului de detaşament şi aşa mai departe. Urmărirea ierarhiei, pe mai multe nivele, implică mai multe sortări ale datelor până să se stabilească cel mai înalt grad din ierarhia respectivă.

De asemenea, dacă se modifică persoana cu rang de comandant dintr-o unitate, atunci este necesară modificarea numelui său în mai multe înregistrări din tabel ceea ce poate să conducă la erori de actualizare. De aceea, este preferată reprezentarea ierarhică, în care orice dată este stocată într-o singură locaţie din BD.

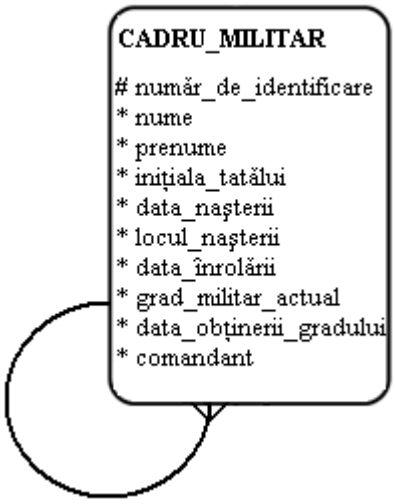

Figura IV.2 Reprezentarea unei relaţii recursive

Reprezentarea ierarhică a entităţilor corespunde unei structuri fixe, iar completarea ei se face cu diferite persoane, care se pot schimba la un anumit moment, fără ca aceasta să afecteze schema în sine (Figura IV.3).

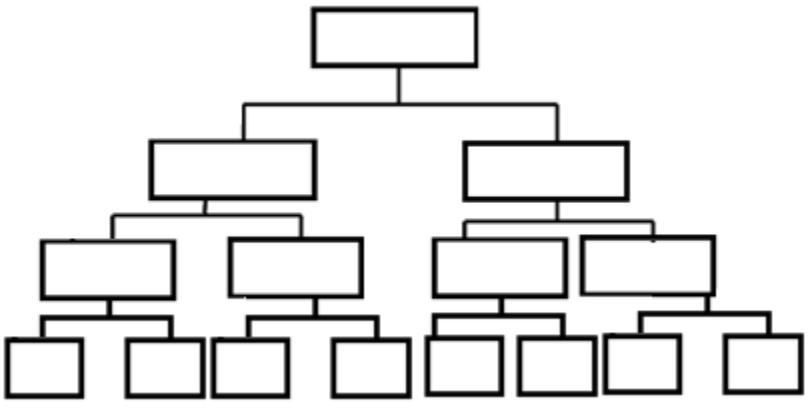

Figura IV.3 Structură ierarhică, cu 4 nivele

Exerciţiu propus: Reprezentaţi ierarhic, ca diagramă ER, pe ani de studiu şi pe grupe, studenţii facultăţilor din cadrul unei universităţi.

IV.4 TRANSFERABILITATEA RELAŢIILOR

Transferabilitatea este o noţiune care exprimă posibilitatea schimbării unor asocieri dintre instanţele a două entităţi relaţionate între ele.
De exemplu, în BD a unui cabinet medical, un pacient este asociat cu un anumit medic de familie. Întrebarea care se pune este aceea dacă pacientul poate să îşi schimbe medicul de familie? Răspunsul poate fi şi da, şi nu, acesta depinzând numai de regulile de afaceri ale cabinetului respectiv. Un răspuns afirmativ reflectă transferabilitatea relaţiei dintre medic şi pacient, permisă de politica aplicată în acel cabinet.

Reţineţi că analiza unei relaţii din modelul ER se face referitor nu numai la opţionalitatea şi cardinalitatea relaţiei, ci şi la transferabilitatea acesteia.

În multe aplicaţii este necesar să se impună netransferabilitatea unor relaţii. De exemplu, relaţia dintre entitatea *CAMERĂ* şi entitatea *CĂMIN* din BD a unui campus universitar, este netransferabilă (Figura IV.4).

Relaţiile netransferabile se marchează cu un romb în reprezentarea grafică.

Modelul de date creat pentru o aplicaţie de BD conţine de foarte multe ori atribute şi relaţii care nu se pot schimba şi pe care le numim **netransferabile.**
De exemplu, ziua, luna şi anul naşterii unei persoane sunt atribute cu caracter permanent, imuabil.
Aceste aspecte particulare trebuie să fie clarificate prin regulile de afaceri, în documentaţia bazei de date, şi să fie asigurată aplicarea lor în momentul implementării BD în limbaj de programare. Neaplicarea caracterului imuabil al unor atribute sau relaţii poate să creeze condiţii favorabile fraudelor informatice, prin falsificarea unor informaţii de identificare, în procesul de accesare a BD.

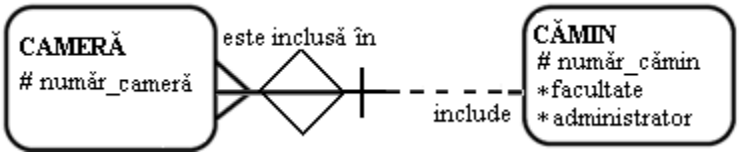

Figura IV.4 Reprezentarea unei relaţii netransferabile între două entităţi

Exerciţiu propus: Analizaţi şi reprezentaţi grafic relaţiile dintre entităţile ABSOLVENT, FACULTATE, SPECIALIZARE, DIPLOMĂ, din BD a unei universităţi (puneţi în evidenţă opţionalitatea, cardinalitatea şi transferabilitatea relaţiilor care se pot stabili între entităţi).

Exerciţiu propus: Daţi şi alte exemple de relaţii transferabile şi de relaţii netransferabile.

IV.5 SUBTIPURI ŞI SUPERTIPURI

În multe modele de date se foloseşte un număr foarte mare de entităţi, ceea ce face deosebit de greu de reprezentat şi de urmărit diagrama ER.
De exemplu, în BD a unui magazin de îmbrăcăminte, se folosesc entităţile *BLUZĂ, FUSTĂ, ROCHIE, CĂMAŞĂ, PANTALON, PALTON, JACHETĂ, COSTUM*, pe lângă celelalte *CLIENT, ANGAJAT, FURNIZOR, ACHIZIŢIE, VÂNZARE, PLATĂ*. Primele opt entităţi au fost definite separat deoarece fiecare reprezintă un articol de îmbrăcăminte cu atribute specifice. Totuşi acestea au în comun multe atribute (*cod, mărime, material, producător, culoare*) şi ar putea fi incluse toate într-un aşa-numit **supertip** sau **superentitate** *ARTICOL*. Astfel numărul de entităţi este simţitor redus. Pe lângă atributele comune, se disting şi atribute care le diferenţiază. De exemplu, o bluză poate avea mânecile scurte sau lungi, în timp ce la o fustă vom fi interesaţi de croiala acesteia iar la rochii ne vor interesa diferite modele (de mireasă, de seară, de plajă etc.) Spunem că entităţile componente ale entităţii supertip sunt **subtipuri** ale acesteia. Un subtip se mai numeşte şi **subentitate.**
În viaţa de zi cu zi, avem de multe ori de a face cu subtipuri, pe care le denumim **categorii**, şi modelarea lor pentru o BD ne obligă să le identificăm corect pe toate şi să le diferenţiem prin aspectele lor particulare.

Exerciţiu propus: Daţi trei exemple de entităţi supertip şi puneţi în evidenţă subtipurile acestora.

Subtipurile sau subentităţile sunt reprezentate grafic toate în acelaşi dreptunghi creat pentru supertip (Figura IV.5).

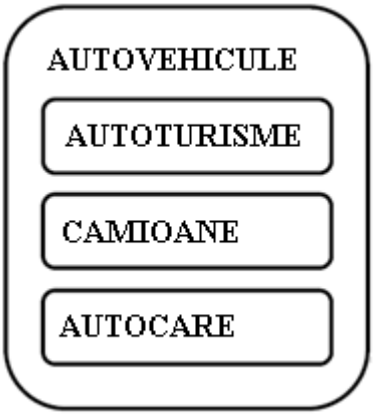

Figura IV.5 Reprezentarea subtipurilor şi a supertipului în aceeaşi casetă

Atributele comune subtipurilor se enumeră unul sub celălalt, sub numele supertipului, în timp ce atributele particulare, specifice fiecărui subtip se înscriu în caseta aferentă acestuia.
Astfel un subtip are toate atributele supertipului şi este implicat în toate relaţiile acestuia cu alte entităţi. Un subtip poate să aibă la rândul său alte subtipuri.

Exerciţiu propus: Completaţi diagrama din figura IV.5 cu atribute comune şi atribute specifice subtipurilor identificate în modelul BD al unui parc auto.

Verificarea corectitudinii alegerii subtipurilor se face în baza următoarelor reguli:

- Orice instanţă a entităţii supertip se încadrează într-un subtip (**regula de exhaustivitate**).
- Orice instanţă a unui subtip nu este instanţă a altui subtip (**regula de exclusivitate mutuală**).

Altfel spus, orice instanţă a supertipului este instanţă a unui şi numai a unui subtip.
Stabilirea subtipurilor se face pe baza regulilor afacerii. Este posibil, ca pe durata utilizării aplicaţiei de BD, să apară noi subtipuri care trebuie introduse în structura acesteia, adică este necesară dezvoltarea BD. Pentru a rezolva cât mai simplu această situaţie, este util ca de la început să se prevadă unul sau mai multe subtipuri suplimentare, cu sau fără un nume specific. Este foarte posibil ca politica firmei să prevadă de la început aceste posibilităţi de dezvoltare care trebuie luate în consideraţie la proiectarea BD sau să apară unele aspecte noi care trebuie incluse în model sub forma unui subtip *ALTELE* sau *DIVERSE*.
De exemplu, un magazin poate să accepte la un anumit moment, ca modalitate de plată, doar numerar şi bonuri valorice. După o perioadă de timp, pe măsură ce afacerea se dezvoltă şi se fac investiţii, acelaşi magazin va accepta şi plata cu carduri bancare. BD a magazinului nu

trebuie complet refăcută, ci numai regândite subtipurile entităţii *PLATĂ*. Este dificil de introdus un nou subtip dar este mai simplu de adaptat numele şi atributele subtipului generic *ALTELE* pentru a înregistra plăţile cu card.

Exerciţiu propus: Reprezentaţi diagrama ER, cu supertipuri şi subtipuri, pentru BD a unui magazin, în care includeţi toate entităţile (BLUZĂ, FUSTĂ, ROCHIE, CĂMAŞĂ, PANTALON, PALTON, JACHETĂ, COSTUM, CLIENT, ANGAJAT, FURNIZOR, ACHIZIŢIE, VÂNZARE, PLATĂ), cu atribute comune şi atribute specifice subtipurilor.

Constrângerea de exclusivitate mutuală se aplică uneori şi relaţiilor dintre entităţi, nu numai subtipurilor unei superentităţi, caz în care relaţiile respective sunt marcate cu un arc.

De exemplu, să considerăm BD pentru planificare activităţilor din sălile de sport ale unui club, care pot găzdui fiecare meciuri de baschet, handbal, fotbal sau concursuri de gimnastică. Bineînţeles că la un anumit moment, doar un singur tip de competiţie se desfăşoară într-o sală, ceea ce impune exprimarea **constrângerii de exclusivitate mutuală** (*exclusive OR*) asupra relaţiilor dintre entităţile *SALĂ* şi *MECI_DE_BASCHET, MECI_DE_HANDBAL, MECI_DE_FOTBAL, CONCURS_DE_GIMNASTICĂ* şi reprezentarea acestei constrângeri în diagrama ER se face cu un arc ce cuprinde toate relaţiile implicate (Figura IV.6).

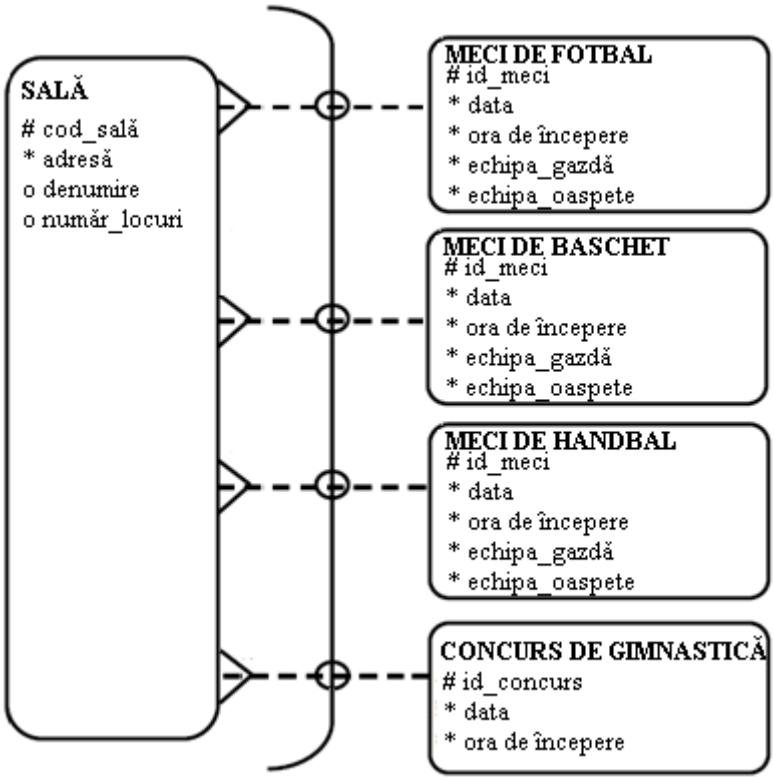

Figura IV.6 Reprezentarea unei constrângeri de tip *exclusive OR*

Trebuie făcută distincţia între regula de exclusivitate mutuală a subtipurilor unei entităţi şi constrângerea de exclusivitate mutuală aplicată unor entităţi distincte, care nu pot fi considerate de acelaşi tip.

Exerciţiu propus: Daţi un exemplu de o BD în care se aplică constrângerea de exclusivitate mutuală între mai multe entităţi şi reprezentaţi grafic diagrama ER a acesteia.

IV.6 MODELAREA ISTORICULUI UNUI ATRIBUT. MODELAREA TIMPULUI

În multe aplicaţii cu BD, valorile unor atribute se modifică în timp şi este nevoie să se păstreze istoricul acestora, pentru a se observa evoluţia lor, pentru a face anumite rapoarte şi comparaţii, precum şi pentru a lua anumite decizii.
De exemplu, managerul unui magazin doreşte să analizeze evoluţia pe o perioadă de un an a preţului unui produs. Dacă folosim entitatea *PRODUS* cu atributul *preţ_de_vânzare*, atunci înregistrarea din BD va reda la un anumit moment preţul curent al produsului, nu şi valorile anterioare. Acestea sunt şterse la fiecare actualizare a preţului. Pentru a păstra toate valorile preţului produsului, este nevoie să se definească o nouă entitate *ISTORICUL PREŢULUI* care să înregistreze evoluţia preţului, data de la care se aplică un anumit preţ şi data de la care acesta nu mai este valabil (Figura IV.7). Relaţia dintre cele două entităţi este 1:M pentru că aceluiaşi produs îi corespund în timp mai multe valori ale preţului. În plus, relaţia dintre cele două entităţi este barată pentru că identificatorul unic al preţului se compune din *id_produs* şi *data_aplicării* preţului deoarece la aceeaşi dată, se stabilesc preţurile la mai multe produse.

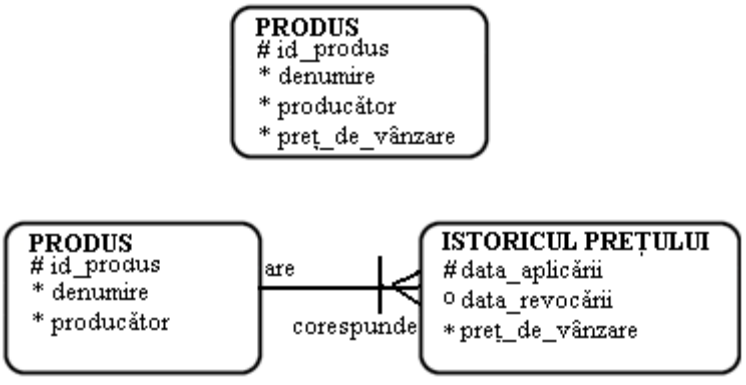

Figura IV.7 Modelarea istoricului unui atribut

Exerciţiu propus: Realizaţi modelul ER pentru BD a unei companii care să înregistreze evoluţia în timp a salariului fiecărui angajat.

Înregistrarea modificărilor suferite în timp de unele atribute ale unor entităţi din model, se poate face folosind atribute de tip dată – timp sau entităţi de timp separate.
În exemplul anterior în care se modela evoluţia preţului în timp, se folosesc atribute de timp.
De nenumărate ori dorim ca BD să păstreze şi înregistrările mai vechi, pentru a putea face o comparaţie între ele şi valorile curente, pentru a observa anumite tendinţe şi pentru a lua decizii optime de eficientizare a proceselor (de producţie, de vânzare, de tratament etc.)
Modelarea timpului ca entitate separata este utilă în multe aplicaţii de planificare a activităţilor. De exemplu, pentru programarea orelor de curs şi de laborator pe grupe de studenţi, se impun anumite **constrângeri sau condiţionări temporale**. O nouă activitate a unei grupe (curs, laborator, seminar) nu poate să înceapă înainte ca o altă activitate să nu se fi încheiat. Exprimăm această constrângere prin impunerea ca ora de început a unei noi activităţi să fie programată după ora de sfârşit a altei activităţi. Dacă la orar apare pentru fiecare activitate *ora_de_început* şi *ora_de_sfârşit*, atunci între aceste atribute apar anumite condiţionări. Realizarea orarului folosind o bază de date trebuie să se facă respectându-se unele constrângeri temporale. Chiar şi o sală în care se desfăşoară anumite activităţi didactice trebuie reprezentată cu aceste condiţionări temporale. Nu se pot planifica activităţi simultane de seminar, cu grupe disctincte, în aceeaşi sală. Adică ora de început a unui nou seminar trebuie să fie egală sau după ora de sfârşit a oricărui alt seminar, programat în aceeaşi sală. De asemenea, nu se pot face modificări în orar referitoare la o activitate, după ora de începere a acelei activităţi. Este un alt tip de condiţionare temporală a tranzacţiilor din BD. Spunem că între entităţile *SALĂ* şi *GRUPĂ* apare o **non-transferabilitate condiţionată.**

Transferabilitatea condiţionată *se referă la faptul că transferabilitatea între entităţi este posibilă la orice moment înainte de momentul de început al unei activităţi, după care nu mai este permisă.*

Reprezentarea entităţilor cu unele atribute de timp poate să cauzeze încălcări ale regulilor de normalizare a BD.

De exemplu, într-o sală de festivităţi, nu se pot programa simultan mai multe spectacole. Adică ora de început a unui nou spectacol trebuie să fie după ora de sfârşit a altui spectacol. Se descrie entitatea *SPECTACOL* prin atributele *cod_spectacol* (cheie primară), *tip, titlu, cod_echipă, data, ora_de_început, ora_de_sfârşit.* Atributele de timp depind de atributul *data*, sau altfel spus, atributul *data* are atribute proprii, deşi nu este cheie primară. Nu putem vorbi independent de ora 20, fără a specifica şi ziua la care facem referire. De aceea spunem că apare dependenţa de atributul *data.* Apare astfel o dependenţă tranzitivă care încalcă forma a treia de normalizare (3NF). Este necesară separarea atributului care determină dependenţa tranzitivă într-o entitate de sine stătătoare *DATĂ_TIMP,* cu atributele *id_dată_oră, data, ora_de_început, ora_de_sfârşit,* astfel încât să se rezolve condiţionarea temporală impusă de regulile de afaceri. Entitatea iniţială va apela numai la cheia primară a entităţii de timp nou create.

Într-un alt exemplu, să considerăm cazul modelării unei BD a unei facultăţi, în care entitatea *ACTIVITATE* are, printre altele, atributele *cod_activitate, număr_grupă, cod_profesor, cod_sală, cod_disciplină, zi, ora_de_început, ora_de_sfârşit.* Întrucât în aceeaşi sală nu se pot programa simultan mai multe activităţi, atributele de timp se condiţionează în funcţie de codul sălii, care la rândul său depinde de cheia primară, *cod_activitate.* De asemenea, atributele de oră nu sunt independente ci trebuie relaţionate cu atributul *zi* pentru a exprima corect intervalul în care se desfăşoară o activitate. Avem de a face aici cu o dependenţă tranzitivă, care încalcă regula formei a treia de normalizare a BD (3NF).

Întrebare: Care sunt condiţionările de timp din exemplul de mai sus? Cum se procedează pentru a trece entitatea în forma 3NF?

IV.7 MODELAREA GENERICĂ

În cazul în care regulile afacerii se schimbă foarte des, este relativ dificil de anticipat şi de modelat entităţile bazei de date. De exemplu, pentru BD a unui magazin de produse de anticariat este dificil de ştiut de la început toate tipurile de produse care se vor comercializa şi nici toate atributele specifice acestor categorii. În astfel de situaţii, cu evoluţii nepredictibile ale afacerii, se poate folosi cu succes un **model generic**.

Modelarea generică a BD este avantajoasă din mai multe motive:

- Este flexibilă, adică permite definirea de noi entităţi şi atribute, pe măsură ce specificul activităţii descrise se schimbă.
- Foloseşte un număr considerabil redus de entităţi, **generice**, cu atribute care pot fi definite pe parcurs.
- Se adaptează uşor şi rapid unor noi condiţii de lucru.

Să considerăm ca exemplu, BD a unui magazin de îmbrăcăminte, în care sunt definite entităţile *BLUZĂ, FUSTĂ, ROCHIE, CĂMAŞĂ, PANTALON, PALTON, JACHETĂ, COSTUM*, pe lângă celelalte specifice oricărui tip de magazin, *CLIENT, ANGAJAT, FURNIZOR, ACHIZIŢIE, VÂNZARE, PLATĂ*. Într-un paragraf anterior, vă arătam că este util să folosim subtipuri astfel încât reprezentarea diagramei ER să fie mai simplă. Dar ce se întâmplă, dacă în acelaşi magazin urmează să se aducă noi categorii de produse de îmbrăcăminte, de exemplu, articole sport, articole de încălţăminte şi accesorii? Trebuie să schimbăm BD pentru a introduce noi subtipuri ale entităţii *PRODUS?* Nu neapărat. Este suficient să avem în vedere o posibilă extindere a activităţii şi să modelăm generic datele de la început.

Modelarea generică *defineşte o entitate generică, cu atribute generice, care urmează să ia valori specifice pe măsură ce activitatea descrisă în BD se diversifică.*

În exemplul considerat, în ipoteza că magazinul îşi păstrează specificul de a comercializa doar articole de îmbrăcăminte, putem considera entitatea generică *PRODUS*, cu un set extins de atribute, în care să includem toate atributele posibile de la toate tipurile de articole de îmbrăcăminte (*id, denumire, mărime, lungime, culoare, material, gen, circumferinţă_talie, bust, lungime_ mânecă, dimensiune_gât, stil* şi altele), bineînţeles toate cu caracter opţional, astfel încât, pentru un anumit tip de produs, să completăm doar câmpurile specifice, iar cele care nu îl caracterizează să admită lipsa valorii (NULL). Se pot defini şi subtipuri ale entităţii generice, cu condiţia să putem anticipa care sunt tipurile pe care le vizăm pentru activitatea descrisă, pe durata de viabilitate a BD. Observăm că acest mod de abordare implică o creştere nedorită a numărului de atribute şi a dimensiunii tabelului asociat entităţii (12 coloane).
Totuşi modelarea generică impune o şi mai mare generalizare a noţiunilor. Mai exact, nu vom specifica denumirile atributelor, ci vom anticipa un număr maxim al acestora. În exemplul bazei de date pentru magazinul de îmbrăcăminte, putem folosim entitatea generică *TIP_PRODUS* cu şase atribute generice pe care le notăm *atribut_1, atribut_2, atribut_3,*

atribut_4, atribut_5, atribut_6. În tabelul asociat se trec denumirile atributelor, urmând ca valorile lor să fie specificate în alt tabel, corespunzător unei entităţi *VALOARE_ATRIBUT* (Figura IV.8). Acest mod de descriere „reciclează" atributele, reducând numărul lor şi menţinând dimensiuni rezonabile ale tabelelor din BD. Conform diagramei din figură, un articol va avea minimum patru atribute specificate şi maximum şase, alese dintr-un set extins de atribute. Tabelul cu tipurile de produse va avea şapte coloane în loc de douăsprezece (vezi, de exemplu, tabelul IV.1). Valorile atributelor pentru diferite tipuri de produse, sunt specificate în alt tabel, cu denumiri generice ale coloanelor (Tabelul IV.2).

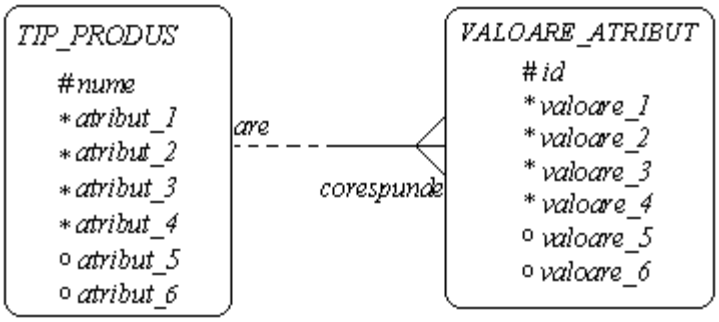

Figura IV.8 Modelarea generică a entităţii *PRODUS*

Tabelul IV.1 Definirea atributelor unor tipuri de produse

nume	*atribut_1*	*atribut_2*	*atribut_3*	*atribut_4*	*atribut_5*	*atribut_6*
FUSTĂ	*id*	*mărime*	*material*	*lungime*	*circumferinţă _talie*	
CĂMAŞĂ	*id*	*stil*	*material*	*culoare*	*lungime_ mânecă*	*dimensiune_ gât*
PANTALONI	*id*	*mărime*	*material*	*culoare*	*gen*	*lungime*

Tabelul IV.2 Valorile atributelor unor tipuri de produse

nume	*atribut_1*	*atribut_2*	*atribut_3*	*atribut_4*	*atribut_5*	*atribut_6*
FUSTĂ	*01*	*40*	*stofă*	*50*	*70*	
CĂMAŞĂ	*02*	*sport*	*elastan*	*negru*	*62*	*41*
PANTALONI	*03*	*38*	*jeans*	*gri*	*de damă*	*scurţi*

Modelarea generică permite folosirea aceleeaşi BD pe măsură ce se extinde activitatea magazinului, prin adăugarea altor game de produse.

Exerciţiu propus: Folosind modelul generic prezentat, adăugaţi în BD a magazinului, alte trei produse de îmbrăcăminte şi două produse de tip accesoriu. Completaţi tabelele cu valori specifice noilor categorii de produse.

Modelul generic prezentat stochează informaţiile despre produse într-un singur tabel, cu foarte multe înregistrări, greu de urmărit, şi este necesară sortarea pe mai multe nivele a datelor pentru separarea informaţiilor despre un anumit tip de produs.
De aceea, în unele cazuri se recurge la folosirea unor relaţii recursive care să permită stocarea informaţiilor despre diferite categorii de obiecte, în tabele diferite.
Pentru exemplul anterior, în care se modelează BD a unui magazin, în loc de două entităţi generice (*TIP_PRODUS, VALOARE_ATRIBUT*), se pot considera mai multe entităţi, cu atribute şi valori specifice:

- *TIP_PRODUS* (*# id_tip_produs, * nume*)
- *PRODUS* (*# id_produs, * id_tip_produs, * nume*)
- *ATRIBUT* (*# id_atribut, # id_tip_produs, * nume*)
- *VALOARE_ATRIBUT*(*# id_produs, # id_atribut, # id_tip_produs, * valoare*)

Relaţiile dintre aceste entităţi (*PRODUS* şi *VALOARE_ATRIBUT, TIP_PRODUS* şi *ATRIBUT*, respectiv *VALOARE_ATRIBUT* şi *ATRIBUT*) sunt barate, în sensul că fiecare instanţă a entităţii *ATRIBUT* se identifică prin *id_atribut* şi *id_tip_produs* iar valoarea atributului va fi identificată în mod unic prin combinaţia cheilor primare ale entităţii *PRODUS* şi *ATRIBUT* (*id_produs, id_atribut, id_tip_produs*). Altfel spus, atributul cu numărul 2, al tipului de produs FUSTĂ, este interpretat ca *mărime* iar valoarea sa diferă de la un produs la altul, în funcţie de producător, model etc.

Exerciţiu propus: Refaceţi modelul generic al magazinului, descris prin tabelele IV.1 şi IV2 şi reprezentaţi grafic diagrama cu noile entităţi. Completaţi cele patru tabele asociate noului model generic, cu valori specifice tuturor categoriilor de produse din exemplul anterior.

Acest model generic „generalizat” este deosebit de flexibil, în sensul că permite adaptarea numărului de atribute la diferitele tipuri de produse sau de entităţi, în general. Spre deosebire de primul model generic care considera un număr maxim prestabilit de atribute, invariabil de la un tip la altul, modelul generalizat poate include un număr variabil de atribute pentru diversele tipuri considerate. Modelul poate fi uşor modificat în timp, pe măsură ce se extimde

activitatea descrisă în BD. Numărul de coloane din tabelele asociate este fix, în timp ce numărul de înregistrări (linii) din tabele va creşte.
Prin realizarea unor modele de date generice, se prelungeşte durata de viaţă a BD şi se simplifică munca administratorilor de date. Totuşi complexitatea şi costurile de implementare a unei BD pe baza unui model generic sunt considerabil crescute.

IV.8 MODELAREA FIZICĂ

Transpunerea modelului conceptual într-un model care să descrie structurile bazei de date reprezintă faza de **modelare fizică** a datelor. Acest proces mai este denumit şi „mapare" (*mapping*).
Proiectarea logică consta în rafinarea modelului conceptual (prin normalizare) şi transpunerea acestuia într-un model de date logic, cunoscând tipul de SGBD ţintă (relaţional, ierarhic, în reţea sau orientat-obiect). Ea viza obţinerea unui model al BD complet, care să permită definirea tuturor vederilor utilizatorilor şi menţinerea integrităţii BD. Prin integrarea în modelul logic a schemelor externe pentru vederile utilizatorilor, respectiv a modelelor de date logice locale, se obţine **modelul de date logic global**. În această etapă se elimină acele probleme care apar atunci când se utilizează aceiaşi termeni pentru obiecte diferite sau termeni diferiţi pentru aceleaşi obiecte.
A treia etapă de proiectare**, proiectarea fizică a BD,** constă în descrierea sub forma unui model fizic, a modului de implementare a BD, a structurilor de stocare în capacitatea de stocare secundară şi a metodelor de acces la date.
În cazul bazelor de date relaţionale, din modelul de date logic global se obţin modelele tabelelor relaţionale, se deduc constrângerile impuse datelor şi necesităţile de securitate ale sistemului.
Mai precis, fiecărei entităţi i se asociază un **tabel**, numit şi **relaţie,** pe care îl descriem prin capul de tabel. Denumirea tabelului este dată, în general, de forma de plural a numelui entităţii. De exemplu, entitatea *STUDENT* se asociază cu tabelul *studenţi.*
Fiecare tabel va avea tot atâtea coloane câte atribute are entitatea respectivă.
În modelul tabelului se înscriu pe lângă denumirile atributelor, constrângerile de opţionalitate şi caracterul de cheie al fiecărui atribut.

Este util ca în modelul fizic să se folosească denumiri prescurtate ale atributelor, astfel încât transcrierea modelului fizic în limbaj de programare, să se facă mai simplu. În acest caz, în modelul fizic este necesar să se includă şi o descriere a fiecărei denumiri prescurtate.
De asemenea, în modelul fizic se mai pot include tipul datelor pentru fiecare atribut şi dimensiunea dorită, opţiuni de tranzacţionare, valori implicite (DEFAULT) şi diferite condiţionări.
De exemplu, să considerăm entitatea *CĂMIN* din figura IV.1. Aceasta are patru atribute care definesc cele patru coloane ale tabelului *cămine.* Modelul fizic la tabelului este următorul:

Tabel IV. 1 Modelul tabelului *cămine*

Tip de cheie	Opţionali-tate	Denumirea coloanei	Descriere	Alte detalii
PK	*	nr_c	număr_cămin	UNIQUE, secvenţă de 3 caractere (litere şi cifre)
	*	adm	id_administrator	Secvenţă de 4 cifre
FK	*	id_f	id_facultate	număr întreg, pozitiv, unic
	o	denum	denumirea căminului	Şir de maximum 30 de caractere, UNIQUE

Relaţiile dintre entităţi, descrise în diagrama ER, se transformă în atribute suplimentare, de tip cheie, cu caracter obligatoriu sau opţional, în funcţie de natura relaţiei.
Se aplică următoarele **reguli de mapare**:
Regula M1: O relaţie simplă de tip 1:M dintre două entităţi determină includerea unui atribut **cheie-străină** (FK – *Foreign Key*)la entitatea cu participare multiplă în relaţie determinat de cheia primară (PK – *Primary Key*) sau o cheie candidat (UK – *Unique Key*) a entităţii cu participare singulară în relaţie. De exemplu, în BD a unei facultăţi, relaţia dintre entitatea *STUDENT* şi entitatea *GRUPĂ* este de tip 1:M deoarece într-o grupă sunt incluşi mai mulţi studenţi. Relaţia aceasta se transformă (mapează) prin atributul *id_grupă* care este cheie primară a entităţii *GRUPĂ* şi devine cheie străină a entităţii *STUDENT.*
Regula M2: O relaţie simplă de tip 1:1 dintre două entităţi determină includerea unui atribut cheie-străină la una din cele două entităţi, în funcţie de regulile de afaceri. Cheia străină este determinată de cheia primară sau de o cheie unică a uneia dintre entităţile implicate în relaţie care poate fi oricare din cele două entităţi. De exemplu, pentru un serviciu de transport, în

care fiecare şofer are repartizată a anumită maşină, relaţia dintre şoferi şi maşini este 1:1 iar cheia străină poate fi fie *id_şofer* inclusă în setul de atribute al fiecărei maşini, fie *id_maşină* inclusă în lista atributelor entităţii *ŞOFER*. De regulă, cheia străină se pune în tabelul care va avea mai puţine înregistrări, pentru a salva spaţiu de memorie.

Exerciţiu propus: Realizaţi modelul fizic pentru diagrama ER corespunzătoare BD a unui magazin, cu relaţiile descrise în paragraful IV.1.

Regula M3: O relaţie barată dintre două entităţi determină apariţia unui nou atribut în setul de atribute al entităţii determinate cu rol de cheie primară sau de componentă a acesteia, respectiv de cheie secretă, cu referire la tabelul entităţii determinante. De exemplu, în BD a unei universităţi, relaţia dintre facultate şi catedre este una barată, în sensul că identificatorul catedrei trebuie asociat cu identificatorul facultăţii, pentru a identifica unic o anumită catedră din universitate. Prin urmare, atributul *id_facultate* este şi cheie străină, şi componentă a cheii primare.

Exerciţiu propus: Realizaţi modelul fizic pentru diagrama din figura IV.7.

Regula M4: Un supertip cu mai multe subtipuri este transformat într-un singur tabel dacă are mai multe atribute comune decât cele specifice fiecărui subtip. În acest caz, subtipurile sunt diferenţiate în modelul fizic, printr-un singur atribut, cu caracter obligatoriu, care specifică subtipul. Relaţiile supertipului se mapează conform regulilor anterioare. Relaţiile subtipurilor se mapează ca şi chei străine opţionale. Atributele subtipurilor devin opţionale, urmând ca la completarea tabelului, să se înscrie date doar în coloanele corespunzătoare atributelor unui singur subtip. Apare aici o constrângere de tip XOR, care se exprimă printr-o condiţionare de forma (de exemplu, considerăm două subtipuri, cu câte un atribut):

```
CHECK (atribut_1 is NOT NULL AND atribut_2 is NULL) OR (atribut_1 is NULL AND atribut_2 is NOT NULL)
```

Exerciţiu propus: Realizaţi modelul fizic pentru modelul conceptual din figura IV.5 corespunzător parcului auto al unei societăţi de transport.

Regula M5: Dacă un supertip are mai multe subtipuri care au mai multe atribute distincte decât comune, atunci fiecare subtip este transformat într-un tabel, în care se includ şi atributele comune pe lângă cele specifice subtipului. Cheia primară a supertipului devine

cheie primară în fiecare tabel de subtip. Cheile unice ale supertipului rămân chei unice în toate tabelele subtipurilor. Orice relaţie a supertipului se marchează printr-o cheie străină în toate tabelele subtipurilor. O relaţie a unui subtip se mapează doar în tabelul acestuia, cu opţionalitatea originală. De exemplu, într-un supermarket se pot comercializa produse alimentare, produse de îmbrăcăminte şi încălţăminte, cărţi, produse electrocasnice. Subtipurile entităţii *PRODUS* sunt foarte diferite şi atunci este indicat să se realizeze tabele distincte pentru subtipuri. În acest caz şi reprezentarea grafică sugerează folosirea mai multor tabele, iar marcarea cu un arc a mai multor relaţii trebuie să se traducă, în modelul fizic şi în programare, într-o constrângere de exclusivitate a subtipurilor (CHECK). Relaţia dintre supertip şi o altă entitate devine relaţie dintre fiecare subtip şi acea entitate, deci relaţia se multiplică şi se exprimă printr-o cheie străină corespunzătoare fiecărui subtip. Aceste chei străine au caracter opţional prin regula de exclusivitate a subtipurilor.

Exerciţiu propus: Reprezentaţi modelul fizic pentru diagrama ER a BD prin care se planifică activităţile dintr-o sală de sport (figura IV.6).

Regula M6: În cazul relaţiilor cu caracter netransferabil (marcate în diagrama ER cu un romb) trebuie să se precizeze în modelul fizic o constrângere referitoare la nemodificarea valorilor înscrise în tabel pe coloana cheii străine. Aceasta se traduce prin nerealizarea de actualizări sau ştergeri ale datelor de pe acea coloană (ON UPDATE NO ACTION, ON DELETE NO ACTION). Observăm că este important să se menţioneze în modelul fizic, toate detaliile impuse de regulile de afaceri pentru ca programatorii să poată înscrie toate constrângerile impuse de regulile de afaceri.

IV.9 REZUMATUL CAPITOLULUI

În acest capitol, sunt prezentate aspecte mai puţin comune şi mai dificile ale modelării datelor, precum:

- identificarea şi soluţionarea relaţiilor redundante sau de tip M:M din diagrama ER
- posibilităţile de reprezentare ierarhică sau recursivă a unor relaţii şi entităţi
- reprezentarea unor constrângeri referitoare la netransferabilitatea unor relaţii
- avantajele definirii unor subtipuri de entităţi
- modalităţi de modelare a istoricului unor variabile şi a timpului în modelul de date

- opţiunile de modelare generică
- regulile modelării fizice a BD.

IV.10 TERMENI SPECIFICI

Relaţii redundante
Relaţii M:M
Intersecţia entităţilor
Relaţii ierarhice
Relaţii recursive
Relaţii barate
Transferabilitate
Relaţii netransferabile
Subtip/Subentitate
Supertip/Superentitate
Regula de exhaustivitate
Regula de exclusivitate mutuală
Constrângere de exclusivitate mutuală
Istoricul atributului
Modelarea generică
Modelarea fizică
Reguli de mapare

IV.11 APLICAŢIE PROPUSĂ

Realizaţi diagrama ER şi modelul fizic al BD a unui supermarket, prin care se gestionează produsele, considerând cel puţin patru subtipuri de produse. Se doreşte să se cunoască amplasarea pe zone a produselor, precum şi angajaţii responsabili de plasarea lor pe rafturi.

IV.12 TEST-GRILĂ

1. Între entităţile STUDENT, DISCIPLINĂ, NOTĂ şi PROFESOR se pot stabili mai multe relaţii. Care dintre următoarele relaţii consideraţi trebuie că este redundantă şi trebuie eliminată?

- ❑ DISCIPLINĂ - NOTĂ
- ❑ DISCIPLINĂ – PROFESOR
- ❑ NOTĂ - PROFESOR
- ❑ STUDENT - NOTĂ

2. Pentru BD a unei facultăţi, care dintre relaţiile următoare este de tip M:M?

- ❑ DISCIPLINĂ – PROFESOR
- ❑ DISCIPLINĂ - SALĂ
- ❑ NOTĂ - PROFESOR
- ❑ STUDENT - DISCIPLINĂ

3. Care dintre următoarele relaţii sunt netransferabile?

- ❑ *client-chitanţă*
- ❑ *student-grupă*
- ❑ *operă-autor*
- ❑ *autovehicul-proprietar*

4. Care dintre următoarele entităţi admite subtipuri? Menţionaţi-le acolo unde este cazul.

- ❑ autovehicul: .....
- ❑ construcţie: .....
- ❑ program software: .....
- ❑ carte: .....

5. Cum se mapează supertipul?

- ❑ cu toate atributele specifice subtipurilor dar cu caracter opţional
- ❑ într-un singur tabel
- ❑ în mai multe tabele asociate subtipurilor
- ❑ printr-o cheie străină

6. Atributul de legătură dintr-o relaţie în buclă devine în modelul fizic:

- ❑ cheie primară
- ❑ cheie străină
- ❑ cheie unică
- ❑ opţional

7. Pentru maparea unei relaţii 1:M dintre două entităţi A şi B, se poate foloseşte:
- ❑ cheia primară a lui A ca şi cheie primară a lui B
- ❑ cheia primară a lui A ca şi cheie străină a lui B
- ❑ cheia primară a lui B ca şi cheie primară a lui A
- ❑ cheia primară a lui B ca şi cheie străină a lui A

8. La maparea unei relaţii barate, atributul de legătură devine:
- ❑ componentă a cheii primare a entităţii determinate
- ❑ cheie străină a entităţii determinate
- ❑ componentă a cheii primare a entităţii determinante
- ❑ cheie străină a entităţii determinante

9. La maparea unei relaţii netransferabile, se impun condiţiile:
- ❑ ON DELETE NO ACTION
- ❑ ON DELETE SET DEFAULT
- ❑ ON UPDATE CASCADE
- ❑ ON UPDATE NO ACTION

10. Se mapează în două tabele, subtipurile entităţii CLIENT, respectiv PERSOANĂ FIZICĂ şi PERSOANĂ JURIDICĂ. Entitatea CLIENT are o relaţie cu entitatea COMANDĂ. Subentitatea PERSOANĂ JURIDICĂ are o relaţie cu entitatea BANCĂ. Care dintre următoarele afirmaţii este adevărată?
- ❑ Atributele comune ale subtipurilor entităţii client apar în fiecare tabel de subtip.
- ❑ Atributele specifice ale fiecărui subtip sunt incluse ca opţionale în tabelul CLIENŢI.
- ❑ Relaţia cu entitatea COMANDĂ se mapează ca şi cheie străină în ambele tabele ale subtipurilor.
- ❑ Relaţia cu entitatea BANCĂ se mapează ca şi cheie străină în ambele tabele ale subtipurilor.

CAPITOLUL V

FINALIZAREA PROIECTĂRII BAZEI DE DATE

DIN CUPRINS:

V.1 ANALIZA „CRUD”

Aşa-numita analiză CRUD (prescurtare a secvenţei CREATE, RETRIEVE, UPDATE, DELETE), face referire la posibilitatea de a crea structura bazei de date, de a regăsi, a modifica ori actualiza datele înregistrate sau de a le şterge, atunci când îşi pierd valoarea sau când devin perimate.

Proiectarea BD trebuie făcută astfel încât să fie posibile aceste patru operaţiuni, pe serverul de BD.

Prin analiza CRUD se urmăreşte dacă modelele şi implementarea BD permit efectuarea tuturor operaţiilor amintite, asupra tuturor datelor şi în mod consistent, cu menţinerea BD într-o stare coerentă.

Prin aceasta, analiza CRUD validează modelul ER creat pentru BD.

Se verifică dacă nu s-au omis din modelul BD, entităţi, atribute sau relaţii care sunt necesare aplicaţiei descrise de BD.

De exemplu, sunt multe relaţii între tabele, exprimate prin atributele de tip cheie străină. Omiterea unei constrângeri de tip cheie străină, conduce la rezultate eronate ale unei interogări a BD, care vizează datele conţinute în tabelele relaţionate.

Se verifică, de asemenea, dacă modelul creat nu include unele aspecte a căror prezenţă nu se justifică, având în vedere cerinţele clientului specificate în scenariul BD. Comunicarea permanentă dintre proiectant şi client, este esenţială pentru crearea unui model complet şi corect dimensionat al BD.

Dacă transpunerea modelului conceptual al BD în modelul fizic şi, ulterior, implementarea BD nu sunt făcute riguros, există riscul ca anumite prevederi din documentaţia BD să nu fie respectate sau îndeplinite şi să se ajungă chiar la încălcări ale unor „reguli ale afacerii”, specifice activităţii descrise în BD.

Analiza BD trebuie făcută cu personal specializat, în mod sistematic, urmărindu-se pas cu pas, fiecare cerinţă din documentaţie.

În specificaţiile clientului trebuie urmărite anumite cuvinte-cheie care exprimă cerinţele acestuia: introducere, înregistrare, încărcare, importare, exportare, vizualizare, actualizare, raport, listare, tipărire, citire, căutare, modificare, schimbare, ştergere etc.

Toate acestea se traduc în operaţii specifice BD, de tip CRUD, şi trebuie să poată fi realizate pe baza modelului creat.

Analiza CRUD a modelului urmăreşte ca asupra oricărei entităţi din model să se poată efectua cele patru operaţii pentru a nu avea aspecte nemodelate ale aplicaţiei.
Testarea cu un set complet de date de test, care să creeze toate situaţiile reale tipice, din perspectiva tuturor tipurilor de utilizatori ai BD şi a tuturor regulilor de afaceri, precum şi corectarea aspectelor nedorite ale BD, va finaliza proiectarea acesteia.

V.2 SECVENŢĂ, INDEX, ROL

Secvenţa este un alt obiect al BD, cu caracter partajat, care permite mai multor utilizatori să atribuie identificatori unici pentru unul şi acelaşi obiect, asupra căruia se efectuează operaţii de inserare de date de către mai multe persoane. Este extrem de utilă o secvenţă de numere unice, de exemplu, de înregistrare a candidaţilor din locaţii diferite, la un concurs de admitere la o universitate, care are sedii în mai multe oraşe. Pentru a nu se crea confuzii, fiecare candidat trebuie să aibă un număr unic de înregistrare, indiferent unde s-a înscris.
Pentru bazele de date distribuite în reţea, generarea secvenţelor unice de înregistrare este esenţială pentru corectitudinea procesului de gestiune a datelor prin intermediul bazelor de date.
O secvenţă este generată independent de tabelele din BD, cu valori minime şi maxime, cu un anumit pas de incrementare, cu posibilitatea memorării unui set de valori iniţial în memoria cache şi cu riscul pierderii acestora la o problemă tehnică a sistemului precum şi cu opţiunea de ciclare a secvenţei, în sensul reluării procesului de generare a valorilor pornind de la valoarea minimă, după ce s-a ajuns la valoarea maximă.

Exerciţiu propus: Daţi trei exemple, de situaţii concrete în care este esenţială crearea şi folosirea secvenţelor în BD.

Indexul este un alt element specific BD, prin care se accelerează procesul de căutare şi de extragere a informaţiilor din tabelele BD. Crearea unui index implică ocuparea unui spaţiu de memorie suplimentar şi, de aceea, este indicat să se creeze un index doar într-un caz bine justificat. Indexul nu ia în considerare decât valorile înscrise într-o coloană, nu şi null-urile.

Securizarea BD impune acordarea de drepturi şi revocarea lor pentru diferiţi utilizatori sau pentru diferite grupuri de utilizatori. Utilizatorii din acelaşi grup sunt asociaţi cu un aşa-numit **rol**, un obiect creat distinct în structura BD (ROLE) pentru a specifica în mod unitar drepturile de accesare a BD şi de manipulare a datelor pentru utilizatorii cu acelaşi rang. Administrarea conturilor utilizatorilor BD se face mai rapid şi mai eficient prin definirea acestor roluri, decât individual, pentru fiecare utilizator în parte.

V.3 TRANZACŢII

Tranzacţia este o unitate logică de lucru care conţine una sau mai multe comenzi SQL, care se execută ca o singură funcţie logică.

Iniţierea tranzacţiei poate fi făcută de către o persoană sau un program, printr-o comandă de iniţiere de tip SELECT, INSERT, UPDATE.

Până la completarea tranzacţiei, efectele ei nu sunt vizibile.

Tranzacţia lasă BD într-o stare coerentă (Fig. V.1). Dacă, dintr-un anumit motiv, una sau mai multe operaţii din cadrul unei tranzacţii eşuează, atunci toate operaţiile efectuate până la acel pas, în cadrul acelei tranzacţii, sunt anulate şi BD revine în starea coerentă, în care se găsea înainte de lansarea tranzacţiei. De fapt, pe toată durata efectuării tranzacţiei, excluzând pasul de finalizare a acesteia, BD nu îşi modifică starea.

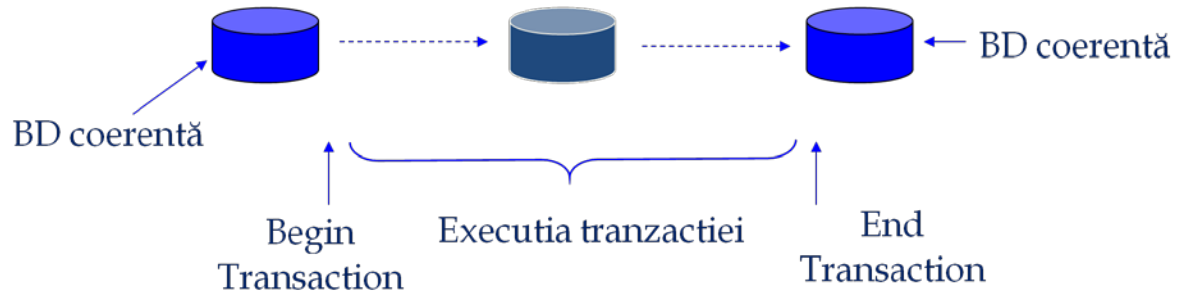

Fig. V.3 Schema de principiu a unei tranzacţii

Tranzacţia include operaţii de acces la BD şi de manipulare a datelor:

- **BEGIN** - începerea efectuării tranzacţiei; tranzacţia intră în **faza activă** de derulare.
- **READ** – citirea datelor
- **WRITE** – scrierea datelor

- **END** – încheierea operaţiilor de manipulare a datelor, specificate în acea tranzacţie
- **COMMIT** – aplicarea efectelor tranzacţiei, după verificarea execuţiei corecte a tuturor operaţiilor tranzacţiei, asupra întregului set de date din BD. Modificările din BD sunt permanente şi nu mai pot fi anulate sau pierdute, la o defectare ulterioară a sistemului.
- **ABORT** – renunţare la efectuarea tranzacţiei; tranzacţia este declarată ca **abandonată (aborted)**. Se execută procesul de revenire la datele iniţiale (**ROLLBACK**), dacă din anumite motive, tranzacţia nu poate fi complet şi corect efectuată (tranzacţie eşuată – **failed).**
- **UNDO** – anularea unei singure operaţii din cadrul tranzacţiei.
- **REDO** – reluarea unor comenzi din tranzacţie, pentru a o putea finaliza.

Tranzacţiile sunt descrise prin patru proprietăţi:

1. **Atomicitate** (*all or nothing*) - acţiunile tranzacţiei se execută toate sau nu se execută deloc.
2. **Consistenţa** – tranzactia menţine BD într-o stare coerentă, după execuţie.
3. **Izolare** - tranzactia este protejată de efectele executării concurente a altor tranzacţii.
4. **Durabilitate** – efectele tranzacţiei comise sunt permanente chiar şi în cazul unei căderi a SGBD (*Database recovery*).

SGBD menţine într-o listă (***log)*** toate acţiunile efectuate asupra datelor, astfel încât poate să refacă datele (undo) după anularea unor operaţii sau tranzacţii.

Asigurarea atomicităţii tranzacţiei în prezenţa căderilor sistemului se numeşte „recuperare la căderi" (c*rash recovery*). Acest procesul de refacere va scana până la cel mai recent punct stabil, *checkpoint*, care conţine lista tranzacţiilor active şi apoi până la punctul de începere a acestora.

O mare problemă a bazelor de date şi a proiectanţilor de BD este aceea de gestionare a conflictelor care apar în cazul tranzacţiilor concurente (care se execută simultan asupra aceloraşi structuri şi date din BD).

Două tranzacţii de citire a aceloraşi date nu sunt conflictuale şi ordinea lor nu contează. Două tranzacţii, care citesc sau scriu obiecte din BD distincte, nu sunt conflictuale şi de asemenea, ordinea de execuţie a lor nu contează. Însă dacă o tranzacţie scrie un obiect de date şi o alta citeşte sau scrie acelaşi obiect, cele două tranzacţii intră în conflict iar ordinea execuţiei lor este esenţială pentru obţinerea rezultatului corect din punctul de vedere al aplicaţiei.

În BD, apar următoarele tipuri de **conflicte**:

- **Scriere - citire** (WR): a doua tranzacţie citeşte un obiect scris anterior de prima tranzacţie.
- **Citire - scriere** (RW): a doua tranzacţie scrie un obiect de date citit anterior de către prima tranzacţie.
- **Scriere – scriere** (WW): a doua tranzacţie scrie un obiect de date scris anterior de prima tranzacţie.

Lipsa unui control asupra ordinii de execuţie a operaţiilor din tranzacţii concurente poate să genereze mari erori în conţinutul BD.

De exemplu, să presupunem, că simultan un contabil efectuează majorarea salariilor cu 20 % iar alt contabil înscrie în BD bonusurile acordate salariaţilor pentru ultima lună. Pentru un salariat, cu un salariu de 2000 RON, căruia i se acordă un bonus de 500 RON, în ordinea specificată a tranzacţiilor, venitul încasat va fi de 2900 RON. Dacă însă se înregistrează mai întâi bonusul şi abia apoi se aplică majorarea de 20 %, salariatul va încasa 3000 RON, pentru că majorarea se va aplica şi bonusului, ceea ce încalcă regulile aplicaţiei.

Este foarte important să fie gestionate corect tranzacţiile concurente şi, pentru aceasta, se folosesc aşa-numitele „**lacăte**" (LOCK), operaţii de blocare a accesului, aplicate obiectelor din BD care sunt folosite într-una din tranzacţii, astfel încât nici o altă tranzacţie să nu poată să le citească sau să le modifice până la finalizarea primei tranzacţii. Lacătele asigură acces exclusiv la anumite obiecte din BD pentru o tranzacţie, pentru o anumită perioadă de timp.

Lacătele pot fi de mai multe tipuri:

- **de blocare a accesului** la obiect: interzice accesul altor tranzacţii la acel obiect, prevenind rezultatele incorecte.
- **de citire** a obiectului (*shared/read lock*) – obiectul poate fi citit, dar nu modificat.
- **de scriere** a obiectului (*exclusive/write lock***) –** tranzacţia care deţine lacătul, poate să citească şi să scrie obiectul pe care este aplicat acesta.

SGBD gestionează, prin componenta sa de management a lacătelor, ordinea de acordare a acestora pentru diferitele tranzacţii.

De exemplu, să presupunem că tranzacţia 1 (T1) este cea de majorare cu 20% a salariilor înscrise în tabelul A iar tranzacţia 2 (T2) este cea de acordare de bonusuri (tabelul B) şi se aplică pe acelaşi tabel din BD. Să urmărim modul de rezolvare a conflitului dintre tranzacţii, prin folosirea lacătelor:

	T1	T2
1	Begin-transaction	

2	Write-lock(A)	Begin-transaction
3	Read(A)	Write-lock(A)
4	A=A*1.20	Wait
5	Write(A)	Wait
6	Unlock(A)	A=A+B

Gestionarea tranzacţiilor concurente revine ca sarcină a programatorului de BD. Tranzacţiile nu pot fi reprezentate în diagram ER dar ordinea de execuţie şi restricţiile de execuţie a lor sunt descrise în documentaţia BD, prin regulile de afaceri ale aplicaţiei.

V.4 ETAPELE CICLULUI DE VIAŢĂ AL UNEI APLICAŢII CU BD

Pentru a realiza o aplicaţie de BD, este necesară urmărirea paşilor următori:

- **Planificarea BD** constă în activităţi administrative care vizează identificarea planurilor de afaceri şi a cerinţelor sistemului informaţional, evaluarea situaţiei curente şi a oportunităţilor IT, şi se bazează pe modelul general de date care include entităţile şi relaţiile dintre ele, reprezentate într-o diagramă ER, în care se specifică şi legăturile cu diferitele zone funcţionale ale întreprinderii.
- **Definirea sistemului** constă în stabilirea scopului şi a limitelor aplicaţiei BD, inclusiv domeniile de utilizare şi grupurile principale de utilizatori.
- **Colectarea şi analiza cerinţelor** se face prin chestionarea persoanelor reprezentative pentru toate domeniile de interes ale întreprinderii, observarea funcţionării acesteia şi analiza documentelor utilizate pentru înregistrarea şi redarea informaţiilor, observarea tranzacţiilor efectuate etc. Instrumentele CASE (*Computer-Aided Software Engineering*) sunt utile pentru specificarea completă şi coerentă a cerinţelor (caracteristicilor) întreprinderii, .
- **Proiectarea BD** începe cu realizarea unor modele de date care conţin entităţi şi relaţii de nivel înalt (esenţiale pentru reprezentarea funcţionalităţii întreprinderii în BD), după care se trece la o analiză detaliată pentru a identifica şi include în model toate entităţile, atributele şi relaţiile de nivel jos.

- **Alegerea unui SGBD** adecvat, care să accepte aplicaţia de tip BD, se poate face între fazele de proiectare logică şi conceptuală a BD. Se au în vedere costurile de achiziţii software şi hardware, cele de instruire a personalului, precum şi performanţele sistemului.
- **Proiectarea programelor de aplicaţie** pentru utilizatori, cu interfeţe prietenoase şi eficiente, cu posibilităţi de accesare a datelor şi efectuare a tranzacţiilor în BD, se face în paralel cu proiectarea propriu-zisă a BD.
- **Realizarea unui prototip al BD** este o etapă opţională dar avantajoasă, întrucât pe baza unui model de lucru simplificat, cu costuri reduse, se testează funcţionalitatea aplicaţiei BD, se identifică eventualele probleme dar şi caracteristici noi care trebuie implementate.
- **Implementarea BD şi a aplicaţiilor** constă în realizarea fizică a proiectelor acestora folosind diverse limbaje. Implementarea BD se face pe baza unui limbaj de definire a datelor (DDL). Instrucţiunile DDL sunt compilate pentru a crea schema BD. Vederile utilizatorilor sunt definite în această etapă. Programele-aplicaţie sunt implementate cu ajutorul altor limbaje, DML, sau de nivel înalt Java, C, C++, Delphi, cu meniuri, formulare, rapoarte şi cu reguli de securitate şi de integritate.
- **Conversia şi încărcarea datelor** constă în transferul în BD a datelor existente, cu eventuala conversie de format, folosind un utilitar de încărcare în BD a fişierelor deja existente.
- **Testarea aplicaţiei BD** constă în executarea programelor de aplicaţie cu scopul depistării erorilor de funcţionare, pe baza unor strategii de testare. **Întreţinerea operaţională** se fac**e** prin monitorizarea continuă a aplicaţiei, remedierea erorilor de funcţionare prin reorganizarea BD şi eventual, implementarea unor cerinţe noi. Este indicată folosirea noilor aplicaţii de tip BD în paralel cu cele vechi, dacă acestea există, pentru o anumită perioadă de timp (de regulă, şase luni) în care să se observe şi să se rezolve disfuncţionalităţile noului sistem.

V.5 REZUMATUL CAPITOLULUI

Proiectarea bazei de date se încheie cu analiza CRUD (CREATE, RETRIEVE, UPDATE, DELETE), care testează posibilitatea de a crea întreaga structură a bazei de date, de a regăsi, a modifica ori a actualiza datele înregistrate sau de a le şterge, atunci când nu mai au valoare. În fapt, aceasta face parte din etapele ciclului de viaţă al unei aplicaţii cu BD.

Optimizarea BD în fazele de modelare dar şi de implementare necesită crearea de noi obiecte şi elemente în BD, precum vederile externe pentru diferite categorii de utilizatori ai BD, secvenţele de numere unice şi indexurile de accelerare a procesului de căutare în BD.

Gestionarea tranzacţiilor şi rezolvarea eventualelor conflicte reprezintă un aspect-cheie al aplicaţiilor cu BD prin care se evită disfuncţionalităţile aplicaţiei, erorile şi inconsistenţa datelor.

Ciclul de viaţă al aplicaţiei de BD începe cu activităţile de **planificare** ce includ identificarea planului de afaceri şi a cerinţelor sistemului informaţional, evaluarea situaţiei curente şi a oportunităţilor IT, urmate de **definirea sistemului, colectarea şi analiza cerinţelor. Proiectarea** propriu-zisă a BD se continuă cu a**legerea unui SGBD** adecvat şi **proiectarea programelor de aplicaţii** pentru utilizatori. **Realizarea unui prototip al BD** permite testarea modelelor folosite şi a eventualelor probleme. **Implementarea BD şi a aplicaţiilor şi testarea lor** este realizată de programatorii BD. Administratorii de date se ocupă apoi de **conversia şi încărcarea datelor.**

Este esenţială **întreţinerea operaţională** a aplicaţiei cu BD şi este recomandat ca pentru un anumit timp, noile aplicaţii de BD să fie rulate în paralel cu cele vechi, astfel încât să se observe şi să se rezolve eventualele disfuncţionalităţi ale noului sistem.

V.6 TERMENI SPECIFICI

Analiza CRUD

Secvenţă

Index

Rol

Tranzacţie

Conflict

Lacăt

Tipuri de lacăte

Checkpoint

Crash recovery

Ciclu de viaţă

V.7 TEST-GRILĂ

1. Care dintre următoarele operaţii sunt vizate prin analiza CRUD?
- ❑ MODIFY
- ❑ SELECT
- ❑ SET
- ❑ TRUNCATE

2. Accelerarea procesului de căutare în BD se face prin crearea:
- ❑ unei interfeţe grafice
- ❑ unui index
- ❑ unei secvenţe
- ❑ unei vederi

3. Scopul unei secvenţe create în BD este acela de a:
- ❑ accelera căutarea datelor
- ❑ crea sinonime ale obiectelor deja definite
- ❑ genera identificatori unici
- ❑ trunchia tabelele din BD

4. Rezolvarea conflictelor dintre tranzacţiile concurente care pot să apară în BD, se face prin definirea unor:
- ❑ indexuri
- ❑ lacăte
- ❑ roluri
- ❑ secvenţe

5. Administrarea drepturilor grupurilor de utilizatori ai BD se face prin intermediul:
- ❑ indexurilor
- ❑ lacătelor
- ❑ rolurilor
- ❑ secvenţelor

BIBLIOGRAFIE

[1] Thomas Connoly, Carolyn Begg, Anne Strachan, *Baze de date – Proiectare, Implementare, Gestionare*, Editura Teora, 2001

[2] Hugh Darwen, *An Introduction to Relational Databases Theory*, Hugh Darwen & Ventus Publishing ApS, 2010

[3] Ramez Elmasri and Shamkant B. Navathe, *Fundamentals of Database Systems*, 5th Edition, Addison-Wesley, 2007

[4] C. J. Date, *An Introduction to Database Systems*, 8th Edition, Addison-Wesley, 2003