Modelul Entitate-Asociere clasic

Cuprins

Introducere

Proiectarea bazelor de date relationale este in acest moment un domeniu in care cercetarile au evoluat spre elaborarea unor metodologii teoretice si a unor instrumente software practice care asigura un grad ridicat de normalizare a schemelor de relatie, pastrarea integritatii, evitarea anomaliilor datorate unei proiectari nesistematice si eliminarea subiectivismului proiectantului prin inlocuirea lui cu exactitatea metodei. Initial se pornea de la o colectie de scheme de relatii si de la dependentele functionale si multivalorice asociate si se ajungea, prin aplicarea unor algoritmi de normalizare la o schema normalizata a bazei de date. Aceasta abordare de tip bottom-up creaza insa dificultati in cazul bazelor de date complexe in care numarul de relatii si de dependente este mare. Probabilitatea ca unele interdependente intre date sa nu fie sesizate in procesul de proiectare este in aceste cazuri ridicat rezultind in exploatare anomalii care duc la reluari ale ciclului: analiza cerintelor $\rightarrow$ schema bazei de date $\rightarrow$ programe de exploatare a datelor.

Cercetarile din ultima decada s-au concentrat pe gasirea unor metodologii top-down pentru proiectarea structurii optime a bazelor de date. Ele au fost influentate de rezultatele obtinute in domenii care folosesc bazele de date: inteligenta artificiala, proiectarea asistata, orientarea pe obiecte. Impactul conceptelor de abstractizare si generalizare precum si elaborarea unui model de descriere informala a datelor, si anume modelul entitate asociere (EL), 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 si simplitatii elementelor sale. Imbunatatiri ulterioare ale sale prin folosirea abstractizarilor si generalizarilor au dus la crearea modelului entitate asociere extins (EA) care este folosit si in aceasta prezentare pentru primul pas al metodologiei descrise.

Extensii ale modelului EA au aparut si pentru alte necesitati:

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

Textul de fata isi propune in prima parte sa prezinte o sinteza integrata a unei metode folosite pentru proiectarea bazelor de date relationale, pornind de la simplu catre complex, acoperind primele doua componente ale ciclului amintit mai sus: analiza cerintelor si proiectarea structurii (schemei) bazei de date.

Capitolul 1. Analiza cerintelor si proiectarea structurii bazei de date

1.1. Etapele elaborarii modelului conceptual al datelor

Descrierea structurii unei baze de date se poate face la trei nivele, pentru fiecare putindu-se elabora o schema a bazei de date respective:

a. nivelul intern, existind o schema fizica a datelor
b. nivelul conceptual, cu schema conceptuala
c. nivelul extern, putind exista mai multe scheme externe.

Primul este singurul care are o existenta materiala, celelalte reprezentind diverse nivele de abstractizare ale acestuia (vezi fig. 1.1).

1.1.1. Nivele de reprezentare pentru baze de date

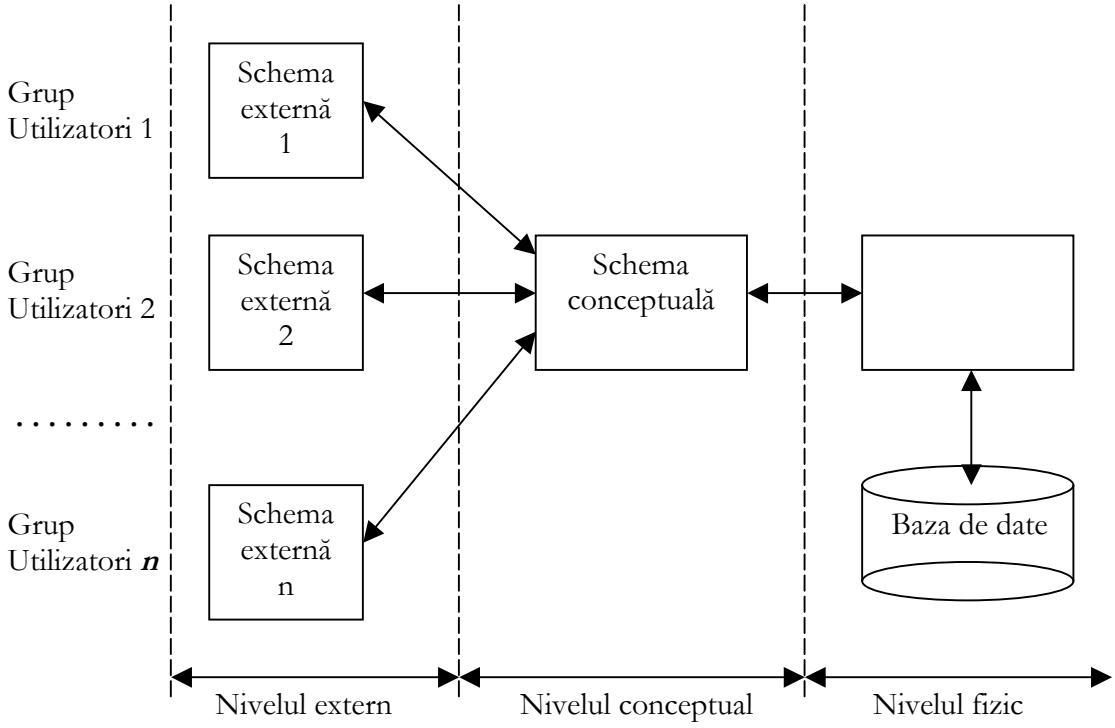


Fig. 1.1. Cele trei nivele de descriere ale unei baze de date

Schema fizica descrie cum sint stocate datele pe dispozitivele sistemului de calcul (discuri, benzi, etc). Specificarea ei se poate face la nivel foarte scazut (bit) dar de obicei este la nivel de inregistrari si fisiere.

Schema conceptuala descrie in termeni abstracti dar exacti o anumita parte a lumii reale, si anume acea parte despre care dorim sa stocam date. Procesul de trecere de la lumea reala la schema conceptuala se face prin clasificarea obiectelor de interes in categorii si asignarea unor nume atit acestor categorii cit si caracteristicilor lor. Acest proces poarta numele de modelare conceptuala.

Schemele externe corespund "ferestrelor" prin care diversele categorii de utilizatori au acces la baza de date. Ele au aparut si din necesitatea unui acces diferentiat la date, fiecare tip de utilizatori putind exploata doar acea portiune a bazei de date care este necesara in activitatea curenta fara a avea acces la intreaga baza de date. Aceste scheme mai poarta numele de vederi.

Textul de fata se ocupa de nivelul conceptual deoarece procesul de proiectare a unei noi baze de date incepe prin definirea schemei conceptuale. Schema fizica rezulta din implementarea acesteia luind in consideratie elementele de hardware si software disponibile iar schemele externe sint subscheme ale schemei conceptuale care se definesc in functie de necesitatile si drepturile de acces la date ale diverselor categorii de utilizatori.

1.1.2. Pasii proiectarii conceptuale

Pasul 1. Modelarea cerintelor folosind modelul EA.

Acest pas implica analiza segmentului din lumea reala supus modelarii si construirea diagramei EA folosind elementele modelului, descrise in subcapitolele urmatoare. Este un pas care nu poate fi automatizat dar exista reguli care ajuta proiectantului pentru o corecta incadrare a datelor elementare. Tot acum are loc si analiza prelucrarilor de efectuat asupra datelor care se specifica intr-o prima forma in limbaj natural. In cazul bazelor de date complexe modelarea se face pe portiuni, fiind urmata de integrarea vederilor rezultate intr-o singura diagrama a bazei de date. Problematica integrarii vederilor va fi discutata la un capitol ulterior.

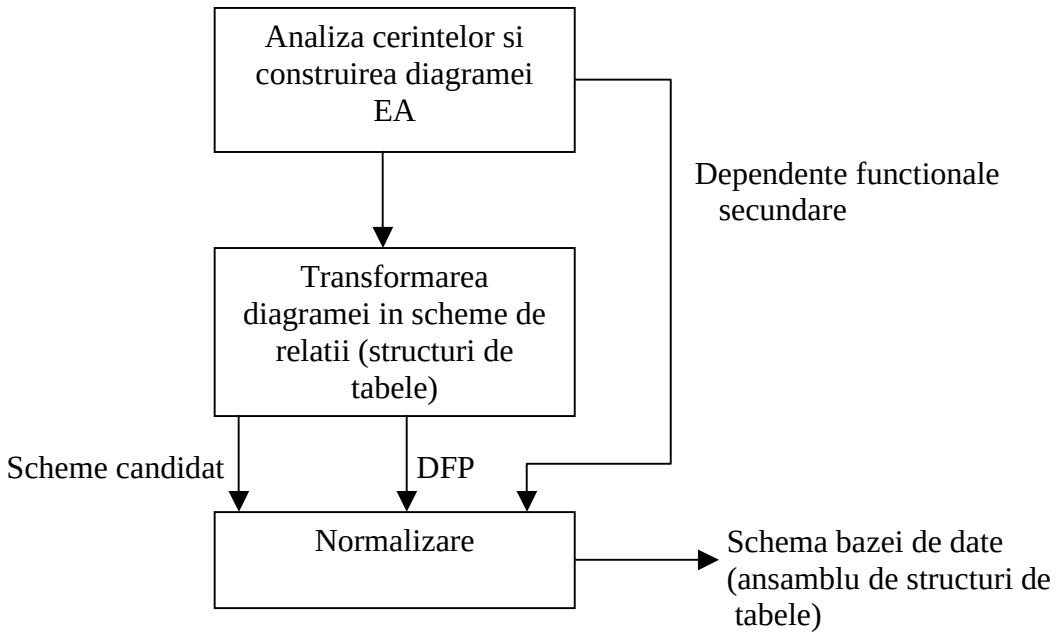


Figura 1.2. Pasii modelarii conceptuale

Pasul 2. Transformarea diagramei EA in scheme de relatii.

Pe baza unor reguli clare se transforma entitatile si asocierile care formeaza diagrama EA in scheme de relatii, rezultind schema preliminara a bazei de date si un set de dependente functionale (DF) asociate, numite dependente functionale primare (DFP). Relatiile redundante sint eliminate. Acest pas este automatizabil, existind posibilitatea scrierii de programe de transformare.

Pasul 3. Normalizarea relatiilor.

Pe baza schemelor si dependentelor functionale primare rezultate din transformare precum si a altor dependente, functionale si multivalorice, rezultate din analiza efectuata la primul pas (numite dependente secundare) se face o normalizare pina la forma normala dorita. Redundantele care eventual apar sint apoi eliminate, cu conditia pastrarii integritatii datelor. Si acest pas poate fi automatizat, existind algoritmi de normalizare bine definiti.

1.2. Modelul entitate asociere extins (EA)

1.2.1. Modelul entitate asociere clasic

Modelul entitate asociere asa cum a fost descris de catre Chen permite reprezentarea informatiilor despre structura bazelor de date folosind trei elemente de constructie: entitati, atribute ale entitatilor si asocieri intre entitati, numite in continuare asocieri. Definitia lor informala este urmatoarea:

Entitatile modeleaza clase de obiecte concrete sau abstracte despre care se colecteaza informatii, au existenta independenta si pot fi identificate in mod unic. Exemple de entitati: studenti, orase, salariati, 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 sint de doua categorii: ***entitati independente (sau tari)*** au existenta independenta de alte entitati pe cind cealalta categorie, ***entitati dependente (sau slabe)*** sint formate din instante care isi justifica incadrarea in clasa respectiva doar atita timp cit intr-o alta entitate (tata) exista o anumita instanta. In cazul unei baze de date de personal, fiecare instanta a entitatii COPII ramine in clasa respectiva (copiii salariatilor unitatii) atit timp cit in entitatea SALARIATI exista instanta reprezentind pe tatal acelui copil.

Atributele modeleaza proprietati distincte ale entitatilor. De exemplu entitatea SALARIATI are ca atribute MARCA, NUME, PRENUME, VIRSTA, FUNCTIA si SALARIUL. In procesul de modelare vor fi luate in considerare doar acele proprietati ale entitatilor care sint semnificative pentru aplicatia respectiva. Din acest motiv, la entitatea SALARIATI nu vom lua in considerare caracteristici ca TALIA, GROSIMEA, NUMAR_INCALTAMINTE ca fiind nesemnificative pentru o baza de date de personal (ele ar putea figura insa in cazul unei baze de date privind personalul militar).

Atributele unei entitati sint de doua feluri: ***atributele de identificare*** (formind impreuna identificatorul entitatii) sint acea multime de atribute care permit distinctia intre instantele aceleiasi entitati si ***atributele de descriere*** (sau descriptori) care sint folositi pentru memorarea caracteristicilor suplimentare ale instantelor. In exemplul de mai sus MARCA este atribut de identificare (deoarece nu pot exista doi salariati cu aceeasi marca intr-o intreprindere) pe cind celelalte atribute sint descriptori.

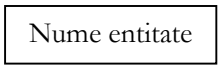
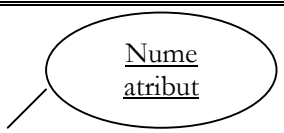
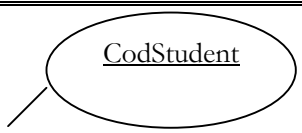
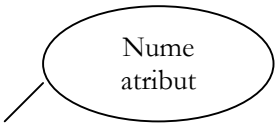
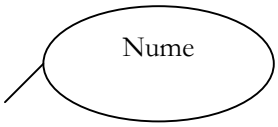
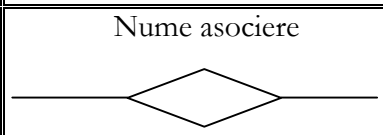
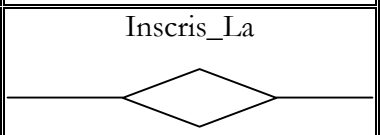
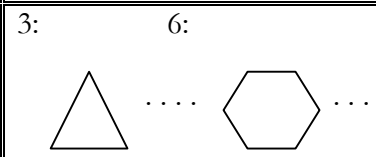
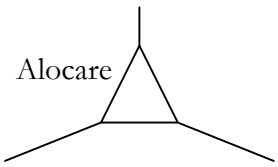
Element al modelului	Tip	Reprezentare	Exemplu
Entitate	Tare		
	Slaba		
Atribut	De identificare		
	De Descriere		
Asociere	Asociaza 1-2 entitati		
	Asociaza mai mult de 2 entitati (exemplu: 3 entitati)		

Fig. 1.3. Conventia de reprezentare a elementelor modelului EA

Asocierile modeleaza interdependentele dintre clasele de obiecte reprezentate prin entitati. De exemplu intre entitatile SALARIATI si SECTII poate figura o asociere LUCREAZA_IN care descrie arondarea salariatilor la sectii. Aceeasi observatie ca mai sus: nu vor fi luate in consideratie decit interdependentele care sint necesare aplicatiei respective, in lumea reala putind exista intre entitatile analizate si alte asocieri care nu sint semnificative pentru acea aplicatie.

Conventia de reprezentare grafica a acestor trei tipuri de constructii care participa la formarea unei diagrame EA este in prezentarea de fata urmatoarea (vezi si figura 1.3.):

a. Entitatile se reprezinta prin dreptunghiuri in care este inscris numele entitatii. In cazul entitatilor dependente, conturul va fi cu linie dubla.

b. Atributele se reprezinta prin cercuri (sau ovale) in interiorul carora apare numele atributului si care sint conectate cu un segment de dreapta cu entitatea de care apartin. Pentru a distinge atributele de identificare de cele de descriere, numele primelor va fi subliniat.

c. 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 pe care le leaga, avind inscris in interior (sau alaturi) numele asocierii. Alte elemente grafice specificind caracteristici suplimentare ale asocierilor vor fi introduse in subcapitolele urmatoare.

1.2.2. Modelul entitate asociere extins

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 si ierarhia de incluziune. Prima defineste partitionarea instantelor unei entitati in n subclase diferite iar a doua permite clasarea instantelor unei entitati in n subclase care nu reprezinta o partitie in sens matematic. Din punct de vedere formal, cele doua noi concepte se pot defini astfel:

Definitia ierarhiei de incluziune: O entitate E1 este o submultime a entitatii E2 (sau este inclusa in entitatea E2) daca fiecare instanta a lui E1 este de asemenea o instanta a lui E2.

Un exemplu de incluziune este definirea in cadrul entitatii SALARIATI a unor subclase modelate prin entitatile MEMBRII_DE_SINDICAT, CU_STUDII_SUPERIOARE si CU_COPII. Caracteristica ierarhiei de incluziune specifica deci ca entitatile fiu pot sa nu fie disjuncte doua cite doua (pentru exemplul dat, exista salariati cu studii superioare care au copii, etc) iar reuniunea lor poate sa nu acopera in intregime entitatea tata (exista salariati care nu sint nici membri de sindicat, nu au nici studii superioare si nici copii).

Definitia ierarhiei de generalizare: O entitate E este generalizarea entitatilor E1, E2, ..., En daca orice instanta a lui E este de asemenea instanta in una si numai una din entitatile E1, E2, ..., En.

Un exemplu de generalizare este clasarea instantelor entitatii SALARIATI in subclasele BARBATI si FEMEI. Caracteristica ierarhiei de generalizare in sens este deci ca din punct de vedere matematic ea reprezinta o partitie:

a. E1 U E2 U ... U En = E
b. Ei n Ej = 0 pentru orice i <> j din intervalul 1..n

Ierarhiile de incluziune si generalizare se folosesc doar in cazul in care pentru subclase ale unor clase modelate prin entitati este nevoie de stocarea unor informatii specifice.

In cazul unei baze de date de personal este nevoie de exemplu de stocat numarul de copii pentru plata alocatiei. Acest fapt se poate modela in doua feluri: fie prin adaugarea la entitatea SALARIATI a unui atribut suplimentar (care va avea valoarea 0 pentru salariatii fara copii) fie prin aparitia unei entitati suplimentare CU_COPII aflata intr-o relatie de incluziune cu entitatea SALARIATI si care va avea ca atribute de identificare pe cele ale tatalui (MARCA de exemplu) iar ca atribut descriptiv numarul de copii (deci atributul/atributele specifice subclasei).

Vom alege a doua varianta in cazul in care numarul salariatilor cu copii este mult mai mic decit numarul total al salariatilor, fapt care va duce la economie de spatiu pe disc: se economiseste spatiul alocat atributului continind numarul de copii care, in ipoteza de mai sus, este in cele mai multe cazuri egal cu valoarea 0.

Conventia grafica de reprezentare a celor doua tipuri de ierarhii se gaseste in fig. 1.4.

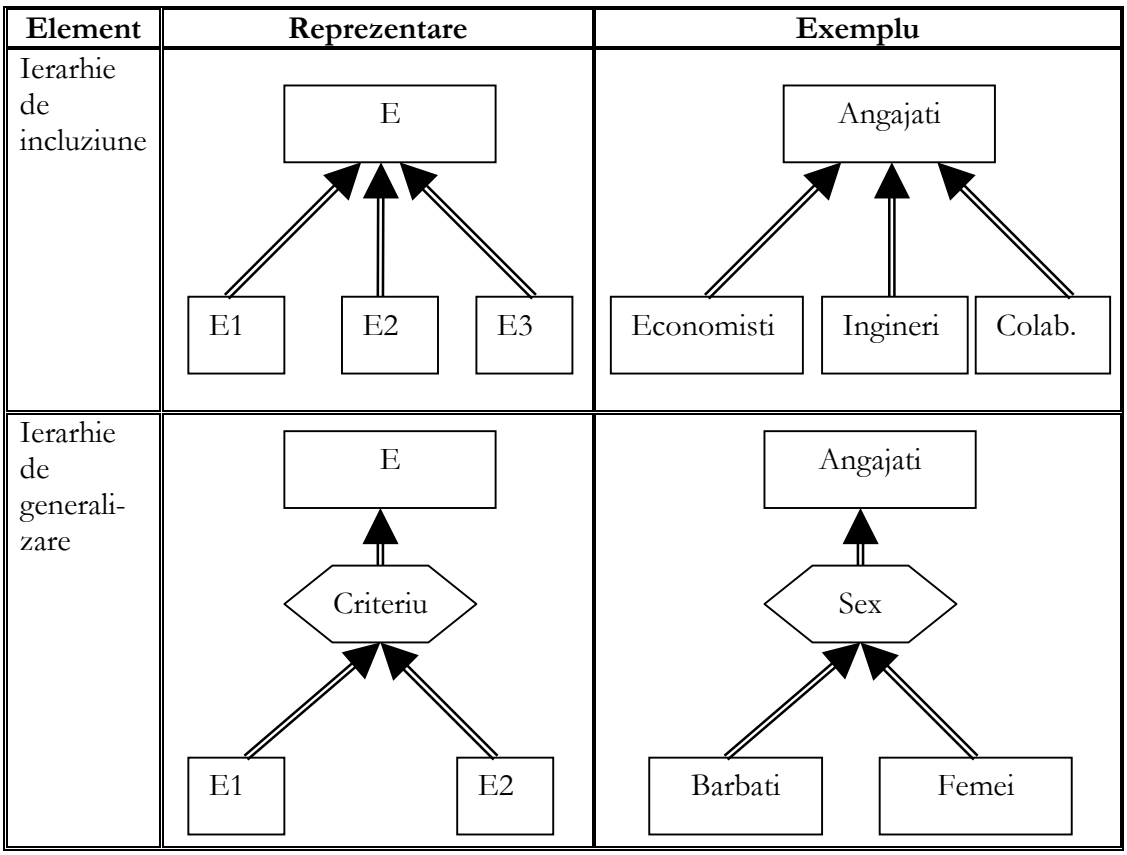


Fig. 1.4. Conventia de reprezentare grafica a ierarhiilor

1.2.3. Caracteristici ale elementelor modelului

Asa cum entitatile au atribute care specifica diversele caracteristici ale clasei de obiecte modelate, si asocierile care reprezinta interdependente intre entitati au caracteristici care aduc informatii suplimentare. Aceste caracteristici sint urmatoarele:

1. ***Gradul asocierii*** este dat ne numarul de entitati care participa la acea asociere. In aceasta prezentare ne vom concentra pe asocierile de grad 1, 2 si 3 dar vor fi abordate si asocierile de grad superior.

Asocierile de grad 1 se mai numesc si ***asocieri unare***. Ele au ca participanta o singura entitate. Reprezentarea lor grafica este un romb conectat prin doua ramuri la o aceeasi entitate. Un exemplu de astfel de asociere este CASATORIT_CU la care participa entitatea SALARIATI si care modeleaza informatii privind casatoriile intre salariatii din baza de date respectiva.

Asocierile de grad doi se mai numesc ***asocieri binare***. La aceste asocieri participa doua entitati distincte. Un exemplu este asociere LUCREAZA_IN intre entitatile SALARIATI si SECTII si care specifica sectia in care este incadrat fiecare salariat.

Asocierile de grad 3 se mai numesc si ***asocieri ternare***. La acestea participa trei entitati distincte. Un exemplu este asociere FOLOSESTE intre entitatile SALARIATI, UNELTE si PIESE care modeleaza ce unelte foloseste fiecare salariat pentru realizarea pieselor pe care le lucreaza in cadrul obligatiilor de serviciu.

Reprezentarea grafica a celor trei exemple de mai sus este data in figura 1.5. Pentru simplificarea figurii, nu s-au reprezentat decit entitatile si asocierile dintre ele si nu si atributele fiecarei entitati in parte.

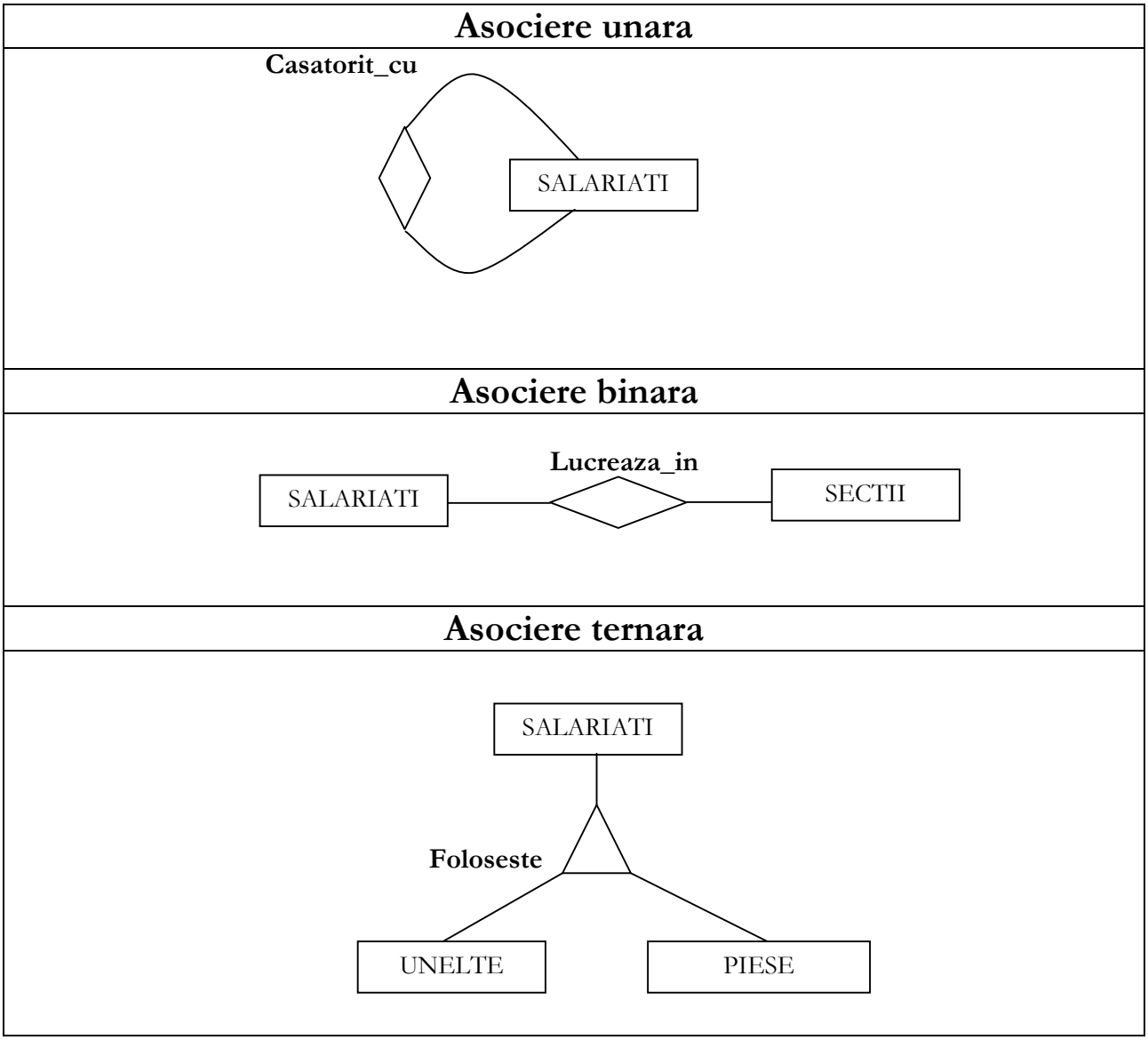


Fig. 1.5. Exemple de asocieri de grad 1, 2 si 3

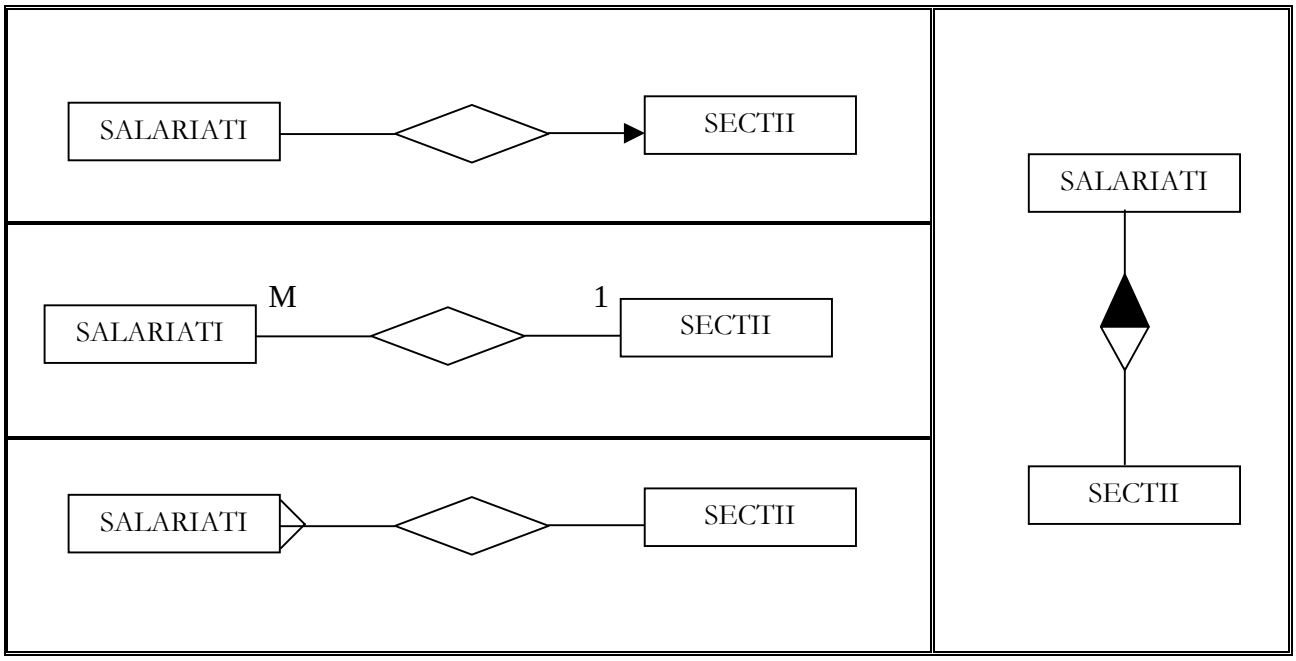
Pentru reprezentarea asocierilor avind crad mai mare ca 3, se folosesc poligoane regulate avind numarul de laturi egal cu gradul asocierii.

2. ***Conectivitatea asocierii*** este specifica fiecarei ramuri a unei asocieri si poate avea una din urmatoarele doua valori: unu sau multi.

Determinarea ei pentru ramura spre Ei se face astfel: fixind arbitrar cite o instanta pentru toate entitatile cu exceptia lui Ei, se pune intrebarea cite instante ale lui Ei sint in interdependenta cu acestea:

- daca poate fi cel mult una, conectivitatea este unu
- daca pot fi mai multe, conectivitatea este multi

Exista mai multe conventii de reprezentare grafica a conectivitatii :



Pentru exemplele din figura 1.5., asociere CASATORIT_CU este unu-unu, asociere LUCREAZA IN este multi-unu (multi spre Salariati, unu spre SECTII).

Determinarea conectivitatii fiecarei ramuri se face conform definitiei. Sa determinam conectivitatea fiecarei ramuri din asociere ternara FOLOSESTE:

- ramura spre SALARIATI: fiind data o piesa si o unealta, citi salariati lucreaza cu acea unealta pentru a realiza acea piesa? De exemplu, in cazul in care fiecare salariat are propria trusa de unelte, raspunsul este: doar unul.
- ramura spre UNELTE: fiind dat un salariat si o piesa, cite unelte foloseste acesta pentru realizarea piesei respective?
  Analizind fiecare caz in parte, raspunsul poate fi de exemplu:

  mai multe (desi pot exista piese care cer o singura unealta, daca pentru altele sint necesare mai multe acesta este raspunsul corect).
- ramura spre PIESE: fiind dat un salariat si o unealta, la relizarea citor piese foloseste acel salariat acea unealta? Sa presupunem ca in cazul luat in discutie raspunsul este: mai multe.

Observam deci ca raspunsul la fiecare din cele trei intrebari se da in functie de realitatea modelata. Aceeasi asociere poate avea conectivitati diferite in cazuri diferite (de exemplu in cazul unei alte sectii sau unitati economice).

In cazul de mai sus vom spune ca asociere este unu-multi-multi (unu spre salariati).

In figura 1.6. sint prezentate cele trei asocieri avind reprezentata si conectivitatea.

3. ***Obligativitatea***, ca si conectivitatea, se determina pentru fiecare ramura si poate avea doar una din urmatoarele valori: obligatorie sau optionala.

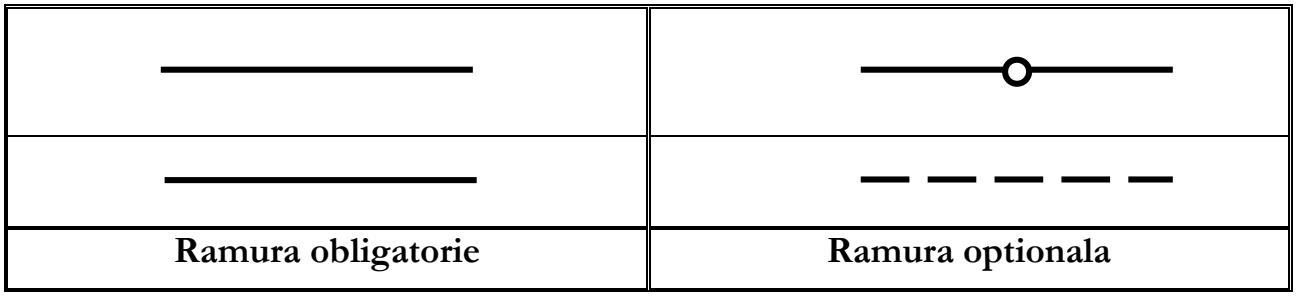
Determinarea ei pentru ramura spre Ei se face astfel: fixind arbitrar cite o instanta pentru toate entitatile cu exceptia lui Ei, se pune intrebarea daca este obligatoriu sa existe o instanta a lui Ei in interdependenta cu acestea:

- daca poate sa nu fie nici una, ramura respectiva este optionala

- daca intotdeauna exista o astfel de instanta, ramura este obligatorie

De exemplu, asociere CASATORIT_CU este optionala intrucit pot sa existe salariati care fie nu sint casatoriti fie sint casatoriti cu persoane din afare realitatii modelate (intreprinderea pentru care se proiecteaza baza de date). Asociere LUCREAZA_IN este obligatorie pe ambele ramuri (in general nu pot exista salariati fara sectie si nici sectie fara salariati). Si in acest caz raspunsul la fiecare intrebare se da in functie de realitatea modelata.

Exista mai multe conventii de representare grafica a obligativitatii:



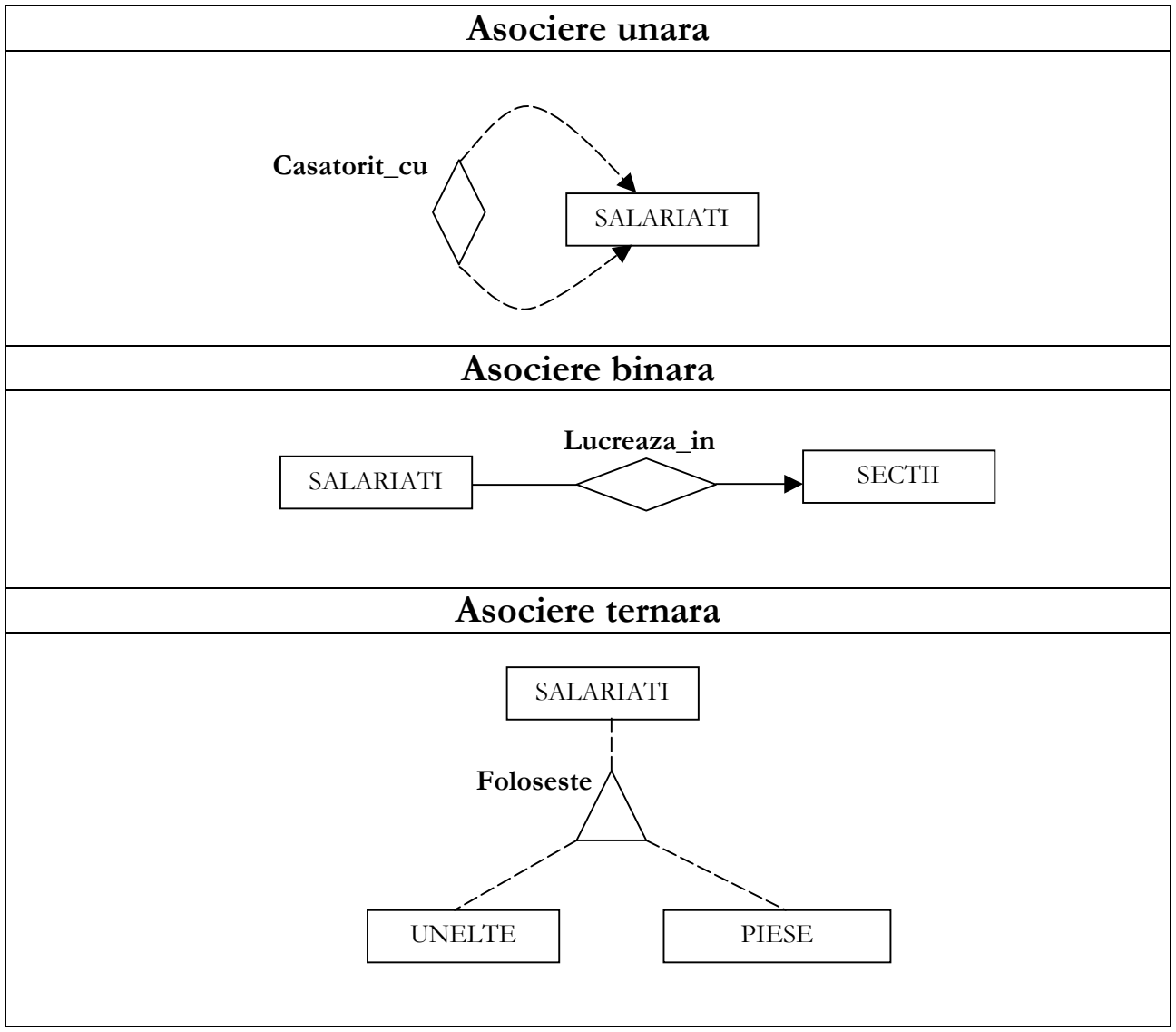


Fig. 1.6. Exemplu de reprezentare grafica a conectivitatii si obligativitatii

Daca gradul si conectivitatea unei asocieri sint folosite in proiectarea conceptuala a schemei bazei de date, obligativitatea se modeleaza pentru definirea unui criteriu de integritate cunoscut si sub numele de posibilitatea de aparitie a valorilor nule. Vom vedea mai tirziu ca in cazul cheilor straine aparute in schemele de relatii dupa transformare si care provin din ramuri optionale ale asocierilor, sint permise valori de NULL, pe cind in cazul celor provenite din ramuri obligatorii nu sint permise astfel de valori.

In figura 1.6. am figurat ca optionale toate ramurile asocierii FOLOSESTE: intr-adevar, avind dat un salariat si o unealta, el poate sa nu lucreze cu acea unealta (care poate nu este in dotarea sa) la nici o piesa. De exemplu directorul, care este si el unul dintre salariati, nu foloseste strungul din sectia de scularie pentru a realiza vreo piesa. Si la celelalte doua intrebari se raspunde analog.

1.3. Modelarea cerintelor folosind modelul EA (pasul 1)

Obiectivele acestui pas sint urmatoarele:

1. Determinarea datelor care trebuiesc stocate in contextul aplicatiei respective.
2. Descrierea informatiilor necesare despre obiectele respective si asocierile intre diversele clase de obiecte.
3. Determinarea tipurilor de prelucrari cere se vor executa asupra bazei de date.

Modelul EA este folosit pentru descrierea claselor de obiecte si a asocierilor dintre acestea. Prelucrarile asupra bazei de date se specifica folosind elementele prezentate in partea a treia a cursului si nu fac obiectul acestui capitol. Datorita posibilitatilor sale de a transmite informatii despre semantica datelor modelate, modelul EA este foarte potrivit pentru partea de proiectare a structurii bazei de date. Metodologia simpla si algoritmizabila prezentata in capitolele urmatoare duce la obtinerea de scheme de relatii in forma normala dorita. Existenta ierarhiilor de generalizare si incluziune este utila in modelarea vederilor utilizator (schema externa) iar determinarea inca din aceasta faza a identificatorilor entitatilor si a obligativitatii ramurilor asocierilor foloseste pentru stabilirea diverselor constringeri de integritate in faza transpunerii schemei conceptuale in schema fizica a bazei de date.

1.3.1. Algoritm si criterii de modelare

Pasul 1.1. Clasificarea in entitati si atribute

Desi definitia notiunilor de entitate, atribut, asociere este destul de simpla, in practica modelarii apar dificultati in clasificarea diverselor informatii in una din aceste categorii. De exemplu, sediile unei banci sint localizate in diverse orase. Obiectul ORAS este entitate distincta sau atribut descriptiv al entitatii SEDIU? Pentru a putea clasifica corect informatiile, exista citeva principii care trebuie respectate si pe care le prezentam in continuare. Prima regula da un criteriu general de impartire in entitati si atribute, urmatoarele doua semnaleaza exceptii de la prima regula iar ultimele doua reguli au un caracter mai putin normativ ci mai degraba orientativ.

a. Entitatile au informatii descriptive, pe cind atributele nu poseda astfel de informatii. Deci daca exista informatii descriptive despre o anumita clasa de obiecte, aceasta va fi modelata ca o entitate. Daca pentru pentru acea clasa de obiecte nu este nevoie decit de un identificator, ea va fi modelata ca un atribut. Daca despre un ORAS este necesara cunoasterea (si memorarea) unor informatii ca JUDET, RESEDINTA, POPULATIE, atunci

ORAS va fi o entitate. Daca singura informatie necesara despre oras este numele sau, NUME_ORAS va fi un atribut al altei entitati.

b. 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 localizare in teritoriu a bancilor, daca se memoreaza informatii doar despre banci care au un singur sediu, LOCALITATE este atribut al entitatii BANCI. Daca insa se memoreaza informatii despre banci care au sucursale si filiale in diverse localitati, deci pentru o singura banca (o valoare a identificatorului entitatii BANCI) avem mai multe localitati in care acea banca are sedii (deci mai multe valori ale descriptorului LOCALITATE), atunci LOCALITATE va fi entitate distincta. Bineinteles, intre cele doua entitati va exista o asociere binara unu-multi (unu spre BANCI) numita de exemplu ARE_SEDIU_IN.

c. Atributele care au asocieri multi-unu cu o entitate sint reclasificate ca entitati. Deci 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 entitatea BANCI cu identificator COD_BANCA si descriptori LOCALITATE si PROFIL si mai exista si entitatea JUDET, deoarece intre descriptorul LOCALITATE si entitatea JUDET exista o asociere multi-unu, LOCALITATE va fi reclasificata ca entitate.

d. Atributele vor fi atasate la entitatile pe care le descriu in mod nemijlocit. De exemplu, MINISTER_COORDONATOR va fi atasat ca atribut al entitatii UNITATE_ECONOMICA si nu al entitatilor SECTIE sau SALARIAT.

e. Folosirea identificatorilor compusi va fi evitata cit mai mult posibil. Am vazut ca identificatorul este acea submultime de atribute ale unei entitati care identifica in mod unic fiecare instanta a sa. Aceasta regula specifica necesitatea ca identificatorii entitatilor sa fie, pe cit posibil, formate dintr-un singur atribut. Respectarea ei se poate face in diverse moduri:
   1. daca identificatorul unei entitati este compus din mai multe atribute care toate sint identificatori si in alte entitati, acea entitate se elimina. Informatia modelata de ea se va regasi la pasul 1.3. sub forma unei asocieri intre aceste alte entitati.
   2. daca identificatorul unei entitati este compus din mai multe atribute care nu sint toate identificatori si in alte entitati, exista doua solutii:
      2.1. entitatea respectiva se elimina si se definesc noi entitati care au ca identificatori elementele componente ale identificatorului compus, urmind ca la pasul 1.3. intre acestea sa se evidentieze asocieri astfel incit pe ansamblu informatia modelata in varianta originara sa fie pastrata.
      2.2. entitatea respectiva ramine in forma originara, cu dezavantaje insa in privinta vitezei operatiilor: identificatorii entitatilor se vor regasi in proiectarea fizica sub forma cheilor primare sau indecsilor, ori folosirea unui index compus duce la timpi de cautare mai mari decit in cazul indecsilor atomici.

Se vede deci ca procedura clasificarii obiectelor in entitati si atribute este iterativa: se face o prima impartire conform regulii a., dupa care o parte din atributele astfel obtinute se reclasifica in entitati conform regulilor b. si c., dupa care se face o rafinare finala conform regulilor d. si e.

Pasul 1.2. Identificarea ierarhiilor de generalizare si incluziune.

In cazul in care despre anumite subclase ale unei clase de obiecte exista informatii specifice, clasa si subclasele (care la pasul 1.1. au fost catalogate ca entitati) sint interconectate intr-o ierarhie de incluziune sau generalizare, dupa cum este cazul. La acest pas se face si o reatasare a atributelor pentru evitarea redundantei, astfel:

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

De aici rezulta trei reguli:

1. Tatal si fii unei ierarhii au acelasi identificator
2. Descriptorii care apar si la tata si la fii se elimina de la fii
3. Descriptorii care apar la toti fii unei ierarhii de generalizare si nu apar la tata se adauga la tata si apoi se aplica regula 2.

Pasul 1.3. Definirea asocierilor

La acest pas se trateaza informatiile care nu au fost clasificate ca entitati sau atribute ci reprezinta asocieri, interdependente intre clase de obiecte. Ele sint modelate ca asocieri intre entitati. Pentru fiecare asociere se specifica gradul, conectivitatea si obligativitatea.

De asemenea, atunci cind este cazul, se specifica atributele asocierii. In cele prezentate mai sus nu am specificat explicit ca o asociere poate avea atribute proprii, dar acest lucru este posibil. Atributele asocierilor specifica informatii descriptive nu despre o clasa de obiecte ci despre asocierea intre obiecte din mai multe clase.
Sa luam de exemplu cazul asocierii multi-multi A_LUCRAT_IN intre entitatile SALARIATI si SECTII. Aceasta modeleaza informatii despre sectiile in care, in decursul timpului, a lucrat fiecare salariat (si respectiv despre salariatii care au lucrat in decursul timpului in fiecare sectie).
In cazul in care se doreste memorarea in baza de date a perioadei in care fiecare angajat a lucrat in diverse sectii, asociere va avea doua atribute: DATA_INCADRARE si DATA_PLECARE care vor memora datele calendaristice la care un angajat a venit si respectiv a plecat dintr-o sectie. Aceste doua atribute nu pot fi atasate nici la salariat (deoarece acesta a trecut prin mai multe sectii si ar fi deci atribute multivalorice) nici la sectie (deoarece intr-o sectie au lucrat in decursul timpului un numar mare de salariati, deci din nou ar fi atribute multivalorice). Ele se ataseaza la asociere A_LUCRAT_IN deoarece ele descriu nu un salariat sau o sectie ci asocierea dintre un salariat si o sectie.

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

a. 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 si ele nu sint considerate redundante atit timp cit au semnificatie diferita.

Un caz des intilnit de redundanta este cel al compunerii (tranzititatii) asocierilor. Prezentam in fig. 1.6. un exemplu:

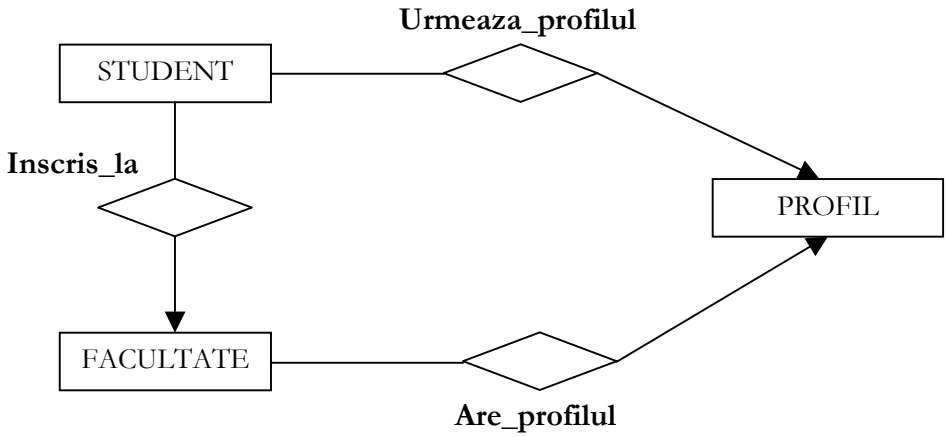


Fig. 1.7. Asocieri redundante

In acest exemplu, asociere INSCRIS_LA modeleaza apartenenta fiecarui student la o facultate a unui institut de invatamint superior. Fiecare facultate are un unic profil, si aceasta informatie (arondarea facultatilor pe profile) este modelata de asociere ARE_PROFILUL. Ambele asocieri sint multi-unu in sensul parcurgerii sirului STUDENT--(m-u)--FACULTATE--(m-u)--PROFIL. Deoarece asocierile multi-unu (ca si cele unu-unu) sint din punct de vedere matematic functii, din compunerea lor rezulta ca putem afla profilul la care este inscris fiecare student. De aici rezulta ca asociere URMEAZA_PROFILUL care are chiar aceasta semnificatie este redundanta si trebuie deci eliminata.

b. Asocierile ternare (sau de grad mai mare ca trei) trebuiesc folosite doar cind sint strict necesare. Este de multe ori posibil ca o aceeasi informatie sa fie modelata cu o asociere ternara sau cu mai multe asocieri binare. In cazul acesta, este de preferat ca sa se opteze pentru a doua varianta. Doar cind asocierile binare nu pot captura intreaga semnificatie dorita se va opta pentru asocieri de grad mai mare ca doi. Aceasta cerinta deriva din faptul ca asocierile de grad superior se transforma (la pasul 2) in scheme de relatii de sine statatoare, marind deci numarul de scheme al bazei de date, pe cind cele de grad unu si doi (cu exceptia celor multi-multi) nu au acest efect.

Pasul 1.4. Integrarea vederilor.

In cazul proiectarii bazelor de date complexe, activitatea se desfasoara uneori de catre mai multe colective simultan, fiecare modelind o portiune distincta a bazei de date. Dupa terminatea modelarii pe portiuni, deoarece trebuie ca in final sa se obtina o singura schema de baza de date, diagramele rezultate trebuiesc integrate avindu-se grija sa nu apara probleme de redundanta sau inconsistenta.

Problematica integrarii vederilor este deosebit de complexa si nu face obiectul acestui capitol.

1.4. Transformarea diagramei EA in schema bazei de date (pasul 2)

Dupa parcurgerea tuturor etapelor pasului 1, se obtine diagrama EA a bazei de date. Aceasta diagrama se foloseste in continuare pentru obtinerea schemelor de relatie, conform metodelor expuse in continuare.

De asemenea, aceasta diagrama, prin calitatile sale deosebite de inteligibilitate si capturarea de informatii privind semantica bazei de date, poate fi folosita si pentru activitatea de

documentare a aplicatiei: ea poate fi prezentata in manualul de utilizare al aplicatiei respective usurind comunicarea unor aspecte importante de structura beneficiarului aplicatiei

1.4.1. Reguli de transformare

In procesul de transformare vom pleca de la o diagrama EA si vom obtine trei tipuri de scheme de relatie:

a. Relatii provenite din entitati. Ele contin aceleasi informatii ca si entitatile din care au rezultat.
b. Relatii provenite din entitati si care contin chei straine.
   Ele contin pe linga informatiile provenite din entitatile din care au rezultat si atribute care in alte entitati sint identificatori. Este cazul acelor entitati care au asocieri multi-unu si partial din cele care au asocieri unu-unu cu alte entitati.
c. Relatii provenite din asocieri. Este cazul celor care apar din transformarea asocierilor binare multi-multi si a asocierilor de grad mai mare ca doi. Ele contin ca atribute reuniunea identificatorilor entitatilor asociate plus atributele proprii ale asocierilor.

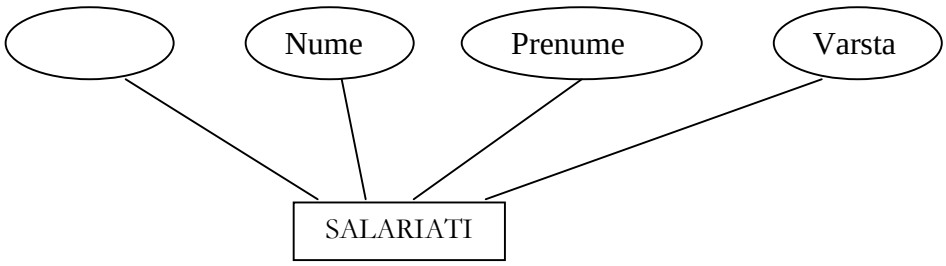
Procesul de transformare are un algoritm foarte precis si este din aceasta cauza pasul care se preteaza cel mai bine pentru crearea de instrumente software care sa-l asiste.

Pasul 2.1. Transformarea entitatilor

Fiecare entitate a diagramei se transforma intr-o schema de relatie avind:

Numele relatiei	=	Numele entitatii
Atributele relatiei	=	Atributele entitatii
Cheia relatiei	=	Identificatorul entitatii

Exemplu: Fie entitatea SALARIATI de mai jos:



Marca = identificator
Nume, Prenume, Varsta = descriptori

Schema rezultata este SALARIATI(Marca, Nume, Prenume, Varsta)

Pasul 2.2. Transformarea asocierilor unare si binare unu-unu si multi-unu

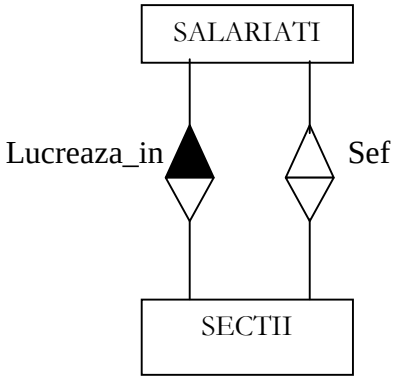
Fiecare asociere din aceasta categorie va avea ca rezultat imbogatirea multimii de atribute descriptive ale uneia dintre cele doua scheme rezultate la pasul 2.1 din entitatile asociate, cu cheia celeilalte scheme. Aceste atribute care se adauga sint denumite in prezentarea de fata cheie straina deoarece ele sint cheie dar in alta schema de relatie.

a. In cazul asocierilor multi-unu, se adauga identificatorul entitatii unu in schema rezultata din entitatea multi

b. In cazul asocierilor unu-unu, se adauga identificatorul unei entitati in schema rezultata din transformarea celeilalte. Alegerea schemei in care se face adaugarea se poate face dupa doua criterii:
   - fie in acea schema care defineste relatia cu cele mai putine tuple din cele doua,
   - fie pastrindu-se, daca exista, filiatia naturala intre cele doua entitati: identificatorul tatalui se adauga la fiu.

In cazul acestui tip de asocieri, la schema de relatie care primeste cheia straina se ataseaza una sau doua dependente functionale primare (vezi tabelul 1.1.)

Exemplu:



Intre entitatile SALARIATI si SECTII exista doua asocieri: una unu-unu, SEF, care modeleaza faptul ca o sectie este condusa de un salariat (seful de sectie) si una multi-unu, LUCREAZA_IN descrisa in paragrafele anterioare.

Rezultatul transformarii este cel de mai jos. Atribulele aflate dupa punctele se suspensie sint cele adaugate (chei straine).

SALARIATI (Marca, Nume, Prenume, Virsta, ..., Cod_Sectie)
SECTII(Cod_Sectie, Profil, ..., Marca_Sef)

Din asociere LUCREAZA_IN

Din asociere SEF

Pe cimpul Cod_Sectie din relatia SALARIATI se va inregistra pentru fiecare salariat codul sectiei in care acesta lucreaza iar pe cimpul Marca din relatia SECTII se va inregistra pentru fiecare sectie marca sefului de sectie. Pentru asociere SEF s-a aplicat primul criteriu (relatia SECTII va avea mult mai putine inregistrari decit SALARIATI), dar si al doilea criteriu este indeplinit.

Pasul 2.3. Transformarea asocierilor unare si binare multi-multi si a celor de grad mai mare ca doi

Fiecare asociere binara multi-multi si fiecare asociere cu grad mai mare ca doi se transforma intr-o schema de relatie astfel:

Nume relatie	=	Nume asociere
Atribute relatie	=	Reuniunea identificatorilor entitatilor asociate la care se adauga atributele

proprii ale asocierii
Cheia relatiei = Conform tabelului 1.2.

Grad	Conectivitate	Dependente Functionale Primare (DFP)
Unare	unu (E1) - unu (E2) Cazul: Cheie(E2) se introduce in E1	Cheie(E1) → CheieStraina(E2) CheieStraina(E2) → Cheie(E1)
	unu (E1) - multi (E2)	Cheie(E2) → CheieStraina(E1)
	multi (E1) - multi (E2)	Cheie(E) + Cheie(E) → AP
Binare	unu (E) - unu (E) Cheie(E) se introduce ca si cheie straina in tabela E, cu redenumire	Cheie(E) → CheieStraina(E) CheieStraina(E) → Cheie(E)
	unu (E) - multi (E)	Cheie(E) → CheieStraina(E)
	multi (E) - multi (E)	Cheie(E) + Cheie(E) → AP
Ternare	unu (E1) - unu (E2) - unu (E3)	Cheie(E1)+Cheie(E2) → Cheie(E3) Cheie(E1)+Cheie(E3) → Cheie(E2) Cheie(E2)+Cheie(E3) → Cheie(E1)
	unu (E1) - unu (E2) - multi (E3)	Cheie(E1)+Cheie(E3) → Cheie(E2) Cheie(E2)+Cheie(E3) → Cheie(E1)
	unu (E1) - multi (E2) - multi (E3)	Cheie(E2)+Cheie(E3) → Cheie(E1)
	multi (E1) - multi (E2) - multi (E3)	Cheie(E2)+Cheie(E3)+Cheie(E1) → AP

Tabelul 1.1 Dependente functionale primare rezultate din transformare

Legenda:
A → B: Multimea de coloane A determina functional multimea de coloane B
AP: Atribute proprii asociere (daca exista) sau multimea vida.
A + B: Coloanele A impreuna cu coloanele B

Grad	Conectivitate	Dependente Functionale Primare (DFP)
Unare	multi (E1) - multi (E2)	Cheie(E1) + Cheie(E2)
Binare	multi (E) - multi (E)	Cheie(E) + Cheie(E)
Ternare	unu (E1) - unu (E2) - unu (E3)	Cheie(E1)+Cheie(E2) sau Cheie(E1)+Cheie(E3) sau Cheie(E2)+Cheie(E3) sau
	unu (E1) - unu (E2) - multi (E3)	Cheie(E1)+Cheie(E3) sau Cheie(E2)+Cheie(E3) sau
	unu (E1) - multi (E2) - multi (E3)	Cheie(E2)+Cheie(E3)
	multi (E1) - multi (E2) - multi (E3)	Cheie(E2)+Cheie(E3)+Cheie(E1)

Tabelul 1.2. Cheile schemelor de relatie rezultate din asocieri

Legenda:
A + B: Coloanele A impreuna cu coloanele B

1.5. Normalizarea schemelor de relatii (pasul 3)

Dupa terminarea pasului 2, fiecare schema obtinuta are eventual atasate dependente functionale primare (rezultate din transformare), dependente functionale secundare si multivalorice (depistate la pasul 1, analiza cerintelor). Aceste dependente determina gradul de normalizare a fiecarei scheme in parte. Daca acesta ne multumeste, putem considera incheiata proiectarea conceptuala. Daca nu, se poate aplica un algoritm de normalizare pina la obtinerea formei normale dorite.

Teoria formelor normale si algoritmii de normalizare sint in afara problematicii acestei prezentari. Se poate folosi orice algoritm dintre cele descrise in literatura de specialitate.

In final obtinem schemele de relatie definitive, cu care se poate trece la proiectarea fizica a bazei de date folosind facilitatile sistemului de gestiune a bazelor de date ales. Urmatorul pas este implementarea prelucrarilor asupra datelor. Acestea au fost definite inca de la pasul 1 iar specificarea lor este obiectul partii a treia a cursului.