

Modelarea datelor

Un **model** este o reprezentare a obiectelor și evenimentelor lumii reale și a asocierilor dintre ele. De fapt, el reprezintă o abstracție asupra aspectelor semnificative ale unei „întreprinderi“, ale unui sistem real, ignorând proprietățile accidentale. Modelul este cel pe care utilizatorii trebuie să-l cunoască; implementarea unui model este cea pe care utilizatorii nu este necesar să o cunoască. Diferența dintre model și implementare este, de fapt, un caz special și important al deosebirii uzuale dintre logic și fizic.

Modelele se impun prin sintaxa și prin semantica lor și, din acest punct de vedere, există trei tipuri fundamentale de modele:

- modele care descriu aspectele statice ale procesului modelat;
- modele care descriu aspectele dinamice ale procesului modelat;
- modele care descriu aspectele funcționale ale procesului modelat.

Un **model de date** reprezintă o colecție integrată de concepte necesare descrierii:

- datelor,
- relațiilor dintre ele,
- constrângerilor existente asupra datelor sistemului real analizat.

Modelarea unei baze de date permite trecerea de la percepția unor fapte din lumea reală la reprezentarea lor prin date. Modelul de date trebuie să reflecte fidel fenomene ale lumii reale, să urmărească evoluția acestei lumi și comunicarea dintre fenomenele lumii reale.

Modelul trebuie să asigure conceptele de bază care permit proiectantului bazei de date și utilizatorilor să comunice, fără ambiguități, cunoștințele lor privind funcționarea și organizarea modelului real analizat. Prin urmare, un model de date trebuie să reprezinte datele și să le facă înțelese.

În esență, modelul de date are trei componente:

- o mulțime de reguli conform cărora sunt construite bazele de date (partea structurală);
- o mulțime de operații permise asupra datelor, care sunt utilizate pentru reactualizarea sau regăsirea datelor (partea de prelucrare);
- o mulțime de reguli de integritate, care asigură coerența datelor.

Abordarea generală a problemei modelării semantice a datelor se face în patru etape.

- Se identifică o mulțime de concepte semantice care sunt utile în descrierea lumii reale. Se presupune că lumea reală (modelul real analizat) este formată din entități care au anumite proprietăți, că fiecare entitate are o identitate, că există legături, corelații între entități. Conceptul de corelație, ca și cel de entitate, este util, în mod intuitiv, la descrierea modelului.
- Se caută o mulțime de obiecte formale, simbolice care sunt utilizate

pentru reprezentarea conceptelor semantice anterioare.

- Se dau reguli de integritate formale și generale (constrângeri) care să reflecte restricțiile la care este supus modelul.
- Se definește o mulțime de operatori formali prin care pot fi prelucrate și analizate obiectele formale.

Modelul entitate-relație

Una dintre cele mai cunoscute abordări ale modelării semantice (cu siguranță una dintre cele mai utilizate) este cea bazată pe **modelul entitate-relație** (E/R). Acesta a fost introdus de către P.P. Chen în 1976 și rafinat de atunci în diverse moduri de către acesta și de mulți alți cercetători, ca un model de date conceptual, pentru a ușura proiectarea bazelor de date. Pentru reprezentarea grafică a modelului sunt utilizate diagramele E/R, care sunt modele neformalizate pentru reprezentarea unui model, unui sistem din lumea reală.

Diagramele E/R constituie o tehnică de reprezentare a structurii logice a bazei de date, într-o manieră grafică. Aceste diagrame oferă un mijloc simplu și inteligibil de comunicare a caracteristicilor importante ale designului unei anumite baze de date.

Diagrama E/R este un model de date conceptual de nivel înalt, independent de platforma *hardware* utilizată și de tipul SGBD-ului. Modelul este constituit din concepte care descriu structura bazei de date și tranzacțiile de regăsire sau reactualizare asociate.

Popularitatea modelului E/R ca modalitate de abordare a proiectării bazelor de date poate fi atribuită în principal tehnicii de realizare a diagramelor E/R. Această tehnică, ca și modelul E/R însuși, a evoluat de-a lungul timpului datorită noilor problematice care au apărut în proiectarea bazelor de date.

Baza de date poate fi definită ca o mulțime de date ce modelează un sistem real. Acest sistem este format din obiecte legate între ele. Modelul E/R împarte elementele unui sistem real în două categorii: entități și relații (legături, asocieri) între aceste entități. Entitățile și legăturile au anumite caracteristici, numite atribute. Nu trebuie confundat conceptul de relație, în sensul de asociere, care intervine în definirea diagramei E/R cu conceptul de relație care este specific modelului relațional.

!!! Studii de caz, exemple – prezentate la curs!

Deficiențe ale modelului E/R

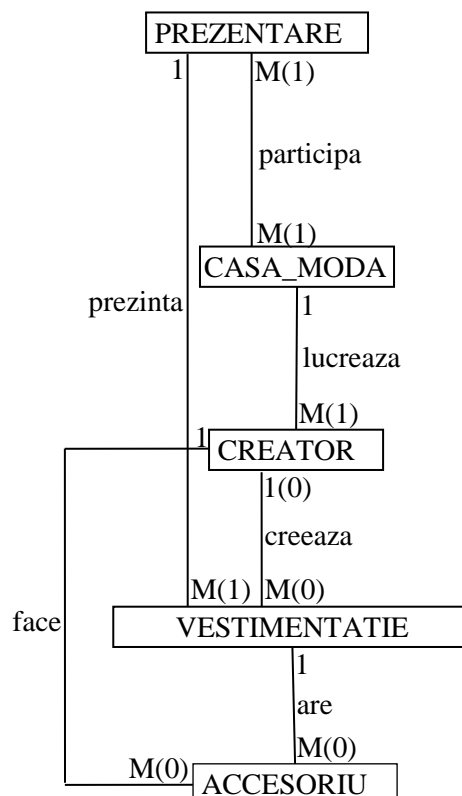
În proiectarea unui model de date pot apărea diverse probleme datorită unei interpretări eronate a sensului unei relații. Aceste probleme sunt denumite **capcane de conectare**.

Unele dintre aceste capcane pot să nu fie semnificative pentru modelul particular considerat, în timp ce altele cer restructurarea modelului. Există două clase importante de capcane: de întrerupere și în evantai.

O capcană de întrerupere poate să apară acolo unde modelul sugerează existența unei relații între entități, dar nu există o cale între anumite apariții ale entităților. Această capcană poate să apară acolo unde există o relație cu participare parțială (0 la cardinalitatea minimă), care face parte din calea dintre entitățile ce sunt legate.

O capcană în evantai poate să apară acolo unde modelul ia în considerare o relație între entități, dar calea dintre anumite apariții ale entităților este ambiguă. Aceste capcane apar când două sau mai multe relații *one_to_many* provin din aceeași entitate.

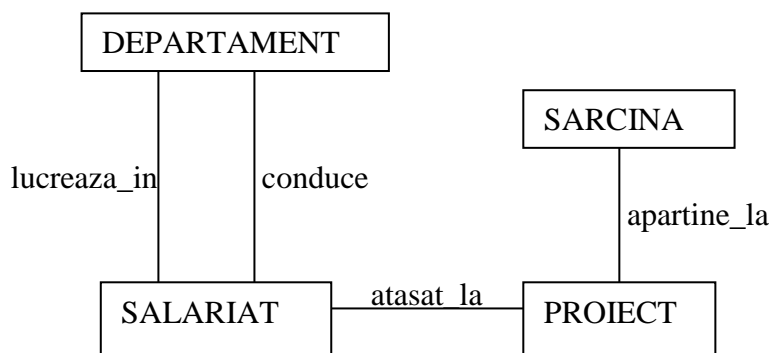
Practic, aceste capcane generează situațiile în care, așa cum a fost proiectat modelul de date, el nu poate să răspundă la anumite interogări. De exemplu, pentru a afla pentru ce prezentare de modă a fost creată o anumită vestimentație, a fost necesară introducerea unei legături între entitățile PREZENTARE și VESTIMENTATIE, care însă a generat redundanță în modelul de date:



Diagrame entitate-relație

Diagrama E/R – model neformalizat pentru reprezentarea unui sistem din lumea reală. Este un model de date conceptual de nivel înalt dezvoltat de Chen (1976).

Entitate: persoană, loc, concept, activitate, eveniment care este semnificativ pentru ceea ce modelăm.



Observații:

- Entitățile devin tabele în modelele relaționale.
- În general, entitățile se scriu cu litere mari.
- Entitățile sunt substantive, dar nu orice substantiv este o entitate.
- Pentru fiecare entitate este obligatoriu să se dea o descriere detaliată.
- Nu pot exista, în aceeași diagramă, două entități cu același nume, sau o aceeași entitate cu nume diferite.

Cheia primară este un identificator unic în cadrul entității, făcând distincție între valori diferite ale acesteia.

Cheia primară:

- trebuie să fie unică și cunoscută la orice moment;
- trebuie să fie controlată de administratorul bazei;
- trebuie să nu conțină informații descriptive, să fie simplă, fără ambiguități;
- să fie stabilă;
- să fie familiară utilizatorului.

Relație (asociere): o comunicare între două sau mai multe entități. Existența unei relații este subordonată existenței entităților pe care le leagă.

Observații:

- În modelul relațional, relațiile devin tabele speciale sau coloane speciale care referă chei primare.
- Relațiile sunt verbe, dar nu orice verb este o relație.
- Pentru fiecare relație este important să se dea o descriere detaliată.
- În aceeași diagramă pot exista relații diferite cu același nume. În acest caz, le diferențiază entitățile care sunt asociate prin relația respectivă.
- Pentru fiecare relație trebuie stabilită cardinalitatea (maximă și minimă) relației, adică numărul de tuple care aparțin relației.

poate (cardinalitate maximă) → trebuie (cardinalitate minimă)

Exemplu:

Câți salariați **pot** lucra într-un departament? Mulți!

În câte departamente **poate** lucra un salariat? În cel mult unul!

→

Relația SALARIAT_lucraza_in_DEPARTAMENT are cardinalitatea maximă **many-one** (n:1).

Exemplu:

Câți salariați **trebuie** să conducă un departament? Cel puțin unul!

Câte departamente **trebuie** să conducă un salariat? Zero!

→

Relația SALARIAT_conduce_DEPARTAMENT are cardinalitatea minimă **one-zero** (1:0).

Atribut: proprietate descriptivă a unei entități sau a unei relații.

Observații:

- Trebuie făcută distincția între tipul atributului (devine coloană în modelele relaționale) și valoarea acestuia (devine valoare în coloane).
- Atributele sunt substantive, dar nu orice substantiv este atribut.
- Fiecărui atribut trebuie să i se dea o descriere completă (exemple, contraexemple, caracteristici).
- Pentru fiecare atribut trebuie specificat numele, tipul fizic (*integer*, *float*, *char* etc.), valori posibile, valori implicite, reguli de validare, tipuri compuse.

Pentru proiectarea **diagramei entitate-relație** au fost stabilite anumite reguli (nu sunt unice):

1. entitățile sunt reprezentate prin dreptunghiuri;
2. relațiile dintre entități sunt reprezentate prin arce neorientate;
3. atributele care reprezintă chei primare trebuie subliniate sau marcate prin simbolul „#”, plasat la sfârșitul numelui acestor atribute;

4. cardinalitatea minimă este indicată în paranteze, iar cardinalitatea maximă se scrie fără paranteze;
5. nu trebuie specificate toate atributele.

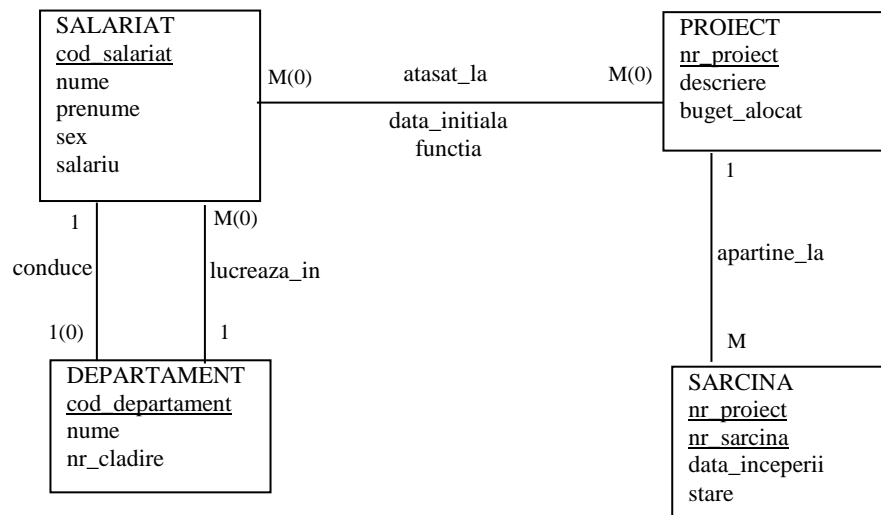


Diagrama E/R.

Cazuri speciale de entități, relații, atribute și modul lor de reprezentare în cadrul diagramei entitate-relație.

1. **Entitate dependentă** – nu poate exista în mod independent (SARCINA depinde de PROIECT). Cheia primară a unei entități dependente include cheia primară a sursei (*nr_proiect*) și cel puțin o descriere a entității (*nr_sarcina*). Entitatea dependentă se desenează prin dreptunghiuri cu linii mai subțiri.
2. Moștenirea atributelor. **Subentitate** (subclasă) – submulțime a unei alte entități, numită **superentitate** (superclasă) (SALARIAT < — > PROGRAMATOR). Subentitatea se desenează prin dreptunghiuri incluse în superentitate. Există o relație între o subentitate și o superentitate, numită ISA, care are cardinalitatea maximă 1:1 și minimă 1:0. Cheile primare, atributele și relațiile unei superentități sunt valabile pentru orice subentitate. Afirmația reciprocă este falsă.
3. Generalizare. Din entități similare care au mai multe atribute comune se pot crea superentități. Aceste superentități conțin atributele comune, iar atributele speciale sunt asignate la subentități. Pentru noile superentități se introduc chei primare artificiale.

4. Specializare. După valorile unor atribute clasificatoare se pot determina **clase**. Un grup de subentități reciproc exclusive definește o clasă. Clasele se aliniază în desen vertical.
5. Într-o diagramă E/R se pot defini relații recursive.
6. Unele relații sunt relative la două entități și le numim de tip 2, iar dacă relațiile implică mai mult de două entități, le vom numi de tip 3. Trei relații de tip 2 sunt diferite de o relație de tip 3. Rupând o relație de tip 3 în trei relații de tip 2, pot apărea informații incorecte.
7. Trebuie excluse din model relațiile indirecte deoarece ele pot conduce la redundanță în baza de date.
8. Atributele derivabile trebuie eliminate și introduse expresii prin care aceste atribute pot fi calculate.
9. Relație sau atribut? Dacă un atribut al unei entități reprezintă cheia primară a unei alte entități, atunci el referă o relație (*cod_departament* în tabelul SALARIAT).
10. Entitate sau relație? Se cercetează cheia primară. Dacă aceasta combină cheile primare a două entități, atunci este vorba de o relație. (cheia primară a relației *asociat_la* combină *cod_salariat* cu *nr_proiect*, prin urmare, *SALARIAT_asociat la_PROIECT* va defini o relație și nu o entitate).
11. Un atribut indirect este inoportun. El nu descrie real relația sau entitatea. Prin urmare, atributele indirecte trebuie reasignate. De fapt, un atribut indirect este un caz special de relație indirectă care trebuie eliminată pentru că introduce redundanță în date (numărul clădirii în care lucrează un salariat este un atribut al entității DEPARTAMENT și nu este o caracteristică a entității SALARIAT).
12. Există atribute opționale, a căror valoare este uneori necunoscută, alteori neaplicabilă. Aceste atribute trebuie introduse la subentități (comisionul pentru deplasare și zona de lucru sunt atribute specifice unui agent teritorial și trebuie introduse la subentitatea AGENT_TERITORIAL).

Algoritmul pentru proiectarea diagramei entitate-relație:

1. identificarea entităților din cadrul sistemului analizat;
2. identificarea relațiilor dintre entități și stabilirea cardinalității;
3. identificarea atributelor aferente entităților și asocierilor dintre entități;
4. stabilirea atributelor de identificare a entităților (stabilirea cheilor).

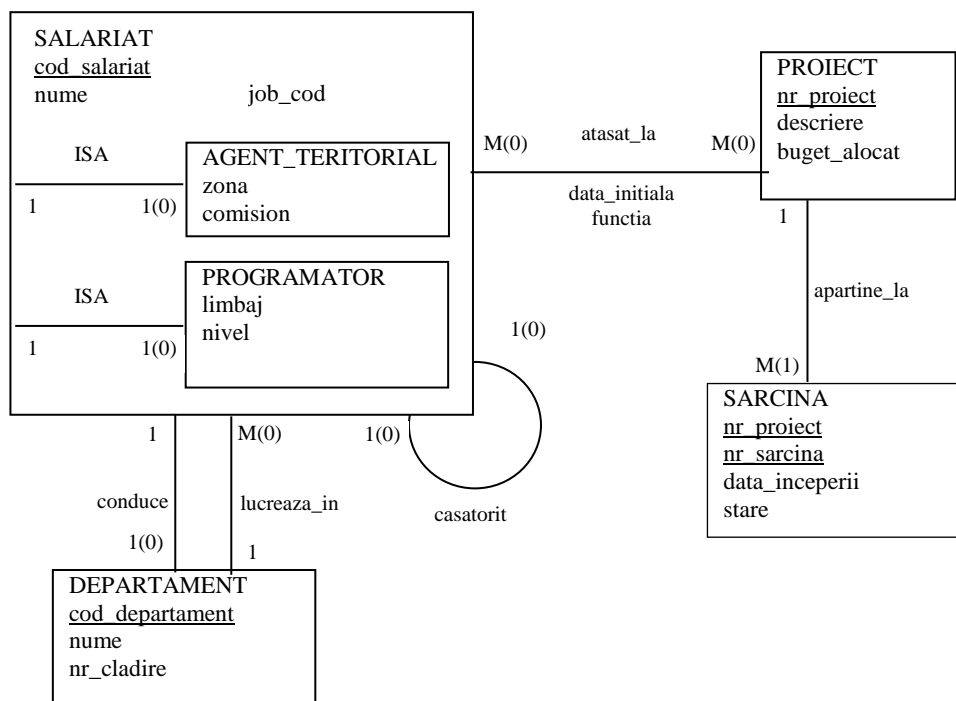


Diagrama E/R.

Modelul EER (modelul E/R extins) = Diagrama E/R + concepte aditionale (subclasă, superclasă, moștenire, specializare, generalizare).

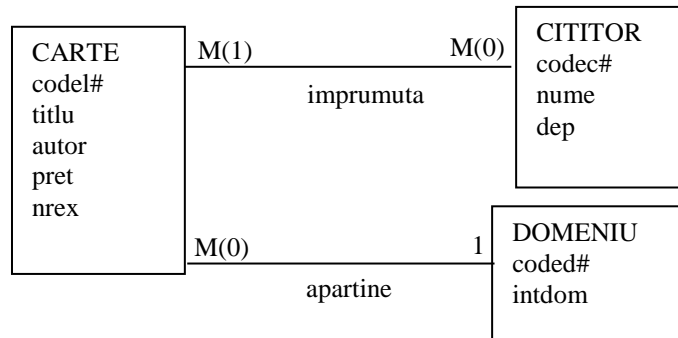
Gestiunea activităților de împrumut dintr-o bibliotecă

S-a presupus (restrictiv) că într-o zi un cititor nu poate împrumuta, de mai multe ori, aceeași carte. Modelul prezintă anomalii (de exemplu, cheia primară de la entitatea carte)! A fost gândit în această manieră cu scop pur didactic. Comentat erori!!!

Entitățile și relațiile care intervin în acest model sunt următoarele:

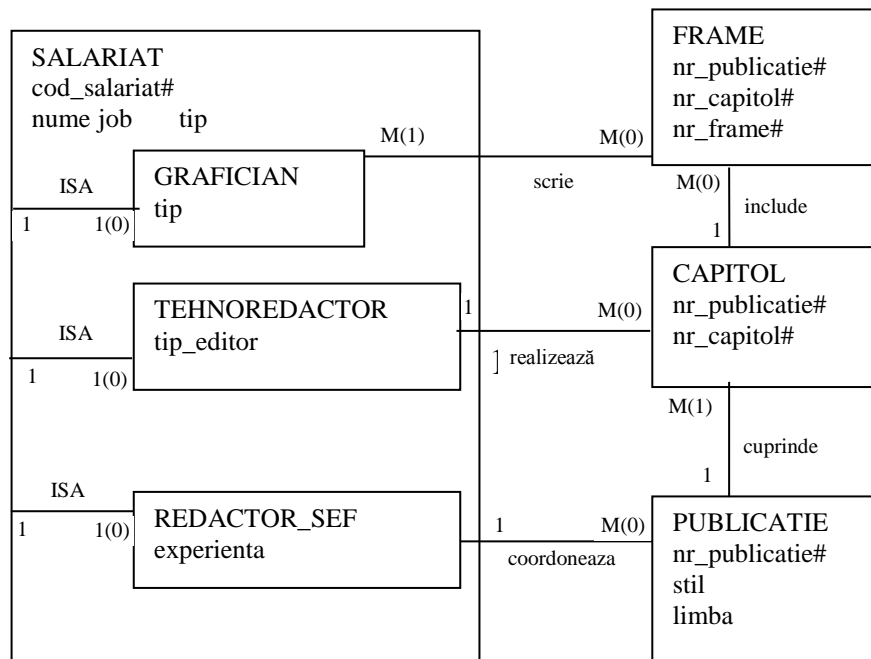
1. CARTE (entitate independentă) – orice carte care se găsește în inventarul bibliotecii. Cheia primară este atributul *codei*.
2. CITITOR (entitate independentă) – orice cititor care poate împrumuta cărți. Cheia primară este atributul *codei*.
3. DOMENIU (entitate independentă) – domeniul căruia îi aparține o carte. Cheia primară este atributul *codei*.
4. IMPRUMUTA – relație având cardinalitatea *m:m* care leagă entitățile CITITOR și CARTE.

5. APARTINE – relație care leagă atributele CARTE și DOMENIU. Relația are cardinalitatea maximă $m:1$, iar cardinalitatea minimă $1:1$.



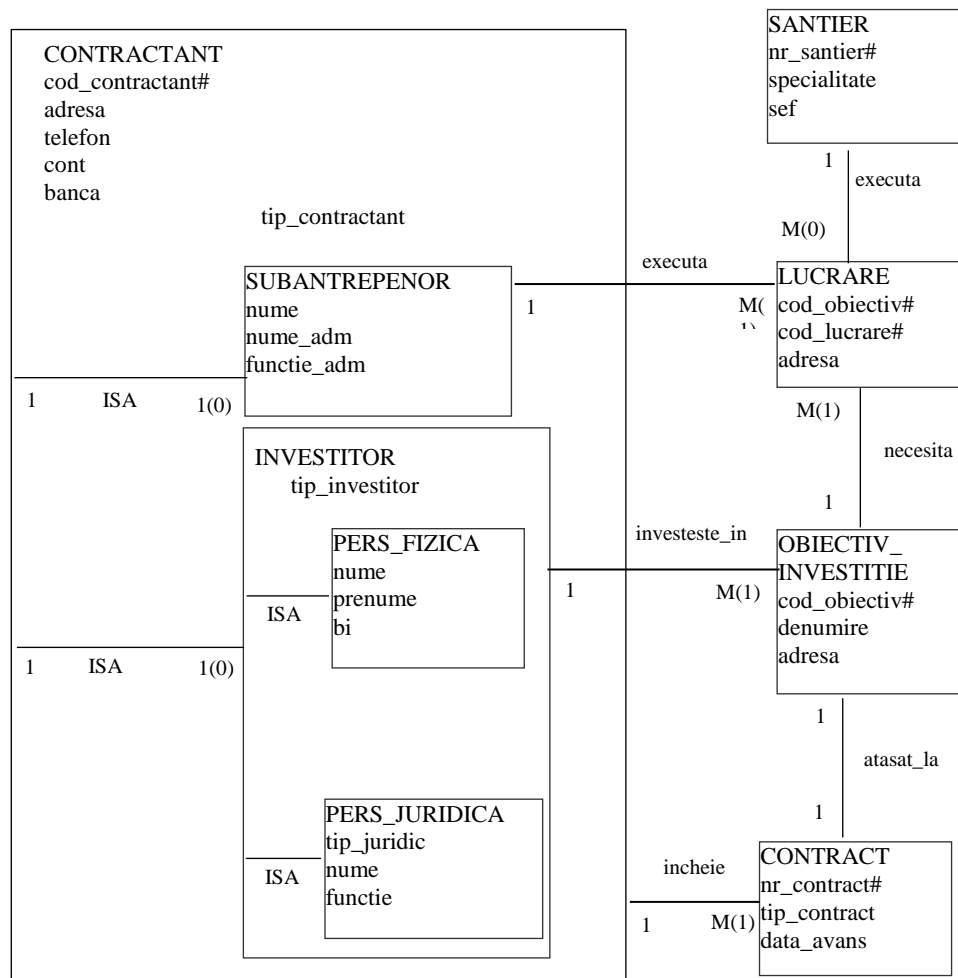
Gestiunea activităților de editare dintr-o editură

Se analizeaza activitatea dintr-o editură referitoare la culegerea textelor, realizarea elementelor grafice, machetarea unor publicații.



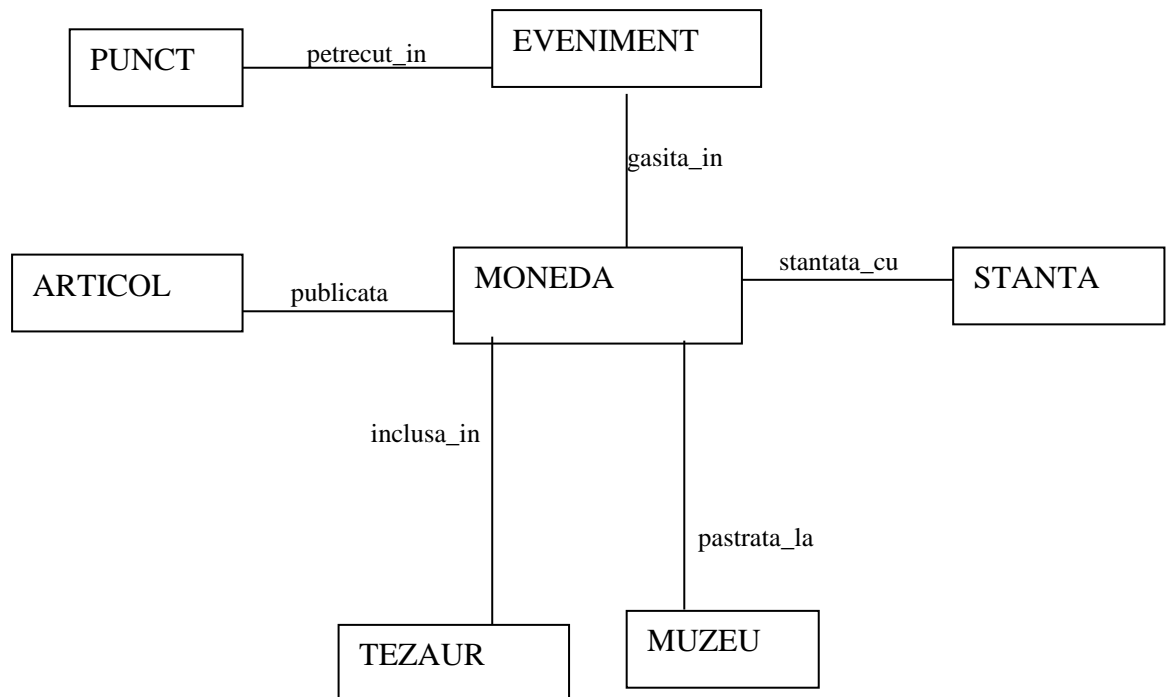
Gestiunea activităților unei firme de construcții

Baza de date construită prin acest model, furnizează informații legate de obiective de execuție, investitori, executanți, șantiere, contracte etc. necesare unui manager al unei firme de construcții.



Vezi erori!

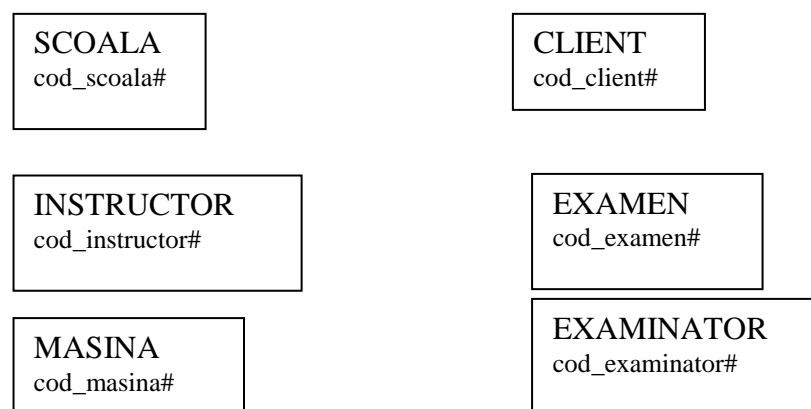
Ordonarea informațiilor cu privire la descoperirile de monede antice din Romania



STANȚA (nr_stanță, împărat emitent, valoare nominală, an emitere, monetăria, legenda de pe avers, legenda de pe revers) == > atribute ale entității **STANTA**

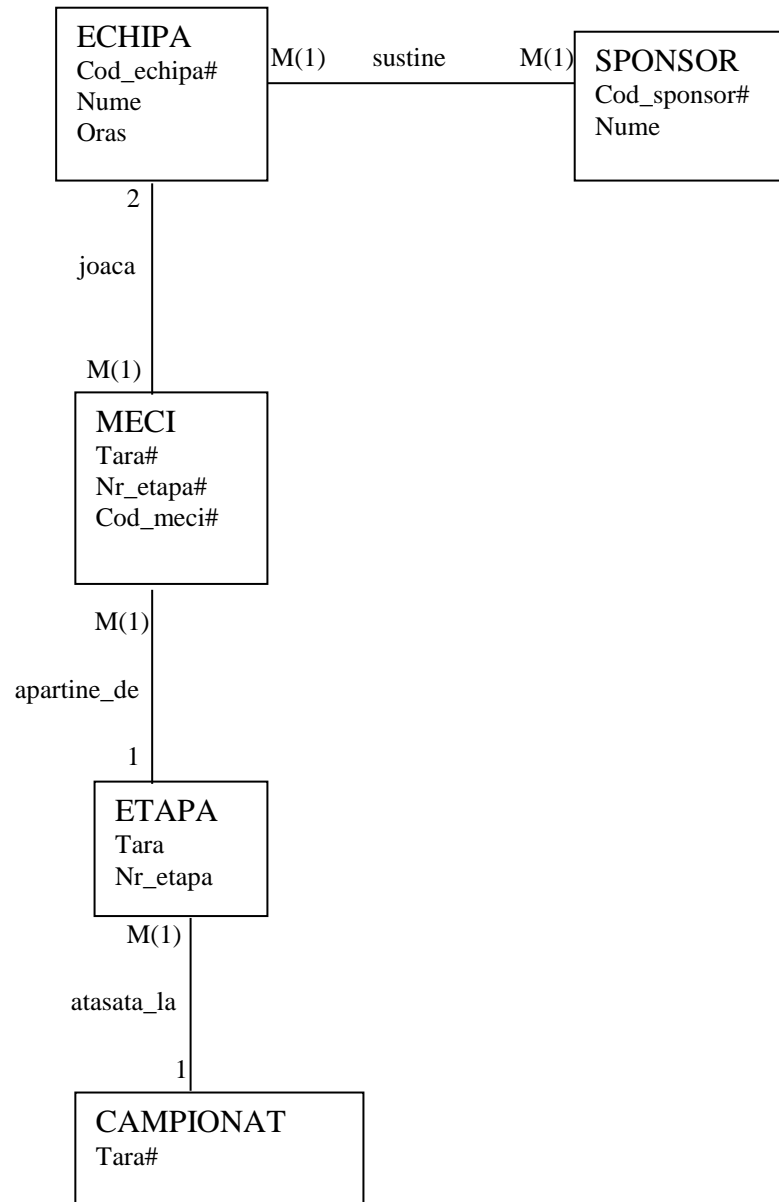
Completați cardinalitatea!

Evidența școlilor de șoferi din Romania



Completați relațiile (*lucreaza_la*, *conduce*, *sustine*, *asista*, *instruieste*) dintre entități și specificați cardinalitatea!

Campionatele de fotbal ale diferitelor țări



Modelul relațional

Modelul relațional a fost conceput și dezvoltat de E.F. **Codd**. El este un model formal de organizare conceptuală a datelor, destinat reprezentării legăturilor dintre date, bazat pe teoria matematică a relațiilor. Modelul relațional este alcătuit numai din relații și prin urmare, orice interogare asupra bazei de date este tot o relație. Cercetarea în domeniu → 3 mari proiecte (*System R*, *INGRES*, *PRTV*)

Calități:

- este simplu;
- riguros din punct de vedere matematic;
- nu este orientat spre sistemul de calcul.

Modalități pentru **definirea unui SGBD relațional**:

- prezentarea datelor în tabele supuse anumitor operații de tip proiecție, selecție, reuniune, compunere, intersecție etc.
- un sistem de baze de date ce suportă un limbaj de tip SQL – *Structured Query Language*;
- un sistem de baze de date care respectă principiile modelului relațional introdus de E.F. Codd.

Caracteristicile unui model relațional:

- structura relațională a datelor;
- operatorii modelului relațional;
- regulile de integritate care guvernează folosirea cheilor în model.

Aceste trei elemente corespund celor trei componente ale ingineriei *software*: informație, proces, integritate.

Structura datelor

Definirea noțiunilor de domeniu, relație, schemă relațională, valoare *null* și tabel vizualizare (*view*).

Conceptele utilizate pentru a descrie formal, uzual sau fizic elementele de bază ale organizării datelor sunt date în următorul tabel:

Formal	Uzual	Fizic
relație	tablou	fișier
tuplu	linie	înregistrare
atribut	coloană	câmp
domeniu	tip de dată	tip de dată

Domeniu – mulțime de valori care poate fi definită fie enumerând elementele componente, fie definind o proprietate distinctivă a domeniului valorilor.

Fie D_1, D_2, \dots, D_n domenii finite, nu neapărat disjuncte. **Produsul cartezian** $D_1 \times D_2 \times \dots \times D_n$ al domeniilor D_1, D_2, \dots, D_n este definit de mulțimea tuplurilor (V_1, V_2, \dots, V_n) , unde $V_1 \in D_1, V_2 \in D_2, \dots, V_n \in D_n$. Numărul n definește **aritatea tuplului**.

O **relație** R pe mulțimile D_1, D_2, \dots, D_n este o submulțime a produsului cartezian $D_1 \times D_2 \times \dots \times D_n$, deci este o mulțime de tupluri. Caracteristicile unei relații \rightarrow comentat curs!

Definirea unei relații se referă la mulțimi care variază în timp. Pentru a caracteriza o relație este necesară existența un element invariant în timp: structura relației (schema relațională). Mulțimea numelor atributelor corespunzătoare unei relații R definește **schema relațională** a relației respective. Vom nota schema relațională prin $R(A_1, A_2, \dots, A_n)$. Exemplu!

Putem reprezenta o relație printr-un tabel bidimensional în care fiecare linie corespunde unui tuplu și fiecare coloană corespunde unui domeniu din produsul cartezian. O coloană corespunde de fapt unui atribut. Numărul atributelor definește **gradul** relației, iar numărul de tupluri din relație definește **cardinalitatea** relației.

Exemplu (crearea unui tabel în *SQL*):

```
CREATE TABLE salariat (
    cod_salariat      SMALLINT,
    nume              VARCHAR(25),
    prenume           VARCHAR(20),
    sex               CHAR(1),
    salariu           INTEGER,
    sot               SMALLINT,
    job_cod            VARCHAR(6),
    cod_departament   SMALLINT );
```

Când se inserează tupluri într-o relație, de multe ori un atribut este necunoscut sau neaplicabil. Pentru a reprezenta acest atribut a fost introdusă o valoare convențională în relație, și anume valoarea **null**.

Este necesară o aritmetică și o logică nouă care să cuprindă acest element. Rezultatul operatorilor aritmetici sau logici este **null** când unul din argumente este **null**. Comentat excepții! Prin urmare, „**null** = **null**“ are valoarea **null**, iar \neg **null** este **null**.

AND	T	F	Null	OR	T	F	Null
T	T	F	Null	T	T	T	T
F	F	F	F	F	T	F	Null
Null	Null	F	Null	Null	T	Null	Null

Tabele de adevăr pentru operatorii AND și OR.

Tabelul **vizualizare** (*view*, filtru, relație virtuală, vedere) constituie un filtru relativ la unul sau mai multe tabele, care conține numai informația necesară unei anumite abordări sau aplicații. Securitate, reactualizări → comentat la curs!

Vizualizarea este virtuală deoarece datele pe care le conține nu sunt în realitate memorate într-o bază de date. Este memorată numai definiția vizualizării. Vizualizarea nu este definită explicit, ca relațiile de bază, prin mulțimea tuplurilor componente, ci implicit, pe baza altor relații prin intermediul unor expresii relaționale. Stabilirea efectivă a tuplurilor care compun vizualizarea se realizează prin evaluarea expresiei atunci când utilizatorul se referă la acest tabel.

Exemplu (crearea unei vizualizări în *SQL*):

```
CREATE VIEW programator(nume,departament)
AS SELECT  nume,cod_departament
FROM      salariat
WHERE     job_cod='programator' ;
```

Reguli de integritate → aserțiuni pe care datele conținute în baza de date trebuie să le satisfacă.

Trebuie făcută distincția între:

- regulile structurale inerente modelării datelor;
- regulile de funcționare specifice unei aplicații particulare.

Există trei tipuri de constrângeri structurale (de cheie, de referință, de entitate) ce constituie mulțimea minimală de reguli de integritate pe care **trebuie** să le respecte un SGBD relațional. Restricțiile de integritate minimale sunt definite în raport cu noțiunea de cheie a unei relații.

O mulțime minimală de attribute ale căror valori identifică unic un tuplu într-o relație reprezintă o **cheie** pentru relația respectivă.

Fiecare relație are cel puțin o cheie. Una dintre cheile candidat va fi aleasă pentru a identifica efectiv tupluri și ea va primi numele de **cheie primară**. Cheia primară nu poate fi reactualizată. Attributele care reprezintă cheia primară sunt fie subliniate, fie urmate de semnul #.

O cheie identifică linii și este diferită de un index care localizează liniile. O **cheie secundară** este folosită ca index pentru a accesa tupluri. Un grup de attribute din cadrul unei relații care conține o cheie a relației poartă numele de **supercheie**.

Fie schemele relaționale $R1(P1, S1)$ și $R2(S1, S2)$, unde $P1$ este cheie primară pentru $R1$, $S1$ este cheie secundară pentru $R1$, iar $S1$ este cheie primară pentru $R2$. În acest caz, vom spune că $S1$ este **cheie externă** (cheie străină) pentru $R1$.

Modelul relațional respectă trei reguli de integritate structurală.

- ❑ **Regula 1** – unicitatea cheii. Cheia primară trebuie să fie unică și minimală.
- ❑ **Regula 2** – integritatea entității. Atributele cheii primare trebuie să fie diferite de valoarea *null*.
- ❑ **Regula 3** – integritatea referirii. O cheie externă trebuie să fie ori *null* în întregime, ori să corespundă unei valori a cheii primare asociate.

Proiectarea modelului relațional (exemple → curs!)

Transformarea entităților

- ❑ Entitățile independente devin **tabele independente**. Cheia primară nu conține chei externe.
- ❑ Entitățile dependente devin **tabele dependente**. Cheia primară a entităților dependente conține cheia primară a entității de care depinde (cheie externă) plus unul sau mai multe atribute adiționale.
- ❑ Subentitățile devin **subtabele**. Cheia externă se referă la supertabel, iar cheia primară este această cheie externă (cheia primară a subentității PROGRAMATOR este *cod_salariat* care este o cheie externă).

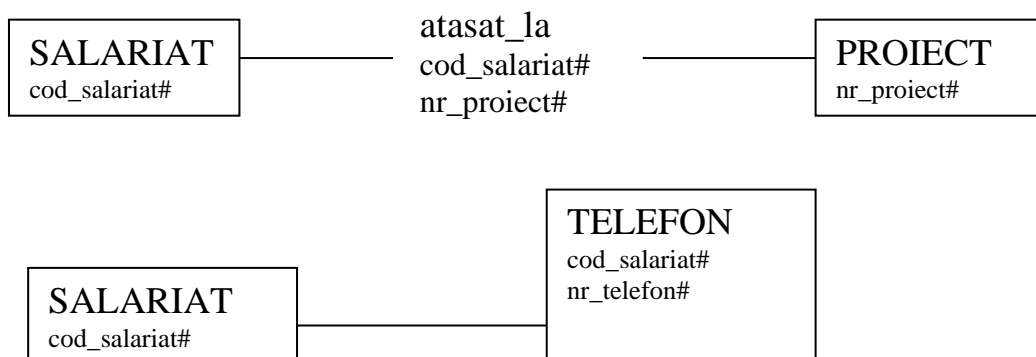
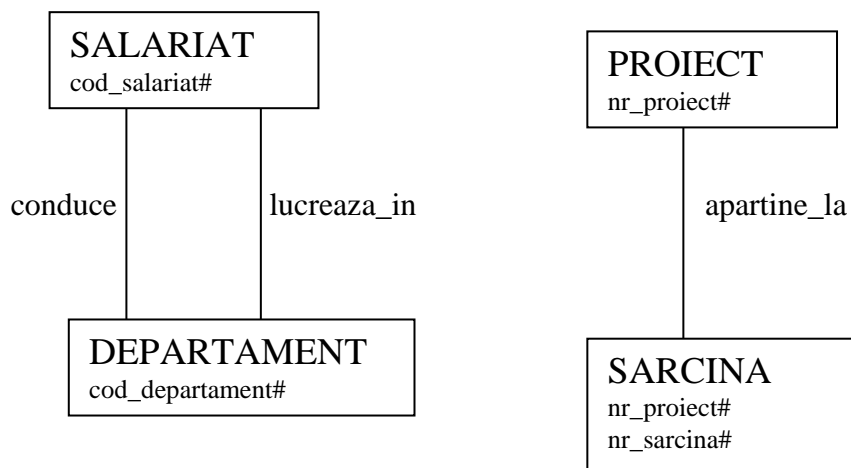
Transformarea relațiilor

- ❑ Relațiile 1:1 și 1:*n* devin chei externe. Relația *conduce* devine coloană în tabelul DEPARTAMENT, iar relația *lucreaza_in* devine coloană în tabelul SALARIAT. Simbolul „×” indică plasamentul cheii externe, iar simbolul „⌋” exprimă faptul că această cheie externă este conținută în cheia primară. Relația 1:1 plasează cheia externă în tabelul cu mai puține linii.
- ❑ Relația *m:n* devine un tabel special, numit tabel **asociativ**, care are două chei externe pentru cele două tabele asociate. Cheia primară este compunerea acestor două chei externe plus eventuale coloane adiționale. Tabelul se desenează punctat.
- ❑ Relațiile de tip trei devin tabele asociative. Cheia primară este compunerea a trei chei externe plus eventuale coloane adiționale.

Transformarea atributelor

- ❑ Un atribut singular devine o coloană.
- ❑ Atributele multiple devin tabele dependente ce conțin cheia primară a entității și atributul multiplu. Cheia primară este o cheie externă, plus una sau mai multe coloane adiționale.

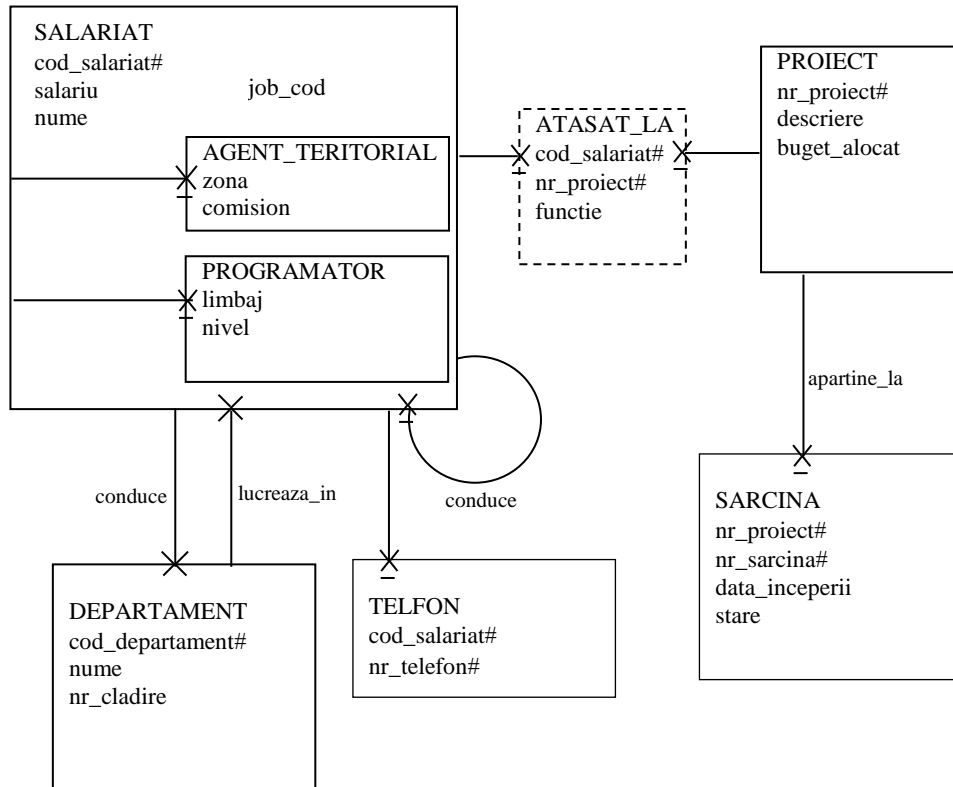
- Entitățile devin tabele, iar atributele lor devin coloane în aceste tabele. Ce devin atributele relațiilor? Pentru relații 1:1 și 1:n, atributele relațiilor vor aparține tabelului care conține cheia externă, iar pentru relații $m:n$ și de tipul trei, atributele vor fi plasate în tabelele asociative.



Cele patru tipuri de tabele (independente, dependente, subtabele și asociative) se deosebesc prin structura cheii primare.

Tabel	Reprezintă	Cheie primară
Independent	entitate independentă	nu conține chei externe
Subtabel	Subentitate	o cheie externă
Dependent	entitate dependentă	o cheie externă și una sau mai multe coloane adiționale
	atribute multiple	
Asociativ	relație $m:n$	două sau mai multe chei externe și (opțional) coloane adiționale
	relații de tip 3	

Diagrama conceptuală pentru proiectarea modelului relațional comentat a fost construită din diagrama E/R prin adăugarea tabelelor asociative și prin marcarea cheilor externe.



Schemele relaționale corespunzătoare acestei diagrame conceptuale sunt următoarele:

- SALARIAT(cod_salariat#, nume, prenume, sex, job_cod, cod_sot, forma_plata, nr_depart);
- DEPARTAMENT(cod_departament#, nume, numar_cladire, cod_sal);
- ATASAT_LA(cod_salariat#, nr_proiect#, functia);
- PROIECT(nr_proiect#, descriere, buget_alocat);
- SARCINA(nr_proiect#, nr_sarcina, data_inceperii, stare);
- AGENT_TERITORIAL(cod_salariat#, zona, comision);
- PROGRAMATOR(cod_salariat#, limbaj, nivel);
- TELEFON(cod_salariat#, nr_telefon#).

Gestiunea activităților unei firme de construcții

CONTRACTANT(cod_contractant#, adresa, telefon, cont, banca, tip_contractant);

SUBANTREPRENOR(cod_contractant#, nume, nr_reg_comert, nume_adm, functie_adm);

INVESTITOR(cod_contractant#, tip_investitor);

PERS_FIZICA(cod_contractant#, nume, prenume, bi);

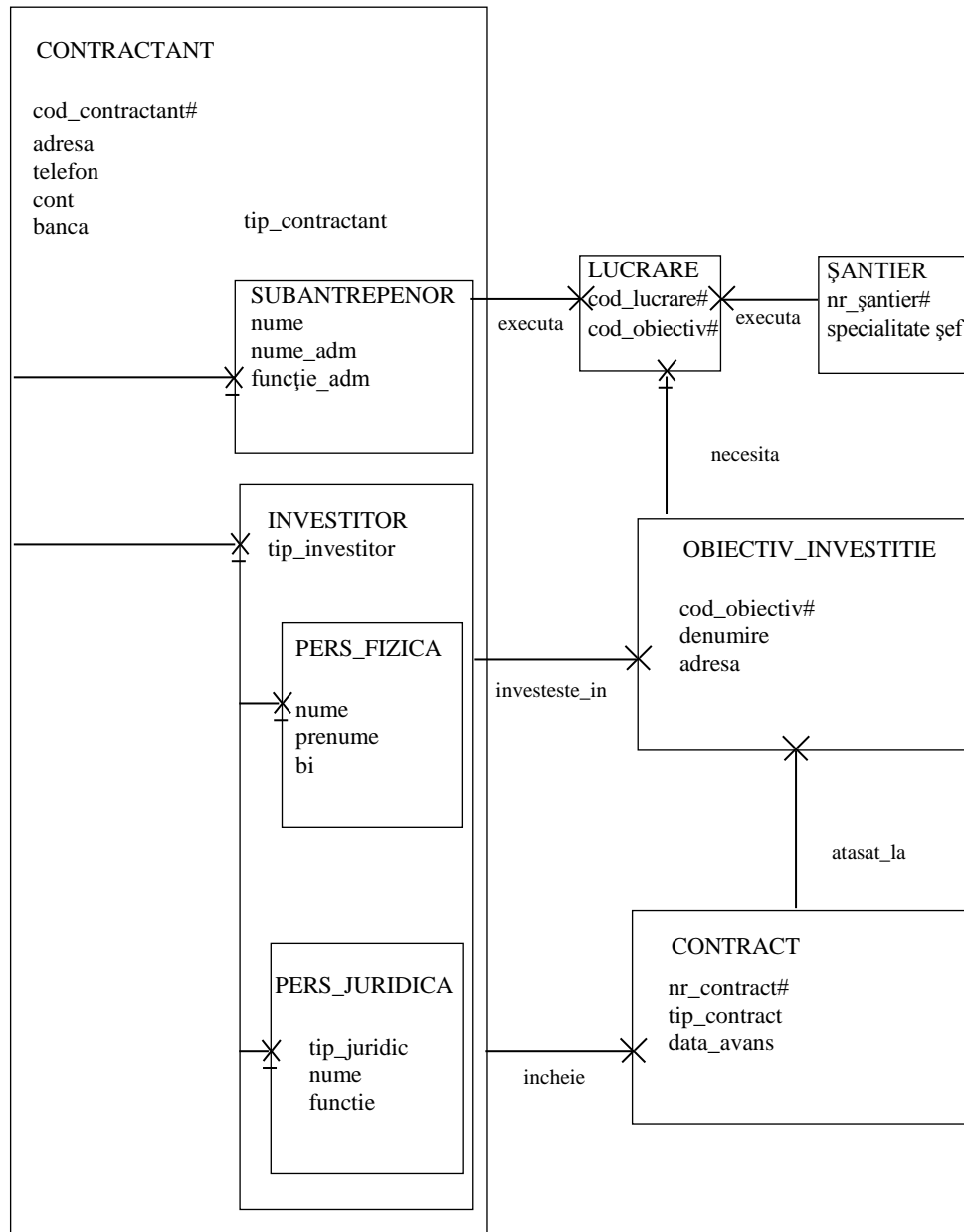
PERS_JURIDICA(cod_contractant#, tip_juridic, nume, reprez_legal, functie);

CONTRACT(nr_contract#, tip_contract, data_incheiere, garantie, val_investitie, durate_executie, cont, banca, perioada, avans, data_avans, cod_contractant);

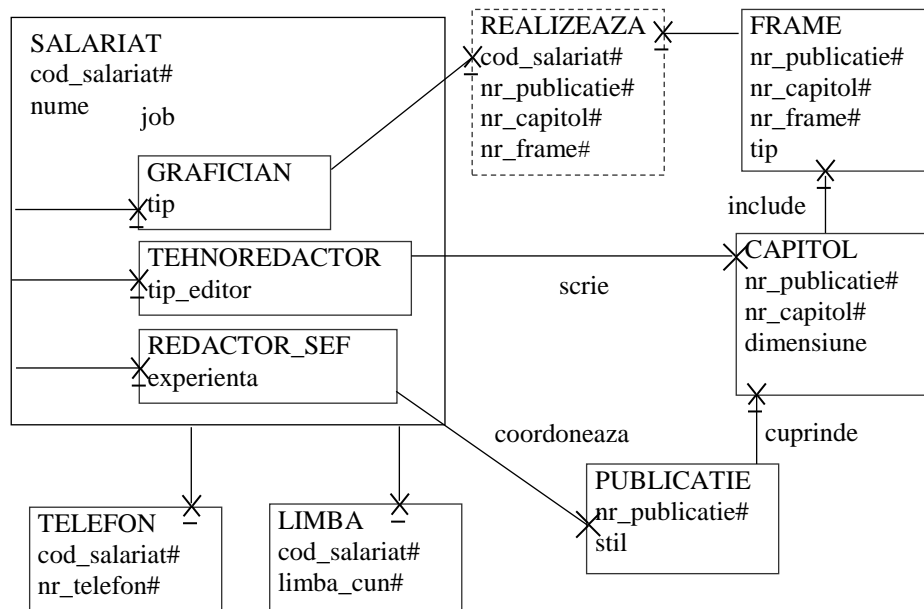
SANTIER(nr_santier#, specialitate, sef);

OBIECTIV_INVESTITIE(cod_obiectiv#, denumire, adresa, adc, nr_cert_urb, nr_aut_constr, nr_contract, cod_contractant);

LUCRARE(cod_lucrare#, cod_obiectiv#, tip_lucrare, nume, data_inc, data_sf, nr_santier, cod_contractant);



Gestiunea activităților de editare dintr-o editură



SALARIAT(cod_salariat#, nume, prenume, vechime, salariu, job);

GRAFICIAN(cod_salariat#, tip);

TEHNOREDACTOR(cod_salariat#, tip_platforma, tip_editor, viteza);

REDACTOR_SEF(cod_salariat#, experienta);

LIMBA(cod_salariat#, limba_cunos#);

TELEFON(cod_salariat#, nr_telefon#);

REALIZEAZA(cod_salariat#, nr_frame#, nr_publicatie#, nr_capitol#, data_inc, data_lim);

FRAME(nr_frame#, nr_publicatie#, nr_capitol#, tip, dim, format);

CAPITOL(nr_publicatie#, nr_capitol#, dimensiune, cod_salariat);

PUBLICATIE(nr_publicatie#, stil, beneficiar, autor, cod_salariat, cost, titlu, limba).

Exemple → curs!