

# 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 apare 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)

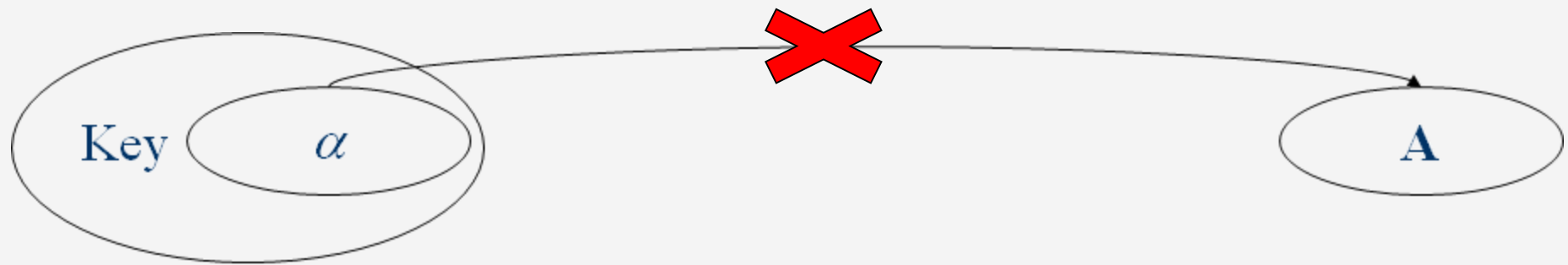
(această condiție este implicită conform definiției modelului relațional)

## 2NF

Spunem că avem o *dependență funcțională parțială* într-o relație atunci când un atribut *ne-cheie* este dependent functional de o parte a cheii primare 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)



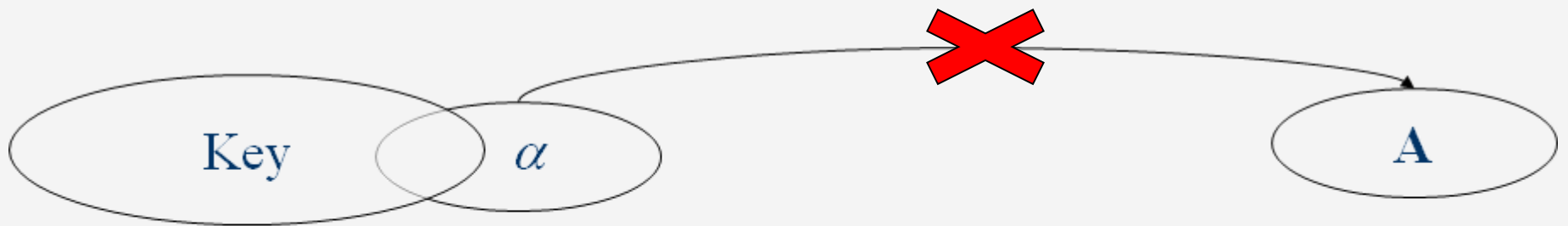
# BCNF

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

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

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

# BCNF



A not in a KEY

# 3NF

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

- $A \in \alpha$  (DF *trivială*), sau
- $\alpha$  este o supercheie pentru  $R$ , sau
- $A$  este un atribut prim.

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

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

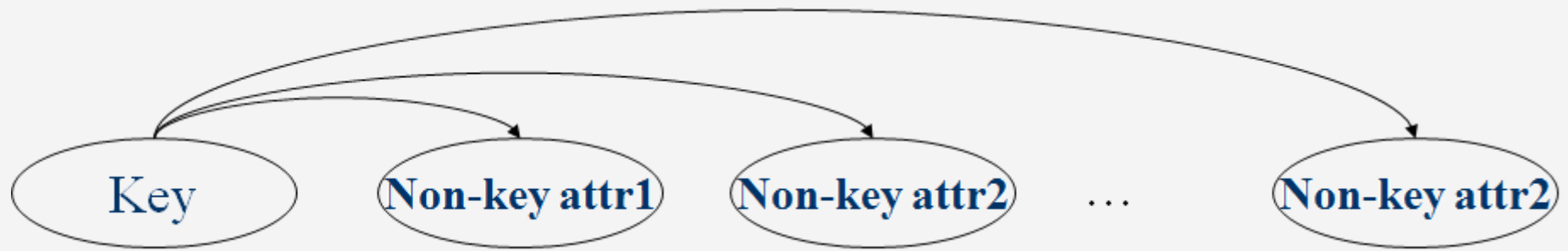
■ Descompunerea *cu joncțiune fără pierderi & cu păstrarea dependențelor* a relației  $R$  într-o mulțime de relații 3NF este întotdeauna posibilă.

3NF



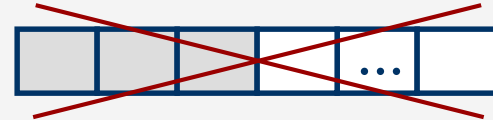
A is in KEY

# BCNF & 3NF

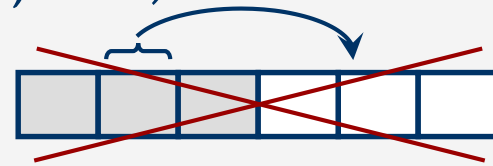
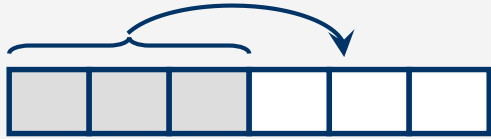


# Forme Normale bazate pe DF

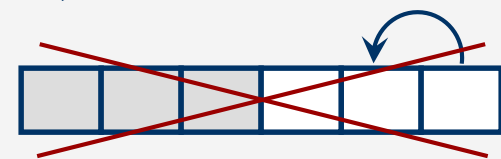
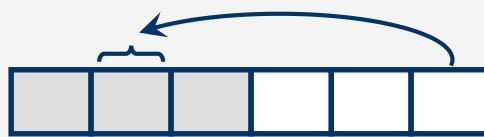
**1NF** - toate valorile atributelor sunt atomice



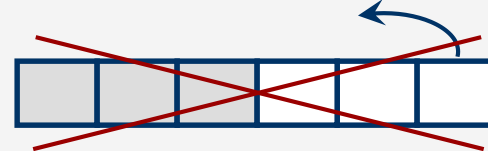
**2NF** - toate attributele non-cheie depind de **întreaga** cheie (nu sunt dependențe parțiale)



**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

# Exemple de nerespectare a FN

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

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



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

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



**BCNF** - toate DF sunt implicate de cheile candidat

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





# "Strategia" de normalizare

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

**3NF** prin descompunere cu joncțiune 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  $\alpha A$ .

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

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

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

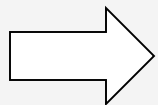
În general, mai multe dependențe pot cauza nerespectarea BCNF. Ordinea în care le ``abordăm`` poate conduce la decompuneri de relații complet diferite!

În general, descompunerea în BCNF nu păstrează dependențele.

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

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

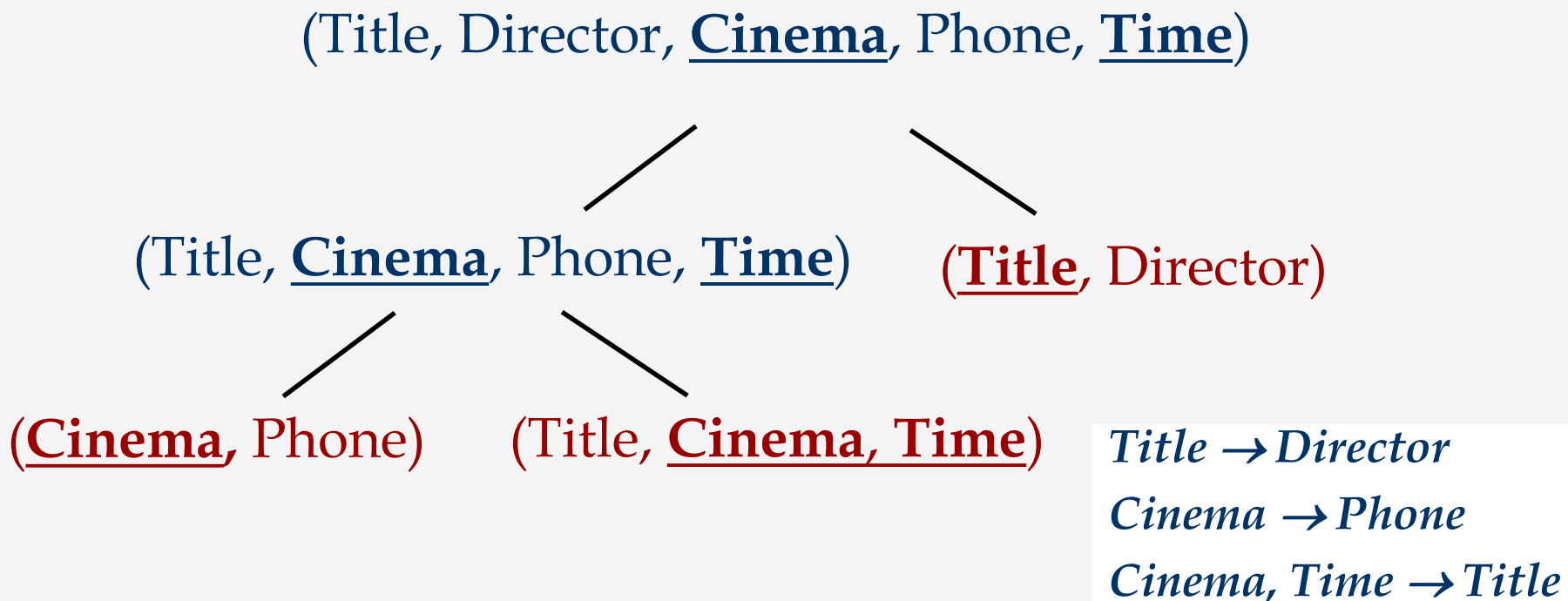
! adăugând JPC la mulțimea de relații obținem *descompunere cu păstrarea dependențelor*.



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  $\alpha^+$  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 mulțime **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**,  $\alpha$  nu are attribute redundante
3. NU sunt DF redundante in **G**
4. **G** și **F** sunt echivalente

Fiecare mulțime de DF are cel puțin o acoperire minimală!

Algoritm de calcul al acoperirii minimale pt F:

1. Folosim descompunerea pentru a obtine DF cu 1 atribut in partea dreapta.
2. Se elimina attributele redundante
3. Se elimina dependentele functionale 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)*

# Decomposition in 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  $\delta$  into  $D$ , where  $\delta$  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

# BCNF vs 3NF

- BCNF: jonctiune fara pierderi (posibil să nu păstreze dependențele)
- 3NF: jonctiune fara pierderi & păstrare dependențe

R(Course, Teacher, Time) cu DF

$\{\text{Course} \rightarrow \text{Teacher}; \text{Teacher, Time} \rightarrow \text{Course}\}$

- Chei:  $\{\text{Course, Time}\}$  și  $\{\text{Teacher, Time}\}$
- R este în 3NF dar nu în BCNF
- descompunere BCNF  $\{R_1(\text{Course, Teacher}), R_2(\text{Course, Time})\}$  este (doar) cu jonctiune fara pierderi

# Din nou despre... descompunere

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

■ Excesul poate fi nociv!

Exemplu:

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

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

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



$R_1 = (\text{Teacher}, \text{Dept})$      $R_2 = (\text{Teacher}, \text{Phone})$      $R_3 = (\text{Teacher}, \text{Office})$

■ Uneori, din motive de performanță se practica denormalizarea

# Multi-valued dependencies

Let's consider a relation with attributes **course**, **teacher**, **book** (CTB). (a teacher T can teach course C and the recommended book for the course is B)

<b>course</b>	<b>teacher</b>	<b>book</b>
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.

This relation is in BCNF



# Multi-valued dependencies

■ Let  $\alpha, \beta$  be subsets of the attributes of  $R$ . The multi-valued dependency  $\alpha \twoheadrightarrow \beta$  is said to hold over  $R$  if, in every legal instance  $r$  of  $R$ , each  $\alpha$  value is associated with a set of  $\beta$  values and this set is independent of the values in other attributes.

■ Formally: If the MVD  $\alpha \twoheadrightarrow \beta$  holds over  $R$  and  $\gamma = R - \alpha - \beta$ , the following must be true for every legal instance  $r$  of  $R$ :

$$t_1, t_2 \in r \text{ and } \pi_{\alpha}(t_1) = \pi_{\alpha}(t_2) \Rightarrow \\ \exists t_3 \in r \text{ such that } \pi_{\alpha\beta}(t_1) = \pi_{\alpha\beta}(t_3) \text{ and } \pi_{\gamma}(t_2) = \pi_{\gamma}(t_3)$$

As a consequence if we take  $t_2$  and  $t_1$  we can deduce that there is also  $t_4 \in r$  such that  $\pi_{\alpha\beta}(t_2) = \pi_{\alpha\beta}(t_4)$  and  $\pi_{\gamma}(t_1) = \pi_{\gamma}(t_4)$

# Multi-valued dependencies (cont)

	<b>X</b>	<b>Y</b>	<b>Z</b>
$t_1 \longrightarrow$	a	$b_1$	$c_1$
$t_2 \longrightarrow$	a	$b_2$	$c_2$
$t_3 \longrightarrow$	a	$b_1$	$c_2$
$t_4 \longrightarrow$	a	$b_2$	$c_1$

$$\forall t_1, t_2 \in r \text{ and } \pi_x(t_1) = \pi_x(t_2) \Rightarrow \\ \exists t_3 \in r \text{ such that} \\ \pi_{XY}(t_1) = \pi_{XY}(t_3), \\ \pi_Z(t_2) = \pi_Z(t_3)$$

Additional rules:

**MVD Complementation:**  $X \twoheadrightarrow Y \Rightarrow X \twoheadrightarrow R - XY$

**MVD Augmentation:**  $X \twoheadrightarrow Y, Z \subseteq W \Rightarrow WX \twoheadrightarrow YZ$

**MVD Transitivity:**  $X \twoheadrightarrow Y, Y \twoheadrightarrow Z \Rightarrow X \twoheadrightarrow Z - Y$

**Replication:**  $X \rightarrow Y \Rightarrow X \twoheadrightarrow Y$

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

# Fourth Normal Form (4NF)

**Definition.** Let  $R$  be a relation schema.

$X, Y \subseteq \text{attributes}(R)$ ,  $F = \{\text{FDs} \cup \text{MVDs}\}$ .

$R$  is said to be in *Fourth Normal Form* (4NF) if  $\forall X \twoheadrightarrow Y \in \text{MVDs}$  that holds over  $R$ , one of the following statements is true:

- $Y \subseteq X$  or
- $XY = R$  or
- $X$  is a super-key

# Fourth Normal Form (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

This relation can be decomposed in:

(Course, Teacher)  
and (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.

# Join Dependencies

■ A join dependency (JD)  $\bowtie\{R_1, \dots, R_n\}$  is said to hold over a relation R if

$$R_1, R_2, \dots, R_n$$

is a lossless-join decomposition of R.

An MVD  $X \twoheadrightarrow Y$  over a relation R can be expressed as the join dependency

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

# Fifth Normal Form (5NF)

A relational schema is said to be in *Fifth Normal Form (5NF)* if  $\forall \otimes\{R_1, \dots, R_n\}$  that holds over  $R$ , one of the following statements is true:

- $R_i = R$  for some  $i$ , or
- the JD is implied by the set of those FDs over  $R$  in which the left side is a key for  $R$

Q&A

You have

Questions

We have

Answers