

BAZE DE DATE

CURS 8

Regulile lui Codd. Normalizare (partea 1).

Regulile lui Codd

Caracteristici ale modelului relațional:

- ▶ nu există tupluri identice;
- ▶ ordinea liniilor și a coloanelor este arbitrară;
- ▶ articolele unui domeniu sunt omogene;
- ▶ fiecare coloană definește un domeniu distinct și nu se poate repeta în cadrul aceleiași relații;
- ▶ toate valorile unui domeniu corespunzătoare tuturor cazurilor nu mai pot fi descompuse în alte valori (sunt atomice).

Regulile lui Codd

Avantajele modelului relațional:

- fundamentare matematică riguroasă;
- independență fizică a datelor;
- posibilitatea filtrărilor;
- existența unor structuri de date simple;
- realizarea unei redundanțe minime;
- suplețe în comunicarea cu utilizatorul neinformatician.

Regulile lui Codd

Limite ale modelului relațional:

- rămâne totuși redundanță,
- ocupă spațiu,
- apar fenomene de inconsistență,
- nu există mecanisme pentru tratarea optimă a cererilor recursive,
- nu lucrează cu obiecte complexe,
- nu există mijloace perfecționate pentru exprimarea constrângerilor de integritate,
- nu realizează gestiunea totală a datelor distribuite,
- nu realizează gestiunea cunoștințelor.

Regulile lui Codd

- ▶ În anul 1985, E.F. Codd a publicat un set de 13 reguli în raport cu care un sistem de gestiune a bazelor de date poate fi apreciat ca relațional.
- ▶ Niciun sistem de gestiune a bazelor de date nu respectă absolut toate regulile definite de Codd, dar acest lucru nu împiedică etichetarea acestor sisteme drept relaționale.
- ▶ Nu trebuie apreciat un SGBD ca fiind relațional sau nu, ci **măsura în care acesta este relațional**, deci numărul regulilor lui Codd pe care le respectă.

Regulile lui Codd

- **Regula 1 – regula gestionării datelor.** *Un SGBD relațional trebuie să fie capabil să gestioneze o bază de date numai prin posibilitățile sale relaționale.*
- **Regula 2 – regula reprezentării informației.** *Într-o bază de date relațională, informația este reprezentată la nivel logic sub forma unor **tabele ce poartă numele de relații**.*
- **Regula 3 – regula accesului garantat la date.** *Fiecare valoare dintr-o bază de date relațională trebuie să poată fi accesată în mod logic printr-o **combinație formată din numele relației, valoarea cheii primare și numele atributului**.*
- **Regula 4 – regula reprezentării informației necunoscute.** *Un sistem relațional trebuie să permită utilizatorului definirea unui tip de date numit „**null**” pentru reprezentarea unei informații necunoscute la momentul respectiv.*

Regulile lui Codd

- **Regula 5 – regula dicționarilor de date.** *Asupra descrierii bazelor de date (informații relative la relații, vizualizări, indecși etc.) trebuie să se poată aplica **aceleași operații** ca și asupra datelor din baza de date.*
- **Regula 6 – regula limbajului de interogare.** *Trebuie să existe cel puțin un **limbaj** pentru prelucrarea bazei de date.*
- **Regula 7 – regula de actualizare a vizualizării.** *Un SGBD trebuie să poată determina dacă o vizualizare poate fi actualizată și să stocheze rezultatul interogării într-un dicționar de tipul unui catalog de sistem.*
- **Regula 8 – regula limbajului de nivel înalt.** *Regulile de prelucrare asupra unei relații luate ca întreg sunt valabile atât pentru operațiile de regăsire a datelor, cât și asupra operațiilor de inserare, actualizare și ștergere a datelor.*

Regulile lui Codd

- **Regula 9** – regula independenței fizice a datelor: *Programele de aplicație și activitățile utilizatorilor nu depind de modul de depunere a datelor sau de modul de acces la date.*
- **Regula 10** – regula independenței logice a datelor. *Programele de aplicație trebuie să fie transparente la modificările de orice tip efectuate asupra datelor.*
- **Regula 11** – regula independenței datelor din punct de vedere al integrității. *Regulile de integritate trebuie să fie **definite într-un sublimbaj relațional**, nu în programul de aplicație.*
- **Regula 12** – regula independenței datelor din punct de vedere al distribuirii. *Distribuirea datelor pe mai multe calculatoare dintr-o rețea de comunicații de date nu trebuie să afecteze programele de aplicație.*
- **Regula 13** – regula versiunii procedurale a unui SGBD. *Orice componentă procedurală a unui SGBD trebuie să respecte aceleași restricții de integritate ca și componenta relațională.*

Regulile lui Codd

- ▶ Deoarece regulile lui Codd sunt prea severe pentru a fi respectate de un SGBD operațional, s-au formulat **criterii minimale de definire a unui sistem de gestiune relațional**.
- ▶ Un SGBD este **minimal relațional** dacă:
 - ▶ toate datele din cadrul bazei sunt reprezentate prin **valori în tabele**;
 - ▶ nu există pointeri observabili de către utilizator;
 - ▶ sistemul suportă operatorii relaționali de **proiecție, selecție și compunere naturală**, fără limitări impuse din considerente interne.
- ▶ Un SGBD este **complet relațional** dacă este minimal relațional și satisface în plus condițiile:
 - ▶ sistemul suportă **restricțiile de integritate de bază** (unicitatea cheii primare, constrângerile referențiale, integritatea entității).
 - ▶ sistemul suportă **toate operațiile de bază ale algebrei relaționale**.

NORMALIZAREA RELAȚIILOR

- În procesul modelării unei baze de date relaționale, o etapă importantă o reprezintă **normalizarea relațiilor conceptuale** (Codd), adică obținerea de relații „moleculare” fără a pierde nimic din informație pentru a elimina:
 - **redundanța;**
 - **anomaliile reactualizării informațiilor.**

NORMALIZAREA RELAȚIILOR

- ▶ Tehnica normalizării permite:
 - ▶ obținerea unei **scheme conceptuale rafinate** printr-un proces de **ameliorare progresivă** a unei scheme conceptuale inițiale a bazei de date relaționale.
- ▶ După fiecare etapă de ameliorare, relațiile bazei de date ating un anumit **grad de perfecțiune**, deci se află într-o anumită **formă normală**.
 - ▶ Trecerea unei relații dintr-o formă normală în alta, presupune **eliminarea unui anumit tip de dependențe** nedorite, care sunt transformate în dependențe admisibile, adică **dependențe care nu provoacă anomalii**.

NORMALIZAREA RELAȚIILOR

- Procesul de ameliorare a schemei conceptuale **trebuie**:
 - să garanteze **conservarea datelor**, adică în schema conceptuală finală trebuie să figureze toate datele din cadrul schemei inițiale;
 - să garanteze **conservarea dependențelor** dintre date, adică în schema finală fiecare dependență trebuie să aibă determinantul și determinatul în schema aceleiași relații;
 - să reprezinte o **descompunere minimală** a relațiilor inițiale, adică niciuna dintre relațiile care compun schema finală nu trebuie să fie conținută într-o altă relație din această schemă.

NORMALIZAREA RELAȚIILOR

Există două metode pentru a modela **baze de date relaționale fără anomalii sau pierderi de informație**.

- **Schema descompunerii** pleacă de la o **schemă relațională universală** ce conține toate attributele BD.
 - Schema se descompune prin **proiecții succesive** în subrelații.
 - Descompunerea se oprește când continuarea ei ar duce la pierderi de informație.
 - Algoritmii de descompunere se bazează, în general, pe descrierea formală a dependenței dintre attribute.
- **Schema sintezei** pleacă de la o mulțime de attribute independente.
 - Utilizând proprietăți de semantică și legături între attribute se pot compune noi relații, astfel încât, acestea să nu sufere de anumite anomalii.
 - Algoritmii se bazează, în general, pe teoria grafurilor pentru a reprezenta legăturile între attribute.

Dependențe funcționale

- ▶ O **relație universală** este o relație ce grupează toate atributele care modelează sistemul real cercetat.
- ▶ Fie E , **mulțimea dependențelor** considerate de proiectantul bazei pentru o schemă relațională sau pentru o relație universală.
 - ▶ Plecând de la o mulțime de proprietăți formale ale dependențelor, proprietăți considerate drept reguli de deducție (**axiome**), poate fi obținută **mulțimea maximală de dependențe asociate lui E** . Această mulțime definește **închiderea** lui E .

Dependențe funcționale

- Fie E mulțimea dependențelor unei relații și $p_1, p_2, \dots, p_r, r \geq 1$, proprietăți formale ale acestor dependențe.
 - Dacă există o mulțime E' , astfel încât orice dependență a mulțimii E este derivabilă din E' prin aplicarea proprietăților p_1, p_2, \dots, p_r , atunci mulțimea E' definește **acoperirea** lui E pentru proprietățile p_1, p_2, \dots, p_r .
- E' este o **acoperire minimală** pentru E , dacă nu există nici o submulțime proprie, nevidă a lui E' care să fie o acoperire pentru E .
Evident, E și E' au închideri identice, deci dispun de același potențial informațional!

Dependențe funcționale

- Fie $R(A_1, A_2, \dots, A_n)$ o schemă relațională și fie X, Y submulțimi de attribute ale lui R .
 - X **determină funcțional** Y sau Y depinde funcțional (FD) de X , dacă pentru orice relație r (valoare curentă a lui R) **nu există două tupluri care să aibă aceleași valori pentru attributele lui X și să aibă valori diferite pentru cel puțin un atribut din Y** . Cu alte cuvinte, o valoare a lui X , determină unic o valoare a lui Y .
- Notăție: $X \rightarrow Y$. X este numit **determinant**, iar Y este numit **determinat** (sau dependent).
 - Dependența funcțională $X \rightarrow Y$ este **trivială** dacă $Y \subseteq X$.

Dependențe funcționale

- ▶ Comparând toate submulțimile de attribute ale unei relații și determinând legăturile dintre ele, se pot obține toate dependențele funcționale pe care o relație le satisface.
 - ▶ Această abordare **nu** este eficientă, consumând mult timp.
- ▶ Există posibilitatea ca, știind **anumite dependențe** funcționale și utilizând **reguli de deducție**, să fie obținute **toate** dependențele funcționale.

Dependențe funcționale

Fie X, Y, Z, W mulțimi de attribute ale unei scheme relaționale R și fie următoarele axiome:

- **Ax1 – reflexivitate.** $X \rightarrow X$. Mai general, dacă $Y \subseteq X$, atunci $X \rightarrow Y$.
- **Ax2 – creșterea determinantului.** Pot fi considerate următoarele formulări echivalente pentru această axiomă.
 - Dacă $X \rightarrow Y$ și $X \subseteq Z$, atunci $Z \rightarrow Y$.
 - Dacă $X \rightarrow Y$ și $W \subseteq Z$, atunci $X \cup Z \rightarrow Y \cup W$.
 - Dacă $X \rightarrow Y$ atunci $X \cup Z \rightarrow Y \cup Z$.
 - Dacă $X \rightarrow Y$ atunci $X \cup Z \rightarrow Y$.
- **Ax3 – tranzitivitate.** Dacă $X \rightarrow Y$ și $Y \rightarrow Z$, atunci $X \rightarrow Z$.

Dependențe funcționale

- ▶ O mulțime de axiome este **completă** dacă și numai dacă plecând de la o mulțime de dependențe E se pot obține toate dependențele închiderii lui E , utilizând axiomele mulțimii.
- ▶ O mulțime de axiome este **închisă** dacă și numai dacă plecând de la o mulțime de dependențe E , nu poate fi dedusă cu ajutorul axiomelor o dependență care nu aparține închiderii lui E . (nu obțin altele!)
- ▶ **Ullman** a demonstrat că axiomele Ax1 – Ax3, numite **axiomele lui Armstrong**, reprezintă o mulțime închisă și completă de axiome. Consecința acestui rezultat este că **închiderea lui E reprezintă mulțimea dependențelor deduse din E , prin aplicarea axiomelor lui Armstrong!!!**

Dependențe funcționale

- ▶ Nu toate dependențele funcționale sunt folositoare pentru modelarea relațională.
- ▶ O dependență funcțională $X \rightarrow Y$ se numește **dependență funcțională totală** (FT), dacă și numai dacă nu există nicio submulțime proprie $X' \subset X$, astfel încât $X' \rightarrow Y$.
 - ▶ Dacă există o submulțime proprie $X' \subset X$, astfel încât $X' \rightarrow Y$, atunci dependența funcțională $X \rightarrow Y$ este **parțială**. În axioma Ax2, dependența $Z \rightarrow Y$ este o dependență funcțională parțială.

Dependențe funcționale

- ▶ În cazul dependenței funcționale totale, axiomele lui Armstrong se reduc la o axiomă unică și anume **pseudo-tranzitivitatea**:
 - ▶ **dacă $X \rightarrow Y$ și $W \cup Y \rightarrow Z$, atunci $W \cup X \rightarrow Z$.**
- ▶ Această axiomă este o regulă de deducție completă pentru total dependențe:
 - ▶ pseudo-tranzitivitatea implică tranzitivitatea ($W = \emptyset$);
 - ▶ reflexivitatea nu poate fi utilizată pentru a obține dependențe totale;
 - ▶ reflexivitatea și pseudo-tranzitivitatea implică creșterea.

Dependențe funcționale

- Dacă F este o mulțime de dependențe funcționale totale, atunci **închiderea pseudo-tranzitivă** F^+ a acestei mulțimi este reuniunea mulțimilor dependențelor funcționale totale care pot fi obținute din F folosind **axioma de pseudo-tranzitivitate**.
- Două mulțimi de dependențe funcționale totale sunt **echivalente** dacă au **închideri pseudo-tranzitive identice**.
 - Pentru a modela scheme relaționale se consideră **mulțimi minimale de dependențe funcționale totale**, capabile să genereze toate închiderile pseudo-tranzitive. Aceste mulțimi definesc **acoperiri minimale**.

Dependențe funcționale

- ▶ O mulțime de dependențe funcționale totale F^* asociată unei mulțimi de attribute A definește o **acoperire minimală** dacă satisface următoarele proprietăți:
 - ▶ nici o dependență funcțională din F^* nu este redundantă;
 - ▶ toate dependențele funcționale totale între submulțimi ale lui A sunt în închiderea pseudo-tranzitivă a lui F^* .
- ▶ Orice mulțime de dependențe totale are cel puțin o acoperire minimală. Alegerea acoperirii minimale este punctul de start în modelarea schemelor relaționale.

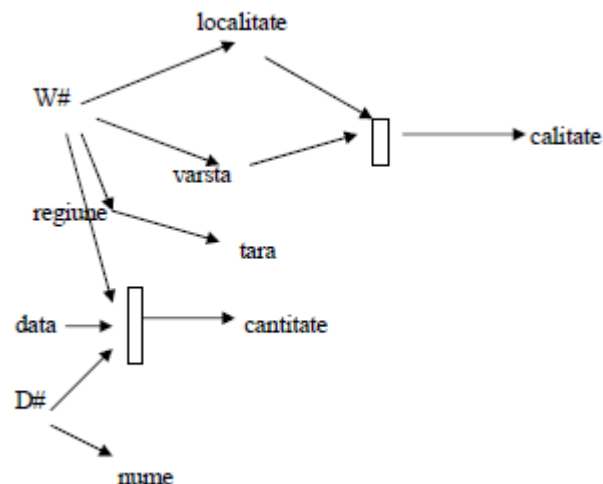
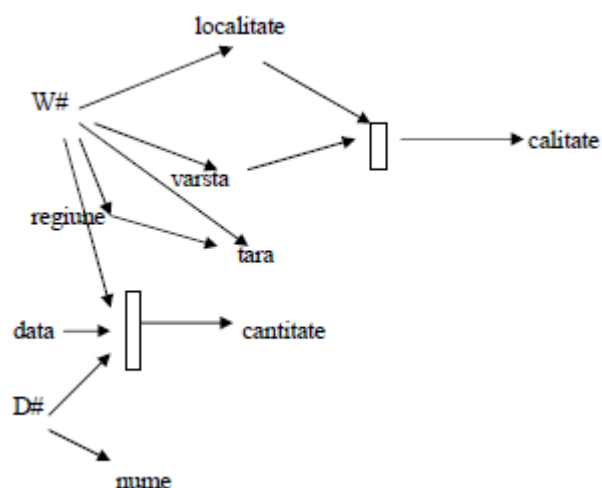
Dependențe funcționale

- ▶ Dependențele funcționale între atributele bazei pot fi reprezentate grafic.
 - ▶ Fie $A = \{A_1, A_2, \dots, A_n\}$ o mulțime de atribute și fie o mulțime de dependențe funcționale $\{X_i \rightarrow A_j\}$, unde X_i este o submulțime a lui A .
- ▶ **Graful dependențelor funcționale** este un **graf direcționat bipartit**, definit astfel:
 1. pentru fiecare atribut A_j există un singur **nod** având eticheta A_j ;
 2. pentru fiecare dependență funcțională de forma $A_i \rightarrow A_j$, există un **arc** de la A_i la A_j ;
 3. pentru fiecare dependență funcțională de forma $X_i \rightarrow A_j$, unde mulțimea X_i este definită de $X_i = \{A_{i_1}, \dots, A_{i_p}\}$ cu $p > 1$, există un **nod auxiliar** etichetat prin X_i și **o mulțime de arce** plecând de la A_{i_1}, \dots, A_{i_p} pentru a obține pe X_i și **printr-un arc adițional** de la X_i la A_j . Nodurile X_i se reprezintă prin dreptunghiuri.

Dependențe funcționale

Exemple:

1. Graful dependențelor funcționale pentru schema relațională CONSUMATOR_DE_VIN(W#, localitate, varsta, calitate, regiune, tara, D#, nume, data, cantitate) și acoperirea minimală.



Dependențe funcționale

2. Graful dependențelor funcționale pentru schema relațională OBIECTIV_INVESTITIE.
Dependențele sunt deduse din regulile impuse de beneficiar!

