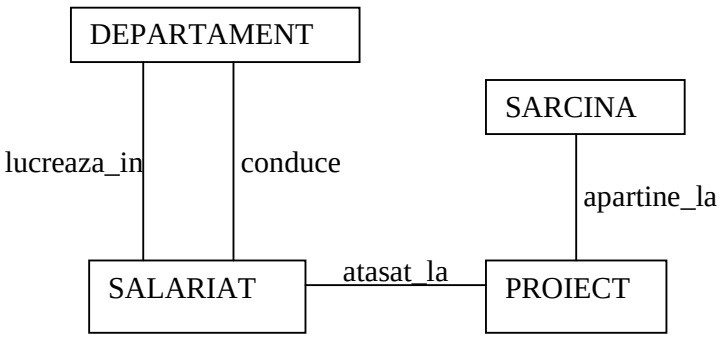
Material incomplet, perfectibil!!! Vezi curs + bibliografie!!!

Diagrame entitate-relaţie

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.



Observaţii:

- Entităţile devin tabele în modelele relaţionale.
- În general, entităţile se scriu cu litere mari.
- Entităţile sunt substantive, dar nu orice substantiv este o entitate.
- Pentru fiecare entitate este obligatoriu să se dea o descriere detaliată.
- Nu pot exista, în aceeaşi diagramă, două entităţi cu acelaşi nume, sau o aceeaşi entitate cu nume diferite.

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

Cheia primară:

- trebuie să fie unică şi cunoscută la orice moment;
- trebuie să fie controlată de administratorul bazei;
- trebuie să nu conţină informaţii descriptive, să fie simplă, fără ambiguităţi;
- să fie stabilă;
- să fie familiară utilizatorului.

Relaţie (asociere): o comunicare între două sau mai multe entităţi. Existenţa unei relaţii este subordonată existenţei entităţilor pe care le leagă.

Observaţii:

- În modelul relaţional, relaţiile devin tabele speciale sau coloane speciale care referă chei primare.
- Relaţiile sunt verbe, dar nu orice verb este o relaţie.
- Pentru fiecare relaţie este important să se dea o descriere detaliată.
- În aceeaşi diagramă pot exista relaţii diferite cu acelaşi nume. În acest caz, le diferenţiază entităţile care sunt asociate prin relaţia respectivă.
- Pentru fiecare relaţie trebuie stabilită cardinalitatea (maximă şi minimă) relaţiei, adică numărul de tupluri ce aparţin relaţiei.

poate (cardinalitate maximă) → trebuie (cardinalitate minima)

Exemplu:

Câţi salariaţi **pot** lucra într-un departament? Mulţi!
În câte departamente **poate** lucra un salariat? In cel mult unul!

→

Relaţia SALARIAT_lucreaza_in_DEPARTAMENT are cardinalitatea maximă ***many-one*** (n:1).

Exemplu:

Câţi salariaţi **trebuie** să conducă un departament? Cel puţin unul!
Câte departamente **trebuie** să conducă un salariat? Zero!

→

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

Atribut: proprietate descriptivă a unei entităţi sau a unei relaţii.

Observaţii:

- Trebuie făcută distincţia între atribut (devine coloană în modelele relaţionale) şi 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-relaţie** au fost stabilite anumite reguli (nu sunt unice):

1. entităţile sunt reprezentate prin dreptunghiuri;
2. relaţiile dintre entităţi sunt reprezentate prin arce neorientate;
3. atributele care reprezintă chei primare trebuie subliniate sau marcate prin simbolul „#“, plasat la sfârşitul numelui acestor atribute;
4. cardinalitatea minimă este indicată în paranteze, iar cardinalitatea maximă se scrie fără paranteze;
5. nu trebuie specificate toate atributele.

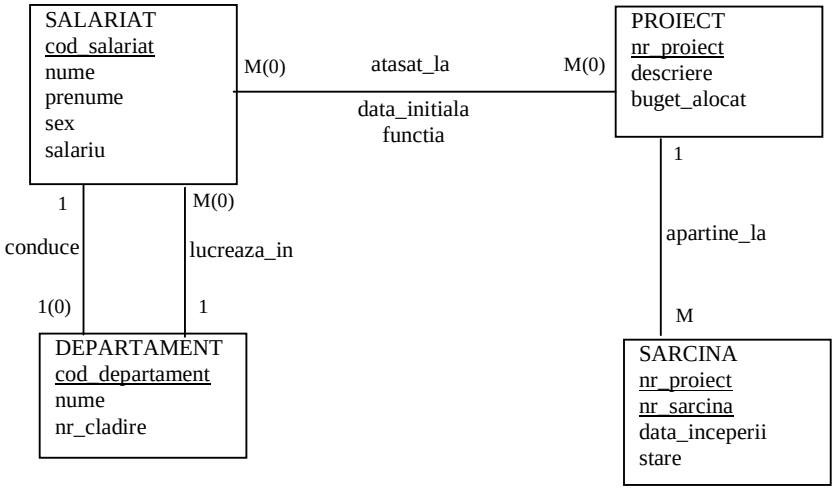


Diagrama E/R.

Cazuri speciale de entităţi, relaţii, atribute şi modul lor de reprezentare în cadrul diagramei entitate-relaţie.

1. **Entitate dependentă** – nu poate exista în mod independent (SARCINA depinde de PROIECT). Cheia primară a unei entităţi dependente include cheia primară a sursei (*nr_proiect*) şi cel puţin o descriere a entităţii (*nr_sarcina*). Entitatea dependentă se desenează prin dreptunghiuri cu linii mai subţiri.
2. Moştenirea atributelor. **Subentitate** (subclasă) – submulţime a unei alte entităţi, numită **superentitate** (superclasă) (SALARIAT < — > PROGRAMATOR). Subentitatea se desenează prin dreptunghiuri incluse în superentitate. Există o relaţie între o subentitate şi o

superentitate, numită ISA, care are cardinalitatea maximă 1:1 şi minimă 1:0. Cheile primare, atributele şi relaţiile unei superentităţi sunt valabile pentru orice subentitate. Afirmaţia reciprocă este falsă.

3. Generalizare. Din entităţi similare care au mai multe atribute comune se pot crea superentităţi. Aceste superentităţi conţin atributele comune, iar atributele speciale sunt asignate la subentităţi. Pentru noile superentităţi se introduc chei primare artificiale.

4. Specializare. După valorile unor atribute clasificatoare se pot determina **clase**. Un grup de subentităţi reciproc exclusive defineşte o clasă. Clasele se aliniază în desen vertical.

5. Într-o diagramă E/R se pot defini relaţii recursive.

6. Unele relaţii sunt relative la două entităţi şi le numim de tip 2, iar dacă relaţiile implică mai mult de două entităţi, le vom numi de tip 3. Trei relaţii de tip 2 sunt diferite de o relaţie de tip 3. Rupând o relaţie de tip 3 în trei relaţii de tip 2, pot apărea informaţii incorecte.

7. Trebuie excluse din model relaţiile indirecte deoarece ele pot conduce la redundanţă în baza de date.

8. Atributele derivabile trebuie eliminate şi introduse expresii prin care aceste atribute pot fi calculate.

9. Relaţie sau atribut? Dacă un atribut al unei entităţi reprezintă cheia primară a unei alte entităţi, atunci el referă o relaţie (*cod_departament* în tabelul SALARIAT).

10. Entitate sau relaţie? Se cercetează cheia primară. Dacă aceasta combină cheile primare a două entităţi, atunci este vorba de o relaţie. (cheia primară a relaţiei *asociat_la* combină *cod_salariat* cu *nr_proiect*, prin urmare, *SALARIAT_asociat la_PROIECT* va defini o relaţie şi nu o entitate).

11. Un atribut indirect este inoportun. El nu descrie real relaţia sau entitatea. Prin urmare, atributele indirecte trebuie reasignate. De fapt, un atribut indirect este un caz special de relaţie indirectă care trebuie eliminată pentru că introduce redundanţă în date (numărul clădirii în care lucrează un salariat este un atribut al entităţii DEPARTAMENT şi nu este o caracteristică a entităţii SALARIAT).

12. Există atribute opţionale, a căror valoare este uneori necunoscută, alteori neaplicabilă. Aceste atribute trebuie introduse la subentităţi (comisionul pentru deplasare şi zona de lucru sunt atribute specifice unui agent teritorial şi trebuie introduse la subentitatea AGENT_TERITORIAL).

Algoritmul pentru proiectarea diagramei entitate-relaţie:

1. identificarea entităţilor din cadrul sistemului analizat;
2. identificarea relaţiilor dintre entităţi şi stabilirea cardinalităţii;
3. identificarea atributelor aferente entităţilor şi asocierilor dintre entităţi;
4. stabilirea atributelor de identificare a entităţilor (stabilirea cheilor).

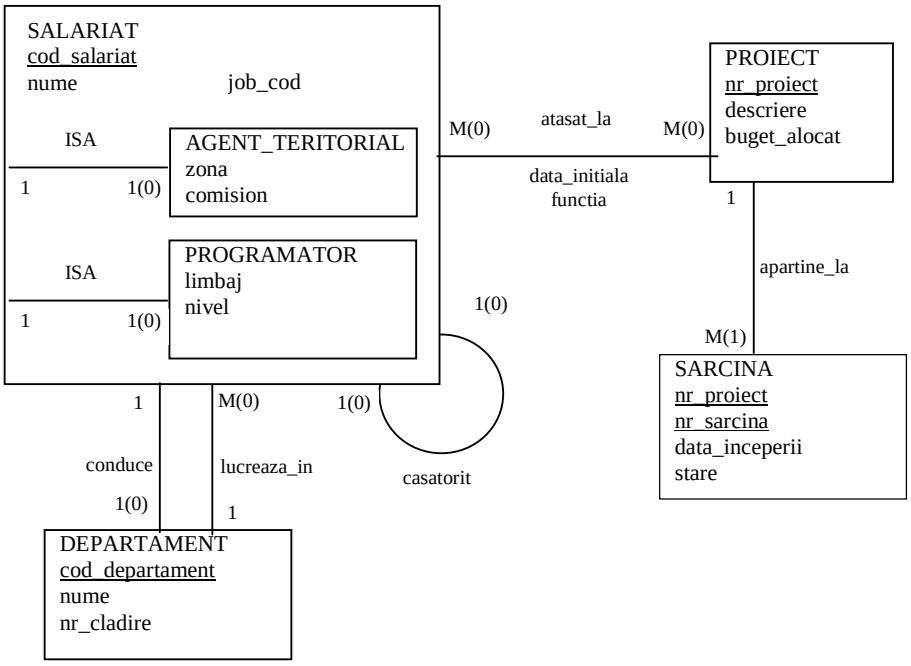
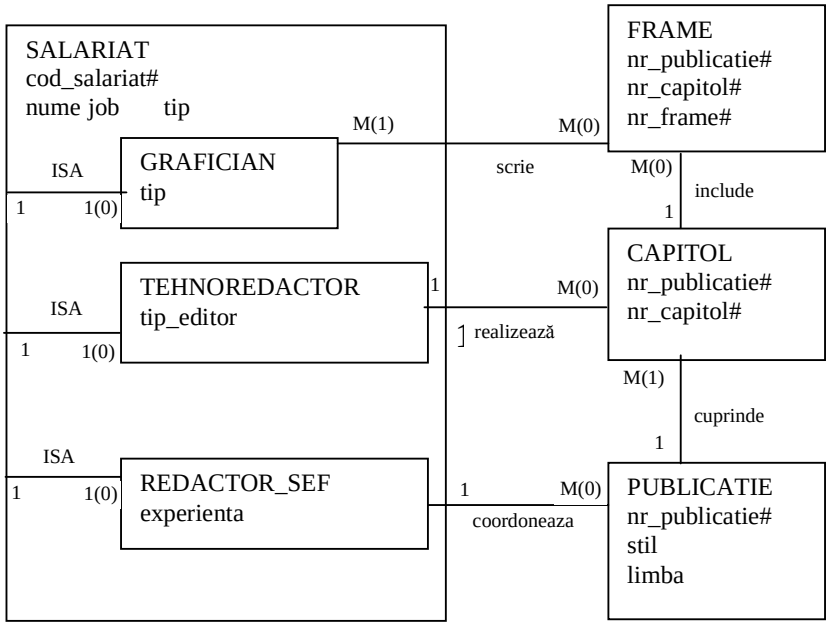


Diagrama E/R.

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

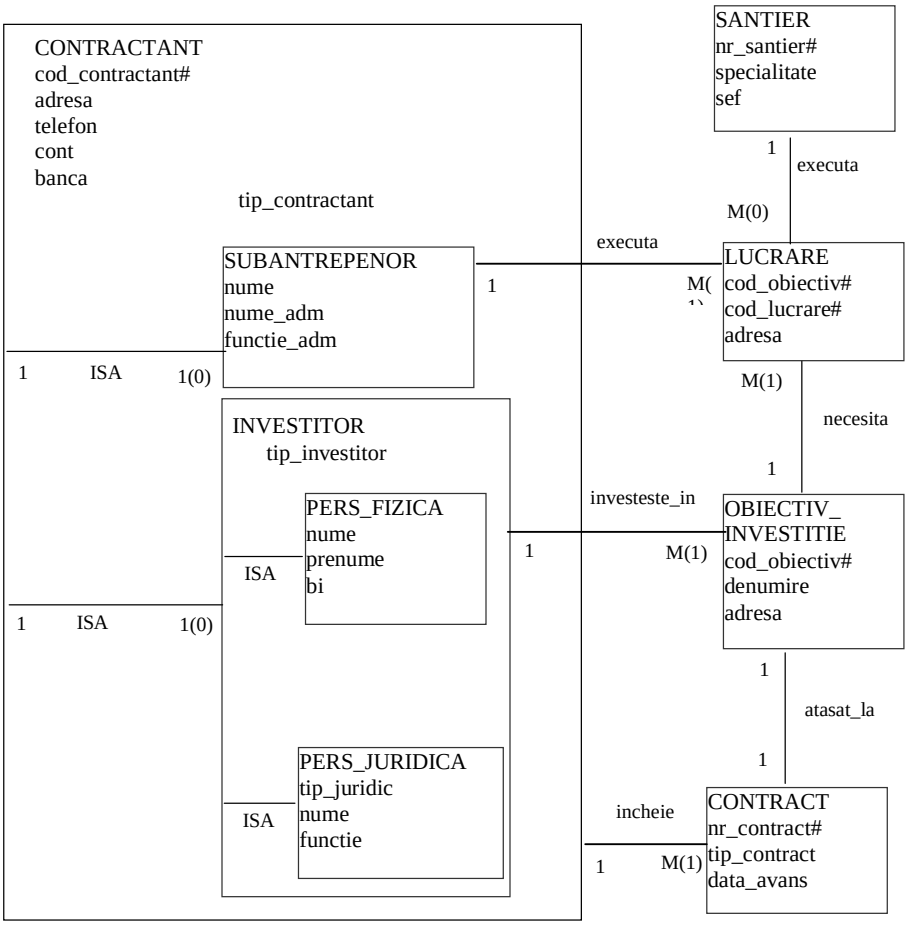
Gestiunea activităţilor de editare dintr-o editură

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



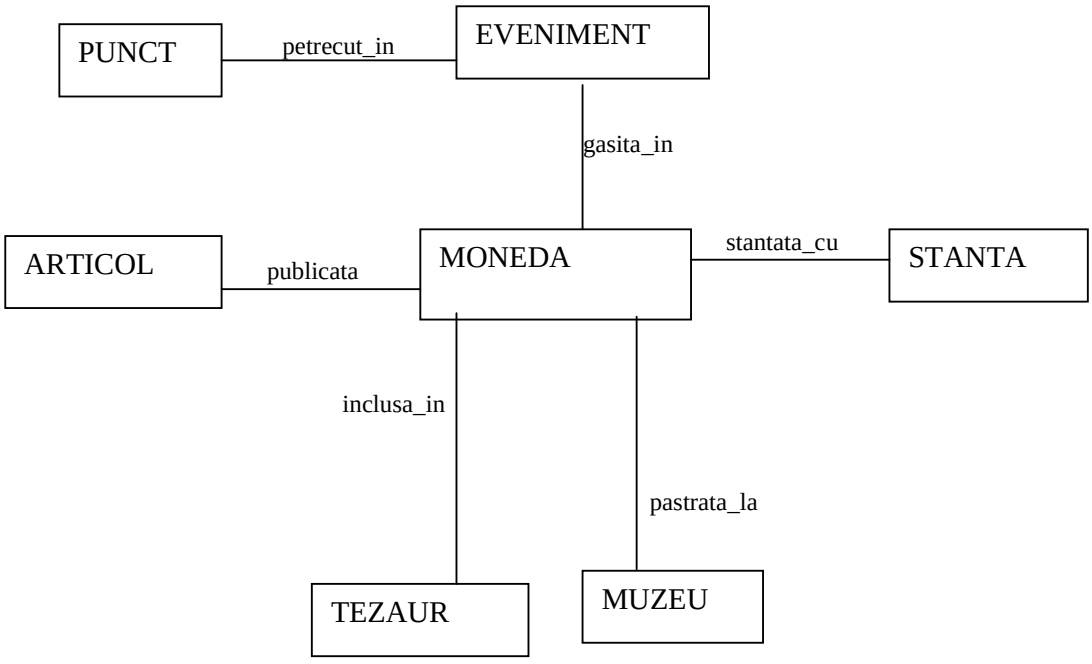
Gestiunea activităţilor unei firme de construcţii

Baza de date construită prin acest model, furnizează informaţii legate de obiective de execuţie, investitori, executanţi, şantiere, contracte etc. necesare unui manager al unei firme de construcţii



Vezi erori!

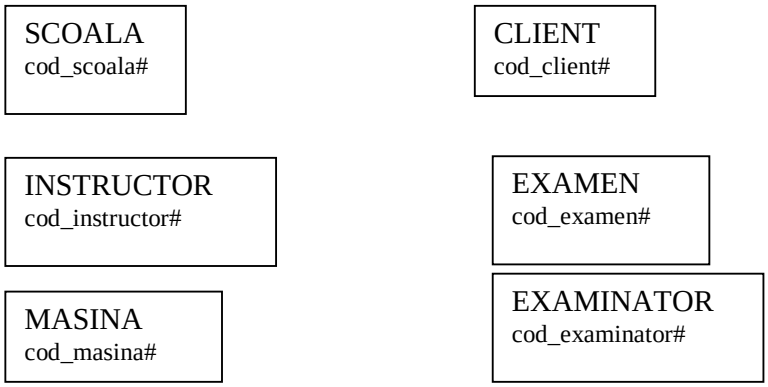
Descoperiri de monede antice din Romania



STANŢA (nr_stanţă, împărat emitent, valoare nominală, an emitere, monetăria, legenda de pe avers, legenda de pe revers) == > atribute ale entităţii **STANTA**

Completaţi cardinalitatea!

Evidenţa şcolilor de şoferi din Romania



Completaţi relaţiile (*lucreaza_la, conduce, sustine, asista, instruieste*) dintre entităţi şi specificaţi cardinalitatea!

Campionatele de fotbal ale diferitelor ţări

Modelul relaţional

Modelul relaţional a fost conceput şi 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 relaţiilor. Modelul relaţional este alcătuit numai din relaţii şi prin urmare, orice interogare asupra bazei de date este tot o relaţie. Cercetarea în domeniu → 3 mari proiecte (*System R*, *INGRES*, *PRTV*)

Calităţi:

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

Modalităţi pentru **definirea unui SGBD relaţional**:

- prezentarea datelor în tabele supuse anumitor operaţii de tip proiecţie, selecţie, reuniune, compunere, intersecţie 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 relaţional introdus de E.F. Codd.

Caracteristicile unui model relaţional:

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

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

Structura datelor

Definirea noţiunilor de domeniu, relaţie, schemă relaţională, valoare *null* şi 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
relaţie	tablou	fişier
tuplu	linie	înregistrare
atribut	coloană	câmp
domeniu	tip de dată	tip de dată

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

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

O **relaţie** *R* pe mulţimile D_1, D_2, ..., D_n este o submulţime a produsului cartezian $D_1 \times D_2 \times ... \times D_n$, deci este o mulţime de tupluri. Caracteristicile unei relaţii → comentat curs!

Definirea unei relaţii se referă la mulţimi care variază în timp. Pentru a caracteriza o relaţie este necesară existenţa un element invariant în timp: structura relaţiei (schema relaţională). Mulţimea numelor atributelor corespunzătoare unei relaţii *R* defineşte **schema relaţională** a relaţiei respective. Vom nota schema relaţională prin $R(A_1, A_2, ..., A_n)$. Exemplu!

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

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 relaţie, de multe ori un atribut este necunoscut sau neaplicabil. Pentru a reprezenta acest atribut a fost introdusă o valoare convenţională în relaţie, şi anume valoarea ***null***.

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

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

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

Tabele de adevăr pentru operatorii AND şi OR.

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

Vizualizarea este virtuală deoarece datele pe care le conţine nu sunt în realitate memorate într-o bază de date. Este memorată numai definiţia vizualizării. Vizualizarea nu este definită explicit, ca relaţiile de bază, prin mulţimea tuplurilor componente, ci implicit, pe baza altor relaţii prin intermediul unor expresii relaţionale. 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 → aserţiuni pe care datele conţinute în baza de date trebuie să le satisfacă.

Trebuie făcută distincţia între:

- regulile structurale inerente modelării datelor;
- regulile de funcţionare specifice unei aplicaţii particulare.

Există trei tipuri de constrângeri structurale (de cheie, de referinţă, de entitate) ce constituie mulţimea minimală de reguli de integritate pe care **trebuie** să le respecte un SGBD relaţional. Restricţiile de integritate minimale sunt definite în raport cu noţiunea de cheie a unei relaţii.

O mulţime minimală de atribute ale căror valori identifică unic un tuplu într-o relaţie reprezintă o **cheie** pentru relaţia respectivă.

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

O cheie identifică linii şi este diferită de un index care localizează liniile. O **cheie secundară** este folosită ca index pentru a accesa tupluri. Un grup de atribute din cadrul unei relaţii care conţine o cheie a relaţiei poartă numele de **supercheie**.

Fie schemele relaţionale *R*1(*P*1, *S*1) şi *R*2(*S*1, *S*2), unde *P*1 este cheie primară pentru *R*1, *S*1 este cheie secundară pentru *R*1, iar *S*1 este cheie primară pentru *R*2. În acest caz, vom spune că *S*1 este **cheie externă** (cheie străină) pentru *R*1.

Modelul relaţional respectă trei reguli de integritate structurală.

- **Regula 1** – unicitatea cheii. Cheia primară trebuie să fie unică şi minimală.
- **Regula 2** – integritatea entităţii. 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 relaţional (exemple → curs!)

Transformarea entităţilor

- Entităţile independente devin **tabele independente**. Cheia primară nu conţine chei externe.
- Entităţile dependente devin **tabele dependente**. Cheia primară a entităţilor dependente conţine cheia primară a entităţii de care depinde (cheie externă) plus unul sau mai multe atribute adiţionale.
- Subentităţile devin **subtabele**. Cheia externă se referă la supertabel, iar cheia primară este această cheie externă (cheia primară a subentităţii PROGRAMATOR este *cod_salariat* care este o cheie externă).

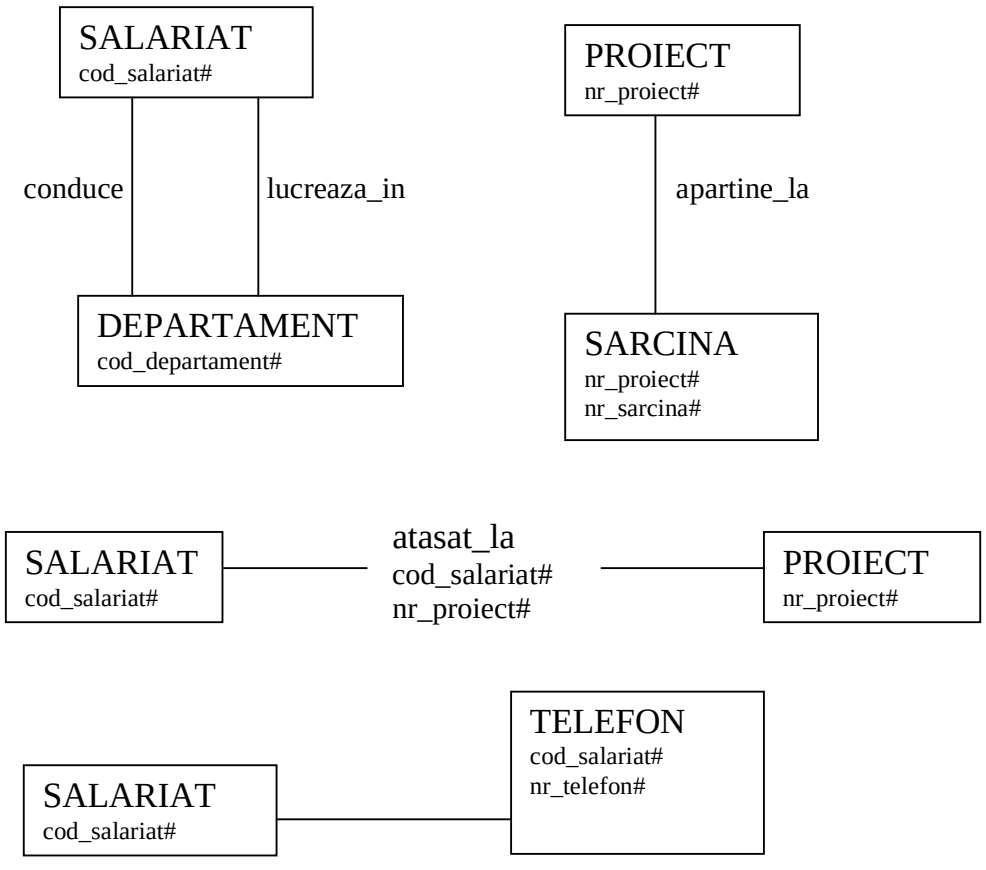
Transformarea relaţiilor

- Relaţiile 1:1 şi 1:*n* devin chei externe. Relaţia *conduce* devine coloană în tabelul DEPARTAMENT, iar relaţia *lucreaza_in* devine coloană în tabelul SALARIAT. Simbolul „×“ indică plasamentul cheii externe, iar simbolul „$\underline{\times}$“ exprimă faptul că această cheie externă este conţinută în cheia primară. Relaţia 1:1 plasează cheia externă în tabelul cu mai puţine linii.

- Relaţia *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 adiţionale. Tabelul se desenează punctat.
- Relaţiile de tip trei devin tabele asociative. Cheia primară este compunerea a trei chei externe plus eventuale coloane adiţionale.

Transformarea atributelor

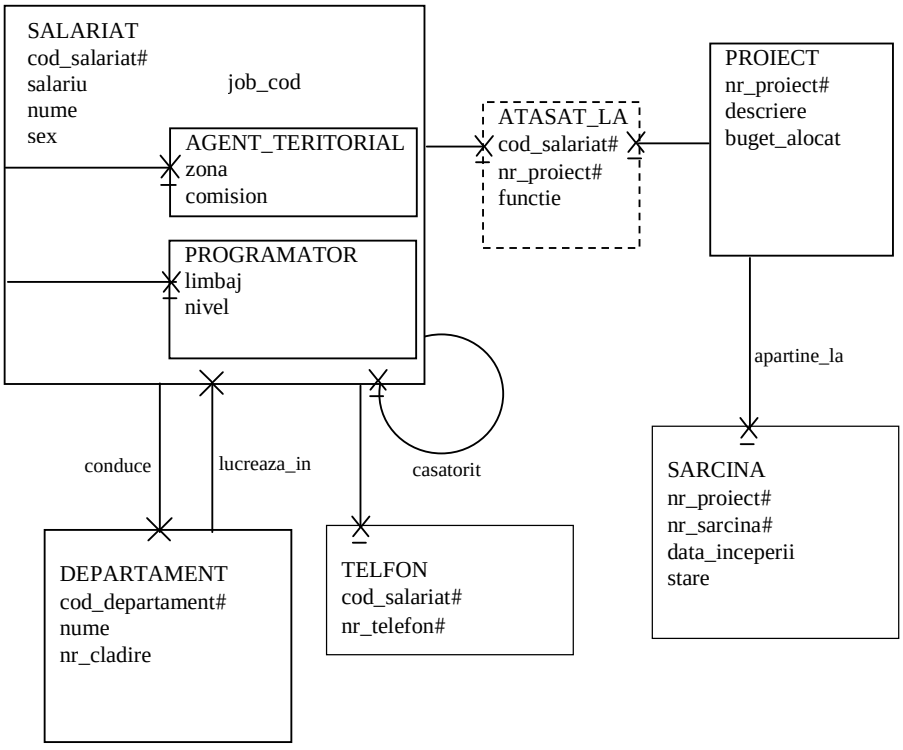
- Un atribut singular devine o coloană.
- Atributele multiple devin tabele dependendente ce conţin cheia primară a entităţii şi atributul multiplu. Cheia primară este o cheie externă, plus una sau mai multe coloane adiţionale.
- Entităţile devin tabele, iar atributele lor devin coloane în aceste tabele. Ce devin atributele relaţiilor? Pentru relaţii 1:1 şi 1:*n*, atributele relaţiilor vor aparţine tabelului care conţine cheia externă, iar pentru relaţii *m*:*n* şi de tipul trei, atributele vor fi plasate în tabelele asociative.



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

Tabel	Reprezintă	Cheie primară
Independent	entitate independentă	nu conţine chei externe
Subtabel	Subentitate	o cheie externă
Dependent	entitate dependentă	o cheie externă şi una sau mai multe coloane adiţionale
	atribute multiple	
Asociativ	relaţie m:n	două sau mai multe chei externe şi (opţional) coloane adiţionale
	relaţii de tip 3	

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



Schemele relaţionale 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 activităţilor unei firme de construcţii

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);

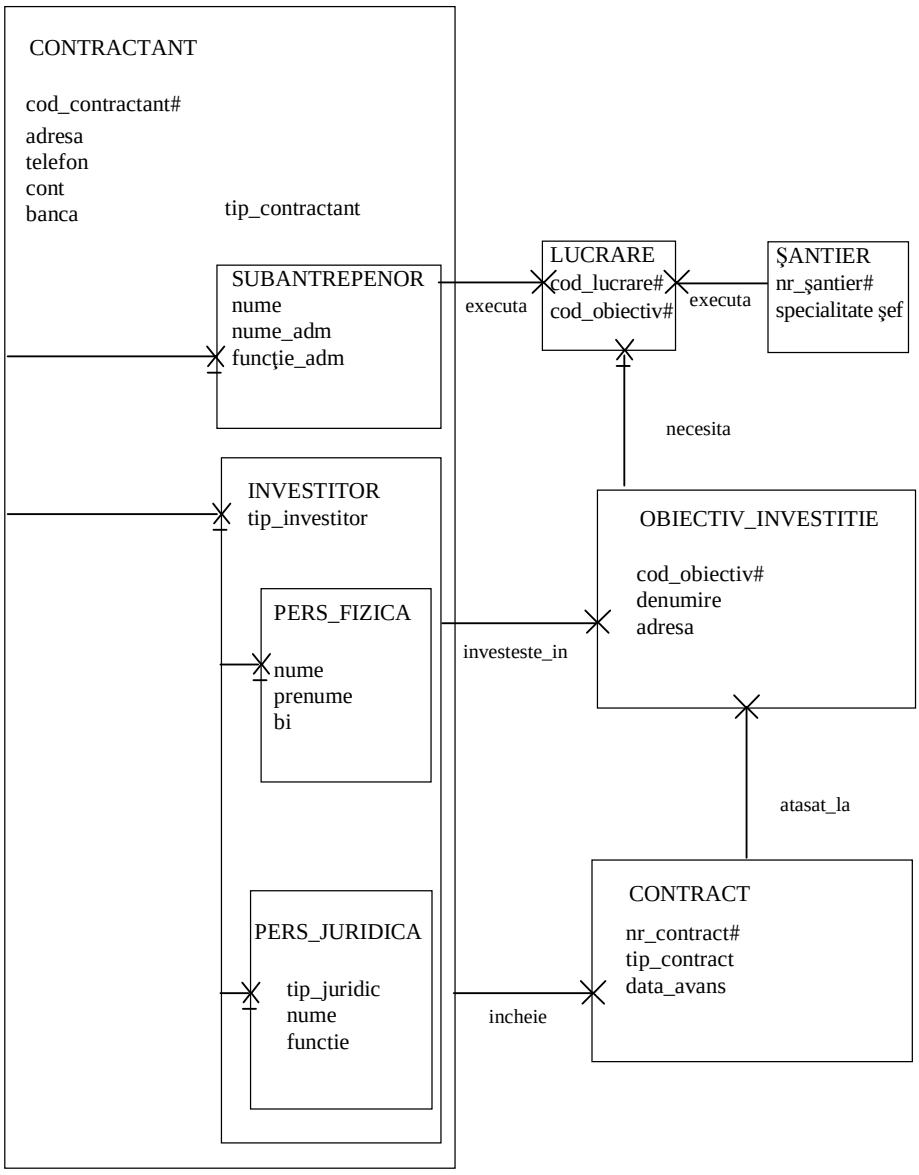
PERS_JURIDICA(cod_contractant#, tip_juridic, nume, reprez_legal, functie);

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

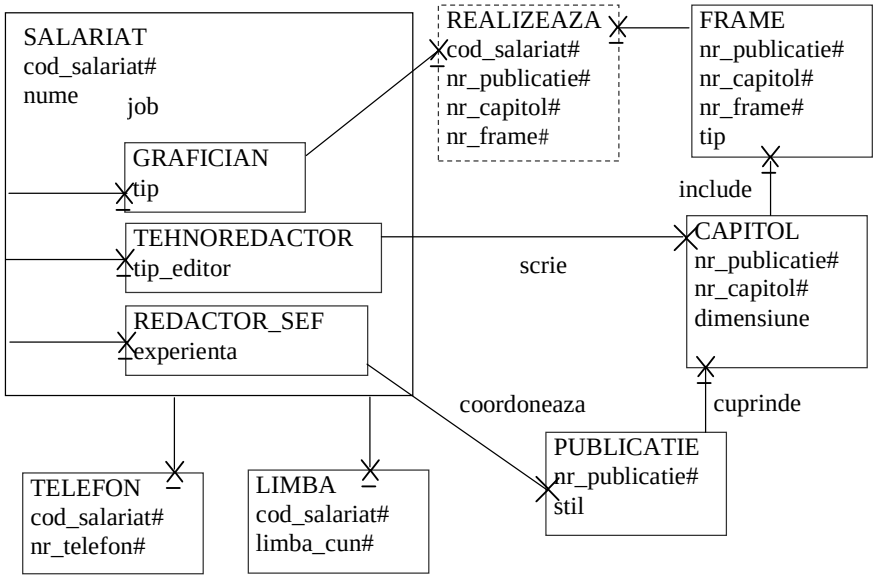
SANTIER(nr_santier#, specialitate, sef);

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);


CONTRACTANT
cod_contractant#
adresa
telefon
cont
banca
tip_contractant
SUBANTREPENOR
nume
nume_adm
funcţie_adm
executa
LUCRARE
cod_lucrare#
cod_obiectiv#
executa
ŞANTIER
nr_şantier#
specialitate şef
necesita
INVESTITOR
tip_investitor
OBIECTIV_INVESTITIE
cod_obiectiv#
denumire
adresa
PERS_FIZICA
nume
prenume
bi
investeste_in
atasat_la
CONTRACT
nr_contract#
tip_contract
data_avans
PERS_JURIDICA
tip_juridic
nume
functie
incheie

Gestiunea activităţilor 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!