CAPITOLUL 2

MODELAREA DATELOR

Proiectarea corectă a structurii unei baze de date relaţionale este o premisă fundamentală în scrierea programelor de aplicaţie şi in functionarea lor fara anomalile care pot apare in cazul unei structuri defectuoase. Acest capitol prezinta un model de date folosit in proiectarea conceptuala de nivel inalt numit **modelul entitate asociere** (EA) in varianta clasica (cu unele extensii). Intr-un alt capitol vor fi prezentate regulile de transformare din modelul entitate-asociere in modelul relational.

2.1. Etapele dezvoltarii unei aplicatii

Proiectarea structurii bazelor de date relationale este un domeniu in care cercetarile au evoluat spre elaborarea unor metodologii teoretice şi a unor instrumente software care asigura un grad ridicat de normalizare a schemelor de relatie, pastrarea integritatii, evitarea anomaliilor datorate unei proiectari nesistematice şi eliminarea subiectivismului proiectantului prin inlocuirea lui cu exactitatea metodei.

Pana la aparitia unor astfel de metodologii se pornea de la o colectie de tabele şi de la dependentele functionale şi multivalorice asociate şi se ajungea, prin aplicarea unor algoritmi sau metode de normalizare la schema dorita a bazei de date. Aceasta abordare de tip bottom-up creaza insa dificultati in cazul bazelor de date complexe in care numarul de tabele şi de dependente este mare. Probabilitatea ca unele interdependente intre date sa nu fie sesizate in procesul de proiectare este in aceste cazuri ridicat rezultand in exploatare anomalii care duc la reluari ale ciclului analiza cerintelor → schema bazei de date → programe de aplicatie.

Cercetarile in domeniul proiectarii schemei conceptuale s-au concentrat pe gasirea unor metodologii top-down pentru obtinerea unei structuri optime a BD. Ele au fost influentate de rezultatele obtinute in domenii care folosesc bazele de date: inteligenta artificiala, proiectarea asistata de calculator, abordarea orientata pe obiecte.

Impactul conceptelor de abstractizare şi generalizare precum şi elaborarea unui model de descriere informala a datelor, şi anume modelul entitate-asociere (EA), au dus la gasirea unor cai algoritmizabile de proiectare optima a bazelor de date. Modelul entitate-asociere este in acest moment cel mai popular model de comunicare a structurii bazelor de date datorita intuitivitatii şi simplitatii elementelor sale. Imbunatatiri sale ulterioare prin

folosirea abstractizarilor şi generalizarilor au dus la crearea de variante ale modelului, doua dintre acestea fiind descrise in acest capitol.

Extensiile modelului EA au aparut şi pentru alte necesitati:

- modelarea cerintelor de secretizare a datelor,
- documentarea programelor de aplicatie şi usurarea comunicarii intre proiectantul de sistem şi utilizatorul obisnuit,
- proiectarea bazelor de date complexe pe portiuni şi integrarea ulterioara a acestora (asa numita integrare a vederilor).

Avantajul principal al modelului EA este acela al simplitatii sale şi al caracterului sau intuitiv. Rezultatul proiectarii consta intr-o diagrama entitate-asociere care poate fi apoi translatata in modelul de date folosit de sistemul de gestiune a bazelor de date ales pentru dezvoltarea aplicatiei.

Figura 2.1. prezinta schematic etapele proiectarii unei noi aplicatii care gestioneaza o baza de date, cu accentul pe partea de proiectare a structurii acesteia. Aceste etape sunt detaliate in paragrafele urmatoare.

2.1.1. Analiza de sistem

In aceasta etapa se realizeaza analiza segmentului din lumea reala care va fi gestionat de aplicatia respectiva. Rezulta o specificatie neformalizata a cerintelor constand din doua componente:

1. ***Cerinte privind continutul bazei de date:*** categoriile de date care vor fi stocate şi interdependentele dintre acestea.
2. ***Cerinte privind prelucrarile efectuate de aplicatie:*** prelucrarile efectuate asupra datelor, arborele de meniuri al aplicatiei, machetele formatelor de introducere şi prezentare a datelor şi ale rapoartelor tiparite de aceasta.

In general aceasta etapa nu poate fi asistata prin programe de tip CASE dar exista reguli care ajuta proiectantul in realizarea sa. Activitatile desfasurate includ:

- Analiza activitatii desfasurate la momentul respectiv de beneficiarul aplicatiei sau de o multime reprezentativa de beneficiari in cazul aplicatiilor de uz general.
- Analiza continutului de date şi a functionalitatii aplicatiilor software - daca exista - care vor fi inlocuite de noua aplicatie incluzand meniuri, machete ecran şi machete de rapoarte.
- Analiza formularelor tipizate şi a altor documente utilizate de beneficiar pentru realizarea activitatii respective.
- Identificarea tuturor interdependentelor dintre datele care vor fi stocate in baza de date şi a restrictiilor privind valorile pe care le pot lua anumite categorii de date.
- Identificarea - daca este cazul - a prelucrarilor care se declanseaza automat atat in cazul modificarii bazei de date cat şi la momente prestabilite de timp (de exemplu sfarsit de luna, de an, etc.)
- Identificarea operatiilor care sunt necesare beneficiarului in activitatea curenta dar care in acest moment nu sunt realizate prin intermediul aplicatiilor software folosite precum si a operatiilor care pot fi incluse in mod natural in noua aplicatie.

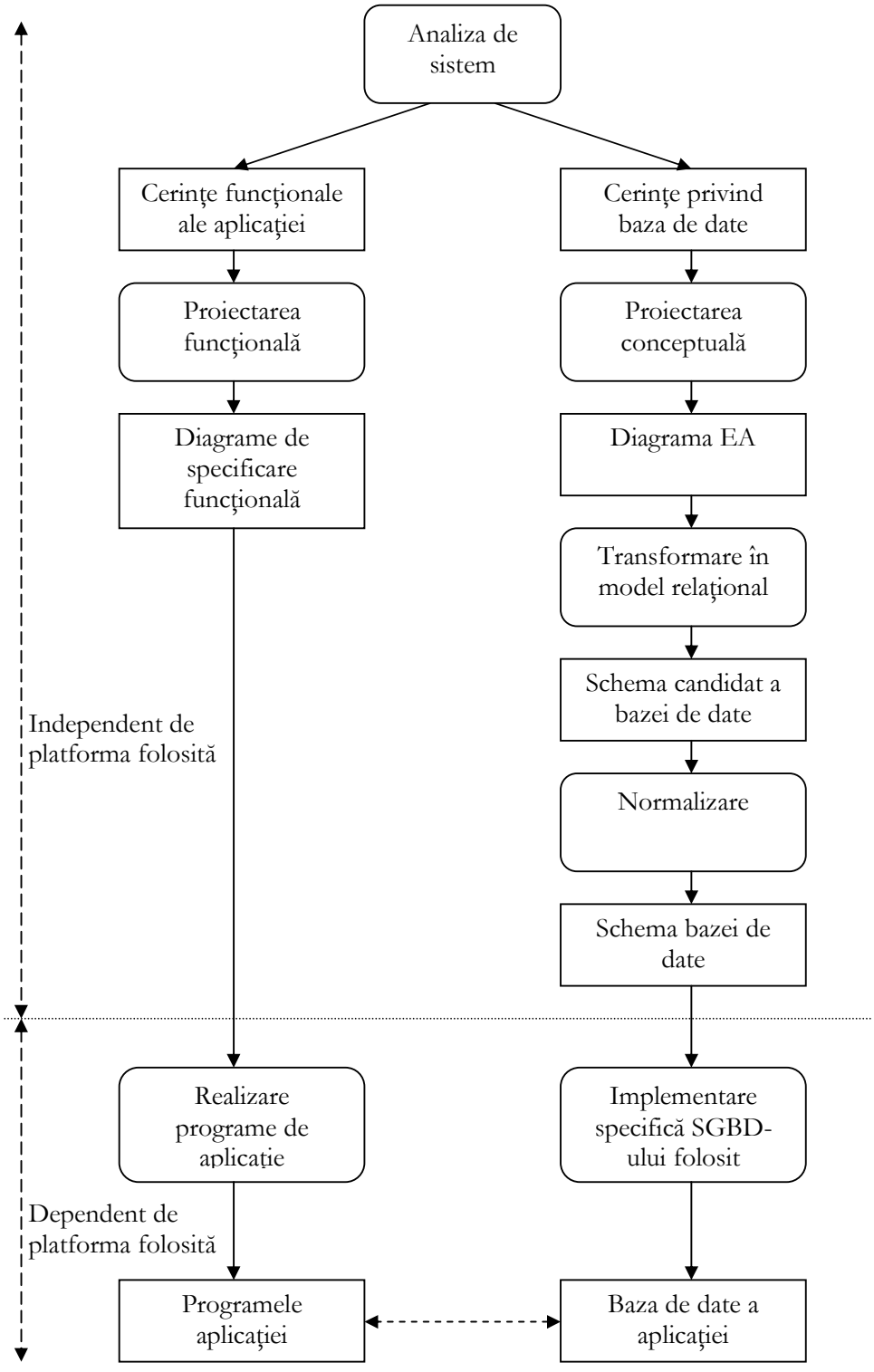


Fig. 2.1. Etapele proiectării unei aplicaţii

- Identificarea bazelor de date existente care pot fi folosite de noua aplicatie - direct sau printr-un import initial de date - evitandu-se in acest fel reintroducerea manuala a acestora.
- Identificarea modalitatilor de transfer de date intre noua aplicatie şi alte aplicatii care ruleaza deja la beneficiar şi care vor fi folosite şi in viitor de catre acesta.
- Identificarea necesitatilor privind datele şi prelucrarile care pot fi in viitor necesare beneficiarului, deci a posibilelor dezvoltari in timp ale aplicatiei.

Aceasta etapa este efectuata de personal calificat avand in vedere ca rezultatele sale sunt baza de la care se pleaca in etapele urmatoare, eventualele erori putand fi corectate ulterior cu costuri semnificative.

2.1.2. Proiectarea conceptuală a bazei de date

In aceasta etapa, pornind de la rezultatele analizei de sistem, se realizeaza modelarea cerintelor privind datele folosind un model de nivel inalt. Cel mai popular model folosit pentru aceasta este modelul entitate-asociere (EA). Actualmente exista pe piata numeroase instrumente CASE care folosesc diverse variante ale modelului. Motivele pentru care a fost ales sunt urmatoarele:

- Nu este legat direct de nici unul dintre modelele folosite de sistemele de gestiune a bazelor de date (relational sau orientat obiect) dar exista algoritmi bine pusi la punct de transformare din model EA in celelalte modele de date.
- Este intuitiv, rezultatul modelarii fiind o diagrama care defineste atat datele stocate in baza de date cat şi interdependentele dintre acestea.
- Poate fi usor de inteles de nespecialisti. Aceasta caracteristica este foarte importanta in momentul in care se face punerea de acord cu beneficiarul asupra structurii bazei de date a aplicatiei, evitandu-se in acest fel o proiectare neconforma cu realitatea sau cu cerintele exprimate de acesta.
- Proiectarea se poate face pe portiuni, diagramele partiale rezultate putand fi apoi integrate pe baza unor algoritmi şi metode bine puse la punct.

2.1.3. Transformare în model relaţional

In aceasta etapa entitatile şi asocierile care formeaza diagrama EA se transforma pe baza unor reguli clare in structura relationala a bazei de date. Rezulta schema preliminara a acesteia formata din tabele (relatii in terminologia relationala), coloanele acestora (atribute ale relatiilor) şi constrangerile de integritate care pot fi deduse automat din diagrama incluzand unele interdependente intre date numite şi ***dependente functionale***. In capitolul 3 este descrisa o metoda de transformare din modelul EA clasic in modelul relational. In cazul variantei specifice uneltelor CASE transformarea se face automat de catre acestea.

2.1.4. Normalizare

Exista o serie de reguli care descriu ce inseamna o structura corecta a unei tabele şi care definesc asa numitele ***forme normale***. Pe baza structurii bazei de date şi a dependentelor rezultate atat din transformarea in model relational cat şi a altor dependente identificate de proiectant in analiza de sistem se poate face o operatie numita ***normalizare*** modificand structura bazei de date astfel incat toate tabelele din aceasta sa fie in forma normala dorita. Capitolul 3 contine definitia formelor normale uzuale şi descrierea unor

algoritmi de normalizare care sunt folositi de uneltele CASE pentru efectuarea automata a operatiei.

2.1.5. Implementarea specifică sistemului de gestiune folosit

In aceasta etapa se realizeaza crearea structurii bazei de date obtinuta in etapa precedenta pe baza facilitatilor oferite de sistemul de gestiune a bazelor de date ales. Rezultatul ei este programul de creare scris in limbajul de definitie a datelor acceptat de SGBD-ul utilizat. Iata un exemplu:
Schema conceptala:

Student(CodStudent:Numeric, Nume:Şir, DataNasterii:Dată)

Program de creare (SQL-Oracle):

```
Create table Student(
        CodStudent   Number(5) Primary Key,
        Nume         Varchar2(40),
        DataNasterii Date);
```

Programul contine atat definirea tabelelor cat şi definirea celorlalte obiecte ale bazei de date şi a interdependentelor dintre acestea (de exemplu constrangerile de integritate intra-tabela şi inter-tabele).
De asemenea aici se fac şi toate operatiile privind proiectarea la nivel fizic a bazei de date.
In cazul folosirii de unor unelte CASE programul de creare poate fi generat şi executat automat.

Prezentam in continuare elementele care definesc modelul entitate asociere in cele doua variante: clasic şi specific instrumentelor CASE.

2.2. Modelul entitate-asociere clasic

Acest model a fost introdus de P. P. Chen in 1976 ([Ch 76]). El se constituie intr-o abordare de tip grafic a proiectarii bazelor de date si a fost adoptat de comunitatea stiintifica precum si de producatorii de software in domeniu datorita simplitatii si expresivitatii sale.

2.2.1. Elementele modelului

Modelul entitate-asociere permite reprezentarea informatiilor despre structura bazelor de date folosind trei elemente de constructie: entitati, atribute ale entitatilor şi asocieri intre entitati. Definitia lor informala este urmatoarea:

Entitatile modeleaza clase de obiecte concrete sau abstracte despre care se colecteaza informatii, au existenta independenta şi pot fi identificate in mod unic. Exemple de entitati: Studenti, Orase, Angajati, etc. Ele definesc de obicei persoane, amplasamente, obiecte sau evenimente cu importanta informationala. Membrii unei clase care formeaza o astfel de entitate poarta numele de **instante** ale acelei entitati. De remarcat ca intreaga literatura de specialitate defineste intii conceptul de multime de entitati (sau tip de entitati) pentru ca apoi sa adopte pentru usurinta exprimarii prescurtarea de entitate pentru acest concept. Deci entitatea este un obiect generic care reprezinta multimea tuturor instantelor sale.
Entitatile sunt de doua categorii:

1. Entitati independente (sau tari) sunt cele care au existenta independenta de alte entitati,
2. Entitati dependente (sau slabe) sunt formate din instante care isi justifica incadrarea in clasa respectiva doar atita timp cit intr-o alta entitate (tata) exista o anumita instanta de care sunt dependente. De exemplu in cazul unei baze de date de personal, fiecare instanta a entitatii COPII ramine in clasa respectiva (copiii angajatilor) atit timp cit in entitatea ANGAJATI exista instanta reprezentand pe tatal/mama acelui copil.

Element al modelului	Tip	Reprezentare	Exemplu
Entitate	Tare	Nume entitate	STUDENT
	Slaba	Nume entitate	COPIL
Atribut	De identificare	Nume atribut	CodStudent
	De Descriere	Nume atribut	Nume
Asociere	Asociaza 1-2 entitati	Nume asociere	Inscris_La
	Asociaza mai mult de 2 entitati (exemplu: 3 entitati)	3: 6:	Alocare

Fig. 2.2. Conventia de reprezentare a elementelor modelului EA

Atributele modeleaza proprietati atomice distincte ale entitatilor. De exemplu entitatea STUDENTI poate avea ca atribute Matricola, Nume, Prenume, Varsta, Anul, Grupa, etc. In procesul de modelare vor fi luate in considerare doar acele proprietati ale entitatilor care sunt semnificative pentru aplicatia respectiva. Din acest motiv, la entitatea STUDENTI nu vom lua in considerare caracteristici ca Talia sau Culoarea_parului acestea nefiind necesare pentru baza de date a universitatii (astfel de atribute ar putea exista de exemplu intr-o baza de date privind personalul militar).

Atributele unei entitati sunt de doua feluri:

- **atributele de identificare** (formand impreuna **identificatorul entitatii**) reprezinta acea multime de atribute care permit distinctia intre instantele aceleiasi entitati

- **atributele de descriere** (sau descriptori) sunt folositi pentru memorarea caracteristicilor suplimentare ale instantelor.

In exemplul de mai sus Matricola este atribut de identificare (deoarece nu pot exista doi studenti cu aceeasi matricola intr-o facultate) pe cand celelalte atribute sunt descriptori.

Asocierile modeleaza interdependentele dintre clasele de obiecte reprezentate prin entitati. De exemplu intre entitatile STUDENTI şi FACULTATI se poate figura o asociere INSCRIS_LA care descrie impartirea studentilor pe facultati.
In crearea diagramei nu vor fi luate in consideratie decit interdependentele care sunt necesare aplicatiei respective, in lumea reala putand exista intre entitatile diagramei şi alte asocieri care nu sunt semnificative in contextul dat.

Figura 2.2. prezinta conventia de reprezentare grafica a celor trei tipuri de constructii care participa la formarea unei diagrame EA. Se observa ca:

- Entitatile se reprezinta prin dreptunghiuri in care este inscris numele entitatii. In cazul entitatilor dependente (slabe), conturul va fi cu linie dubla.
- Atributele se reprezinta prin cercuri (sau ovale) in interiorul carora apare numele atributului. Ele sunt conectate cu un segment de dreapta la entitatea de care apartin. Pentru a distinge atributele de identificare de cele de descriere, numele primelor va fi subliniat.
- Asocierile se reprezinta prin romburi (daca conecteaza una sau doua entitati) sau poligoane regulate (daca conecteaza mai mult de doua entitati) conectate prin segmente de dreapta la entitatile asociate, avand inscris in interior (sau alaturi) numele asocierii. Alte elemente grafice specificand caracteristici suplimentare ale asocierilor vor fi introduse in paragrafele urmatoare.

2.2.2 Extensii ale modelului

Modelul entitate-asociere clasic are unele lipsuri in ceea ce priveste posibilitatea modelarii caracteristicilor asociate unor subclase de obiecte modelate prin simple entitati. Pentru aceasta, la modelul original au fost adaugate doua noi concepte: **ierarhia de generalizare** şi **ierarhia de incluziune**. Prima defineste partitionarea instantelor unei entitati in **n** subclase diferite iar a doua permite clasarea unora dintre instantele unei entitati in **m** subclase care nu reprezinta o partitie in sens matematic. Din punct de vedere formal, cele doua concepte se pot defini astfel:

Definitie (ierarhia de incluziune): O entitate E_1 este o submultime a entitatii E (sau este inclusa in entitatea E) daca fiecare instanta a lui E_1 este de asemenea o instanta a lui E.
Un exemplu de incluziune este definirea in cadrul entitatii ANGAJATI a unor subclase modelate prin entitatile INGINERI, ECONOMISTI si COLABORATORI.
In cazul ierarhiei de incluziune entitatile fiu pot sa nu fie disjuncte doua cite doua: pentru exemplul dat, exista angajati ingineri si care sunt incadrati cu contract de colaborare. De asemenea reuniunea lor poate sa nu acopera in intregime entitatea tata: exista angajati care nu sunt nici ingineri, nici economisti si nici colaboratori.

Definitie (ierarhia de generalizare): O entitate E este generalizarea entitatilor E_1, E_2, ..., E_n daca orice instanta a lui E este de asemenea instanta in una şi numai una din entitatile E_1, E_2, ..., E_n.
Un exemplu de generalizare este clasarea instantelor entitatii ANGAJATI in subclasele BARBATI şi FEMEI.
Caracteristica ierarhiei de generalizare este ca din punct de vedere matematic entitatile fiu reprezinta o partitie a entitatii tata:

a. $E_1 \cup E_2 \cup ... \cup E_n = E$ şi

b. $E_i \cap E_j = \varnothing$ pentru orice $i \neq j$ din intervalul 1..n

Ierarhiile de incluziune şi generalizare se folosesc doar in cazul in care pentru subclasele unor clase modelate prin entitati este nevoie de stocarea unor informatii suplimentare specifice.
In cazul unei baze de date de personal este nevoie de exemplu sa fie memorat numarul de copii ai fiecarui angajat. Acest fapt se poate modela in doua feluri: fie prin adaugarea la entitatea ANGAJATI a unui atribut suplimentar *NumarCopii* (care va avea valoarea 0 pentru angajatii fara copii) fie prin aparitia unei entitati suplimentare CU_COPII aflata intr-o relatie de incluziune cu entitatea ANGAJATI şi care va avea ca atribute de identificare pe cele ale tatalui iar ca atribut descriptiv numarul de copii, acesta fiind atributul specific subclasei.

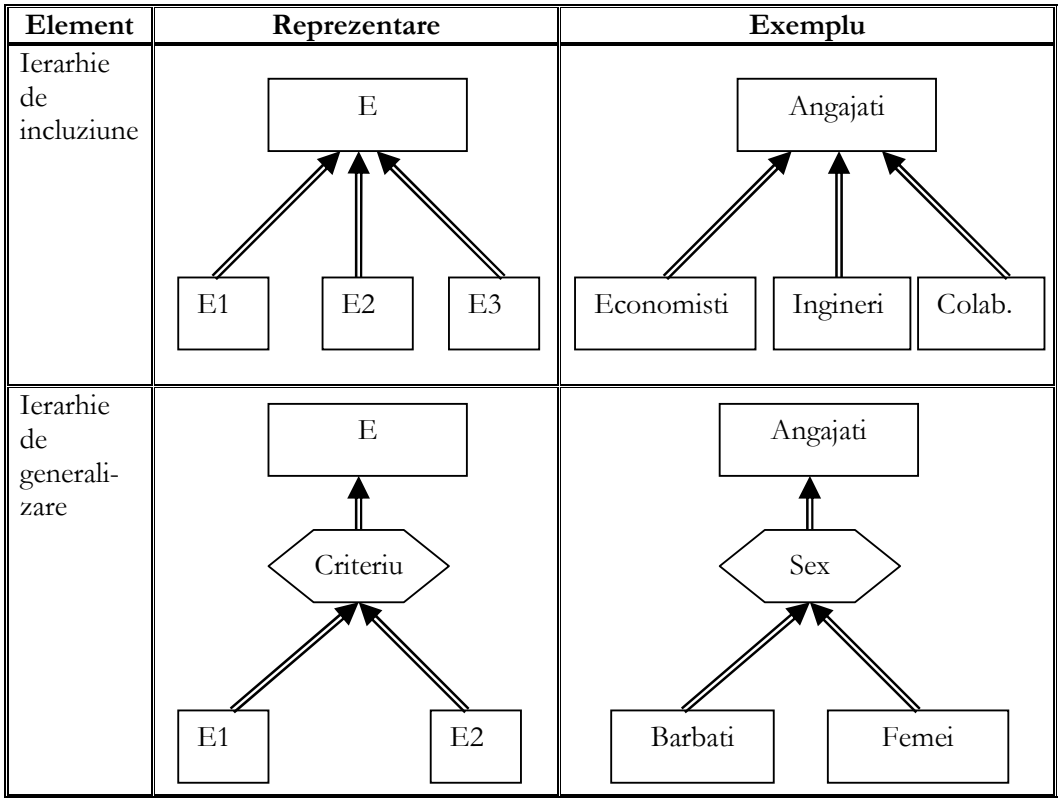


Fig. 2.3. Conventia grafica de reprezentare grafica a ierarhiilor

Vom alege a doua varianta in cazul in care numarul angajatilor cu copii este mult mai mic decit numarul total al angajatilor, fapt care va duce la economie de spatiu pe disc: in acest

caz o noua entitate - din care va rezulta o tabela a bazei de date – va ocupa mai putin spatiu decat un atribut suplimentar - din care rezulta o coloana suplimentara a tabelei SALARIATI.
Conventia grafica de reprezentare a celor doua tipuri de ierarhii se gaseste in fig. 2.3.

2.2.3. Caracteristici ale elementelor modelului

Asa cum entitatile au atribute care specifica diversele proprietati ale clasei de obiecte modelate, şi asocierile au caracteristici care aduc informatii suplimentare. Acestea sunt urmatoarele:

Gradul asocierii

Este o valoare numerica intreaga si este dat de numarul de entitati care participa la acea asociere. Asocierile de grad 1, 2 şi 3 se mai numesc si asocieri ***unare***, ***binare*** si respectiv ***ternare***.

Pentru exemplificare vom considera o baza de date continand informatii despre studenti, proiectele realizate de acestia, calculatoarele pe care au alocate ore de lucru si facultatile la care sunt inscrisi. De asemenea vom considera ca unii dintre studenti au un ***tutor*** care ii indruma, acesta fiind un student dintr-un an mai mare.
Diagrama EA a bazei de date este prezentata in figura 2.4. Pentru simplificarea figurii, nu s-au reprezentat decit entitatile şi asocierile dintre ele nu şi atributele fiecarei entitati in parte.

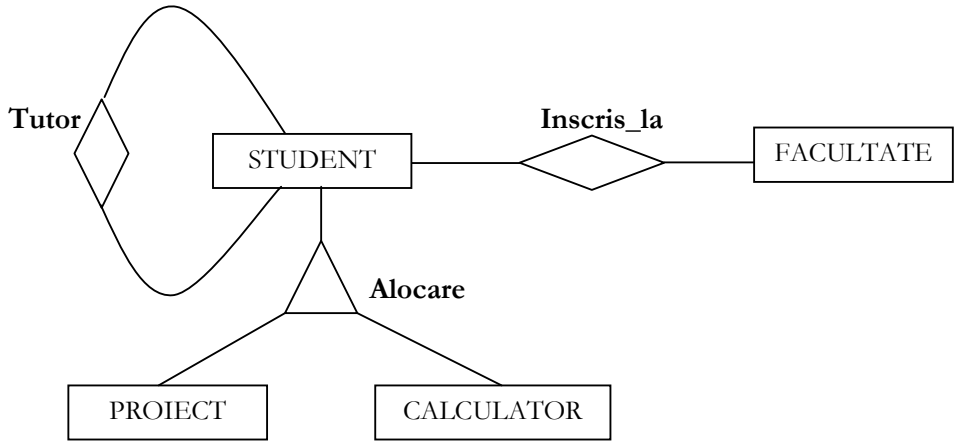


Fig. 2.4. Exemple de asocieri de grad 1, 2 si 3

Asocierea TUTOR este o asociere unara deoarece la ea participa doar entitatea STUDENT. Aceasta asociere arata cine este tutorul fiecarui student (daca exista).
Asocierea INSCRIS_LA este o asociere binara intre entitatile STUDENT şi FACULTATE si arata la ce facultate/facultati este inscris fiecare student.
Asocierea ALOCARE este o asociere ternara intre entitatile STUDENT, PROIECT şi CALCULATOR. Ea modeleaza pe ce calculatoare are alocate ore de lucru fiecare student pentru fiecare proiect.

Un exemplu de asociere de grad mai mare ca trei este orarul unui an de studiu al unei facultati. Acesta este o asociere intre urmatoarele entitati:

- GRUPE. Fiecare grupa are un cod unic.
- SALI. Salile sunt etichetate printr-un indicativ alfanumeric.
- INTERVALE ORARE. Un interval orar este un triplet (Zi, De la ora, La ora)
- ACTIVITATE. Este o activitate prezenta in orar (curs, laborator, seminar, proiect).
- PROFESOR. Este cadrul didactic titular pentru o activitate

Modelarea acestor clase de obiecte ca entitati presupune faptul ca in baza de date sunt stocate si alte informatii despre ele. Diagrama EA este prezentata in figura 2.5. Ea modeleaza programarea activitatilor didactice efectuate de profesori pe intervale orare, sali si grupe.

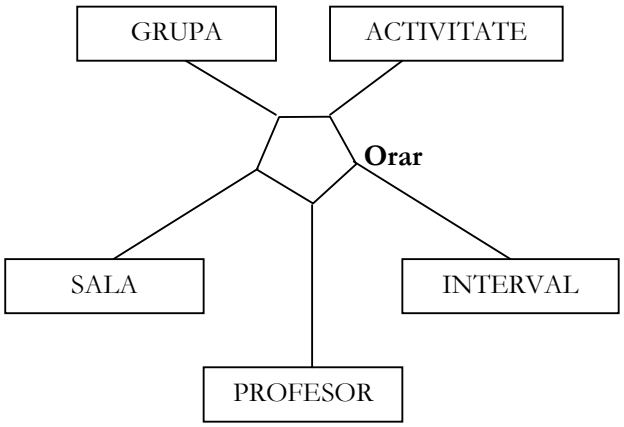


Fig. 2.5. Asociere de grad 5

Conectivitatea asocierii

Este specifica fiecarei ramuri a unei asocieri şi poate avea una din urmatoarele doua valori: **unu** sau **multi**. Determinarea ei pentru ramura spre o entitate E se face astfel: fixand arbitrar cite o instanta pentru celelalte entitati care participa la asociere se pune intrebarea: cite instante ale lui E pot fi conectate cu acestea? Daca poate fi cel mult una, conectivitatea ramurii este **unu**, altfel conectivitatea este **multi**.

Pentru exemplul din figura 2.4.:

- Asocierea TUTOR este unu-unu sau multi-uni dupa cum un student poate fi tutor pentru un singur alt student sau pentru mai multi studenti de an inferior.
- Asocierea INSCRIS_LA este multi-unu (multi spre STUDENT) sau multi-multi dupa cum un student poate fi inscris la una sau mai multe facultati.
- Asocierea ternara ALOCARE (aplicam definitia):
  - ramura spre STUDENT: fiind dat un proiect si un calculator, citi studenti au ore alocate pe acel calculator pentru respectivul proiect? Presupunand ca mai multi studenti lucreaza pentru acelasi proiect pe acelasi calculator ramura va fi ***multi***.
  - ramura spre PROIECT: fiind dat un student şi un calculator, la cite proiecte are acesta alocate ore pe acel calculator? Presupunand ca pentru fiecare proiect exista un calculator dedicat, ramura va fi ***unu***.
  - ramura spre CALCULATOR: fiind dat un student şi un proiect, pe cate calculatoare are alocate acesta ore pentru realizarea proiectului? Presupunand ca la un proiect se lucreaza pe un singur calculator, ramura va fi ***unu***.

Deci asocierea ALOCARE este multi-unu-unu.

Observam ca raspunsul la fiecare din cele trei intrebari se da in functie de realitatea modelata. Aceeasi asociere poate avea conectivitati diferite in cazuri diferite: daca exista chiar si un singur proiect la care un student are ore alocate pe mai mult de un calculator, ramura spre CALCULATOR va fi ***multi*** iar asocierea va fi multi-unu-multi.

Conventia de reprezentare grafica a conectivitatii folosita in aceasta prezentare este urmatoarea: ramurile 'unu' vor fi reprezentate sub forma de sageata. In figura 2.6. sunt prezentate cele trei asocieri avand figurata şi conectivitatea. S-a presupus ca asocierile TUTOR si INSCRIS_LA sunt unu-unu respectiv multi-unu.

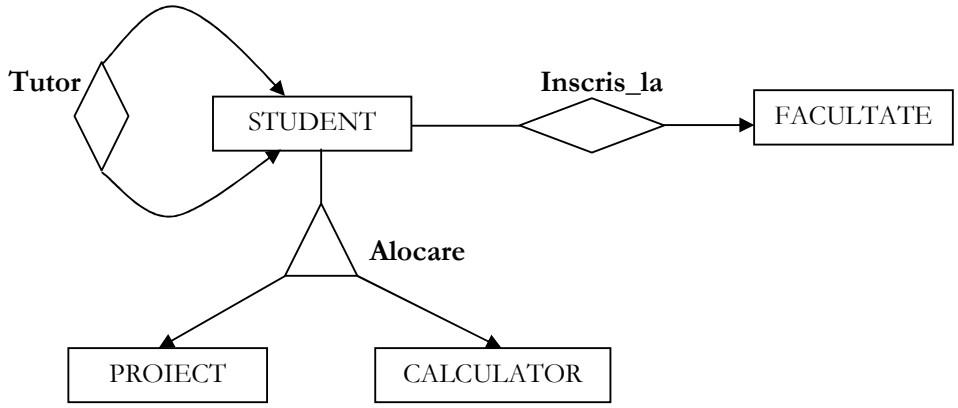


Fig. 2.6. Reprezentarea conectivitatii

Obligativitatea asocierii

Ca şi conectivitatea, aceasta se determina pentru fiecare ramura şi poate avea doar una din urmatoarele valori: **obligatorie** sau **optionala**. Determinarea ei pentru ramura spre o entitate E se face astfel: fixand arbitrar cite o instanta pentru celelalte entitati care participa la asociere se pune intrebarea: este obligatoriu sa existe o instanta a lui E asociata cu acestea? Daca raspunsul este 'Da' ramura este **obligatorie** altfel este **optionala**.

In exemplul anterior ramurile asocierilor TUTOR si ALOCARE sunt optionale iar cele ale asocierii INSCRIS_LA sunt obligatorii deoarece:

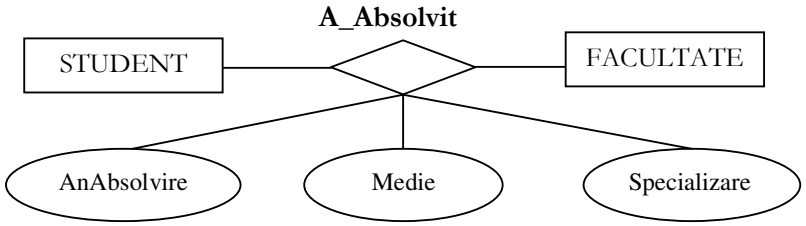
- Pentru asocierea TUTOR: exista studenti care nici nu au un tutor nici nu sunt tutori pentru alti studenti
- Pentru asocierea ALOCARE:
  - Un student la un proiect poate sa nu aiba alocate ore pe nici calculator (de exemplu in cazul unui proiect la o materie umanista)
  - Un student si un calculator respectiv un calculator si un proiect pot sa nu fie asociati prin alocare de ore de lucru (de exemplu pentru calculatoarele din birourile cadrelor didactice)
- Pentru asocierea INSCRIS_LA: nu exista studenti care nu sunt inscrisi la nici o facultate si nici facultati fara studenti inscrisi.

Conventia de reprezentare grafica a clasei de apartenenta folosita in continuare este urmatoarea: ramurile obligatorii vor fi reprezentate prin linie continua iar cele optionale prin linie intrerupta. In figura 2.7 este prezentata diagrama anterioara avand figurata si obligativitatea.

Daca gradul şi conectivitatea unei asocieri sunt folosite in proiectarea conceptuala a schemei bazei de date, obligativitatea se modeleaza pentru definirea unui criteriu de integritate specificand posibilitatea de aparitie a valorilor nule: la transformarea diagramei EA in model relational atributele tabelelor care modeleaza informatia reprezentata de asocieri pot avea sau nu valori nule dupa cum ramurile acestora sunt optionale sau obligatorii.

Atributele asocierilor

In unele cazuri o anumita informatie descriptiva nu este asociata cu o clasa de obiecte ci cu un ansamblu de clase diferite modelate fiecare prin entitati. In acest caz aceasta va fi modelata ca un atribut al asocierii dintre entitatile respective. Sa luam de exemplu cazul unei asocieri multi-multi A_ABSOLVIT intre entitatile STUDENT şi FACULTATE care contine informatii privind facultatile absolvite anterior de unii studenti.



In acest caz informatii ca anul absolvirii, media, specializarea nu pot fi conectate nici la STUDENT (pentru ca un student poate fi absolventul mai multor facultati in ani diferiti, cu medii diferite, etc.) si din motive similare nici la FACULTATE. Ele descriu asocierea unui student cu o facultate si de aceea vor fi atasate asocierii A_ABSOLVIT. Toate atributele unei asocieri sunt atribute descriptive, neexistand in acest caz un identificator al asocierii.

Rolul

In cazul in care de la o asociere pornesc mai multe ramuri catre aceeasi entitate, fiecareia dintre acestea i se poate asocia un rol. Acesta arata semnificatiile diferite pe care le are aceeasi entitate in cadrul asocierii respective.

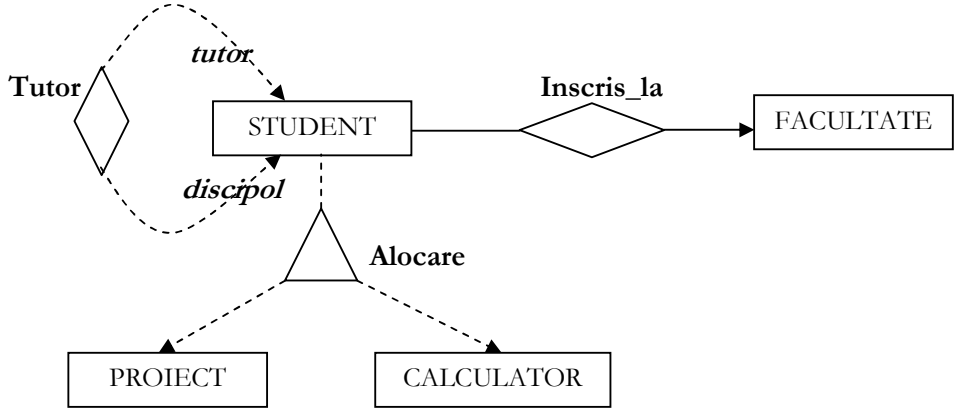


Fig. 2.7. Reprezentarea obligativitatii. Roluri

In cazul asocierii TUTOR cele doua ramuri pot fi etichetate de exemplu cu **tutor** si **discipol** aratand ca instante diferite ale aceleiasi entitati au rolurile respective (fig. 2.7.).

2.2.4. Criterii de modelare

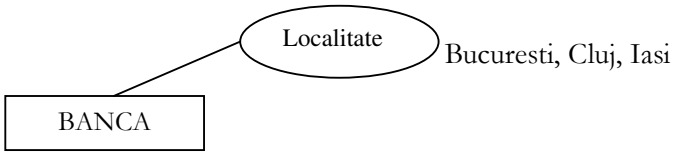
a. Clasificarea in entitati şi atribute

Desi definitia notiunilor de entitate, atribut, asociere este destul de simpla, in practica modelarii apar dificultati in clasificarea diverselor informatii intr-una din aceste categorii. De exemplu in cazul sediilor unei banci localizate in diverse orase: obiectul ORAS este entitate distincta sau atribut descriptiv al entitatii SEDIU?

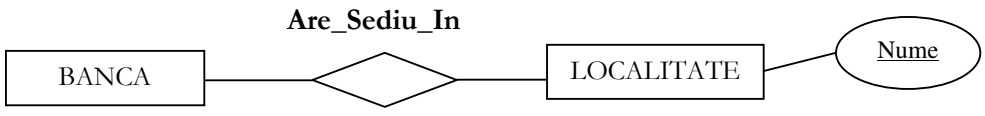
Pentru a putea clasifica corect informatiile, exista citeva reguli care trebuie respectate şi pe care le prezentam in continuare. Prima regula da un criteriu general de impartire in entitati şi atribute, urmatoarele doua semnaleaza exceptii iar ultimele doua reguli au un caracter mai putin normativ ci mai degraba orientativ.

Regula 1. Entitatile au informatii descriptive, pe cand atributele nu poseda astfel de informatii.
Daca exista informatii descriptive despre o anumita clasa de obiecte, aceasta va fi modelata ca o entitate. In cazul in care pentru pentru acea clasa de obiecte nu este nevoie decit de un identificator (codul, denumirea, etc), ea va fi modelata ca un atribut. De exemplu daca despre un ORAS este necesara stocarea in baza de date unor informatii ca JUDET, POPULATIE, etc. atunci ORAS va fi o entitate. Daca singura informatie necesara este numele sau atunci NUME_ORAS va fi un atribut al altei entitati.

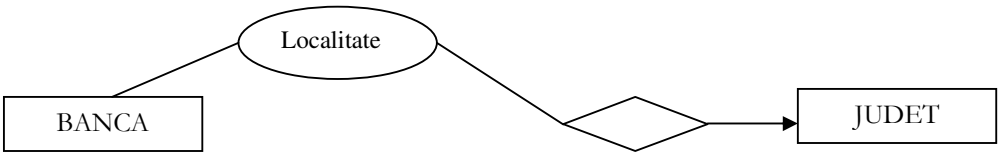
Regula 2. Atributele multivalorice vor fi reclasificate ca entitati.
Daca la o valoare a unui identificator corespund mai multe valori ale unui descriptor, acesta va fi clasat ca entitate. De exemplu, in cazul unei baze de date privind localizarea in teritoriu a unor banci, daca se memoreaza informatii doar despre banci care au un singur sediu, LOCALITATE este atribut al entitatii BANCA.



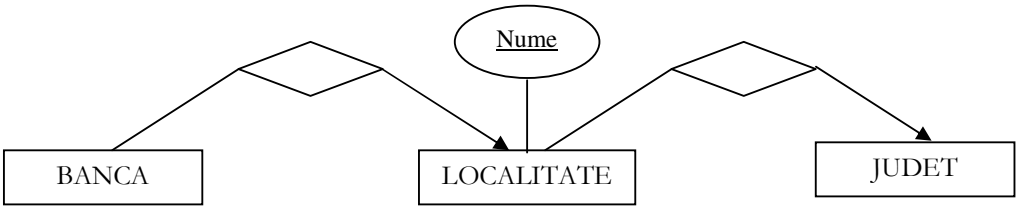
Daca insa se memoreaza informatii despre banci care au sucursale şi filiale in diverse localitati, deci pentru o singura banca (o valoare a identificatorului entitatii BANCA) avem mai multe localitati in care aceasta are sedii (mai multe valori ale descriptorului LOCALITATE), atunci LOCALITATE va fi entitate distincta desi nu are decat un singur atribut. Pentru a modela localizarea sediilor in diverse localitati intre cele doua entitati va exista o asociere binara unu-multi (unu spre BANCI) numita de exemplu ARE_SEDIU_IN.



Regula 3. Atributele unei entitati care au o asociere multi-unu cu o alta entitate vor fi reclasificate ca entitati.
Asa cum am vazut asocierile pot lega doar entitati. Daca un descriptor al unei entitati este intr-o relatie multi-unu cu o alta entitate acel descriptor va fi trecut in categoria entitatilor. De exemplu, daca avem entitatile BANCA avand ca atribut descriptiv monovaloric LOCALITATE şi JUDET, daca se doreste modelarea apartenentei la judete a localitatilor va exista o asociere multi-unu intre atributul LOCALITATE şi entitatea JUDET.



In acest caz LOCALITATE va fi reclasificata ca entitate desi nu sunt necesare alte informatii in afara numelui localitatii.



Regula 4. Atributele vor fi atasate la entitatile pe care le descriu in mod nemijlocit. De exemplu, UNIVERSITATE va fi atasat ca atribut al entitatii FACULTATE şi nu al entitatilor STUDENT sau PROFESOR.

Regula 5. Folosirea identificatorilor compusi va fi evitata. Identificatorul unei entitati este acea submultime de atribute ale acesteia care identifica in mod unic fiecare instanta a sa. In model relational pentru atributele de acest fel se construiesc de regula structuri de cautare rapida (indecsi) care functioneaza cu atat mai lent cu cat complexitatea indecsului creste. Aplicarea acestei reguli se poate face in diverse moduri:

1. Daca identificatorul unei entitati este compus din mai multe atribute care sunt toate identificatori in alte entitati, acea entitate se elimina. Informatia continuta de aceasta va fi modelata sub forma unei asocieri intre acele entitati.
2. Daca identificatorul unei entitati este compus din mai multe atribute care nu sunt toate identificatori in alte entitati, exista doua solutii:
   - Entitatea respectiva se elimina şi este inlocuita prin alte entitati si asocieri astfel incit pe ansamblu informatia modelata in varianta originara sa fie pastrata.
   - Entitatea respectiva ramine in forma originara, cu dezavantaje insa in privinta vitezei operatiilor.

Se vede ca procedura clasificarii obiectelor in entitati şi atribute este iterativa:

- se face o prima impartire conform primei reguli
- parte din atributele astfel obtinute se reclasifica in entitati conform regulilor 2 si 3
- se face o rafinare finala conform regulilor 4 si 5.

b. Identificarea ierarhiilor de generalizare şi incluziune.

In cazul in care despre anumite subclase ale unei clase de obiecte exista informatii specifice, clasa şi subclasele (care la pasul anterior au fost catalogate ca entitati) sunt interconectate intr-o ierarhie de incluziune sau generalizare, dupa cum este cazul. La acest pas se face şi o reatasare a atributelor pentru evitarea redundantei, astfel:

- La entitatea tata vor fi atasate atributele care formeaza identificatorul şi descriptorii care modeleaza informatii specifice intregii clase.
- La entitatile fiu vor fi atasate atributele de identificare (aceleasi ca ale tatalui) şi atributele care modeleaza informatii specifice doar acelei subclase de obiecte.

Sa consideram o ierarhie care imparte studentii dupa in caministi si necaministi (fig. 2.7.). In acest caz atributul NUME trebuie eliminat de la entitatea NECAMINIST deoarece el este prezent deja la tata.

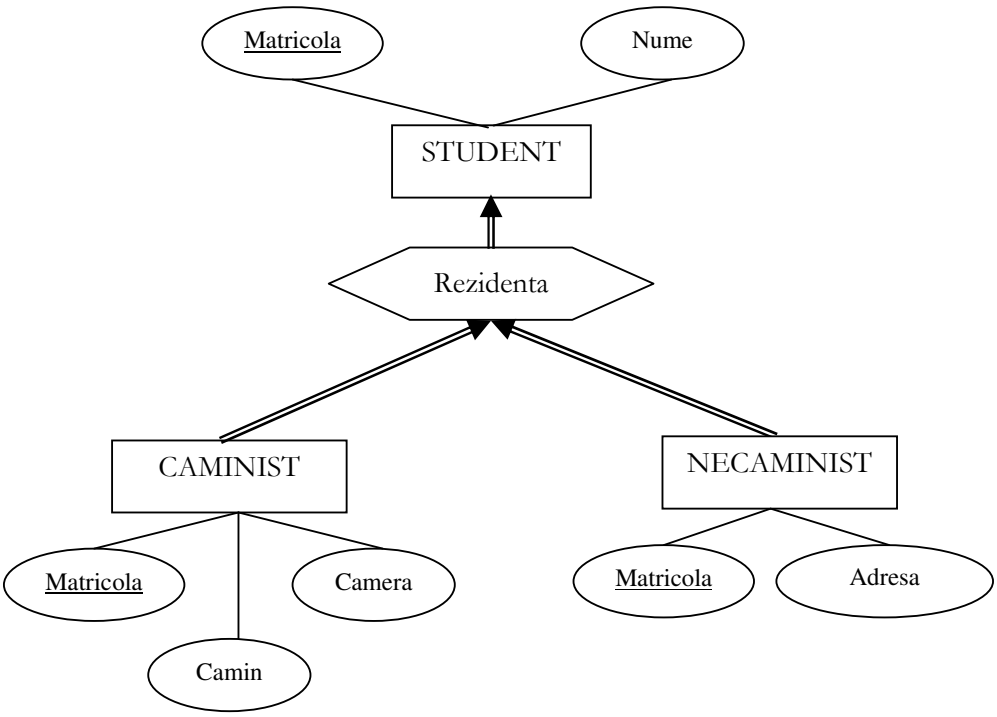


Fig. 2.8. Atributele entitatilor unei ierarhii

Rezulta urmatoarele reguli:

1. Tatal şi fii unei ierarhii au acelasi identificator.
2. Descriptorii care apar şi la tata şi la fii se elimina de la fii.
3. Descriptorii care apar la toti fii unei ierarhii de generalizare şi nu apar la tata se muta la tata.

c. Identificarea asocierilor

In aceasta etapa se trateaza informatiile care nu au fost clasificate ca entitati sau atribute ci reprezinta interdependente intre clase de obiecte. Ele sunt modelate ca asocieri intre entitati. Pentru fiecare asociere se specifica gradul, conectivitatea, obligativitatea si daca este cazul si atributele asocierii precum si rolurile ramurilor sale.

Ca şi in cazul clasificarii in entitati şi atribute, existe citeva reguli de urmat in operatia de definire a asocierilor:

Eliminarea asocierilor redundante. In cazul in care o asociere poate fi dedusa din alte asocieri deja catalogate, aceasta se elimina. De retinut ca intre doua entitati pot sa existe oricite asocieri şi ele nu sunt considerate redundante atit timp cit au semnificatie diferita. Un caz des intilnit de redundanta este cel al compunerii (tranzititatii) asocierilor. Prezentam un exemplu:

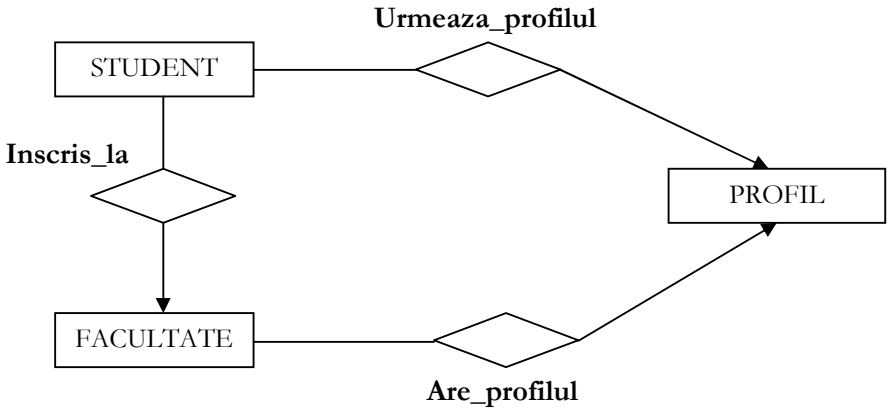


Fig. 2.9. Asocieri redundante

In acest exemplu, asocierea INSCRIS_LA modeleaza apartenenta fiecarui student la o facultate a unui institut de invatamint superior. Fiecare facultate are un profil unic descris de asocierea ARE_PROFILUL. Ambele asocieri sunt multi-unu in sensul STUDENT→FACULTATE→PROFIL. Deoarece asocierile multi-unu (ca şi cele unu-unu) sunt din punct de vedere matematic functii, din compunerea lor putem afla profilul la care este inscris fiecare student. Rezulta ca asocierea URMEAZA_PROFILUL care are chiar aceasta semnificatie este redundanta şi trebuie eliminata.

Asocieri de grad mai mare ca 2. Asocierile ternare (sau de grad mai mare ca trei) se folosesc doar atunci cand sunt strict necesare. Este de multe ori posibil ca o aceeasi informatie sa fie modelata ca o asociere ternara sau ca un ansamblu de asocieri binare si unare. In cazul acesta, este de preferat ca sa se opteze pentru a doua varianta. Doar cand asocierile binare nu pot modela intreaga semnificatie dorita se va opta pentru asocieri de grad mai mare ca doi. Aceasta cerinta deriva din faptul ca la trecerea in modelul relational asocierile de grad superior devin scheme de relatii de sine statatoare, marind numarul de tabele din baza de date pe cand cele de grad unu şi doi (cu exceptia celor multi-multi) nu au acest efect.

d. Integrarea vederilor.

In cazul proiectarii bazelor de date complexe, activitatea se desfasoara uneori de catre mai multe colective simultan, fiecare modeland o portiune distincta a bazei de date. Deoarece in final trebuie sa se obtina o singura diagrama a bazei de date, dupa terminatea modelarii pe portiuni diagramele rezultate sunt integrate eliminandu-se redundantele si inconsistentele.