Baze de date

Universitatea “Transilvania” din Brasov

Lect.dr. Costel Aldea

costel.aldea@gmail.com

Integritatea relaţională

- Pe lângă partea structurală, modelul de date relaţional mai are
  - **o parte de manipulare**, care defineşte tipurile de operaţii permise asupra datelor
  - un set de **reguli de integritate**, care asigură corectitudinea datelor
- Vom analiza care sunt regulile relaţionale de integritate

Integritatea relaţională

- Din moment ce fiecare atribut are un domeniu asociat, există anumite constrângeri - constrângeri de domeniu - sub formă de restricţii asupra mulţimii de valori permise pentru atributele relaţiilor
- Pe lângă acestea, există două reguli de integritate care reprezintă restricţii sau constrângeri ce se aplică tuturor instanţelor unei baze de date. Pentru modelul relaţional, acestea sunt:
  - integritatea entităţilor
  - integritate referenţială

Integritatea relaţională - null

- Null-ul reprezintă valoarea unui atribut care este în mod curent necunoscută sau nu este aplicabilă tuplului respectiv
- Un null nu este acelaşi lucru cu o valoare numerică egală cu 0 sau cu un text completat cu spaţii
  - zerourile şi spaţiile sunt valori, pe când null-ul semnifică absenţa unei valori

Integritatea relaţională - null

- Null-urile pot crea probleme privind implementarea
- Această dificultate apare deoarece modelul relaţional se bazează pe calculul predicativ de ordinul întâi, care reprezintă o logică bazată pe două valori, sau booleană, unde singurele valori admise sunt adevărat sau fals
- Introducerea null-urilor înseamnă că trebuie să lucrăm cu o logică polivalentă

Integritatea relaţională - null

- Încorporarea null-urilor în modelul relaţional constituie o chestiune controversată
  - Codd tratează null-urile ca parte integrantă a modelului
  - Alţi autori consideră această abordare greşită, fiind de părere că
    - problema informaţiilor lipsă nu este complet înţeleasă
    - că nu s-a găsit încă o soluţie complet satisfăcătoare
    - că încorporarea null-urilor în modelul relaţional este prematură
- Nu toate sistemele relaţionale acceptă null-urile

Constrângeri de domeniu

- Constrângerile de domeniu: constrângerea NOT NULL, constrângerea de valoare implicită (DEFAULT), constrângerea de verificare (CHECK)
- Constrângerea NOT NULLînsemna că atributul respectiv nu poate lua valoarea NULL în nici un tuplu al relaţiei.
- Valoarea NULL a unui atribut într-un tuplu semnifică faptul că valoarea acelui atribut nu este cunoscută pentru acel tuplu. Exemple:
  - nu se cunoaste deloc data de nastere a unei personalitati istorice;
  - nu se cunoaşte valoarea unui atribut in momentul inserarii tuplului, dar aceasta va fi cunoscuta si completată ulterior
- La crearea unui tabel opţiunea NULL este implicită (nu se specifică nimic), sau se poate introduce explicit; optiunea NOT NULL se introduceexplicit.
- Optiunile NULL si NOT NULL se introduc ca si constrangeri de coloana in instructiunea SQL CREATE TABLE.
- Exemplu:

```
CREATE TABLE ANGAJATI(
Nume varchar(20) NOT NULL,
Prenume varchar(20) NOT NULL,
DataNasterii date NULL,
Adresa varchar(50) NOT NULL,
Functie varchar(20),
Salariu numeric);
```

Constrangeri de domeniu

- Constrangerea de valoare implicita a unui atribut (DEFAULT): daca la inserarea unui tuplu nu se specifică valoarea unui atribut, atunci:
  - atributul primeşte valoarea implicită (dacă a fost definită) sau valoarea NULL (dacă nu a fost definită o valoare implicită, dar sunt admise valori NULL);
  - dacă nu a fost definită o valoare implicită şi nici nu sunt admise valori NULL, se generează o eroare. Exemplu:

CREATE TABLE STUDENT(

Nume varchar (20) NOT NULL,

Prenume varchar (20) NOT NULL,

Tara varchar (20) DEFAULT ('Romania') NULL) ;

- Constrângerea de verificare (CHECK) - pentru verificarea valorilor atributelor printr-o conditie care trebuie sa ia valoarea TRUE.
- Se introduce ca o constrangere de tabel in instructiunea CREATE TABLE:

[CONSTRAINT nume_constrangere] CHECK (conditie);

Exemplu:

CREATE TABLE ANGAJAT (

Nume varchar(20) NOT NULL,

Prenume varchar(20) NOT NULL,

Salariu numeric,

CONSTRAINT Verificare_Salariu CHECK (Salariu >= 1500));

Constrangeri de tuplu

- O relaţie = mulţime de tupluri, astfel ca tuplurile unei relaţii trebuie să fie distincte (nu pot exista două sau mai multe tupluri identice)
- Pentru ca tuplurile unei relații să fie distincte se folosește câte o cheie primară (primary key) în fiecare relație
- O cheie primară a unei relații este un atribut (simplu sau compus) al acelei relații care are propietatea de unicitate, adică fiecare valoarea cheii primare este unică în acea relație. Aceasta înseamnă că:
  - Nu există două tupluri distincte (diferite) care să aibă aceeași valoare a cheii primare (sau combinație de valori, dacă cheia primară este un atribut compus) pentru orice stare a relației, adică:

    ti[PK] ≠ tj[PK] dacă i ≠j, unde ti si tj sunt 2 tupuri ale relatiei
  - Fiecare tuplu poate fi identificat dacăse cunoaște valoarea cheii primare
  - Proprietatea de unicitate a cheii primare este o constrângere de integritate a tuplurilor
- Cheia primară se definește la crearea tabelului corespunzător
- Se pot defini chei primare naturale sau chei primare artificile, cu condiția ca acestea să îndeplinească condiția de unicitate

Integritatea relaţională a entităţilor

- Prima regulă de integritate se aplică cheilor primare ale relaţiilor de bază
- Definim o relaţie de bază ca o relaţie ce corespunde unei entităţi în schema conceptuală
- Regula integrităţii entităţilor
  - Într-o relaţie de bază, nici un atribut al unei chei primare nu poate fi null

Integritatea entităţilor

- Explicaţie
  - Prin definiţie, o cheie primară este un identificator minim, utilizat pentru identificarea unică a tuplurilor
  - Aceasta înseamnă că nici un subset al cheii primare nu este suficient pentru a permite identificarea unică a tuplurilor
  - Dacă admitem un null pentru orice parte a unei chei primare, aceasta implică faptul că nu toate atributele sunt necesare pentru a deosebi tuplurile, ceea ce contrazice definiţia cheii primare

Cheia primară (1)

- O cheie primară naturalăeste un atribut (simplu sau compus) al relației care are în mod natural valori unice, adică nu există două tupluri cu aceeasi valoare a cheii primare
- O cheie primară artificială este un atribut (de obicei simplu) care se adaugă în schema relaţiei special pentru identificarea unică a tuplurilor
- Exemplu:

ANGAJAT(IdAngajat, CNP, Nume, Prenume, DataNasterii, Adresa, Functia, Salariu):

- IdAngajat este o cheie primarăartificială
- Ar mai putea fi definite ca și chei primare atributul simplu {CNP} sau atributul compus {Nume, Prenume, DataNasterii, Adresa}, care au proprietatea de unicitate
- Din motive de eficiență a operațiilor de identificare a tuplurilor se preferă chei primare cu un număr cât mai mic de atribute (atribut simplu)

Cheia primară (2)

- Cheia primară are următoarele restricţii:
  - Nici o valoare a atributelor cheii primare nu poate fi modificată prin operaţii de actualizare(UPDATE)
  - Nu se admit valori de NULL pentru nici unul dintre atributele cheii primare
- Cheia primară se defineste prin instructiunea CREATE TABLE
  - In general, cheia primară se defineste ca o constrângere de tabel sub forma: [CONSTRAINT nume_constr] PRIMARY KEY (lista_atribute)
  - Dacă cheia primară este simplă (formată dintr-un singur atribut), ea se poate specifica si ca o constrângere de coloana
  - Exemplu:

```
CREATE TABLE SECTIE (
IdSectie int,
Nume varchar(50) NOT NULL,
Buget numeric
CONSTRAINT PK PRIMARY KEY (IdSectie)
);
```

Cheia primară (3)

- Exemplu:

```
CREATE TABLE ANGAJAT(
IdAngajat int PRIMARY KEY,
Nume varchar(20) NOT NULL,
Prenume varchar(20) NOT NULL,
DataNasterii Date,
Adresa varchar(50),
Salariu numeric
);
```

- Modul de asigurare a unicităţii valorii cheii primare artificiale depinde de sistemul SGBD folosit. De exemplu:
  - In Microsoft SQL Server se pot obţine valori unice ale cheii primare folosind parametrul IDENTITY, care asigură incrementarea valorii atributului cheii la introducerea fiecărei linii noi.
  - În sistemele Oracle se pot genera chei artificiale folosind obiecte SEQUENCE; un obiect SEQUENCE geneaza un numar unic la fiecare apel al metodei NEXTVAL
  - In MySQL, se foloseste parametrul AUTO_INCREMENT pentru generarea numerelor unice pentru cheile primare.
- SGBD-urile interzic introducerea liniilor (tuplurilor) care au valori identice ale cheilor primare sau secundare (daca nu exista valori NULL)

Superchei, chei candidat candidate

- O supercheie(superkey) este o submulţime SK de atribute ale relaţiei care prezintă proprietatea de unicitate (orice combinaţie de valori ale atributelor supercheii este unică pentru orice stare a relaţiei)
  - Rezulta că, dacă se cunoaşte valoarea (combinaţia de valori ale atributelor) supercheii, atunci acel tuplu poate fi identificat în mod unic
  - Orice relație are cel puțin o supercheie, care este mulțimea tuturor atributelor sale
- O cheie candidată(candidate key) este o supercheie ireductibilă:
  - Unicitate: nu există două tupluri diferite ale relaţiei care să conţină aceeaşi combinaţie de valori ale atributelor cheii CK;
  - Ireductibilitate: nu există nici o submulţime proprie, nevidă a cheii CK care să aibă proprietatea de unicitate.
- O cheie candidată este o supercheie minimală (ireductibilă)si poate fi simplă (un singur atribut), sau compusă (mai multe atribute)
- Exemplu: ANGAJATI(IdAngajat, CNP, Nume, Prenume, DataNasterii, Adresa, Functia, Salariu)

SK1 = {IdAngajat, CNP, Nume, Prenume, DataNasterii, Adresa,Functia, Salariu}

SK2 = {CNP, Nume, Prenume, DataNasterii, Adresa};

SK3 = {IdAngajat, CNP}

CK1={Nume, Prenume, DataNasterii, Adresa};

CK2 = {CNP};

CK3={IdAngajat}

Chei secundare

- Atunci când există mai multe chei candidate, una dintre ele se alege ca şi cheie primară, celelalte chei candidate fiind numite chei secundare
- O cheie secundară (alternativă, unică) (secondary, alternate, unique key) este o cheie candidată care nu a fost desemnată ca şi cheie primară; cheile secundare compuse admit valori NULL pentru unele din atributele lor
- Alegerea cheii primare dintre mai multe chei candidate este arbitrară, dar, din motive de eficienta, se alege cheia cu cel mai mic număr de atribute
- Cheile secundare se definesc în instrucțiunea CREATE TABLE folosind specificatorul UNIQUE [KEY] în loc de PRIMARY KEY

Integritatea referenţială

- A doua regulă de integritate se aplică cheilor străine
- Regula integrităţii referenţiale
  - Dacă într-o relaţie există o cheie străină, valoarea acesteia trebuie ori să coincidă cu valoarea unei chei candidat a unui tuplu în relaţia sa de bază, ori să fie în întregime null

Integritatea referenţială

- Explicaţie
  - Atributul NrFil din relaţia Personal este o chei străină, care ţinteşte atributul NrFil din relaţia de bază Filiala
  - Nu trebuie să fie posibilă crearea unei înregistrări de personal cu numărul de filială C20, de exemplu, decât dacă există deja o înregistrare corespunzătoare numărului C20 în relaţia Filiala
  - Totuşi trebuie să existe posibilitatea de a crea o nouă înregistrare de personal cu număr de filială null
  - Aceasta corespunde situaţiei în care un nou membru al personalului a fost angajat în companie, fără a i se atribui încă o anumită filială

Constrângeri inter-relații

- Asocierile(relationships) 1:N între multimile de entităţi (din modelul Entitate-Asociere) se realizează în modelul relaţional prin chei străine
- Exemplu: Pentru a realiza asocierea 1:N dintre relaţiile SECTII si ANGAJATI, se adaugă în relaţia ANGAJATI cheia străină IdSectie, care reprezintă identificatorul (numărul) secţiei în care lucrează angajatul respectiv:

ANGAJATI(IdAngajat, Nume, Prenume, DataNasterii, Adresa, Salariu, IdSectie)

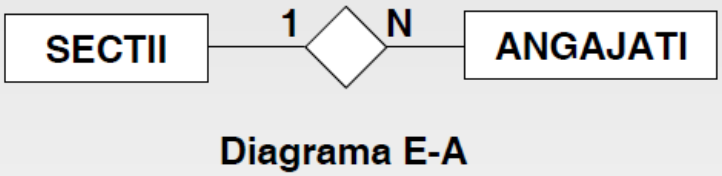

Diagrama E-A

SECTII

IdSectie	Nume	Buget
1	Productie	400000
2	Proiectare	300000
3	Cercetare	200000
4	Documentare	100000

ANGAJATI

IdAngajat	Nume	Prenume	DataNasterii	Adresa	Functia	Salariu	IdSectie
1	Ionescu	Ion	1960.01.05	Bucuresti	inginer	4000	1
2	Popescu	Petre	1965.02.97	Bucuresti	tehnician	3200	1
3	Vasilescu	Ana	1961.03.06	Bucuresti	secretara	2000	2
4	Ionescu	Ion	1970.03.98	Bucuresti	NULL	NULL	NULL

Cheia straina

- Fie doua relatii R1 si R2, intre care exista o asociere cu raportul 1:N O cheie străină(foreign key) este o submulţime FK de atribute ale relaţiei R2 care referă cheia CK din relaţia R1 (relatia referita) şi satisface urm.condiţii:
  - atributele cheii străine FK sunt definite pe domenii compatibilecu cele ale atributelor unei cheii candidate CK a relaţiei R1
  - valorile atributelor FK într-un tuplu din relaţia R2, fie sunt identice cu valorile atributelor CK a unui tuplu oarecare din starea curentă a relaţiei R1, fie sunt NULL
- Cheia străină reprezintă o constrângere referenţială intre cele 2 relatii
- Două domenii sunt compatibile dacă ele sunt comparabile din punct de vedere semantic (are sens sa fie comparate)
- În limbajul SQL verificarea domeniilor se rezumă la verificarea tipurilor de date, iar compatibiltatea semantică trebuie să fie asigurată de proiectant
- Cheia străină se specifică la crearea tabelului printr-o constrângere de tabel:

[CONSTRAINT nume_constr] FOREIGN KEY (cheie_straina) REFERENCES relatia_referita (cheie_candidata)

CREATE TABLE ANGAJAT (
IdAngajat int PRIMARY KEY,
Nume varchar(20) NOT NULL,
Prenume varchar(20) NOT NULL,
IdSecţie int,
UNIQUE (Nume, Prenume),
CONSTRAINT FK FOREIGN KEY (IdSectie) REFERENCES SECTIE(IdSectie));

Menținerea integrității referențiale a relațiilor (1)

- Integritatea referenţială(referential integrity) este proprietatea bazei de date prin care orice cheie străina:
  - fie are o valoare care se regăseşte printre valorile cheii candidate referite
  - fie are valoarea NULL
- Pentru mentinerea integritatii referentiale trebuie sa fie inpuse restrictii operaţiilor de modificare a stării relaţiilor (INSERT, DELETE, UPDATE)
- Restricţiile care se impun operaţiilor de modificare a relaţiilor depind de rolul relaţiei (relaţie care referă, relaţie referită, sau poate avea ambele roluri)
- Operaţia INSERT:
  - într-o relaţie care nu referă altă relaţie se poate face fără restricţii
  - într-o relaţie care referă (care conţine o cheie străină): trebuie să se verifice că în relaţia referită există un tuplu care are valorile atributelor cheii referite egale cu valorile atributelor cheii străine a tuplului de introdus; dacă această condiţie nu este satisfăcută, operaţia de introducere este refuzată.
- Operatia DELETE:
  - Intr-o relatie care nu este referită se poate face fara restrictii
  - Intr-o relatie referita se poate face ştergere restricţionatăsau ştergere în cascadă.

Menținerea integrității referențiale a relațiilor (2)

- Ştergerea restricţionată interzice ştergerea unui tuplu din relaţia referită dacă acesta este referit de un tuplu din relaţia care o referă
- Ştergerea în cascadă permite ştergerea unui tuplu din relaţia referită; dacă tuplul şters era referit de unul sau mai multe tupluri, atunci se şterg şi acestea din relaţia care o referă; dacă tuplurile şterse din relaţia care referă sunt, la rândul lor referite de alte tupluri, atunci trebuie să fie şterse şi acestea, ş.a.m.d.; se execută deci o ştegere în cascadă
- Operaţia UPDATE poate fi privită ca o ştergere urmată de o introducere, deci restricţiile de actualizare reprezintă combinaţia restricţiilor de introducere şi de ştergere
- In limbajul SQL se specifica opţiunile ON DELETE si ON UPDATE constrîngerii de cheie străină; valorile posibile ale acestor opţiuni sunt:
  - RESTRICT-pentru ştergerea restricţionată (este valoare implicita)
  - CASCADE-pentru ştergerea în cascadă;

CREATE TABLE ANGAJAT (

IdAngajatint PRIMARY KEY,

Nume varchar (20) NOT NULL,

. . . . . .

CONSTRAINT FK FOREIGN KEY (IdSectie) REFERENCES SECTIE(IdSectie),

ON DELETE CASCADE);

Vederi

- În arhitectura ANSI-SPARC cu 3 nivele vederea externă este structura bazei de date aşa cum apare ea unui anumit utilizator
- În modelul relaţional, noţiunea de vedere are un înţeles uşor diferit
- O vedere este o relaţie virtuală – o relaţie care nu este de sine stătătoare, ci este derivată în mod dinamic din una sau mai multe relaţii de bază
- O vedere poate fi construită prin efectuarea unor operaţii sau calcule cu valorile relaţiilor de bază existente
- Un model extern poate consta atât în relaţii de bază (la nivel conceptual), cât şi în vederile derivate din acestea

Vederi - terminologie

- **Relaţie de bază**

  este o relaţie cu o anumită denumire, corespunzătoare unei entităţi din schema conceptuală, ale cărei tupluri sunt stocate fizic în bază de date

- Vederile se pot defini prin intermediul relaţiilor de bază

Vederi - terminologie

- Vederea
  - este rezultatul dinamic al uneia sau mai multor operaţii relaţionale, care acţionează asupra relaţiilor de bază pentru a realiza o altă relaţie
- O vedere este o relaţie virtuală care nu există în realitate în baza de date
- Ea este produsă la cererea unui utilizator

Vederi - terminologie

- O vedere este o relaţie care pentru utilizator pare să existe şi poate fi manipulată ca şi cum ar fi o relaţie de bază, dar care nu există în dispozitivul de stocare în sensul admis pentru relaţiile de bază
- Conţinutul unei vederi este definit ca o interogare asupra uneia sau mai multor relaţii de bază
- Orice operaţii efectuate asupra unei vederi sunt automat transpuse în operaţii asupra relaţiilor din care este derivată
- Vederile sunt dinamice, adică modificările din relaţiile de bază care o afectează, sunt imediat reflectate de către acestea

Scopul vederilor

- Furnizează un **mecanism de securitate puternic şi flexibil**, prin ascunderea unor părţi ale bazei de date faţă de anumiţi utilizatori
  - Utilizatorul nu este conştient de existenţa atributelor tuplurilor care lipsesc din vederea respectivă
- Permite **accesarea datelor într-un mod personalizat**, conform cu cerinţele utilizatorilor, astfel încât, aceleaşi date pot fi vizualizate simultan, de către utilizatori diferiţi, în diverse moduri;

Scopul vederilor

- Poate simplifica operaţii complexe asupra relaţiilor de bază
  - Exemplu
  - Dacă o vedere este definită ca o uniune a două relaţii, utilizatorul poate efectua operaţiile unare mai simple, de selecţie şi proiecţie a vederii, iar acestea vor fi traduse de SGBD în operaţii echivalente asupra uniunii

Scopul vederilor

- O vedere trebuie proiectată astfel încât să accepte modelul extern, cu care este familiarizat utilizatorul
  - Exemplu
    - Un utilizator poate avea nevoie de înregistrările din relaţia *Filiale* care conţin numele managerilor, împreună cu celelalte atribute
      - Această vedere se creează prin uniunea relaţiilor *Filiale* şi *Angajati*, urmată de proiectarea asupra atributelor care mai interesează

Scopul vederilor

- Exemplu (continuare)
  - Un alt utilizator ar putea dori să vadă înregistrările din relaţia Angajati, dar fără atributul Salariul
    - Pentru aceasta se efectuează o proiecţie ce creează o vedere care nu include atributul Salariul
  - Atributele pot fi redenumite, astfel încât utilizatorul să poată folosi denumirile care îi sunt lui familiare
  - Un membru al personalului poate vizualiza numai informaţii din filiala în care lucrează
    - În acest caz, trebuie efectuată o operaţie de selecţie, astfel încât să se poată vizualiza numai un subset din relaţia Angajati

Reactualizarea vederilor

- Toate reactualizările unei relaţii din baza de date trebuie să fie imediat reflectate în toate vederile care se referă la aceasta
- Similar, dacă o vedere este reactualizată, atunci relaţia de bază corespunzătoare trebuie să reflecte modificarea

Reactualizarea vederilor

- Există restricţii privind tipurile de modificări care pot fi efectuate prin intermediul vederilor:
  - sunt permise reactualizările prin vederi definite prin utilizarea unei interogări simple, care implică o singură relaţie de bază şi conţine fie cheia primară, fie cheia candidat
  - nu sunt permise reactualizările prin vederi care implică relaţii de bază multiple
  - nu sunt permise reactualizările prin vederi care implică operaţii de acumulare sau de grupare

Crearea vederilor

- Tabelele create cu instrucţiunea CREATE TABLE:
  - se numesc şi tabele de bază (base tables)
  - ele sunt memorate în fişierele bazei de date şi pot fi accesate pentru introducerea, modificarea şi regăsirea (interogarea) datelor
- Un tabel vedere(view) este un tabel virtual care:
  - nu este memorat fizic în fişiere
  - reprezintă o selecţie (după un anumit criteriu) a datelor memorate în unul sau mai multe tabele de bază
- Un tabel vedere secreeaza cu instrucţiunea SQL:

  CREATE VIEW nume_vedere AS (SELECT...);
- Datele (valorile atributelor) sunt memorate o singură dată, în tabelele de bază, dar pot fi accesate atât prin tabelele de bază cât şi prin tabelele vederi
- Un tabel vedere este întotdeauna actualizat ("la zi"), adică orice modificare efectuată în tabelele de bază se regăseşte imediat în orice tabel vedere creat pe baza acestora

Modificarea si stergerea tabelelor si a vederilor

- Comanda de modificare a tabelelor (ALTER TABLE) permite:
  - adăugarea sau ştergerea unor atribute
  - modificarea domeniilor unor atribute
  - adăugarea, modificarea sau ştergerea unor constrângeri ale tabelului
- Pentru adăugare unei coloane intr-un tabel se foloseşte clauza ADD, urmata de numele coloanei si numele domeniului (tipul SQL) atributului corespunzător. Exemplu:

  ALTER TABLE ANGAJATI ADD DataAngajarii date;
- Pentru ştergerea unei coloane dintr-un tabel se foloseşte clauza DROP, urmata de numele coloanei care se va şterge. Exemplu:

  ALTER TABLE ANGAJATI DROP DataAngajarii;
- Instrucţiunile de ştergere a tabelelor de bază şi a vederilor sunt:

  DROP TABLE nume_tabel;

  DROP VIEW nume_vedere;

Exemplu – bd Intreprindere

- cu tipurile de entitati tari:
  - SECTII(Nume, Buget)
  - ANGAJATI(Nume, Prenume, DataNasterii, Adresa, Functie, Salariu)
  - PROIECTE(Denumire,Termen, Buget)
  - PRODUSE(Denumire, Descriere)
  - COMPONENTE(Denumire, Descriere)
  - FURNIZORI(Nume, Prenume, Adresa) ;
  - CLIENTI(Nume, Prenume, Adresa)
- Diagrama entitate-relatie