

BAZE DE DATE

CURS 9

Normalizare (partea 2)

Necesitatea normalizării

- Anomaliile care apar în lucrul cu baza de date se produc din cauza dependențelor care există între datele din cadrul relațiilor bazei de date.
 - Dependențele sunt plasate greșit în tabele!!!
- O dependență poate provoca:
 - **anomalii** la inserare, modificare sau ștergere
 - **redundanță** în date
 - probleme de **reconexiune**.

Necesitatea normalizării

- **Normalizarea** are drept **scop**:
 - suprimarea **redundanței** logice,
 - evitarea **anomaliilor** la reactualizare,
 - rezolvarea **problemei reconexiunii**.
- Există o **teorie matematică a normalizării** al cărei autor este E.F. Codd.
 - Soluția: **construirea unor tabele standard** (forme normale).
- Normalizarea este **procesul reversibil de transformare a unei relații**, în relații de structură mai simplă.
 - Procesul este reversibil în sensul că **nicio informație nu este pierdută în timpul transformării**.
 - O relație este într-o formă normală particulară dacă ea satisface o **mulțime specificată de constrângeri**.

Necesitatea normalizării

► Relație universală + mulțime de anomalii

- Orice formă normală se obține aplicând o **scemă de descompunere**. Există două tipuri de descompuneri.

► Descompuneri ce conservă dependențele.

- descompunerea relației R în proiecțiile R_1, R_2, \dots, R_k , a.î. dependențele lui R sunt echivalente (au închideri pseudo-tranzitive identice) cu reuniunea dependențelor lui R_1, R_2, \dots, R_k .

► Descompuneri fără pierderi de informație (*L-join*).

- descompunerea relației R într-o mulțime de proiecții R_1, R_2, \dots, R_j , a.î. pentru orice instanță a lui R este adevărată relația:

$$R = \text{JOIN}(\Pi_{B_1}(R), \Pi_{B_2}(R), \dots, \Pi_{B_j}(R))$$

- Relația inițială = **compunerea naturală a relațiilor obținute prin descompunere**.

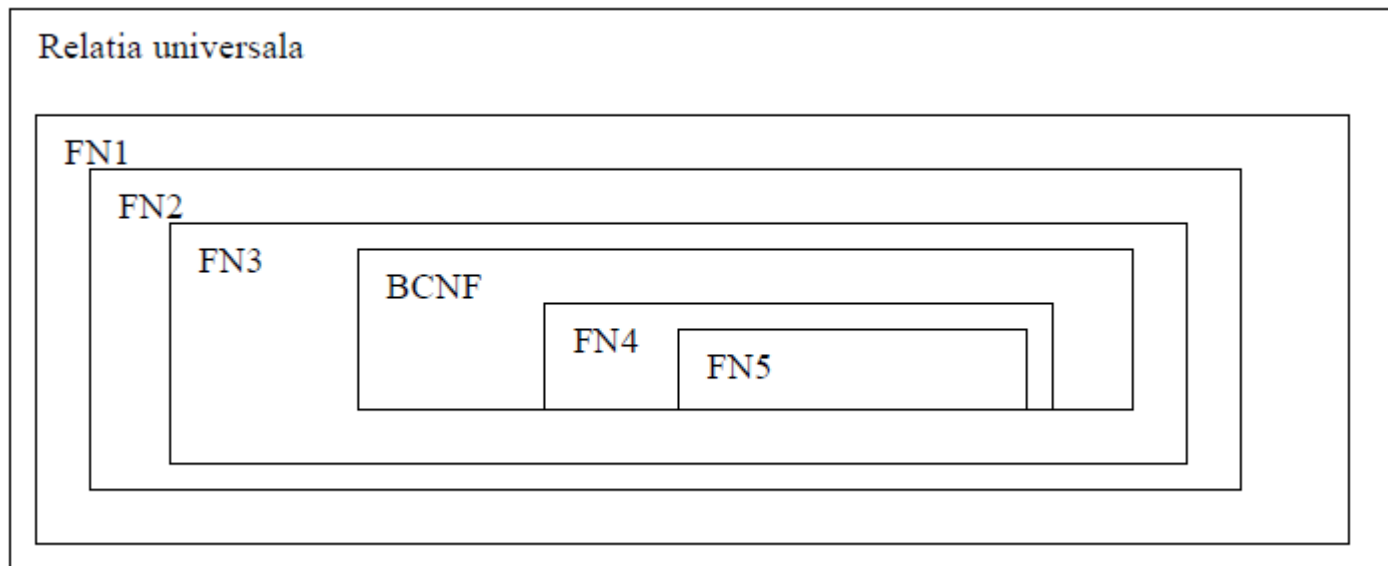
Necesitatea normalizării

- **Formele normale sunt obținute prin descompuneri fără pierderi de informație.**
- O descompunere fără pierdere de informație, utilizată în procesul normalizării, este dată de **regula Casey-Delobel:**
- Fie $R(A)$ o schemă relațională și fie α, β, γ o partiție a lui A . Presupunem că α determină funcțional pe β . Atunci:

$$R(A) = \text{JOIN}(\Pi_{\alpha \cup \beta}(R), \Pi_{\alpha \cup \gamma}(R)).$$

- $\alpha \cup \beta \rightarrow$ mulțimea atributelor care intervin în dependențele funcționale;
- $\alpha \cup \gamma \rightarrow$ reprezintă reuniunea determinantului cu restul atributelor lui A .

Necesitatea normalizării



Forma normală 1 (FN1)

- O relație este în prima formă normală dacă fiecărui atribut care o compune îi corespunde o valoare indivizibilă (atomică).
- Exemplu:

VEHICUL (Non FN1)

| cod_persoana# | vehicule |
|----------------------|-----------------|
| P1 | DL, RC, FF |
| P2 | RM, VW |
| P3 | DL |

Forma normală 1 (FN1)

Varianta 1:

VEHICUL (FN1)

| cod_persoana# | vehicul# |
|----------------------|-----------------|
| P1 | DL |
| P1 | RC |
| P1 | FF |
| P2 | RM |
| P2 | VW |
| P3 | DL |

Varianta 2:

VEHICUL (FN1)

| cod_persoana# | vehicul1 | vehicul2 | vehicul3 |
|----------------------|-----------------|-----------------|-----------------|
| P1 | DL | RC | FF |
| P2 | RM | VW | null |
| P3 | DL | null | null |

Forma normală 1 (FN1)

Varianta 3:

VEHICUL 1

| cod_persoana# | vehicul |
|----------------------|----------------|
| P1 | DL |
| P2 | RM |
| P3 | DL |

VEHICUL 2

| cod_persoana# | vehicul |
|----------------------|----------------|
| P1 | RC |
| P2 | VW |

VEHICUL 3

| cod_persoana# | vehicul |
|----------------------|----------------|
| P1 | FF |

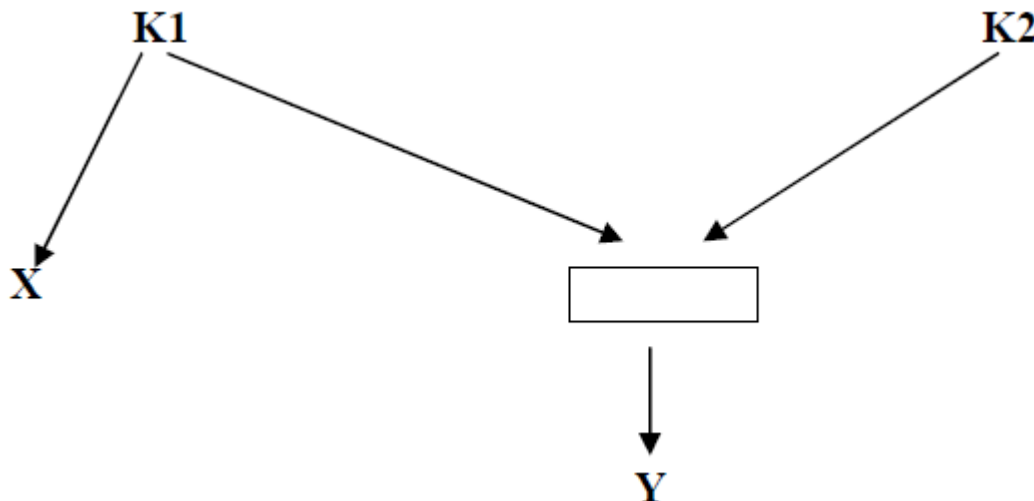
Forma normală 2 (FN2)

- O relație R este în a doua formă normală dacă și numai dacă:
 - relația R este în FN1;
 - fiecare atribut care nu este cheie (nu participă la cheia primară) este dependent de întreaga cheie primară.
- FN2 interzice manifestarea unor dependențe funcționale parțiale în cadrul relației R !!!
- Ce consecință imediată se observă?

Forma normală 2 (FN2)

► Aplicarea regulii Casey-Delobel pentru FN2

- Fie relația $R(K1, K2, X, Y)$, unde $K1$ și $K2$ definesc cheia primară, iar X și Y sunt mulțimi de atribute, astfel încât $K1 \rightarrow X$. Din cauza dependenței funcționale $K1 \rightarrow X$ care arată că R nu este în FN2, se înlocuiește R (fără pierdere de informație) prin două proiecții $R1(K1, K2, Y)$ și $R2(K1, X)$.



Forma normală 2 (FN2)

► Exemplu:

atasat_la

| Cod_salariat# | Job_cod | Nr_proiect# | Functia | Suma |
|---------------|-------------|-------------|------------|------|
| S1 | Programator | P1 | Supervizor | 60 |
| S1 | Programator | P2 | Cercetator | 25 |
| S1 | Programator | P3 | Auxiliar | 10 |
| S3 | Vanzator | P3 | Supervizor | 60 |
| S5 | Inginer | P3 | Supervizor | 60 |

De ce nu este în FN2?

Forma normală 2 (FN2)

► Transformarea în FN2:

atasat_2a

| Cod_salariat# | Nr_proiect# | Funcția | Suma |
|---------------|-------------|------------|------|
| S1 | P1 | Supervizor | 60 |
| S1 | P2 | Cercetator | 25 |
| S1 | P3 | Auxiliar | 10 |
| S3 | P3 | Supervizor | 60 |
| S5 | P3 | Supervizor | 60 |

atasat_2b

| Cod_salariat# | Job_cod |
|---------------|-------------|
| S1 | Programator |
| S3 | Vanzator |
| S5 | Inginer |

Forma normală 2 (FN2)

► Exemplu:

Presupunem că un șantier poate executa mai multe lucrări de bază și că o lucrare poate fi executată de mai multe șantiere.

LUCRARE(cod_obiectiv#, cod_lucrare#, nume);

SANTIER(nr_santier#, specialitate, sef);

EXECUTA(cod_obiectiv#, cod_lucrare#, nr_santier#, descriere, functie, conducator, data_inceput, data_sfarsit).

- Pentru relația EXECUTA sunt evidente dependențele:

$\{\text{cod_obiectiv\#, cod_lucrare\#}\} \rightarrow \{\text{data_inceput, data_sfarsit}\},$

$\{\text{cod_obiectiv\#, cod_lucrare\#, nr_santier\#}\} \rightarrow \{\text{descriere, functie, conducator}\}.$

- Relația EXECUTA este în FN1, dar nu este în FN2. Se aplică regula Casey Delobel:

EXECUTA_1(cod_obiectiv#, cod_lucrare#, nr_santier#, descriere, functie, conducator)

EXECUTA_2(**cod_obiectiv#, cod_lucrare#**, data_inceput, data_sfarsit).

Forma normală 3 (FN3)

- **Intuitiv**, o relație R este în a **treia formă normală** dacă și numai dacă:
 - relația R este în **FN2**;
 - fiecare atribut care nu este cheie (nu participă la o cheie) **depinde direct de cheia primară**.
- Fie R o relație, X o submulțime de attribute ale lui R și A un atribut al relației R . A este **dependent tranzitiv** de X dacă există Y astfel încât $X \rightarrow Y$ și $Y \rightarrow A$ (A nu aparține lui Y și Y nu determină pe X). X nu este dependent funcțional de Y sau A !
 - De exemplu, dacă $K_1, K_2, K_3 \rightarrow A_1$ și dacă $K_1, K_2, A_1 \rightarrow A_2$, atunci $K_1, K_2, K_3 \rightarrow K_1, K_2, A_1$ și $K_1, K_2, A_1 \rightarrow A_2$. Prin urmare, A_2 este dependent tranzitiv de K_1, K_2, K_3 .

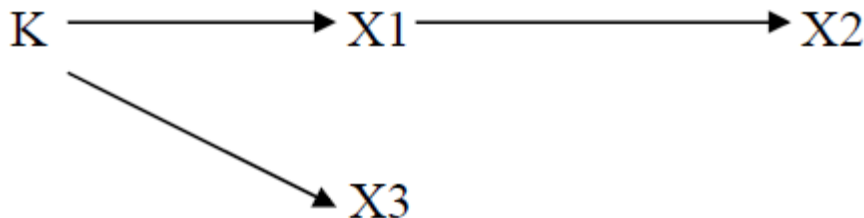
Forma normală 3 (FN3)

- **Formal**, o relație R este în **a treia formă normală** dacă și numai dacă:
 - relația R este în **FN2**;
 - fiecare atribut care nu este cheie (nu participă la o cheie) **nu este dependent tranzitiv de nici o cheie a lui R .**
- O relație este în FN3 dacă și numai dacă fiecare atribut (coloană) care nu este cheie, **depinde de cheie, de întreaga cheie și numai de cheie.**

Forma normală 3 (FN3)

► Aplicarea regulii Casey-Delobel pentru FN3

- Fie relația $R(K, X_1, X_2, X_3)$, unde atributul X_2 depinde tranzitiv de K , iar K este cheia primară a lui R . Presupunem că $K \rightarrow X_1 \rightarrow X_2$.
- Din cauza dependenței funcționale $X_1 \rightarrow X_2$ care arată că R nu este în FN3, se înlocuiește R (fără pierdere de informație) prin două proiecții $R1(K, X_1, X_3)$ și $R2(X_1, X_2)$.



Forma normală 3 (FN3)

► Exemplu:

atasat_2a

| Cod_salariat# | Nr_proiect# | Funcția | Suma |
|---------------|-------------|------------|------|
| S1 | P1 | Supervizor | 60 |
| S1 | P2 | Cercetator | 25 |
| S1 | P3 | Auxiliar | 10 |
| S3 | P3 | Supervizor | 60 |
| S5 | P3 | Supervizor | 60 |

De ce nu este în FN3?

Forma normală 3 (FN3)

► Transformarea în FN3:

atasat_3a

| Cod_salariat# | Nr_proiect# | Funcția |
|---------------|-------------|------------|
| S1 | P1 | Supervizor |
| S1 | P2 | Cercetator |
| S1 | P3 | Auxiliar |
| S3 | P3 | Supervizor |
| S5 | P3 | Supervizor |

atasat_3b

| Funcția | Suma |
|------------|------|
| Supervizor | 60 |
| Cercetator | 25 |
| Auxiliar | 10 |

Forma normală 3 (FN3)

► Exemplu:

În tabelul EXECUTA1(cod_obiectiv#, cod_lucrare#, nr_santier#, descriere, functie, conducator) continuă să existe redundanță în date.

- Atributul *conducator* depinde indirect de cheia primară prin intermediul atributului *functie*. Între attributele relației există dependențele:

$\{\text{cod_obiectiv\#}, \text{cod_lucrare\#}, \text{nr_santier\#}\} \rightarrow \{\text{descriere}\},$

$\{\text{cod_obiectiv\#}, \text{cod_lucrare\#}, \text{nr_santier\#}\} \rightarrow \{\text{functie}\} \rightarrow \{\text{conducator}\}.$

- Se aplică regula Casey-Delobel. Relația se descompune, prin eliminarea dependențelor funcționale tranzitive, în proiecțiile:

EXECUTA11(cod_obiectiv#, cod_lucrare#, nr_santier#, descriere, **functie**)

EXECUTA12(**functie**, **conducator**).

Schema de sinteză pentru obținerea lui FN3

- **Algoritmul de sinteză** construiește o acoperire minimală F a dependențelor funcționale totale.
 - Se elimină attributele și dependențele funcționale **redundante**.
 - Mulțimea F este partiționată în grupuri F_i , astfel încât în fiecare grup F_i sunt dependențe funcționale care au **același membru stâng** și nu există două grupuri având același membru stâng.
 - Fiecare grup F_i produce o **schemă FN3**.
- Algoritmul realizează o descompunere ce conservă dependențele.

Schema de sinteză pentru obținerea lui FN3

► **Algoritm SNF3** (aducerea unei relații în FN3 prin utilizarea unei scheme de sinteză):

1. Se determină F o acoperire minimală a lui E (mulțimea dependențelor funcționale).
2. Se descompune mulțimea F în grupuri notate F_i , astfel încât în cadrul fiecărui grup să existe dependențe funcționale având aceeași parte stângă.
3. Se determină perechile de chei echivalente (X, Y) în raport cu F (două mulțimi de attribute X, Y sunt chei echivalente dacă în mulțimea de dependențe E există atât dependența $X \rightarrow Y$, cât și dependența $Y \rightarrow X$).

Schema de sinteză pentru obținerea lui FN3

4. Pentru fiecare pereche de chei echivalente: se identifică grupurile F_i și F_j care conțin dependențele funcționale cu partea stângă X și respectiv Y ; se formează **un nou grup de dependențe F_{ij}** , care va conține dependențele funcționale având **membrul stâng (X, Y)** ; se elimină grupurile F_i și F_j , iar locul lor va fi luat de grupul F_{ij} .
5. Se determină o **acoperire minimală a lui F** , care va include toate dependențele $X \rightarrow Y$, unde X și Y sunt chei echivalente (celelalte dependențe sunt redundante).
6. Se construiesc **relații FN3** (câte o relație pentru fiecare grup de dependențe funcționale).

Schema de sinteză pentru obținerea lui FN3

- Se observă că **algoritmul solicită**:
 - determinarea unei **acoperiri minimale** (algoritmii **EAR** și **EDF**);
 - determinarea **închiderii (A^+) unei mulțimi de attribute **A**** în raport cu mulțimea de dependențe funcționale **E** (algoritm **AIDF**).
- Determinarea acoperirii minimale presupune eliminarea atributelor și dependențelor redundante.
- Acoperirea minimală **nu este unică** și depinde de ordinea în care sunt eliminate aceste attribute și dependențe redundante.

Schema de sinteză pentru obținerea lui FN3

- **Algoritm EAR** (elimină attributele redundante din determinantul dependențelor funcționale)
- Pentru fiecare dependență funcțională din E și pentru fiecare atribut din partea stângă a unei dependențe funcționale:
 - **Pas1.** Se elimină atributul considerat.
 - **Pas2.** Se calculează închiderea părții stângi reduce.
 - **Pas3.** Dacă închiderea conține toate attributele din determinantul dependenței, atunci atributul eliminat la pasul 1 este redundant și rămâne eliminat. În caz contrar, atributul nu este redundant și se reintroduce în partea stângă a dependenței funcționale.

Schema de sinteză pentru obținerea lui FN3

- **Algoritm EDF** (elimină dependențele funcționale redundante din E)
- Pentru fiecare dependență funcțională $X \rightarrow Y$ din E:
 - **Pas1.** Se elimină dependența din E.
 - **Pas2.** Se calculează închiderea X^+ , în raport cu mulțimea redusă de dependențe.
 - **Pas3.** Dacă Y este inclus în X^+ , atunci dependența $X \rightarrow Y$ este redundantă și rămâne eliminată. În caz contrar, se reintroduce în E.

Schema de sinteză pentru obținerea lui FN3

- **Algoritm AIDF** (determină închiderea lui A)
 - **Pas1.** Se caută dacă există în E dependențe $X \rightarrow Y$ pentru care determinantul X este o submulțime a lui A, iar determinatul Y nu este inclus în A.
 - **Pas2.** Pentru fiecare astfel de dependență funcțională se adaugă mulțimii A attributele care constituie determinatul dependenței.
 - **Pas3.** Dacă nu mai există dependențe funcționale de tipul de la pasul 1, atunci $A^+ = A$.
- Exemplu!

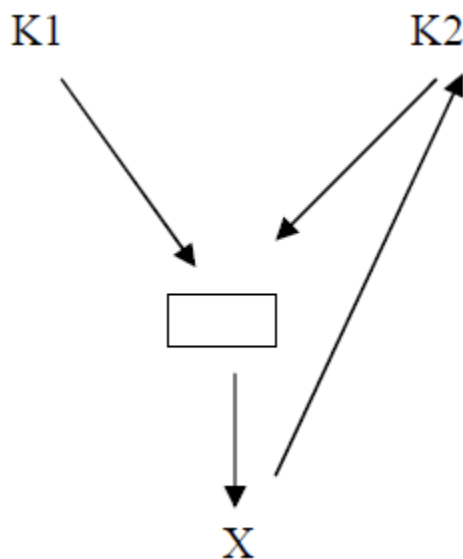
Forma normală Boyce-Codd (BCNF)

- Determinantul este un atribut sau o mulțime de attribute neredundante, care constituie un identificator unic pentru alt atribut sau altă mulțime de attribute ale unei relații date.
- **Intuitiv**, o relație R este în forma normală Boyce-Codd dacă și numai dacă **fiecare determinant este o cheie candidat**.
- **Formal**, o relație R este în forma normală Boyce-Codd dacă și numai dacă pentru orice dependență funcțională totală $X \rightarrow A$, X este o cheie (candidat) a lui R .

Forma normală Boyce-Codd (BCNF)

- **Regula Casey Delobel** pentru $R(K1\#, K2\#, X)$ presupunând că există dependența: $X \rightarrow K2$.

→ $R1(K1\#, X)$ și $R2(X\#, K2)$



Forma normală Boyce-Codd (BCNF)

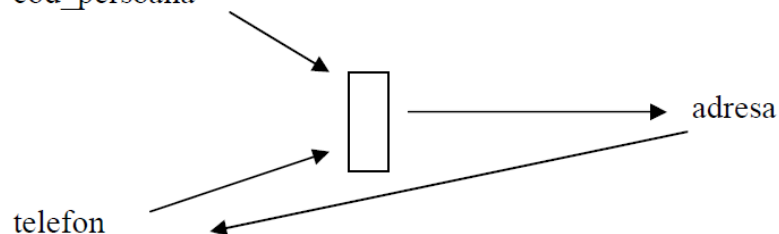
- Pentru ca o relație să fie adusă în BCNF **nu trebuie în mod obligatoriu să fie în FN3.**
- Se pot aduce în BCNF și relații aflate în FN1 sau FN2.
 - Acest lucru este posibil întrucât dependențele funcționale parțiale și cele tranzitive sunt tot dependențe noncheie, adică dependențe ai căror determinanți nu sunt chei candidat.

Forma normală Boyce-Codd (BCNF)

► Exemplu:

ADRESA(cod_persoana#, telefon#, adresa)

cod_persoana



- În dependența *adresa _ Telefon* se observă că determinantul nu este o cheie candidat. Relația ADRESA se descompune în:

ADRESA_1(cod_persoana#, adresa);

ADRESA_2(adresa#, telefon).

- Relațiile sunt în BCNF, se conservă datele, dar nu se conservă dependențele (s-a pierdut *cod_persoana, telefon → adresa*).

Forma normală Boyce-Codd (BCNF)

► Exemplu:

- Relația `INVESTESE_IN` leagă entitățile `INVESTITOR` și `OBIECTIV_INVESTITIE`. Ea are schema relațională:

`INVESTESE_IN(cod_contractant#, cod_obiectiv#, nr_contract, cota_parte).`

- Între atributele relației există dependențele:

`{cod_contractant#, cod_obiectiv#} → {nr_contract, cota_parte},`

`{nr_contract} → {cod_obiectiv}.`

- Se aplică regula Casey-Delobel și se aduce relația în BCNF.

`INVESTESE_IN_1(cod_obiectiv, nr_contract#);`

`INVESTESE_IN_2(cod_contractant#, nr_contract, cota_parte).`

Forma normală Boyce-Codd (BCNF)

► **Algoritm TFBCNF** (aducerea unei relații R din FN1 în BCNF)

1. Dacă relația conține cel mult două atribute, atunci R este în BCNF și algoritmul s-a terminat.
2. Dacă relația conține mai mult de două atribute, se consideră toate perechile (X, Y) de atribute distincte din A .
3. Se determină A_1^+ , închiderea mulțimii $A_1 = A - \{X, Y\}$.
4. Dacă pentru orice pereche (X, Y) , $X \notin A_1^+$ atunci relația R este în BCNF și algoritmul s-a terminat.
5. În caz contrar (pentru cel puțin o pereche (X, Y) , X aparține lui A_1^+), relația R nu este în BCNF.
6. Se reduce progresiv schema relației și se reia algoritmul, exploatând relația redusă. Orice relație obținută prin reducerea lui R și care este în BCNF se consideră ca făcând parte din descompunerea lui R în procesul aducerii sale în BCNF.

Forma normală 4 (FN4)

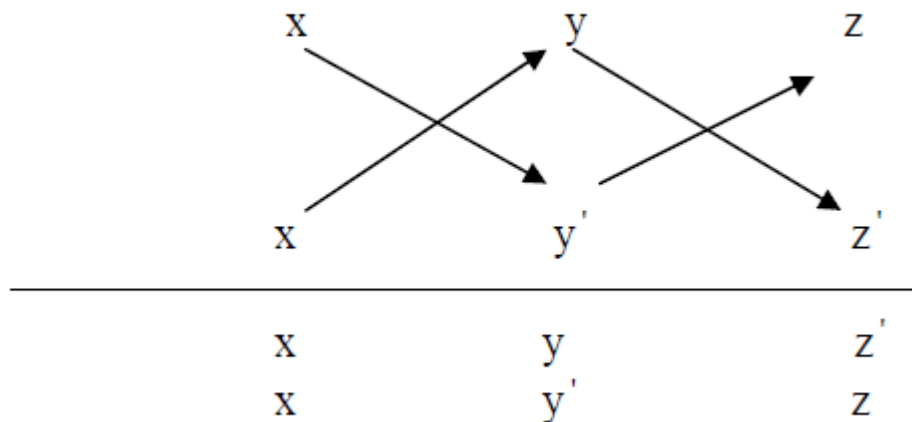
- FN4 elimină redundanțele datorate relațiilor $m:n$, adică datorate **dependenței multiple**.
- **Intuitiv**, o relație R este în a patra formă normală dacă și numai dacă relația este în BCNF și nu conține relații $m:n$ independente.

Forma normală 4 (FN4)

- Fie R o relație definită pe o mulțime de attribute $A = \{A_1, A_2, \dots, A_n\}$ și fie $X, Y, Z \subset A$. Se spune că X **multidetermină** pe Z sau că Z este multidependent de X :
 - dacă pentru fiecare valoare a lui Z în R există numai o valoare pentru perechea (X, Y) ;
 - dacă valoarea lui Z depinde numai de valoarea lui X .
- Acest tip de dependență, numită și multivaloare sau multidependență (MVD) se notează prin $X \twoheadrightarrow Z$.
- **Intuitiv, multidependența reprezintă situația în care valoarea unui atribut (sau a unei mulțimi de attribute) determină o mulțime de valori a altui atribut (sau mulțimi de attribute)!!!**

Forma normală 4 (FN4)

- Multidependența $X \twoheadrightarrow Y$ poate fi gândită ca o **regulă de deducție**:
 - dacă tuplurile $\langle x, y, z \rangle$ și $\langle x, y', z' \rangle$ sunt în relație la un moment r , atunci la momentul r sunt în relație și tuplurile $\langle x, y, z' \rangle$ și $\langle x, y', z \rangle$.



Forma normală 4 (FN4)

- Orice dependență funcțională este o multidependență.
- Afirmația inversă nu este adevărată.
- Dacă $X \rightarrow Y$ (FD), atunci pentru oricare două tupluri $\langle x, y, z \rangle$ și $\langle x, y', z' \rangle$, se obține $y = y'$. Prin urmare în relație apar tuplurile $\langle x, y', z \rangle$ și $\langle x, y, z' \rangle$ și deci $X \twoheadrightarrow Y$ (MVD).

Forma normală 4 (FN4)

- Fie W, V, X, Y și Z submulțimi de attribute ale unei scheme relaționale R . Fiind dată o mulțime T de multidependențe există o mulțime completă de axiome (Ax1–Ax8) care permit obținerea tuturor multidependențelor ce se pot deduce din mulțimea T . -> *Vezi curs!*
- O **multidependență elementară** este o multidependență care are părți stângi și drepte minimale (nu există $X' \subset X$ și $Y' \subset Y$ a.i. $X' \twoheadrightarrow Y'$).
- **Formal**, relația R este în a patra formă normală dacă și numai dacă:
 - R este în BCNF;
 - orice dependență multivaloare este o dependență funcțională.

Forma normală 4 (FN4)

- O relație BCNF este în FN4 dacă pentru orice multidependență elementară de forma $X \twoheadrightarrow Y$, X este o supercheie a lui R .
- **Regula de descompunere în relații FN4.**
 - Fie $R(X, Y, Z)$ o schemă relațională care nu este în FN4 și fie $X \twoheadrightarrow Y$ o multidependență elementară care nu este de forma „CHEIE \twoheadrightarrow atribut”.
 - Această relație este descompusă prin proiecție în două relații:

$$R = \text{JOIN}(\Pi_{X \cup Y}(R), \Pi_{X \cup Z}(R)).$$

Forma normală 4 (FN4)

- O relație BCNF este în FN4 dacă pentru orice multidependență elementară de forma $X \twoheadrightarrow Y$, X este o supercheie a lui R .
- **Regula de descompunere în relații FN4.**
 - Fie $R(X, Y, Z)$ o schemă relațională care nu este în FN4 și fie $X \twoheadrightarrow Y$ o multidependență elementară care nu este de forma „CHEIE \twoheadrightarrow atribut”.
 - Această relație este descompusă prin proiecție în două relații:

$$R = \text{JOIN}(\Pi_{X \cup Y}(R), \Pi_{X \cup Z}(R)).$$

Forma normală 4 (FN4)

► Exemplu:

► Fie relația `INVESTITIE(cod_contractant#, denumire#, telefon#)` și presupunem că un investitor poate avea mai multe numere de telefon și că poate investi în mai multe obiective.

► Între attributele relației există multidependențele:

`cod_contractant# →→ denumire;`

`cod_contractant# →→ telefon.`

► Relația `INVESTITIE` este în BCNF. Pentru a aduce relația în FN4 o vom descompune prin proiecție în două relații:

`INVESTITIE_1(cod_contractant#, denumire#),`

`INVESTITIE_2(cod_contractant#, telefon#).`

► `INVESTITIE = JOIN(INVESTITIE_1, INVESTITIE_2).`

Forma normală 5 (FN5)

- FN5 își propune eliminarea redundanțelor care apar în relații $m:n$ dependente.
 - În general, aceste relații nu pot fi descompuse.
 - S-a arătat că o relație de tip 3 este diferită de trei relații de tip 2. Există totuși o excepție, și anume, dacă relația este ciclică
- **Intuitiv**, o relație R este în forma normală 5 dacă și numai dacă:
 - relația este în FN4;
 - nu conține dependențe ciclice.

Concluzii

1. **FN1 \rightarrow FN2** elimină redundanțele datorate dependenței netotale a atributelor care nu participă la o cheie, față de cheile lui R . Se suprimă dependențele funcționale care nu sunt totale.
2. **FN2 \rightarrow FN3** elimină redundanțele datorate dependenței tranzitive. Se suprimă dependențele funcționale tranzitive.
3. **FN3 \rightarrow BCNF** elimină redundanțele datorate dependenței funcționale. Se suprimă dependențele în care partea stângă nu este o supercheie.
4. **BCNF \rightarrow FN4** elimină redundanțele datorate multidependenței. Se suprimă toate multidependențele care nu sunt și dependențe funcționale.
5. **FN4 \rightarrow FN5** elimină redundanțele datorate dependenței ciclice. Se suprimă toate *join*-dependențele care nu sunt implicate de o cheie.
6. **BCNF, FN4 și FN5** corespund la regula că orice determinant este o cheie, dar de fiecare dată dependența cu care se definește determinantul este alta și anume dependența funcțională, multidependența sau *join*-dependența).
7. Descompunerea unei relații FN2 în FN3 conservă datele și dependențele, pe când descompunerea unei relații FN3 în BCNF și, respectiv, a unei relații BCNF în FN4 conservă doar datele.