Baze de date

Curs 3 – Proiectarea bazelor de date relaționale (cont.)

Sorina Preduț
sorina.predut@my.fmi.unibuc.ro
Universitatea din București

Cuprins

1. Crearea schemei conceptuale
    a. Modelul entitate-legătură (entitate-relaţie)
2. Crearea design-ului logic al bazei de date
3. Crearea design-ului fizic al bazei de date

1. Crearea schemei conceptuale

- Procesul de design al schemei conceptuale începe prin determinarea datelor necesare activităţilor din antrepriză.
- Este creată o echipă de design a schemei conceptuale care se ocupă cu determinarea datelor necesare, eventual prin folosirea de interviuri cu managerii antreprizei.
- După ce echipa proiectează datele, ea le revizuieşte şi le organizează.

1.a. Modelul entitate-legătură (entitate-relaţie)

- Una dintre tehnicile folosite pentru organizarea rezultatelor din etapa de colectare a datelor este modelul entitate-legătură, care **împarte elementele unui sistem real în 2 categorii** şi anume **entităţi şi legături (relaţii) între aceste entităţi**.
- Principalele concepte folosite în acest model sunt cele de entitate, relaţie (legătură) şi atribut.
- **Notă**: Nu trebuie confundat conceptul de relaţie în sensul de legătură sau asociere, care intervine în modelul entitate-legătură cu cel definit în cursul 1.

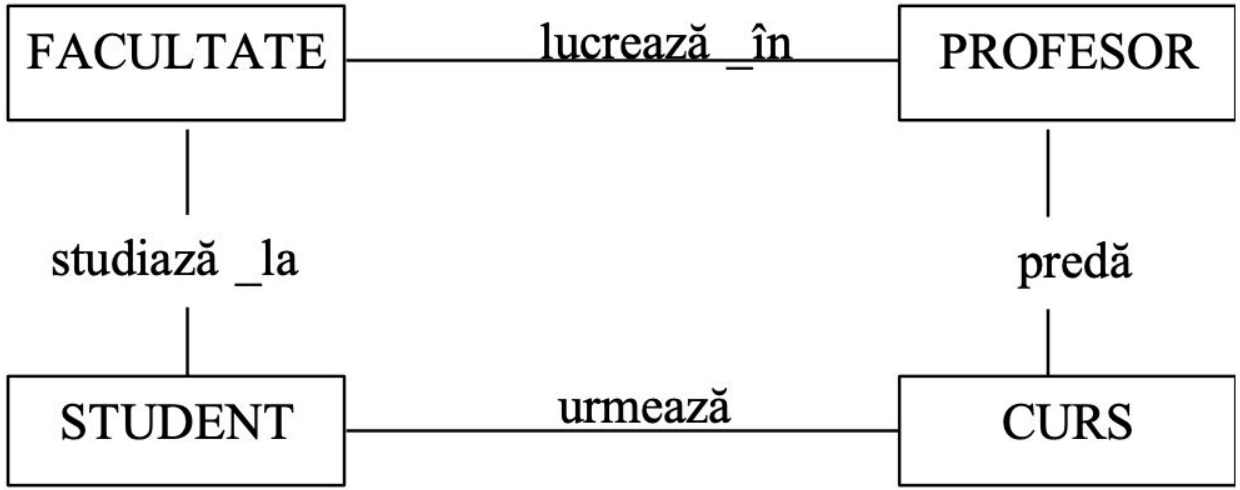
Entitate

- Este un obiect de interes şi pentru care trebuie să existe date înregistrate.
- Poate fi atât un obiect tangibil – precum **persoane, locuri sau lucruri** - cât şi abstracte – precum **comenzi, conturi bancare**, etc.

Entitate - cont.

- De exemplu, să considerăm o universitate formată din mai multe facultăţi; în fiecare **facultate** studiază mai mulţi **studenţi** şi predau mai mulţi **profesori**. Fiecare **student** urmează mai multe **cursuri**, după cum un **profesor** poate preda unul sau mai multe cursuri.
- Elementele semnificative ale acestui sistem sunt: facultate, student, profesor şi curs; acestea sunt entităţile sistemului.
- Ele sunt reprezentate în figura de pe slide-ul următor împreună cu relaţiile dintre ele.
- Remarcaţi că entităţile se reprezintă prin dreptunghiuri, iar relaţiile dintre ele prin arce neorientate.

Entitate - cont.



Entitate - cont.

- Ideile de bază pentru identificarea şi reprezentarea entităţilor sunt următoarele:
  - Fiecare entitate este denumită în mod unic; nu pot exista 2 entităţi cu acelaşi nume sau o entitate cu 2 nume diferite.
  - Entităţile sunt reprezentate întotdeauna prin substantive, dar nu orice substantiv folosit în descrierea sistemului este o entitate a acestuia.
  - Entităţile sistemului sunt doar acele substantive care au o semnificaţie deosebită în descrierea sistemului.

Entitate - cont.

De exemplu, chiar dacă suntem interesaţi de nr. de ore de predare efectuate de un profesor pe săptămână, aceasta nu înseamnă că vom crea o entitate pentru aceasta. De fapt, vom vedea în continuare că nr. de ore predate va fi un atribut al entităţii PROFESOR.

- De asemenea, pentru fiecare entitate trebuie să se dea o descriere detaliată, de exemplu, putem spune că un PROFESOR este un cadru didactic angajat al universităţii pe o perioadă nedeterminată, din această categorie făcând parte atât profesorii permanenţi cât şi cei asociaţi, dar fiind excluşi cei care predau la universitate numai o perioadă limitată.

Relație (legătură)

- Entităţile pot forma relaţii între ele. O relaţie este o asociere nedirecţionată între 2 entităţi. Ea exprimă **un raport care există între entităţile respective**.

- De exemplu, „lucrează_în” este o relaţie între entităţile PROFESOR şi FACULTATE, iar „predă” este o relaţie între entităţile PROFESOR şi CURS.

- Principalele idei pentru identificarea şi reprezentarea relaţiilor sunt următoarele:
  - Relaţiile sunt reprezentate prin verbe, dar nu orice verb este o relaţie.

Relație (legătură) - cont.

- Între 2 entităţi poate exista mai mult decât o singură relaţie.
  De exemplu, dacă luăm în vedere că fiecare facultate este condusă de un decan şi că acesta este ales din rândurile profesorilor, atunci între entităţile PROFESOR şi FACULTATE va mai exista o relaţie numită „conduce”.

- Pot exista relaţii cu acelaşi nume, dar relaţiile care asociază aceleaşi entităţi trebuie să poarte nume diferite.

Relație (legătură) - cont.

- **Cardinalitatea unei relaţii** indică nr. maxim de instanţe din fiecare entitate care poate participa la relaţie.
- Cu alte cuvinte, cardinalitatea unei relaţii reprezintă răspunsul la întrebări de genul: Câţi studenţi pot studia la o facultate? Mulţi.
  Dar la câte facultăţi poate studia un student? La cel mult una.
  Deci cardinalitatea relaţiei „studiază_la” este de mulţi-la-unu.

- Cardinalitatea unei relaţii poate fi de trei feluri: **mulţi-la-unu**, **unu-la-unu** sau **mulţi-la-mulţi**.

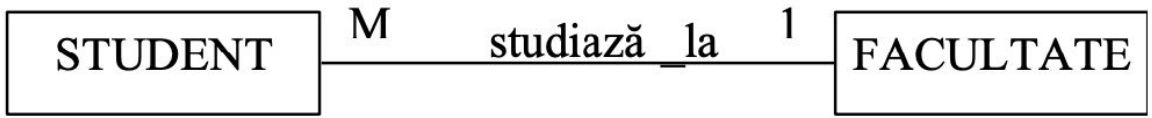
Relație (legătură) - cont.

- **mulţi-la-unu (many-to-one, N:1)**: Relaţia dintre entităţile A şi B este de tipul mulţi-la-unu dacă fiecărei instanţe din A îi poate fi asociată cel mult o singură instanţă din B şi fiecărei instanţe din B îi pot fi asociate mai multe instanţe din A.

- De exemplu, relaţiile „lucrează_în” dintre PROFESOR şi FACULTATE şi „studiază_la” dintre STUDENT şi FACULTATE sunt de tipul N:1.

Relație (legătură) - cont.

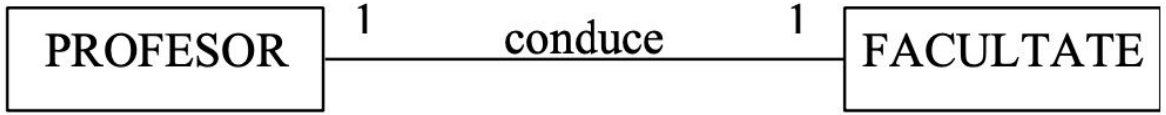
➢ O relaţie mulţi-la-unu se reprezintă în modul următor:



Relație (legătură) - cont.

- **unu-la-unu (one-to-one, 1:1)**: Relaţia dintre entităţile A şi B este de tipul unu-la-unu dacă fiecărei instanţe din A îi poate fi asociată cel mult o singură instanţă din B şi fiecărei instanţe din B îi poate fi asociată cel mult o singură instanţă din A.

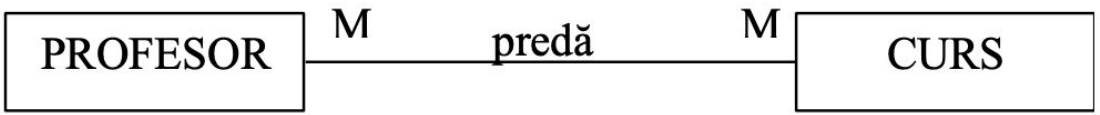
- De exemplu, relaţia „conduce” dintre PROFESOR şi FACULTATE este o relaţie 1:1.



Relație (legătură) - cont.

- **mulţi-la-mulţi (many-to-many, N:M)**: Relaţia dintre entităţile A şi B este de tipul mulţi-la-mulţi dacă fiecărei instanţe din A îi pot fi asociate mai multe instanţe din B şi fiecărei instanţe din B îi pot fi asociate mai multe instanţe din A.

- De exemplu, relaţiile „predă” dintre PROFESOR şi CURS şi „urmează” dintre STUDENT şi CURS sunt de tipul N:M. O relaţie N:M se reprezintă în modul următor:



Relație (legătură) - cont.

- Valorile discutate până acum (N:1, 1:1, N:M) reprezintă **cardinalitatea maximă** a unei relaţii. Pe de altă parte, o relaţie este caracterizată şi de o **cardinalitate minimă**, care indică obligativitatea participării entităţilor la relaţie.
  Cu alte cuvinte, aceasta furnizează răspunsul la întrebări de genul:

  Câţi studenţi trebuie să studieze la o facultate? Zero.
  Dar la câte facultăţi trebuie să studieze un student? Cel puţin una.

  Deci cardinalitatea minimă a relaţiei „studiază_la” dintre STUDENT şi FACULTATE este de 0:1.

Relație (legătură) - cont.

- În mod similar, relaţia „predă” dintre PROFESOR şi CURS are cardinalitatea minimă 0:0 (un profesor trebuie să predea zero cursuri şi un curs trebuie să fie predat de zero profesori – de exemplu dacă cursul este nou şi nu s-a stabilit încă titularul de curs).
  Deci **cardinalitatea minimă** a unei relaţii poate avea valorile **0:0, 0:1 şi 1:1**.
- Dacă participarea unei entităţi la o relaţie este obligatorie (cardinalitatea minimă respectivă este 1) se mai spune şi că **participarea** acesteia la relaţie este **totală**.
- În caz contrar (cardinalitatea minimă respectivă este 0), participarea entităţii la relaţie se numeşte **parţială**.

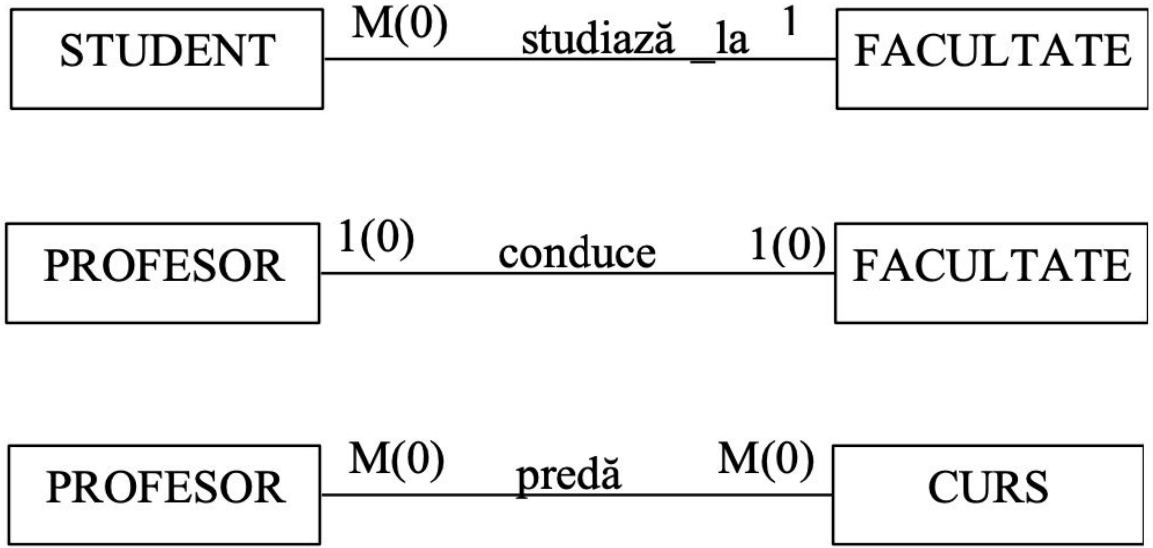
Relație (legătură) - cont.

- De exemplu participarea entităţii STUDENT la relaţia „studiază_la” este parţială, pe când participarea entităţii FACULTATE la aceeaşi relaţie este totală.

- În cadrul reprezentării grafice, **cardinalitatea maximă** a unei relaţii se va indica **fără paranteze**, în timp ce c**ardinalitatea minimă**, dacă este diferită de cea maximă, se va scrie **în paranteze**, vezi figurile de pe slide-ul următor.

- De multe ori, cardinalitatea minimă nu este indicată în diagrama entitate-legătură, pe când cardinalitatea maximă trebuie indicată întotdeauna, ea fiind esenţială.

Relație (legătură) - cont.

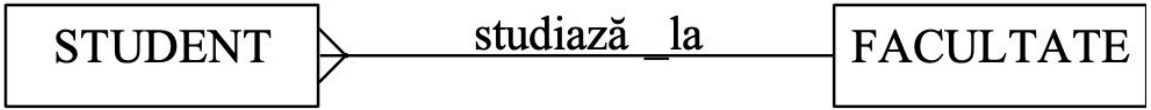


Relație (legătură) - cont.

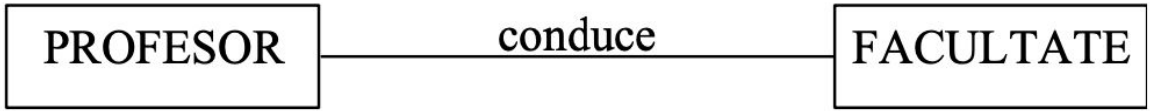
- Un alt mod de a reprezenta relaţiile, indicând doar cardinalitatea lor maximă este următorul:

Relație (legătură) - cont.

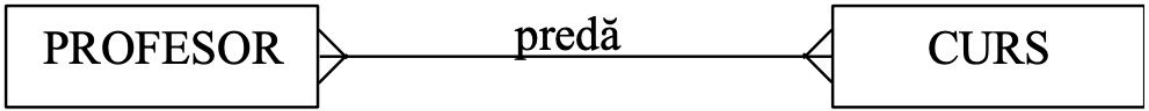
- relaţii mulţi-la-unu



- relaţii unu-la-unu



- relaţii mulţi-la-mulţi



Atribut

- Un atribut este **o caracteristică a unei entităţi sau a unei relaţii**.
  Fiecare entitate are un anumit număr de atribute despre care sunt înregistrate date.

- De exemplu, numele, prenumele, vârsta şi numărul de ore predate sunt atribute ale entităţii PROFESOR.

- Fiecare atribut poate lua o valoare care furnizează informaţii despre entitatea respectivă.

Atribut

- Exemple de valori de atribute sunt „Ionescu” pentru nume, „Mihai” pentru prenume etc.

- Pe de altă parte şi relaţiile pot avea atribute.
  De exemplu, relaţia „urmează” dintre STUDENT şi CURS poate avea ca atribute nota obţinută la examen şi nota obţinută la restanţă - pentru cei care nu au promovat examenul - iar relaţia „lucrează_în” dintre PROFESOR şi FACULTATE poate avea ca atribut data angajării.

Atribut - cont.

- Principalele idei pentru identificarea şi reprezentarea atributelor sunt următoarele:
  - Numele unui atribut este unic în cadrul unei entităţi sau al unei relaţii.
  - Atributele sunt întotdeauna substantive, dar nu orice substantiv este un atribut.
  - Pentru fiecare atribut, trebuie furnizată o descriere, împreună cu domeniul de valori (întreg, şir de caractere, dată, etc.).

Atribut - cont.

- Alegerea atributelor trebuie făcută în aşa fel încât să se evite aşa-numitele atribute indirecte.

- Un **atribut indirect** al unei entităţi sau relaţii este un atribut care nu aparţine în mod real acelei entităţi sau relaţii, fiind o caracteristică a unui alt obiect al sistemului.

- De exemplu, numele facultăţii este un atribut indirect al entităţii STUDENT, el descriind de fapt o proprietate a entităţii FACULTATE. De aceea, el va trebui redistribuit acestei entităţi.

Modelul entitate-legătură şi modelul relaţional

- Modelul entitate-legătură poate fi transformat în mod natural într-o bază de date relaţională. Fără a intra deocamdată în amănuntele acestei transformări, enunţăm în continuare principalele idei ale acestei transformări:
  - O entitate devine un tabel.
  - Un atribut al unei entităţi devine o coloană a tabelului respectiv.
  - O relaţie va fi reprezentată fie printr-un tabel special, fie printr-o cheie străină într-unul dintre cele două tabele entitate, care face referire la cheia primară a celuilalt tabel entitate.

Chei primare. Chei naturale şi chei artificiale

- În concordanţă cu terminologia folosită în cursul 1, o cheie a unei entităţi va fi un atribut sau un set de atribute care identifică în mod unic o instanţă a acelei entităţi.
- Cu alte cuvinte, o cheie face distincţie între oricare două rânduri diferite ale tabelului provenit din entitatea respectivă.
- De exemplu, putem presupune că fiecare student va fi identificat în cadrul universităţii printr-un cod unic; atunci codul studentului este o cheie a entităţii STUDENT.

Chei primare. Chei naturale şi chei artificiale

- Pe de altă parte, numele studentului nu poate fi cheie a acestei entităţi deoarece pot exista mai mulţi studenţi cu acelaşi nume.

- Dacă însă presupunem că nu pot exista studenţi cu acelaşi nume, prenume şi dată de naştere atunci combinaţia acestor atribute este la rândul ei cheie a entităţii STUDENT.

Chei primare. Chei naturale şi chei artificiale

- Există 2 tipuri de chei: naturale şi artificiale.
- O **cheie naturală** este constituită dintr-un atribut sau o combinaţie de atribute cu semnificaţie reală pentru entitatea în cauză.
  De exemplu, combinaţia nume, prenume, dată de naştere este o cheie naturală a entităţii STUDENT.
- O **cheie artificială** este un atribut al unei entităţi care nu are semnificaţie reală pentru entitatea în cauză, fiind folosită doar pentru a face distincţie între instanţele entităţii.
  De exemplu, codul studentului este o cheie artificială a entităţii STUDENT.

Chei primare. Chei naturale şi chei artificiale

- Una dintre cheile entităţii va fi declarată cheie primară (notăm cu CP sau PK).
- Deci, în principiu, oricare dintre cele 2 chei ale entităţii STUDENT poate fi declarată CP.
- Pe de altă parte însă, este preferată folosirea CP artificiale, excepţie făcând cazul când CP respectivă nu va fi stocată în alte tabele ca şi cheie străină.
- Principalele avantaje ale CP artificiale faţă de cele naturale sunt următoarele:

Chei primare. Chei naturale şi chei artificiale

- Principalele avantaje ale CP artificiale faţă de cele naturale sunt următoarele:
  - **Stabilitatea**. Valoarea unei chei artificiale rămâne aceeaşi pe parcursul funcţionării sistemului, în timp ce valoarea unei chei naturale poate fi în general modificată, această modificare atrăgând, la rândul ei, schimbarea cheilor străine care fac referire la ea.
    De exemplu, numele unei studente se poate schimba prin căsătorie; dacă se consideră combinaţia nume, prenume şi data naşterii ca fiind CP a entităţii STUDENT, atunci orice schimbare a numelui va impune modificarea valorilor cheilor străine corespunzătoare.

Chei primare. Chei naturale şi chei artificiale

- Ca o regulă generală, valoarea CP a unui tabel nu trebuie să poată fi modificată, aceasta creând probleme privind păstrarea integrităţii datelor - cu alte cuvinte schimbarea CP a unui tabel va trebui însoţită de schimbarea cheilor străine care fac referire la aceasta.
- **Simplitatea**. În general, o cheie artificială este mai simplă decât una naturală. Cheile naturale sunt mai complexe, atât dpdv fizic (nr. de octeţi) cât şi al nr. de coloane.
  De exemplu, este mult mai comodă stocarea codului studentului ca şi cheie străină, decât a combinaţiei dintre numele, prenumele şi data naşterii.

Chei primare. Chei naturale şi chei artificiale

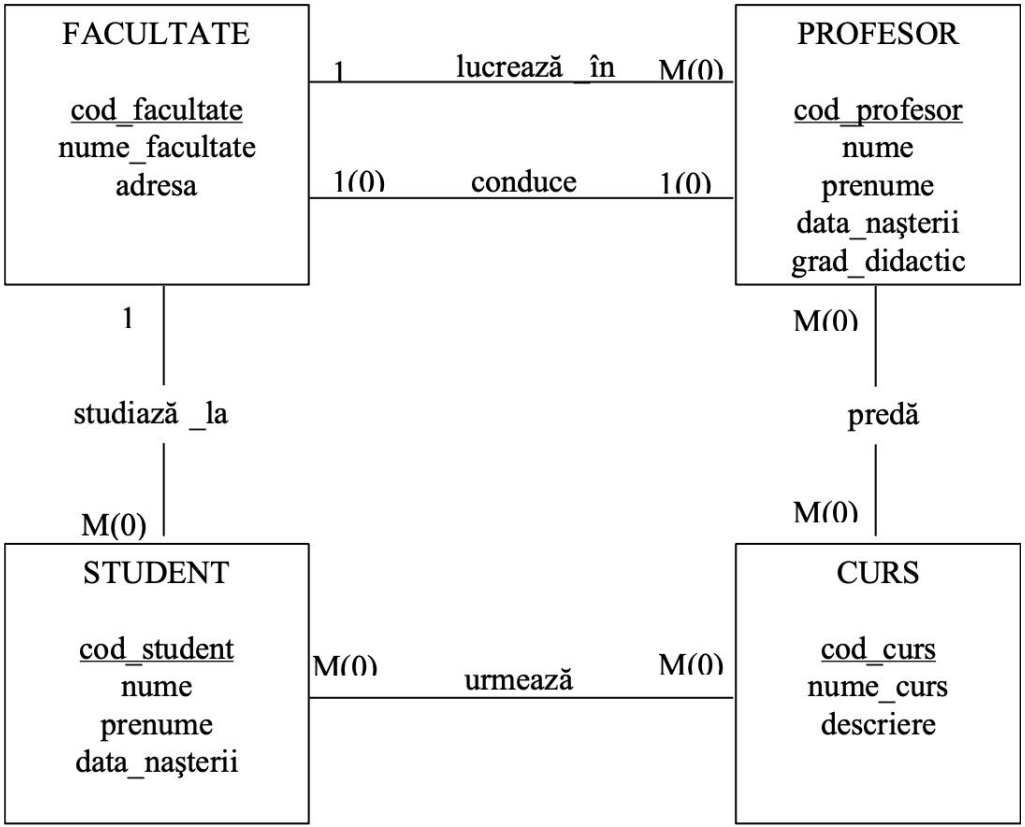
- **Nu prezintă ambiguităţi**. O CP trebuie să nu prezinte ambiguităţi, a. î. să poată fi folosită cu uşurinţă de către dezvoltator sau utilizator în filtrările efectuate pe tabele.
  Şi în această privinţă, o cheie naturală creează probleme.
  De exemplu, numele şi prenumele unui student pot fi formate dintr-unul sau mai multe cuvinte care pot fi despărţite de un spaţiu sau de o linie, etc.

Chei primare. Chei naturale şi chei artificiale

- **Elimină valorile Null**. În cazul CP naturale, valorile Null reprezintă o problemă.
  De exemplu, aceasta înseamnă că un student nu poate fi înregistrat dacă nu i se ştie data de naştere.
- În concluzie, o **CP trebuie să fie unică, diferită de Null, scurtă, simplă, fără ambiguităţi, să nu conţină informaţii descriptive, să fie uşor de manipulat, să fie stabilă şi familiară utilizatorului**.
  Cheile artificiale îndeplinesc toate aceste condiţii în afară de ultima, fiind preferate aproape întotdeauna celor naturale.

Diagrama entitate-legătură (ERD)

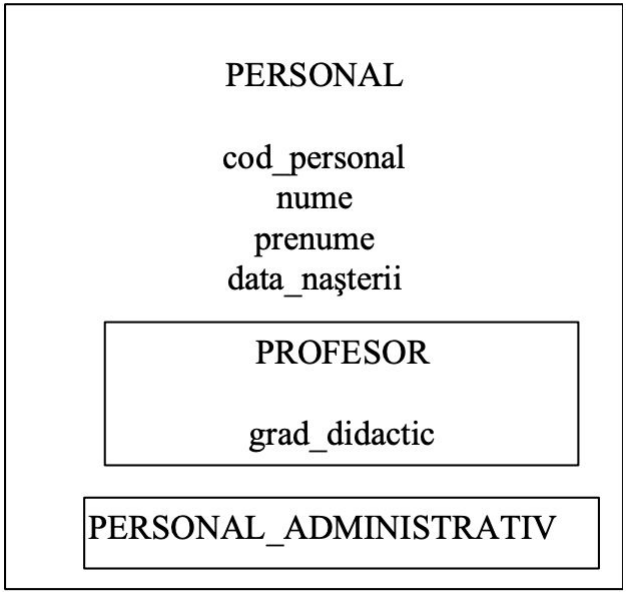
- Entităţile sistemului, împreună cu relaţiile dintre ele se reprezintă prin aşa numita diagramă entitate-legătură (eng. **entity relationship diagram**), în care **entităţile** sunt reprezentate prin **dreptunghiuri**, iar **relaţiile** dintre acestea prin **arce neorientate**, specificându-se şi **cardinalitatea** acestora.
- Pentru fiecare entitate se specifică **CP** şi eventual atributele mai semnificative, atributele care reprezintă CP trebuind să fie **subliniate**.
- Diagrama entitate-legătură a sistemului descris la începutul cursului este reprezentată astfel:


FACULTATE
cod_facultate
nume_facultate
adresa
1
lucrează _în
M(0)
PROFESOR
cod_profesor
nume
prenume
data_naşterii
grad_didactic
1(0)
conduce
1(0)
1
studiază _la
M(0)
M(0)
predă
M(0)
STUDENT
cod_student
nume
prenume
data_naşterii
M(0)
urmează
M(0)
CURS
cod_curs
nume_curs
descriere

Cazuri speciale de entităţi, relaţii şi atribute

- În continuare vom considera câteva cazuri speciale de entităţi, relaţii şi atribute, încercând în acelaşi timp o clasificare a acestora.
- **Subentitate/Superentitate**. O subentitate este o submulţime a unei alte entităţi, numită superentitate. De exemplu, să presupunem că în sistemul prezentat mai înainte nu vom reţine date numai despre profesorii, universităţii, ci şi despre tot personalul din universitate. Atunci vom crea o superentitate PERSONAL, pentru care PROFESOR este o subentitate. O altă subentitate a acestei superentităţi va fi PERSONAL_ADMINISTRATIV. O **subentitate** se reprezintă printr-un **dreptunghi inclus în dreptunghiul care reprezintă superentitatea corespunzătoare**, ca în figura următoare:

Subentitate/Superentitate



Cazuri speciale de entităţi, relaţii şi atribute

- CP, atributele şi relaţiile unei superentităţi sunt valabile pentru orice subentitate, reciproca fiind evident falsă.
  De exemplu, CP a entităţii PROFESOR va fi acum „cod_personal”, care este CP a entităţii PERSONAL, în timp ce unele dintre atributele subentităţii PROFESOR (de exemplu „nume”, „prenume”, „data_nasterii”) se regăsesc printre atributele entităţii PERSONAL.
  Pe de altă parte însă, subentitatea PROFESOR poate avea şi alte atribute decât cele specifice superentităţii PERSONAL, de exemplu gradul didactic. Cu alte cuvinte, **atributele comune** vor fi repartizate superentităţii, în timp ce **atributele specifice** vor fi repartizate subetităţilor.

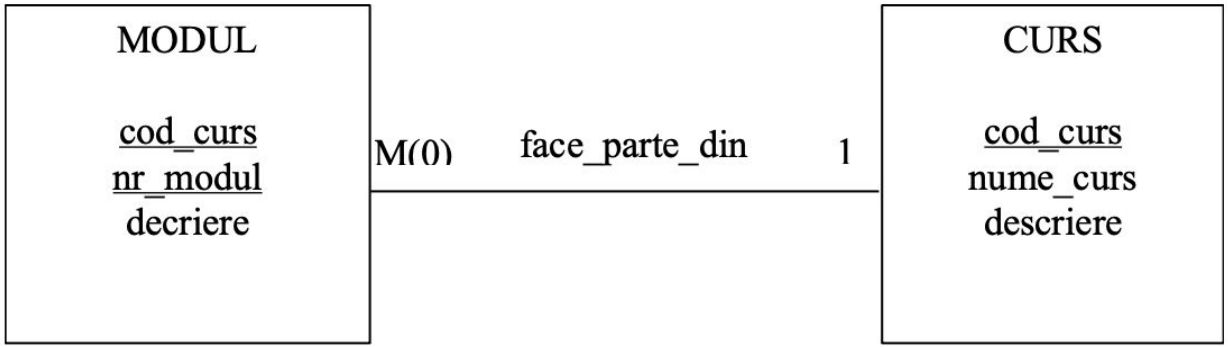
Cazuri speciale de entităţi, relaţii şi atribute

- **Între o subentitate şi superentitatea corespunzătoare există întotdeauna o relaţie 1:1, având cardinalitatea minimă 1:0.**
  Uneori este convenabil să se creeze superentităţi din entităţi cu mai multe atribute comune.
  De exp., din entităţile PROFESOR şi PERSONAL_ADMINISTRATIV s-a creat entitatea PERSOANA. Superentitatea creată va conţine atributele comune, iar cele specifice vor fi repartizate subentităţilor componente.
  În plus, se va crea o nouă cheie artificială pentru superentitatea nou formată. De exemplu, pentru PERSOANA s-a creat un cod personal, care a devenit CP a acestei entităţi.

Cazuri speciale de entităţi, relaţii şi atribute

- **Entitate dependentă (detaliu)/entitate master**.
  O **entitate dependentă** (detaliu) este o entitate care **nu poate exista de sine stătătoare, ci numai ataşată unei alte entităţi**, aceasta din urmă fiind numită entitatea master a acestei legături.
  De exemplu, dacă presupunem că fiecare curs poate fi constituit dintr-unul sau mai multe module, atunci entitatea MODUL va fi o entitate dependentă de CURS, ca în figura:

Entitate dependentă (detaliu)/entitate master

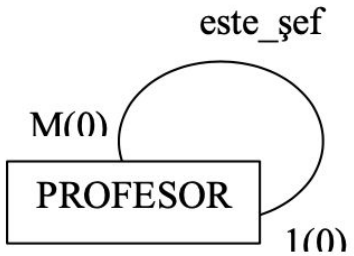


Cazuri speciale de entităţi, relaţii şi atribute

- **Între entităţile master şi detaliu va exista întotdeauna o relaţie 1:N, având cardinalitatea minimă 1:0**.
  CP a unei entităţi detaliu va fi formată din CP a entităţii master plus una sau mai multe atribute ale entităţii detaliu.
  De exemplu, cheia entităţii MODUL poate fi aleasă ca fiind combinaţia dintre cod_curs şi nr_modul, acesta din urmă specificând numărul de ordine al unui modul în cadrul unui curs.

Cazuri speciale de entităţi, relaţii şi atribute

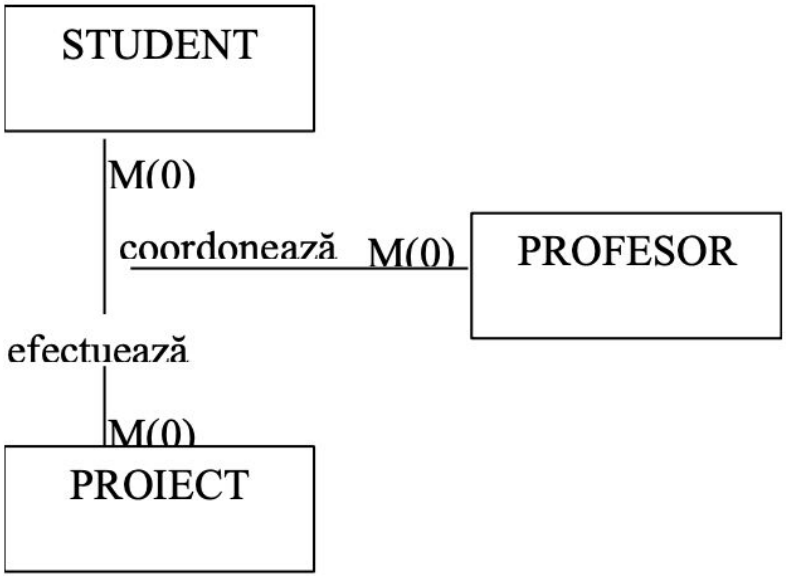
- **Relaţii recursive.**
  Pot exista **relaţii** nu numai între două entităţi diferite, ci şi **între o entitate şi ea însăşi**; acestea se numesc relaţii recursive.
  De exemplu, dacă presupunem că activitatea de cercetare în universitate este organizată pe o structură ierarhică, adică un profesor poate avea un şef şi poate fi la rândul lui şeful mai multor profesori, atunci entitatea profesor admite o relaţie recursivă de tipul N:1, ca în figura



Cazuri speciale de entităţi, relaţii şi atribute

- **Relaţii binare (de tip 2)/ relaţii între mai mult de două entităţi (de tip 3)**.
  Până acum am discutat doar despre relaţiile dintre 2 entităţi, numite relaţii binare sau de tip 2.
  Pot însă exista şi relaţii între mai mult de două entităţi, pe care le vom numi relaţii de tip 3.
  De exemplu, să presupunem că fiecare student trebuie să efectueze mai multe proiecte, iar pentru fiecare proiect el poate să-şi aleagă unul sau mai mulți profesori coordonatori. Deci relaţia „efectuează_coordonează” este o relaţie de tip 3 între entităţile STUDENT, PROIECT şi PROFESOR, ca în figura:

Relaţii binare (de tip 2)/ relaţii între mai mult de două entităţi (de tip 3)



Cazuri speciale de entităţi, relaţii şi atribute

- **O relaţie de tip 3 nu poate fi spartă în relaţii binare între entităţile componente**, un exemplu este oferit în figura următoare, unde prin spargerea relaţiei „efectuează_coordonează” în 3 relaţii binare prin proiecţie se obţin informaţii eronate, relaţia iniţială nemaiputând fi reconstituită din relaţiile componente.

STUDENT	PROIECT	PROFESOR
s1	p1	x2
s1	p2	x1
s2	p1	x1

a) Relaţia de tip 3 iniţială

STUDENT	PROIECT
s1	p1
s1	p2
s2	p1

STUDENT	PROFESOR
s1	x2
s1	x1
s2	x1

PROIECT	PROFESOR
p1	x2
p2	x1
p1	x1

b) Descompunerea relaţiei de tip 3 în 3 relaţii binare prin proiecţie

STUDENT	PROIECT	PROFESOR
s1	p1	x1
s1	p1	x2
s1	p2	x1
s2	p1	x1

c) Reconstituirea eronată a relaţiei iniţiale

Cazuri speciale de entităţi, relaţii şi atribute

- **Atribute simple/ compuse/ repetitive (multivaloare)/ calculate (deduse)**.
  Atributele pot fi de 4 feluri:
  - simple,
  - compuse,
  - repetitive (multivaloare) şi
  - calculate (deduse).
- Unui atribut simplu îi corespunde o singură valoare, atomică.
  De exemplu, numele şi prenumele unui student sunt atribute simple.

Cazuri speciale de entităţi, relaţii şi atribute

- Un atribut compus este format din mai multe atribute simple, numite componentele sale.
- Valoarea unui atribut compus este reprezentată de valorile atributelor componente.
  Dacă presupunem, de exemplu, că o adresă se poate descompune în componentele ţară, oraş, stradă, număr şi cod, atunci adresa este un atribut compus din 5 componente.

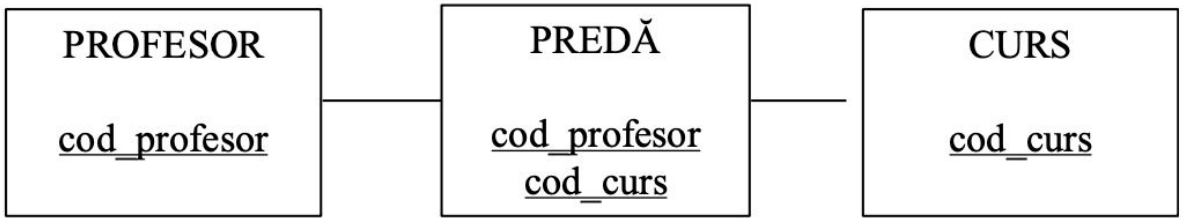
Cazuri speciale de entităţi, relaţii şi atribute

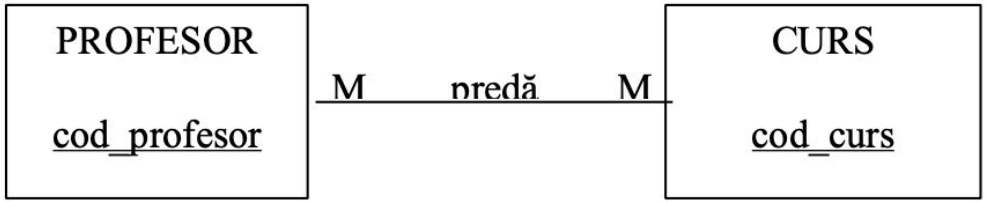
- Un **atribut repetitiv (multivaloare)** este un atribut care poate avea mai multe valori, numărul acestora variind de la o instanţă la alta.
  De exemplu, un student poate avea mai multe numere de telefon, deci acesta este un atribut repetitiv.
- Un **atribut calculat** reprezintă un atribut a cărui valoare nu este cunoscută direct, ci calculată pe baza valorilor altor atribute.
  De exemplu atributul valoare este calculat ca produs între atributele cantitate şi preţ. **Atributele calculate se folosesc foarte rar deoarece ele reprezintă de fapt o redundanţă a datelor**.

Probleme în identificarea entităţilor, relaţiilor şi atributelor

- **Relaţie sau entitate?**
  Uneori este greu de identificat dacă o componentă a sistemului este relaţie sau entitate.
  Dacă o entitate are o cheie provenită din combinaţia cheilor primare a două sau mai multe entităţi, atunci trebuie definită o relaţie.
  Deci entitatea PREDĂ va avea semnificaţia unei relaţii între entităţile PROFESOR şi CURS, reprezentată astfel:

Relaţie sau entitate?

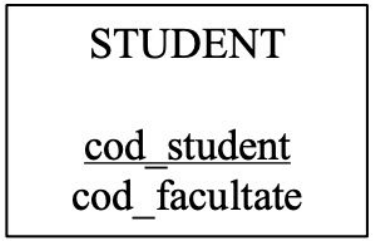


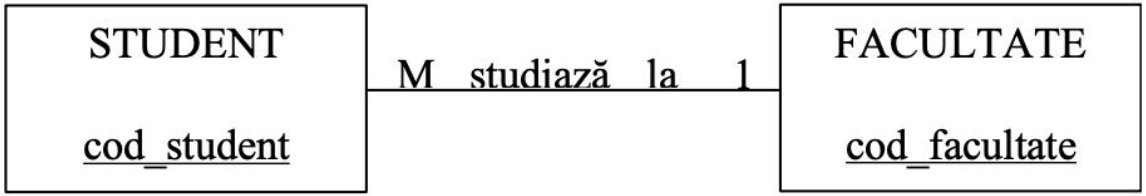


Probleme în identificarea entităţilor, relaţiilor şi atributelor

- **Relaţie sau atribut?**
- După cum am văzut până acum, o relaţie poate fi reprezentată ca un atribut al unei entităţi, după cum atributele unei entităţi pot fi înlocuite cu relaţii.
  Deci, care este diferenţa între o relaţie şi un atribut?
  Atunci când un atribut al unei entităţi reprezintă CP a altei entităţi, el face referinţă la o relaţie. Deci atributul cod_facultate din prima entitate STUDENT va reprezenta o relaţie între entităţile STUDENT şi FACULTATE, așa cum se poate vedea în figura:

Relaţie sau atribut?



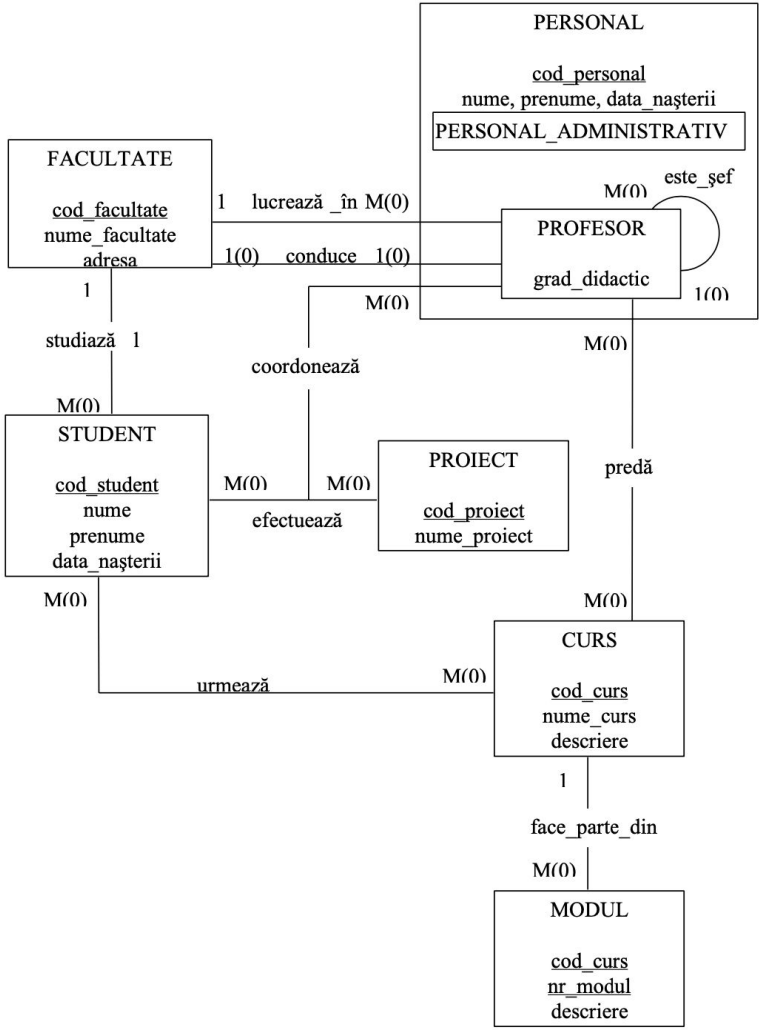


Etapele obţinerii modelului entitate-legătură

- Pentru realizarea modelului entitate-legătură a sistemului analizat sunt parcurse următoarele etape:
  - Identificarea **entităţilor** sistemului.,
  - Identificarea **relaţiilor** sistemului şi stabilirea **cardinalităţii** acestora.
  - Identificarea **atributelor** entităţilor şi relaţiilor sistemului.
  - Stabilirea **cheilor primare** ale entităţilor.
  - Trasarea **diagramei entitate-legătură**.

Exemplu de ERD

Diagrama entitate-legătură a sistemului prezentat ca exemplu în aceast curs, incluzând entităţile şi relaţiile menţionate mai înainte:



Etapele obţinerii modelului entitate-legătură

- Trebuie remarcat că aceeaşi realitate poate fi percepută diferit de către analişti diferiţi, aşa că **este posibilă obţinerea de modele diferite pentru acelaşi sistem**, după cum şi un sistem poate să se modifice în timp, ceea ce va atrage la rândul său modificarea modelului asociat.
- În sfârşit, există şi alte moduri grafice de prezentare a diagramei entitate-legătură, cum ar fi aceea din fișierul Diagrame E-R.pdf, în acest curs prezentându-se doar una dintre notaţiile existente.

2. Crearea design-ului logic al bazei de date

- Pentru realizarea design-ului logic al bazei de date, **schema conceptuală este transformată într-un design al BD care va funcţiona într-un SGBD specific**.
  Designul logic al bazei de date este o rafinare a modelului iniţial furnizat de schema conceptuală.
  Aceasta nu înseamnă că modelul conceptual nu este corect, dar trebuie stabilite detalii suplimentare necesare dezvoltării proiectului.

Transformarea modelului entitate legătură în modelul relaţional

- Pentru obţinerea design-ului logic al unei BD relaţionale se porneşte de la schema conceptuală, mai precis de la modelul entitate-legătură, şi se încearcă **reprezentarea entităţilor şi a legăturilor sub formă de tabele relaţionale**.
- Regulile de conversie ale entităţilor, legăturilor şi atributelor sunt următoarele:

Transformarea entităţilor

- Regula generală este că entităţile devin tabele, distingându-se următoarele subcazuri:
  - **Entităţile independente devin tabele independente**, adică tabele a căror CP nu conţine chei străine. De exemplu, entitatea STUDENT va deveni un tabel a cărui cheie primară este “cod_student”.
  - **Entităţile dependente devin tabele dependente** (tabele detaliu) adică tabele a căror CP conţine cheia străină ce face referinţă la CP a entităţii de care depinde entitatea în cauză.
    De exemplu, CP a entităţii MODUL va fi formată din „cod_curs”, care reprezintă o cheie străină pentru entitatea CURS, plus „nr_modul”.

Transformarea entităţilor

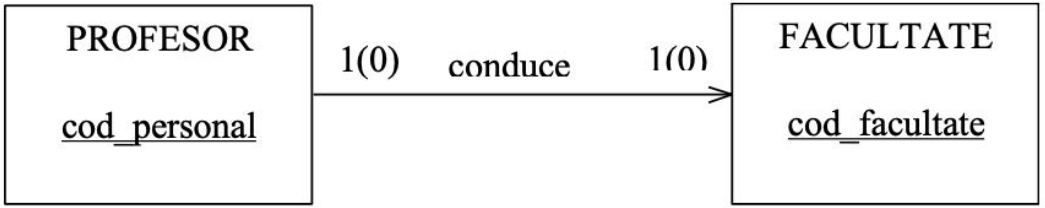
- **Subentităţile** devin **subtabele** adică tabele a căror CP este cheia străină pentru tabelul superentitate. De exemplu, CP a tabelului PROFESOR este „cod_personal”, care este o cheie străină ce face referinţă la CP „cod_personal” din tabelul PERSONAL.
- Uneori, se preferă construirea unor supertabele, formate atât din atributele superentităţii - cele comune tuturor subentităţilor - cât şi din atributele specifice fiecărei subentităţi.
  Avantajul unor astfel de supertabele este simplificarea programelor de manipulare a datelor.

Transformarea entităţilor

- Pe de altă parte însă, ele creează **probleme suplimentare privind integritatea datelor**, de exemplu dacă vom avea un singur tabel pentru tot personalul din facultate, atunci când se inserează în tabel un rând corespunzător unui profesor **numai atributele specifice pot avea valori diferite de Null**.
  În plus, subtabelele obţinute din descompunerea unui astfel de supertabel sunt mai stabile, mai flexibile, ocupă spaţiu fizic mai mic şi conţin mai puţine valori Null.

Transformarea relațiilor

- **Relaţiile 1:1 devin chei străine, cheia străină fiind plasată în tabelul cu mai puţine linii**. De exemplu, relaţia “conduce” dintre PROFESOR şi FACULTATE se realizează prin inserarea unei chei străine în tabelul FACULTATE care face referinţă la CP a tabelului PROFESOR, ca în figura:

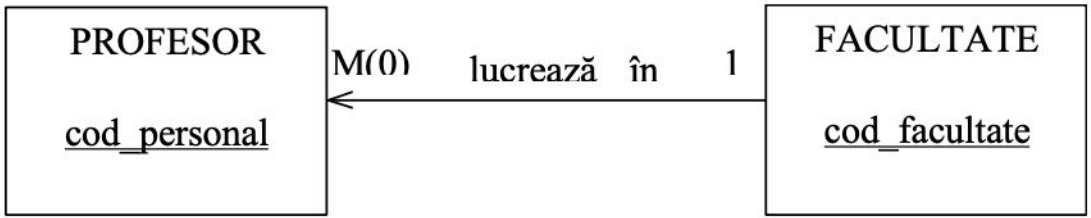


Transformarea relațiilor

- Notă: Plasamentul cheii străine va fi indicat printr-o săgeată (⟶), iar când cheia străină va fi conţinută în CP, atunci vârful săgeţii va fi umplut (⟶).
- Deci într-o relaţie 1:1 **poziţia cheii străine depinde de cardinalitatea minimă a relaţiei**. Dacă aceasta este tot 1:1, atunci cheia străină poate fi plasată în oricare din cele două tabele. Dacă însă această cardinalitate minimă este 1:0, atunci cheia străină este plasată în tabelul a cărui cardinalitate minimă în relaţie este 0.

Transformarea relațiilor

- **Relaţiile N:1 devin chei străine plasate în tabelul care se află de partea „mulţi” a relaţiei.**
  De exemplu relaţia „lucrează_în” va fi realizată prin inserarea unei chei străine în tabelul PROFESOR care va face referinţă la CP a tabelului FACULTATE, ca în figura:



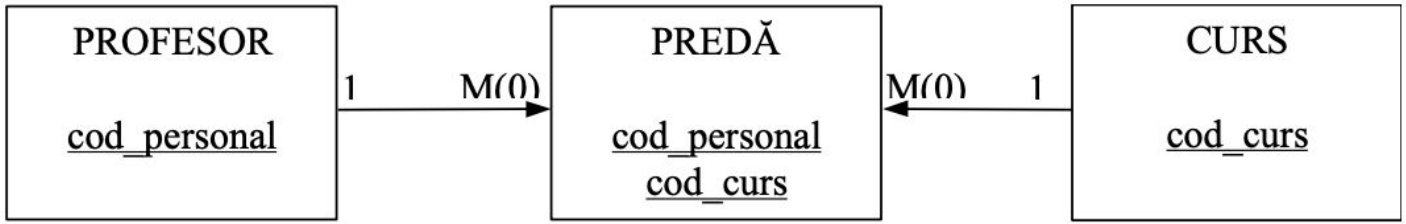
Transformarea relațiilor

- Şi în cazul relaţiilor N:1 se disting 2 cazuri în funcţie de cardinalitatea minimă a relaţiei.
  Dacă aceasta este 0:1, atunci cheia străină respectivă nu poate avea valoarea Null, iar în cazul entităţilor dependente ea va face chiar parte din CP a tabelului.
  Dacă însă cardinalitatea minimă a relaţiei este 0:0 atunci cheia străină poate avea valoarea Null şi nu poate face parte din CP.

Transformarea relațiilor

- **O relaţie mulţi-la-mulţi se transformă într-un tabel special, numit tabel asociativ**, care are 2 chei străine pentru cele 2 tabele asociate;
  CP a tabelului asociativ este compusă din aceste 2 chei străine plus eventual alte coloane adiţionale. În acest caz se spune că **o relaţie mulţi-la-mulţi se sparge în 2 relaţii mulţi-la-unu**, tabelul asociativ fiind în relaţie de mulţi-la-unu cu fiecare dintre cele 2 tabele entitate.
  De exemplu, relaţia „predă” dintre PROFESOR şi CURS se realizează printr-un tabel a cărui CP este combinaţia cheilor străine ale acestor 2 entităţi, ca în figura:

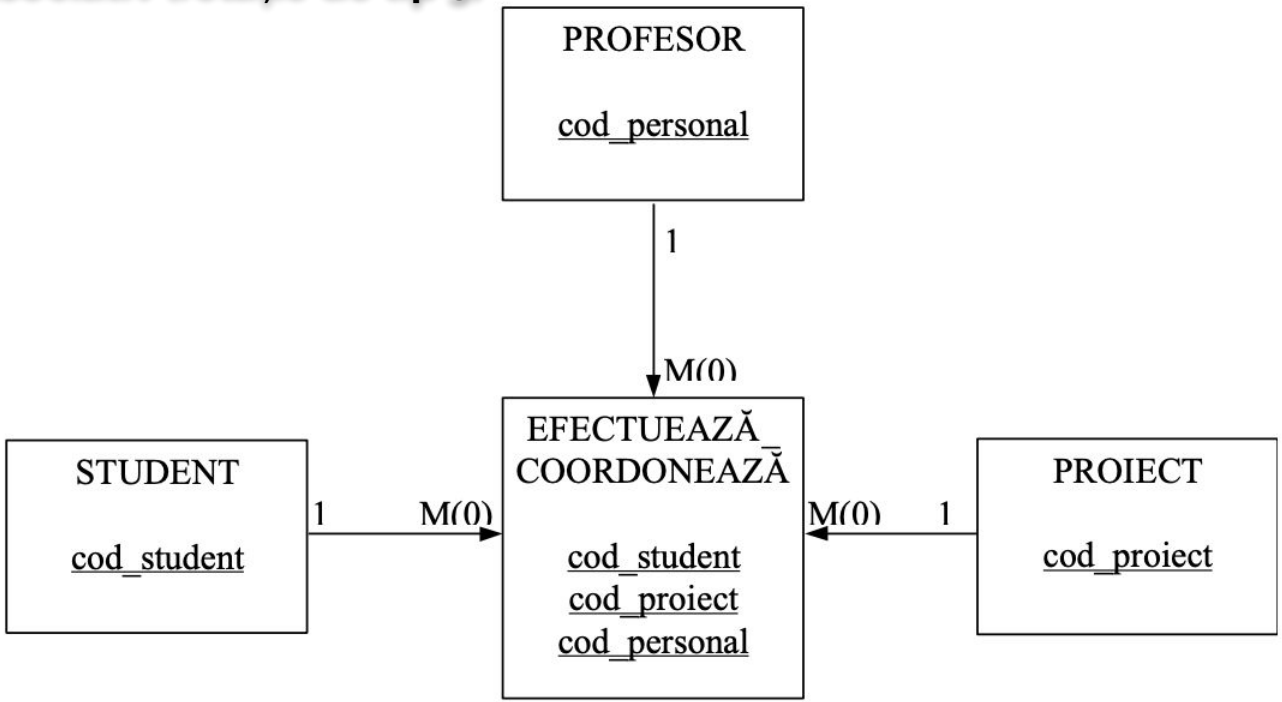
Tabel asociativ (relație M:M)



Transformarea relațiilor

- **O relaţie de tip 3 (relaţie între mai mult de două entităţi) devine un tabel asociativ care are câte o cheie străină pentru fiecare dintre tabelele asociate;**
  CP este compusă din aceste chei străine plus eventual alte coloane adiţionale.
  De exemplu, tabelul reprezentat în figura următoare exprimă relaţia „efectuează_coordonează” dintre STUDENT, PROIECT şi PROFESOR.
  În acest caz, CP este combinaţia cheilor străine corespunzătoare celor 3 entităţi.

Tabel asociativ (relație de tip 3)



Transformarea atributelor

- **Atributele simple ale unei entităţi devin coloane în tabelul provenit din entitatea corespunzătoare.**
  De asemenea, fiecare componentă a unui atribut compus devine o coloană în tabel.
  De exemplu, pentru atributul compus adresă, format din ţară, oraş, stradă, număr şi cod, vom avea cinci coloane, câte una pentru fiecare componentă a sa.

Transformarea atributelor

- **Atributele repetitive (multivaloare) ale unei entităţi devin tabele dependente ce conţin o cheie străină (care face referinţă la CP a entităţii) şi atributul multivaloare;**
  CP a acestui nou tabel este formată din cheia străină plus una sau mai multe coloane adiţionale.
  De exemplu, dacă presupunem că un student poate avea mai multe numere de telefon, atunci „nr_telefon” este un atribut multivaloare al entităţii STUDENT, care va da naştere unui tabel TELEFON, a cărui CP va fi combinaţia dintre „cod_student” şi „nr_telefon”, ca în figura:

Atribute repetitive (multivaloare) ale unei entități

Transformarea atributelor

- **Atributele simple ale unei relaţii 1:1 sau 1:N vor deveni coloane ale tabelului care conţine cheia străină**.
  De exemplu, data angajării, care este un atribut al relaţiei „lucrează_în” dintre PROFESOR şi FACULTATE, va fi reprezentată ca o coloană în tabelul PROFESOR.
  De asemenea, fiecare atribut compus al unei relaţii 1:1 sau 1:N va deveni o coloană în tabelul care conţine cheia străină.

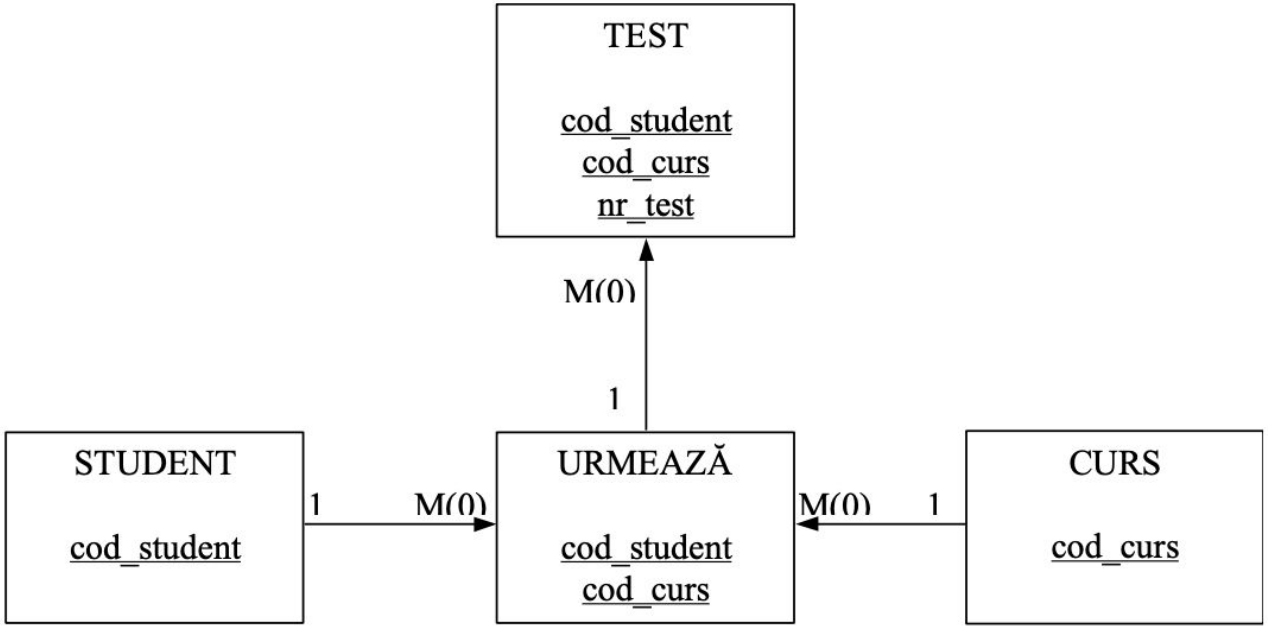
Transformarea atributelor

- **Atributele simple ale unei relaţii N:M vor deveni coloane ale tabelului asociativ**. De exemplu, nota obţinută la examen, care este un atribut al relaţiei „urmează” dintre STUDENT şi CURS va fi reprezentată ca o coloană în tabelul asociativ corespunzător acestei relaţii.
  De asemenea, fiecare componentă a unui atribut compus al unei relaţii N:M va deveni o coloană în tabelul asociativ.

Transformarea atributelor

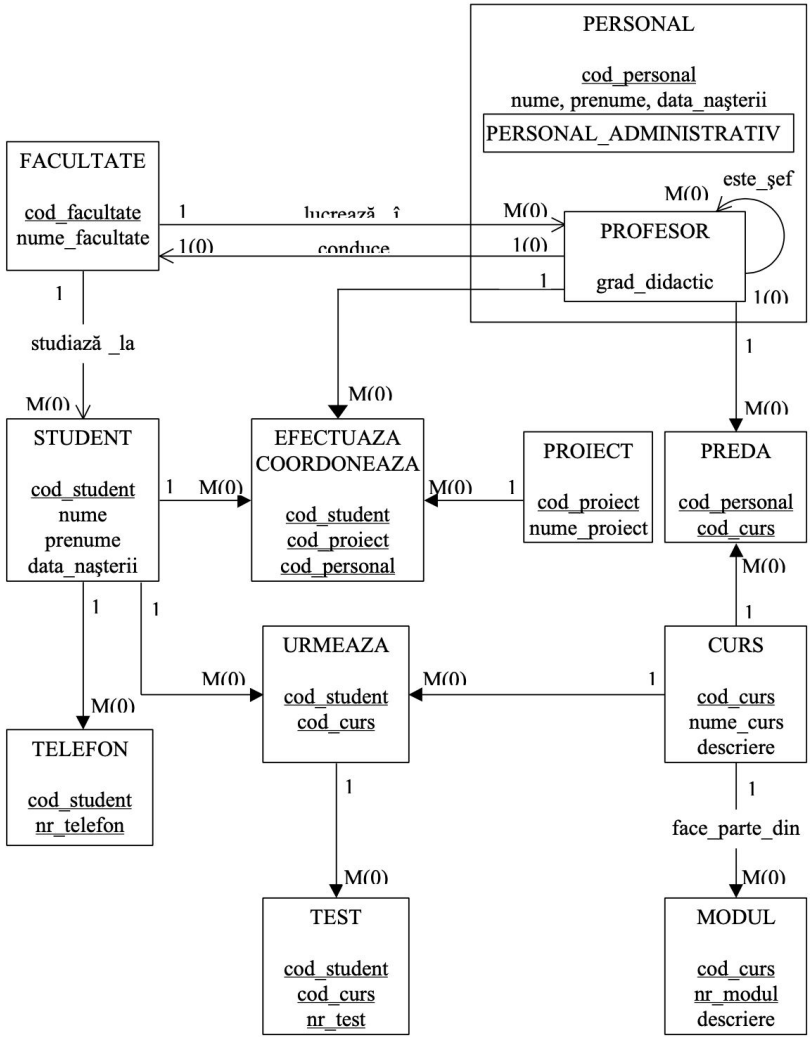
- **Atributele repetitive (multivaloare) ale unei relaţii 1:1 sau 1:N vor deveni tabele dependente de tabelul care conţine cheia străină, iar atributele repetitive ale unei relaţii N:M vor deveni tabele dependente de tabelul asociativ corespunzător relaţiei**.
  Evident, CP a acestor tabele dependente va fi o combinaţie formată din cheia străină respectivă şi una sau mai multe coloane adiţionale.
  De exemplu, dacă presupunem în cadrul anumitor cursuri că studenţii trebuie să dea un nr. de teste, atunci „test" va fi un atribut multivaloare al relaţiei „urmează" dintre STUDENT şi CURS şi care va da naştere unui tabel dependent de tabelul asociativ al acestei relaţii, ca în figura:

Atribute repetitive (multivaloare) ale unei relații N:M



Exemplu de diagramă logică

Diagrama logică a BD pentru sistemul descris ca exemplu în acest curs care a rezultat din ERD, în urma transformărilor prezentate mai înainte:



Exemplu de diagramă logică - cont.

➢ **Tabelele asociate acestei diagrame sunt următoarele:**

PERSONAL (cod_personal, nume, prenume, data_nastere, sex, stare_civila)
PERSONAL_ADMINISTRATIV (*cod_personal*, profesie, funcţie)
PROFESOR (*cod_personal*, grad_didactic, titlu, *sef*, ore_predate, data_angajării, *cod_facultate*)
CURS (cod_curs, nume_curs, descriere, nr_ore)
PREDA (*cod_personal*, *cod_curs*)
MODUL (*cod_curs*, nr_modul, descriere)
FACULTATE (cod_facultate, nume_facultate, localitate, strada, nr, cod_postal *cod_decan*)

Exemplu de diagramă logică - cont.

STUDENT (cod_student, nume, prenume, data_nasterii, tara, localitate, strada, nr, cod_postal, studii_anterioare)
TELEFON (*cod_student*, nr_telefon, tip_telefon)
PROIECT (cod_proiect, nume_proiect, domeniu)
EFECTUEAZA_COORDONEAZA (*cod_student*, *cod_proiect*, *cod_personal*)
URMEAZA (*cod_student*, *cod_curs*, nota_examen, nota_restantă, observatii)
TEST (*cod_student*, *cod_curs*, nr_test, nota_test, observatii)

➢ **Notă:** Atributele subliniate constituie CP a tabelului iar cele italice constituie chei străine.

Bibliografie

F. Ipate, M. Popescu, *Dezvoltarea aplicațiilor de baze de date în Oracle 8 și Oracle Forms 6*, Editura ALL, 2000.