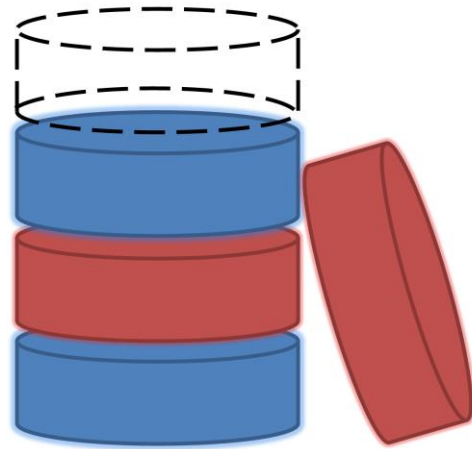


Forme normale



Redundanța

Redundanța este cauza principală a majorității problemelor legate de structura bazelor de date relaționale:

- *spațiu utilizat,*
- *anomalii de inserare / stergere / actualizare*

Redundanța

- *Dependențele funcționale* pot fi utilizate pentru identificarea problemelor de proiectare și sugerează posibile îmbunătățiri
- Fie relația R cu 3 attribute, ABC.
 - **Nici o DF:** nu avem redundanțe.
 - **Pentru $A \rightarrow B$:** Mai multe înregistrări pot avea aceeași valoare pentru A, caz în care avem valori identice pentru B!

Tehnica de rafinare a structurii: *descompunerea*

Descompunerea trebuie folosită cu "măsură":

- Este necesară o rafinare? Există motive de descompunere a relației?
- Ce probleme pot să apară prin descompunere?

Forme Normale

■ Dacă o relație se află într-o *formă normală* particulară avem certitudinea că anumite categorii de probleme sunt eliminate/minimizate → ne ajută să decidem dacă descompunerea unei relații este necesară sau nu.

■ Formele normale bazate pe DF sunt:

- *prima formă normală (1NF),*
- *a doua formă normală (2NF),*
- *a treia formă normală (3NF),*
- *forma normală Boyce-Codd (BCNF).*

$$\{BCNF \subseteq 3NF, 3NF \subseteq 2NF, 2NF \subseteq 1NF\}$$

1NF

Definiție. O relație se află în *Prima Formă Normală* (**1NF**) dacă fiecare atribut al relației poate avea doar valori atomice (deci listele și mulțimile sunt excluse)

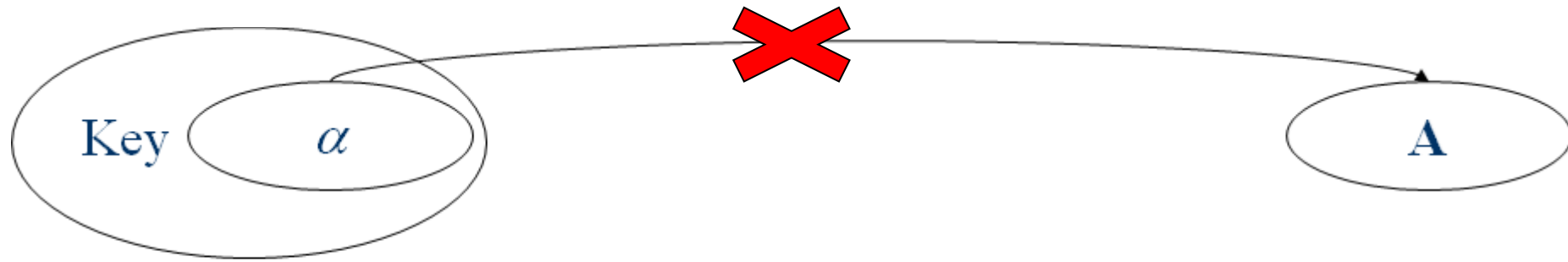
(această condiție este implicită conform definiției modelului relațional)

2NF

Spunem că avem o *dependență funcțională parțială* într-o relație atunci când un atribut *ne-cheie* este dependent funcțional de o parte a unei chei candidat a relației (dar nu de întreaga cheie).

Definiție. O relație se află în *A Doua Formă Normală* (**2NF**) dacă este 1NF și nu are dependențe parțiale.

2NF



Partial dependencies (A not in a KEY)

BCNF

Definiție. O relație R ce satisface dependențele funcționale F se află în *Forma Normală Boyce-Codd* (BCNF) dacă se află în 2NF și pentru toate $\alpha \rightarrow A$ din F^+ :

- $A \in \alpha$ (DF *trivială*), sau
- α conține o cheie a lui R.

R este în BCNF dacă singurele dependențe funcționale satisfăcute de R sunt cele corespunzătoare constrângerilor de cheie.

BCNF



A not in a KEY

3NF

Definitie. O relație R ce satisface dependențele funcționale F se află în *A Treia Formă Normală (3NF)* dacă se află în 2NF și pentru toate $\alpha \rightarrow A$ din F^+

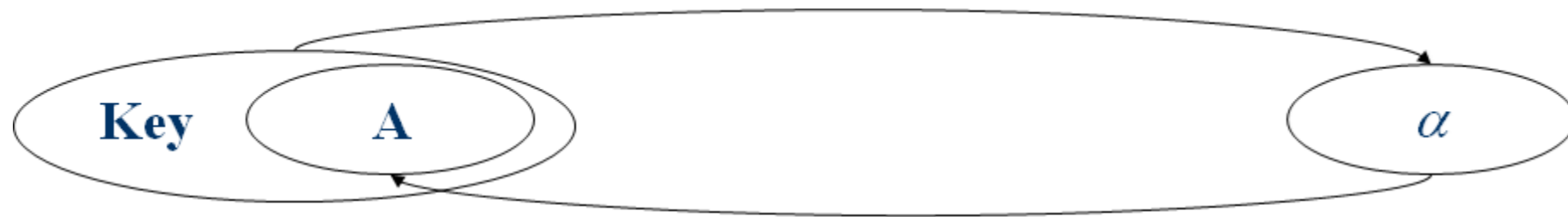
- $A \in \alpha$ (DF *trivială*), sau
- α conține o cheie de-a lui R, sau
- A este un atribut prim.

■ Dacă R este în BCNF, evident este și în 3NF.

■ Dacă R este în 3NF, este posibil ca să apară anumite redundanțe. Este un compromis, utilizat atunci când BCNF nu se poate atinge.

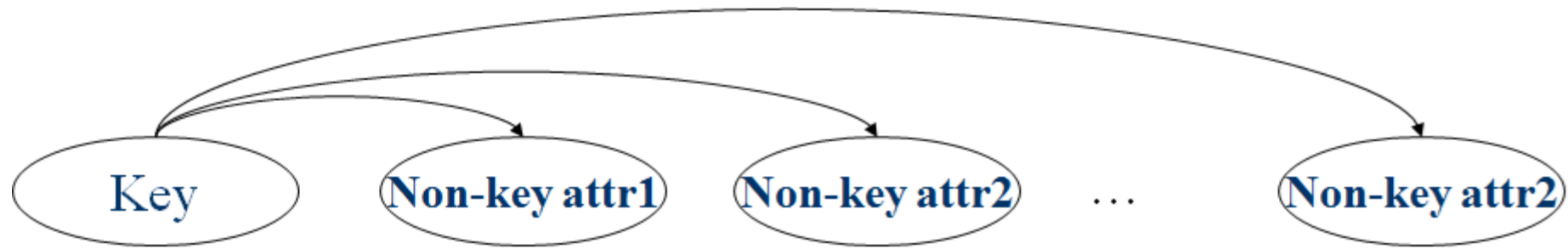
■ Descompunerea *cu joncțiune fără pierderi* & *cu păstrarea dependențelor* a relației R într-o mulțime de relații 3NF este întotdeauna posibilă.

3NF



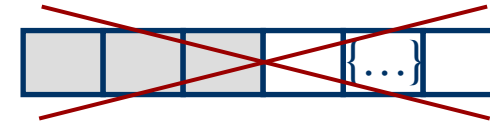
A is in KEY

BCNF & 3NF

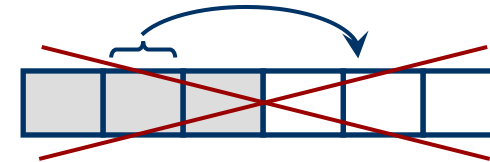
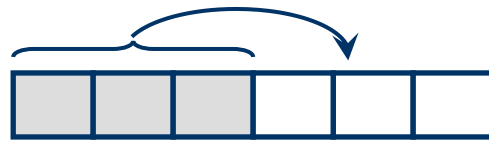


Forme Normale bazate pe DF

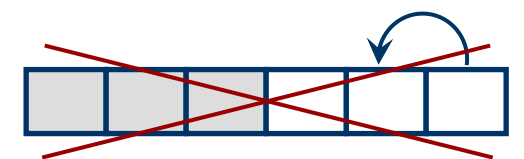
1NF - toate valorile atributelor sunt atomice



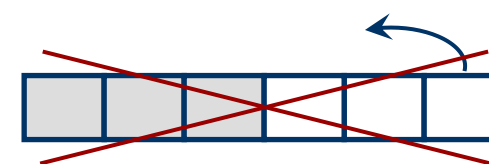
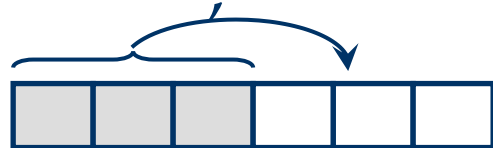
2NF - toate attributele non-cheie depind de **întreaga** cheie (nu sunt dependențe parțiale)



3NF - tabele în 2NF și toate attributele non-prime depind **doar** de cheie (nu sunt dependențe tranzitive)



BCNF - Toate dependențele sunt date de chei



Normalizarea pe scurt

Fiecare atribut depinde:

de cheie,► definiție cheie

de întreaga cheie,► 2NF

și de nimic altceva

decât de cheie► BCNF

Normalizarea pe scurt

Fiecare atribut ^{neprim} depinde:

- de cheie,► definiție cheie
- de întreaga cheie,► 2NF
- și de nimic altceva
decât de cheie► 3NF

Exemple de nerespectare a FN

2NF - toate attributele neprime trebuie să depindă de **întreaga** cheie

Exam (*Student*, *Course*, Teacher, Grade)



```
graph LR; SC["(Student, Course)"] --> T["Teacher"]
```

3NF - toate attributele neprime trebuie să depindă **doar** de cheie

Dissertation(*Student*, Title, Teacher, Department)



```
graph LR; ST["(Student, Title)"] --> D["Department"]
```

BCNF - toate DF sunt implicate de cheile candidat

Schedule (*Day*, *Route*, *Bus*, Driver)



```
graph LR; DR["(Day, Route)"] --> Bus_D["Driver"]
```

"Strategia" de normalizare

BCNF prin descompunere cu jonctiune fără pierderi și păstrarea dependențelor
(prima alegere)

3NF prin descompunere cu jonctiune fără pierderi și păstrarea dependențelor
(a doua alegere)

*deoarece uneori dependențele
nu pot fi păstrate pt a obține BCNF*

Descompunerea în BCNF

Fie relația R cu dependențele funcționale F . Dacă $\alpha \rightarrow A$ nu respectă BCNF, descompunem R în $R - A$ și αA .

Aplicarea repetată a acestei idei va conduce la o colecție de relații care

- sunt în BCNF;
- conduc la joncțiune fără pierderi;
- garantează terminarea.

Descompunerea în BCNF

Exemplu:

$R(\underline{C}, S, J, D, P, Q, V)$, **C** cheie,

$\{JP \rightarrow C, SD \rightarrow P, J \rightarrow S\}$

Alegem $SD \rightarrow P$, decompunând în

$(\underline{S}, \underline{D}, P), (\underline{C}, S, J, D, Q, V)$.

Apoi alegem $J \rightarrow S$, decompunând $(\underline{C}, S, J, D, Q, V)$ în

(\underline{J}, S) și $(\underline{C}, J, D, Q, V)$

În general, mai multe dependențe pot cauza nerespectarea BCNF. Ordinea în care le ``abordăm'' poate conduce la decompuneri de relații complet diferite!

În general, descompunerea în BCNF nu păstrează dependențele.

Exemplu. $R(C, S, Z)$, $\{CS \rightarrow Z, Z \rightarrow C\}$

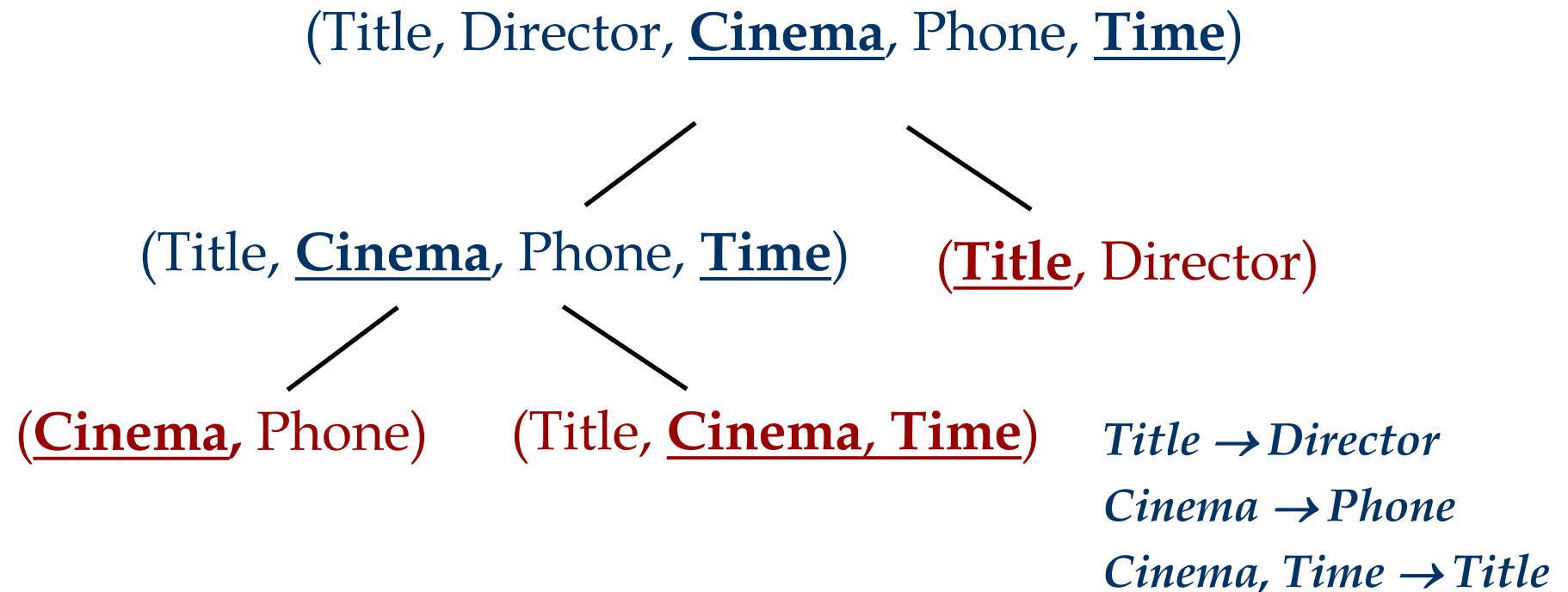
Exemplu. $R(C, S, J, P, D, Q, V)$ descompus în
 (S, D, P) , (J, S) și (C, J, D, Q, V)
nu păstrează dependențele inițiale $\{JP \rightarrow C, SD \rightarrow P, J \rightarrow S\}$).

! adăugând JPC la mulțimea de relații obținem *descompunere cu păstrarea dependențelor*.

⇒ BCNF & redundanță

Exemplu

1. Fie $\alpha \rightarrow A$ o DF din F ce nu respectă BCNF
2. Descompunem R în $R_1 = \alpha A$ și $R_2 = R - A$.
3. Dacă R_1 sau R_2 nu sunt în BCNF, descompunerea continuă



Descompunerea în 3NF

Evident, procedeul descompunerii din BCNF poate fi utilizat și pentru descompunerea 3NF.

- Cum asigurăm păstrarea dependențelor?
 - Dacă $X \rightarrow Y$ nu se păstrează, adăugăm XY .
 - Problema este că XY e posibil să nu respecte 3NF! (pp. că adăugăm CJP pt `păstrarea' $JP \rightarrow C$. Dacă însă are loc și $J \rightarrow C$ atunci nu e corect.)
- *Rafinare:* În loc de a utiliza mulțimea inițială F , folosim o *acoperire minimală a lui F* .

Redundanța în DF

■ Un atribut $A \in \alpha$ e redundant în DF $\alpha \rightarrow B$ dacă

$$(F - \{\alpha \rightarrow B\}) \cup \{\alpha - A \rightarrow B\} \equiv F$$

■ Pentru a verifica dacă $A \in \alpha$ e redundant în $\alpha \rightarrow B$, calculăm $(\alpha - A)^+$. Apoi $A \in \alpha$ e redundant în $\alpha \rightarrow B$ dacă $B \in (\alpha - A)^+$

■ *Exercițiu:* Care sunt attributele redundante în $AB \rightarrow C$ având:
 $\{AB \rightarrow C, A \rightarrow B, B \rightarrow A\}$?

Redundanța în DF

- O DF $f \in F$ e **redundantă** dacă $F - \{f\}$ e echivalent cu F
- Verificăm că $\alpha \rightarrow A$ e redundantă în F , calculând α^+ pe baza $F - \{\alpha \rightarrow A\}$. Atunci $\alpha \rightarrow A$ e redundantă în F dacă $A \in \alpha^+$
- *Exercițiu:* Care sunt dependențele funcționale redundante în:
 $\{A \rightarrow C, A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow A\}$?

Acoperire minimală

■ O **acoperire minimală** pentru mulțimea **F** de dependente functionale este o multime **G** de dependente functionale pentru care:

1. Fiecare DF din **G** e de forma $\alpha \rightarrow A$
2. Pt fiecare DF $\alpha \rightarrow A$ din **G**, α nu are attribute redundante
3. Nu sunt DF redundante in **G**
4. **G** și **F** sunt echivalente

Fiecare multime de DF are cel puțin o acoperire minimală!

Algoritm de calcul al acoperirii minime pt F:

1. Folosim descompunerea pentru a obține DF cu 1 atribut în partea dreaptă.
2. Se elimină attributele redundante
3. Se elimină dependențele funcționale redundante

Calcul Acoperire Minimală

Fie $F = \{ABCD \rightarrow E, E \rightarrow D, A \rightarrow B, AC \rightarrow D\}$

Atributele BD din $ABCD \rightarrow E$ sunt redundante:

$\Rightarrow F = \{AC \rightarrow E, E \rightarrow D, A \rightarrow B, AC \rightarrow D\}$

$AC \rightarrow D$ este redundanta

$\Rightarrow F = \{AC \rightarrow E, E \rightarrow D, A \rightarrow B\}$

care este o acoperire minimala

*Acoperirile minimale nu sunt unice
(depind de ordinea de alegere a DF/atr. redundante)*

Decompunere în 3NF

Input: Schema R with F which is a minimal cover

Output: A dependency-preserving, lossless-join 3NF decomposition of R

Initialize $D = \emptyset$

Apply *union rule* to combine FDs in F with same L.H.S. into a single FD

For each FD $\alpha \rightarrow \beta$ in F do

 Insert the relation schema $\alpha\beta$ into D

 Insert δ into D , where δ is some key of R

Remove redundant relation schema from D as follows:

 delete R_i from D if $R_i \subseteq R_j$, where $R_j \in D$

return D

Exemplu

Fie $R(A,B,C,D,E)$ cu dependentele functionale:

$$F = \{ABCD \rightarrow E, E \rightarrow D, A \rightarrow B, AC \rightarrow D\}$$

- Acoperirea minimala a F este $\{AC \rightarrow E, E \rightarrow D, A \rightarrow B\}$
- Unica cheie: AC
- R nu e in 3NF deoarece $A \rightarrow B$ nu respecta 3NF
- descompunerea 3NF a R :
 - Relatii pentru fiecare DF: $R_1(A, C, E)$, $R_2(E,D)$, si $R_3(A,B)$
 - Relatie pentru cheia lui R : $R_4(A, C)$
 - Eliminare relatie redundanta: R_4 (deoarece $R_4 \subseteq R_1$)
 - \Rightarrow descompunerea 3NF este $\{R_1(A,C,E), R_2(E,D), R_3(A,B)\}$
- Descompunerea 3NF nu este unică. Depinde de:
 - Alegerea acoperirii minimale sau
 - Alegerea relatiei redundante care va fi eliminata

Din nou despre... descompunere

■ Descompunerea este ultima solutie de rezolvare a problemelor generate de redundanțe & anomalii

■ Excesul poate fi nociv!

Exemplu:

$R = (\text{Teacher}, \text{Dept}, \text{Phone}, \text{Office})$

cu DF $F = \{\text{Teacher} \rightarrow \text{Dept Phone Office}\}$

$R = (\text{Teacher}, \text{Dept}, \text{Phone}, \text{Office})$



■ Uneori, din motive de performanță se practica de-normalizarea

Dependențe multivaloare

| course | teacher | book |
|---------------|----------------|--------------|
| alg101 | Green | Alg Basics |
| alg101 | Green | Alg Theory |
| alg101 | Brown | Alg Basics |
| alg101 | Brown | Alg Theory |
| logic203 | Green | Logic B. |
| logic203 | Green | Logic F. |
| logic203 | Green | Logic intro. |

relația e în BCNF

Dependențe multivaloare

■ Fie α, β două submulțimi de attribute din R . Dependența multivaloare $\alpha \twoheadrightarrow \beta$ este respectată de R dacă, pentru fiecare instanță validă r a lui R , fiecare valoare α este asociată cu o mulțime de valori pentru β și această mulțime valori este independentă de valorile altor attribute.

■ Formal: dacă $\alpha \twoheadrightarrow \beta$ e respectată de R și $\gamma = R - \alpha\beta$, următoarea afirmație e adevărată pentru orice instanță validă r a lui R :

$$\begin{aligned} & t_1, t_2 \in r \text{ și } \pi_\alpha(t_1) = \pi_\alpha(t_2) \Rightarrow \\ & \exists t_3 \in r \text{ a.î. } \pi_{\alpha\beta}(t_1) = \pi_{\alpha\beta}(t_3) \text{ și } \pi_\gamma(t_2) = \pi_\gamma(t_3) \end{aligned}$$

Ca și consecință, pentru t_2 și t_1 se poate deduce că există și $t_4 \in r$ a.î. $\pi_{\alpha\beta}(t_2) = \pi_{\alpha\beta}(t_4)$ și $\pi_\gamma(t_1) = \pi_\gamma(t_4)$

Dependențe multivaloare

| | X | Y | Z |
|---------|---|-------|-------|
| t_1 → | a | b_1 | c_1 |
| t_2 → | a | b_2 | c_2 |
| t_3 → | a | b_1 | c_2 |
| t_4 → | a | b_2 | c_1 |

$$\forall t_1, t_2 \in r \text{ și } \pi_x(t_1) = \pi_x(t_2) \Rightarrow \\ \exists t_3 \in r \text{ astfel încât} \\ \pi_{XY}(t_1) = \pi_{XY}(t_3), \\ \pi_Z(t_2) = \pi_Z(t_3)$$

Reguli adiționale:

Complementare: $X \twoheadrightarrow Y \Rightarrow X \twoheadrightarrow R - XY$

Augumentare: $X \twoheadrightarrow Y, Z \subseteq W \Rightarrow WX \twoheadrightarrow YZ$

Tranzitivitate: $X \twoheadrightarrow Y, Y \twoheadrightarrow Z \Rightarrow X \twoheadrightarrow Z - Y$

Replicare: $X \rightarrow Y \Rightarrow X \twoheadrightarrow Y$

Fuzionare: $X \twoheadrightarrow Y, W \cap Y = \emptyset, W \rightarrow Z, Z \subseteq Y \Rightarrow X \rightarrow Z$

A patra formă normală (4NF)

Definiție. Fie R o schemă relațională și F o mulțime de dependențe funcționale și multivaloare pe R .

Spunem că R este în a patra forma normală NF4 dacă, pentru orice dependență multivaloare $X \twoheadrightarrow Y$:

- $Y \subseteq X$ sau
- $XY = R$ sau
- X e super-cheie

A patra formă normală (4NF)

| course | teacher | book |
|----------|---------|--------------|
| alg101 | Green | Alg Basics |
| alg101 | Green | Alg Theory |
| alg101 | Brown | Alg Basics |
| alg101 | Brown | Alg Theory |
| logic203 | Green | Logic B. |
| logic203 | Green | Logic F. |
| logic203 | Green | Logic intro. |

course \twoheadrightarrow teacher

Relatia se poate descompune in:

(Course, Teacher)

si (Course, Book)

| course | teacher |
|----------|---------|
| alg101 | Green |
| alg101 | Brown |
| logic203 | Green |

| course | book |
|----------|--------------|
| alg101 | Alg Basics |
| alg101 | Alg Theory |
| logic203 | Logic B. |
| logic203 | Logic F. |
| logic203 | Logic intro. |

Dependența *Join*

■ Spunem ca R satisface dependența *join*

$\otimes\{R_1, \dots, R_n\}$ dacă

R_1, R_2, \dots, R_n

este o descompunere cu joncțiuni fără pierderi a lui R.

O dependență multivaloare $X \twoheadrightarrow Y$ poate fi exprimată ca o dependență join:

$\otimes\{XY, X(R-Y)\}.$

A cincea formă normală (5NF)

O relație R este în NF5 dacă și numai dacă pentru orice dependență *join* a lui R :

- $R_i = R$ pentru un i oarecare, sau
- dependența este implicată de o mulțime de dependențe functionale din R în care partea stângă e o cheie pentru R