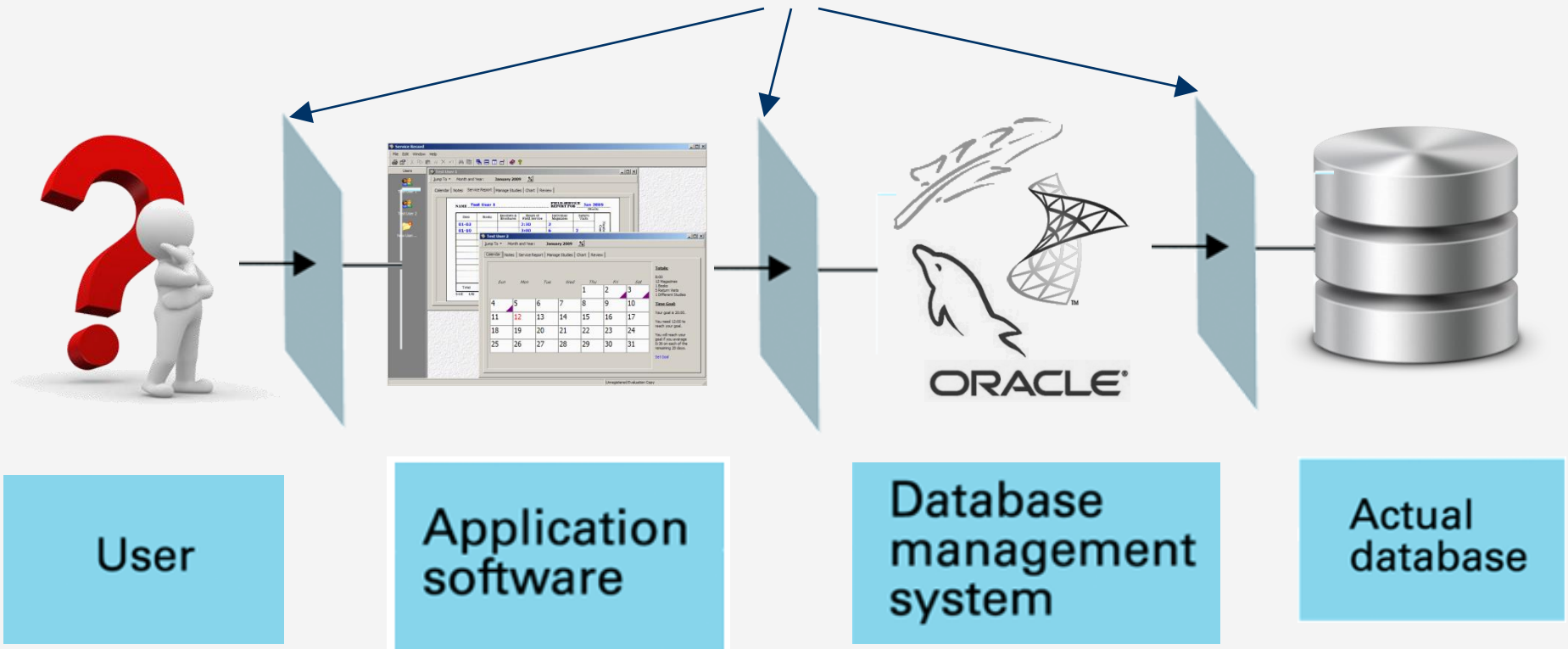


# Rafinarea structurii bazelor de date

(Dependențe funcționale)

# Nivelele de abstractizare

Nivele diferite  
de abstractizare



**STUDENT**

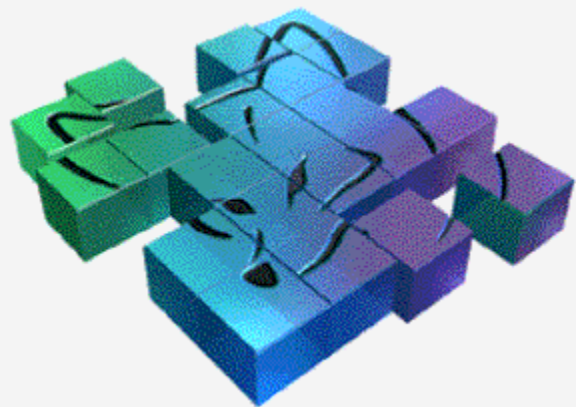
Name

Date of birth

Sex

CNP

Group



# Structura fizică

Faculty.dbc

42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

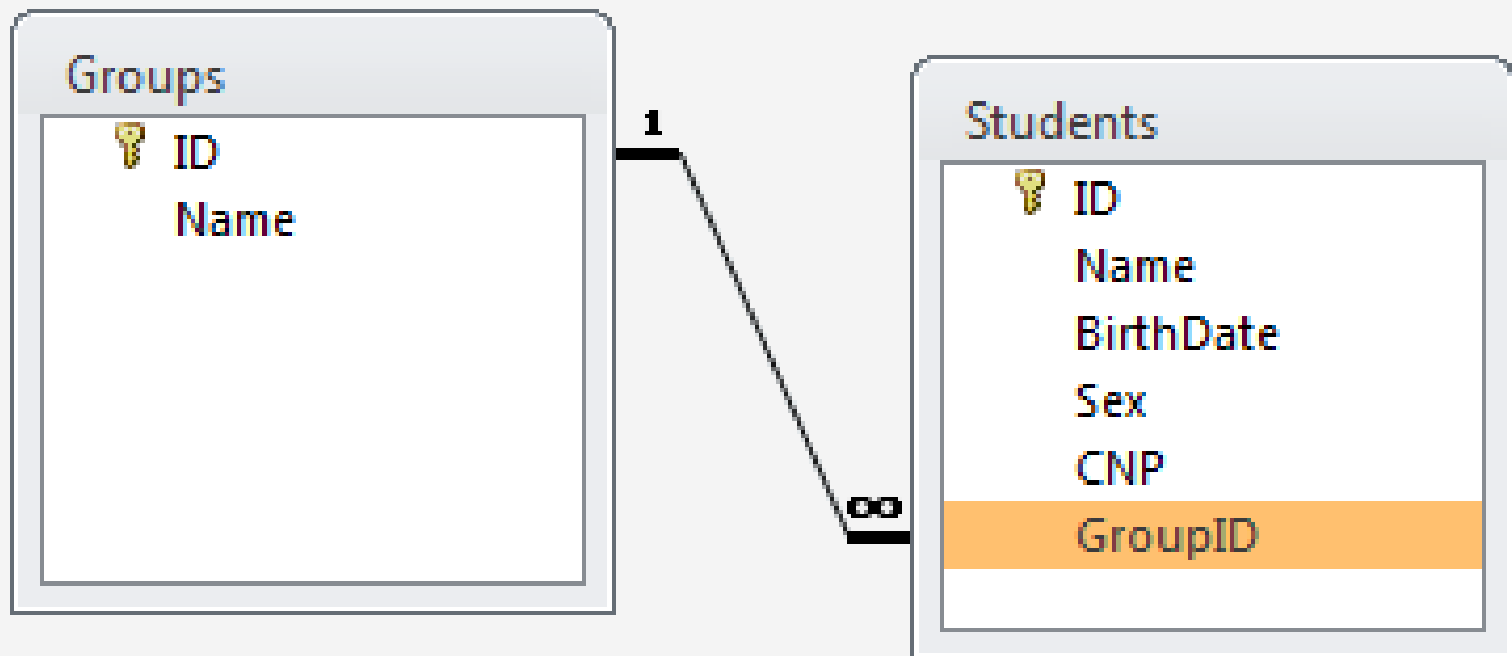
Students.dbf

20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

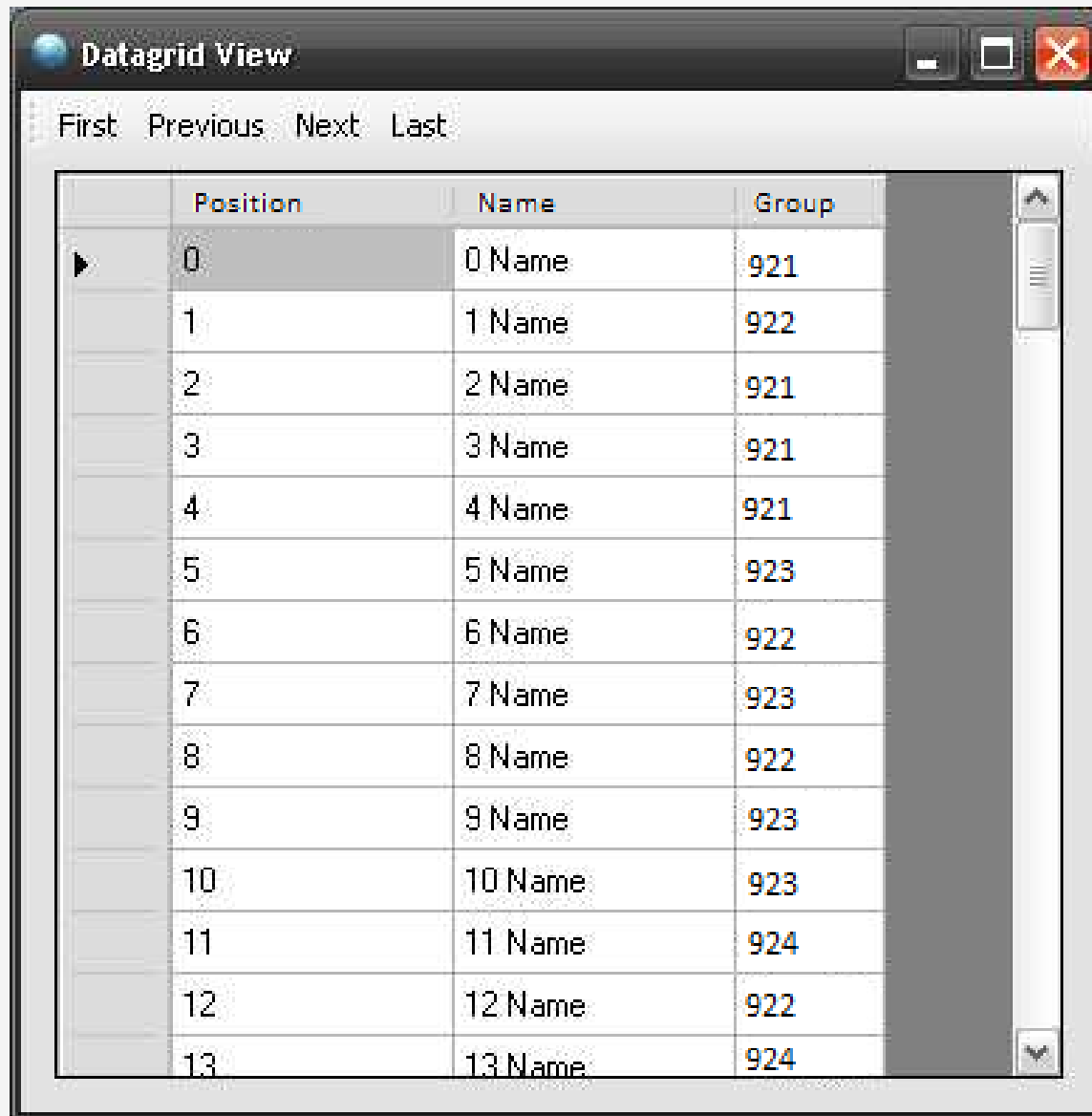
Groups.dbf

20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

# Structura conceptuală



# Vizualizare pentru utilizator



The image shows a software window titled "Datagrid View". Inside the window, there is a table with the following data:

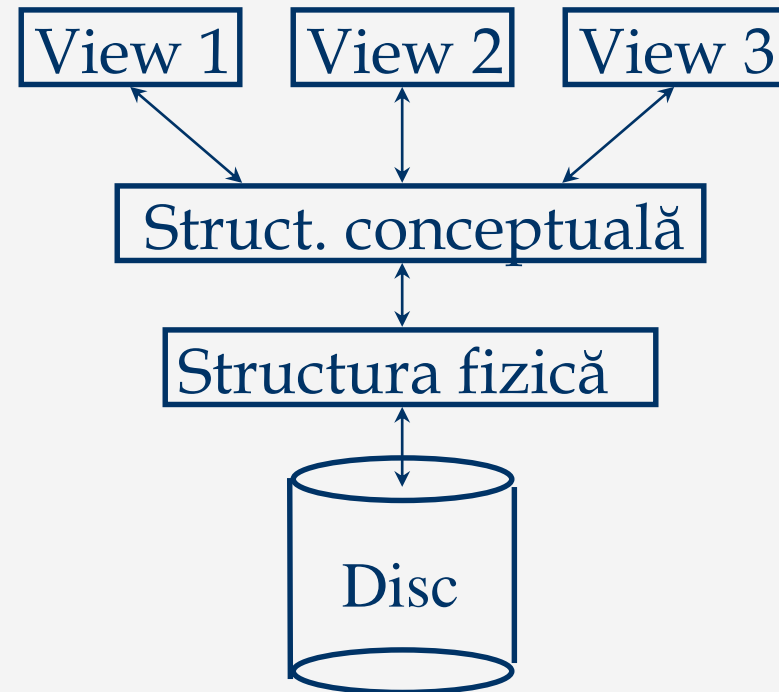
	Position	Name	Group
▶	0	0 Name	921
	1	1 Name	922
	2	2 Name	921
	3	3 Name	921
	4	4 Name	921
	5	5 Name	923
	6	6 Name	922
	7	7 Name	923
	8	8 Name	922
	9	9 Name	923
	10	10 Name	923
	11	11 Name	924
	12	12 Name	922
	13	13 Name	924

Navigation buttons: First, Previous, Next, Last.

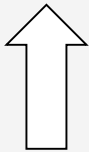
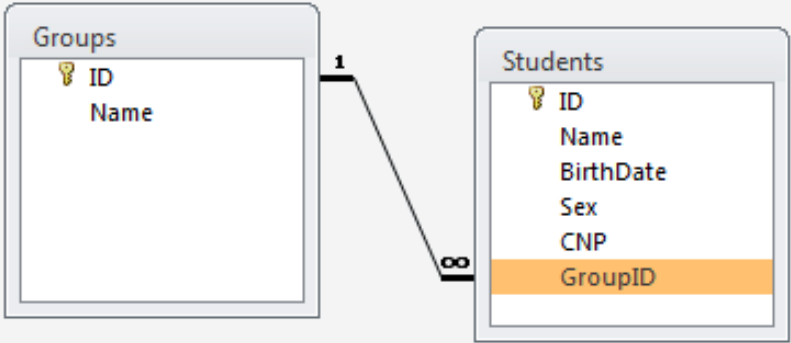
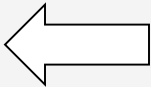
# Nivelele de abstractizare

- Mai multe structuri externe (views), câte o singură structură conceptuală (logică) și o structură fizică (internă).

- *Views* – cum văd utilizatorii datele.
- *Conceptual* - modelul logic compus din relații, attribute, etc
- *Fizic* - fișierele de date și indecși



Datagrid View			
First Previous Next Last			
	Position	Name	Group
▶	0	0 Name	921
	1	1 Name	922
	2	2 Name	921
	3	3 Name	921
	4	4 Name	921
	5	5 Name	923
	6	6 Name	922
	7	7 Name	923
	8	8 Name	922
	9	9 Name	923
	10	10 Name	923
	11	11 Name	924
	12	12 Name	922
	13	13 Name	924



Faculty.dbc

42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

Students.dbf

20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

Groups.dbf

20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

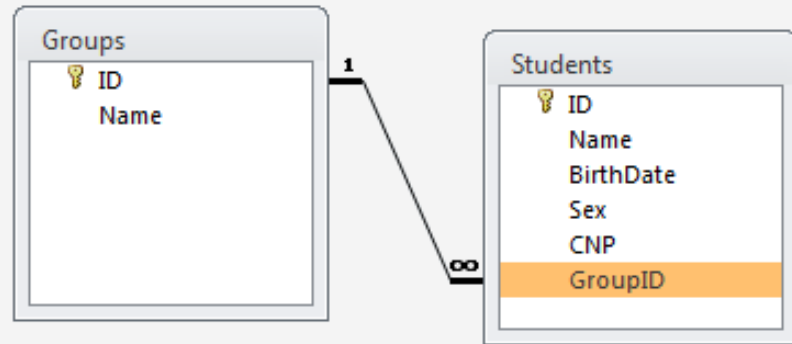
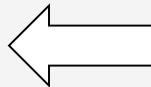


# Independența fizică a datelor

Datagrid View

First Previous Next Last

	Position	Name	Group
0		0 Name	921
1		1 Name	922
2		2 Name	921
3		3 Name	921
4		4 Name	921
5		5 Name	923
6		6 Name	922
7		7 Name	923
8		8 Name	922
9		9 Name	923
10		10 Name	923
11		11 Name	924
12		12 Name	922
13		13 Name	924



Faculty.mdf

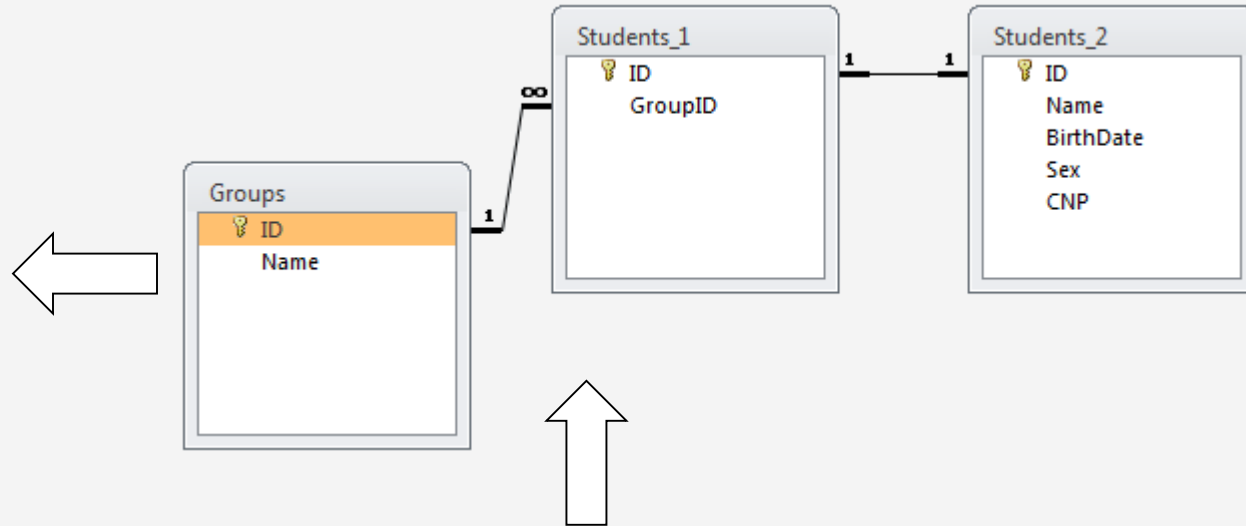
20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

# Independența logică a datelor

Datagrid View

First Previous Next Last

	Position	Name	Group
▶	0	0 Name	921
	1	1 Name	922
	2	2 Name	921
	3	3 Name	921
	4	4 Name	921
	5	5 Name	923
	6	6 Name	922
	7	7 Name	923
	8	8 Name	922
	9	9 Name	923
	10	10 Name	923
	11	11 Name	924
	12	12 Name	922
	13	13 Name	924



Faculty.mdf

20	20	20	31	56	31	2e	30
38	31	39	32	44	65	66	61
61	67	65	20	53	65	74	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	00	ff	01	00	7c	80	00
45	41	44	45	52	34	0f	53
4e	55	4d	42	45	52	14	34
34	21	0a	20	20	20	20	20
42	53	54	42	20	2e	30	37
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
20	20	20	20	20	20	20	20
03	50	81	01	f0	06	4f	52
4c	45	2d	49	44	01	14	3c
54	2d	4e	41	4d	45	01	13
45	54	2d	4e	55	4d	42	45

Structura bazei de date

=

Structura relațiilor

+

Constrângeri

## Exemplu: relația *MovieList*

<i>Title</i>	<i>Director</i>	<i>Cinema</i>	<i>Phone</i>	<i>Time</i>
The Hobbit	Jackson	Florin Piersic	441111	11:30
The Lord of the Rings 3	Jackson	Florin Piersic	441111	14:30
Adventures of Tintin	Spielberg	Victoria	442222	11:30
The Lord of the Rings 3	Jackson	Victoria	442222	14:00
War Horse	Spielberg	Victoria	442222	16:30

### Constrângeri:

- Fiecare film are un regizor
- Fiecare cinematograf are un număr de telefon
- Fiecare cinematograf începe proiecția unui singur film al un moment dat

# Proiectare defectuoasă!

<i>Title</i>	<i>Director</i>	<i>Cinema</i>	<i>Phone</i>	<i>Time</i>
The Hobbit	Jackson	Florin Piersic	441111	11:30
The Lord of the Rings 3	Jackson	Florin Piersic	441111	14:30
Adventures of Tintin	Spielberg	Victoria	442222	11:30
The Lord of the Rings 3	Jackson	Victoria	442222	14:00
War Horse	Spielberg	Victoria	442222	16:30

Anomalie de **inserare**

Anomalie de **ștergere**

Anomalie de **actualizare**

# Rafinarea unei structuri defectuoase prin *descompunerea* în mai multe structuri “bune”

## *Movies*

<i>Title</i>	<i>Director</i>
The Hobbit	Jackson
The Lord of the Rings 3	Jackson
Adventures of Tintin	Spielberg
War Horse	Spielberg

## *Screens*

<i>Cinema</i>	<i>Time</i>	<i>Title</i>
Florin Piersic	11:30	The Hobbit
Florin Piersic	14:30	The Lord of the Rings 3
Victoria	11:30	Adventures of Tintin
Victoria	14:00	The Lord of the Rings 3
Victoria	16:30	War Horse

## *Cinema*

<i>Cinema</i>	<i>Phone</i>
Florin Piersic	441111
Victoria	442222

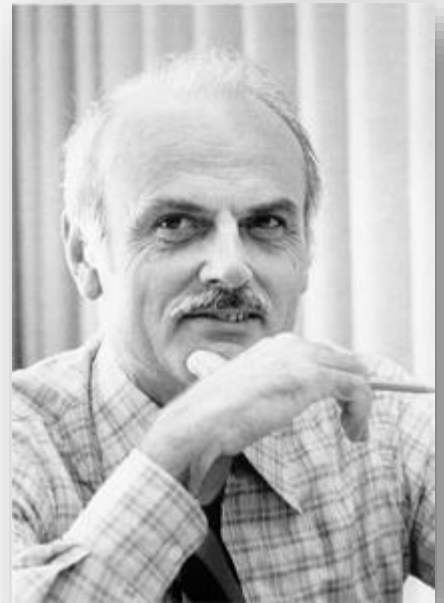
- ✓ Anomalie de **inserare**
- ✓ Anomalie de **ștergere**
- ✓ Anomalie de **actualizare**

Cum determinăm  
dacă o structură este  
“*bună*” sau “*defectuoasă*”?

Cum transformăm  
o structură *defectuoasă*  
într-una *bună*?

Teoria *dependențelor funcționale* furnizează o abordare sistematică a celor două întrebări

Introdusă de  
Edgar Frank Codd în:



*“A relational model for large shared data banks”,*  
Com. of the ACM, 13(6), 1970, pp.377-387.



# Dependențe funcționale

$$\alpha \rightarrow \beta$$

$\alpha, \beta$  sunt submulțimi de attribute ale R

“ $\alpha$  determină funcțional  $\beta$ ”

sau

“ $\beta$  depinde functional de  $\alpha$ ”

# Definiție dependențe funcționale

Dependența funcțională  $\alpha \rightarrow \beta$  este satisfăcută de R dacă și numai dacă

pentru *orice* instanță a lui R,

oricare două tupluri  $t_1$  și  $t_2$  pentru care  
valorile lui  $\alpha$  sunt identice

vor avea de asemenea valori identice pentru  $\beta$ .

O dependență funcțională

$$\alpha \rightarrow \beta$$

este **trivială** dacă

$$\alpha \supseteq \beta.$$

<i>Title</i>	<i>Director</i>	<i>Cinema</i>	<i>Phone</i>	<i>Time</i>
The Hobbit	Jackson	Florin Piersic	441111	11:30
The Lord of the Rings 3	Jackson	Florin Piersic	441111	14:30
Adventures of Tintin	Spielberg	Victoria	442222	11:30
The Lord of the Rings 3	Jackson	Victoria	442222	14:00
War Horse	Spielberg	Victoria	442222	16:30

Dependențe funcționale pentru relația *MovieList*:

1. Title → Director
2. Cinema → Phone
3. Cinema, Time → Title

Fie  $r$  instanța unei relații  $R$

- Spunem că  $r$  **satisface DF**  $\alpha \rightarrow \beta$  dacă pentru orice pereche de tupluri  $t_1$  și  $t_2$  din  $r$  astfel încât  $\pi_\alpha(t_1) = \pi_\alpha(t_2)$ , este de asemenea adevărat că  $\pi_\beta(t_1) = \pi_\beta(t_2)$ .

sau

$$\forall t_1, t_2 \in r$$

$$\pi_\alpha(t_1) = \pi_\alpha(t_2) \Rightarrow \pi_\beta(t_1) = \pi_\beta(t_2) *$$

\*  $\pi_\alpha(t)$  este proiecția atributelor  $\alpha$  pentru tuplul  $t$

Fie  $r$  instanța unei relații  $R$

- o DF  $f$  este satisfăcută pe  $R$  dacă și numai dacă orice instanță  $r$  a lui  $R$  satisface  $f$
- $r$  nu respectă o DF  $f$  dacă  $r$  nu satisface  $f$ .
- $r$  este o instanță legală a lui  $R$  dacă  $r$  satisface toate dependențele funcționale definite pentru  $R$ .

## Exemplu: *Movie*(*Title*, *Director*, *Composer*)

<i>Title</i>	<i>Director</i>	<i>Composer</i>
Schindler's List	Spielberg	Williams
Saving Private Ryan	Spielberg	Williams
North by Northwest	Hitchcock	Herrmann
Angela's Ashes	Parker	Williams
Vertigo	Hitchcock	Herrmann

- DF *composer*  $\rightarrow$  *director* nu este respectată de relația *Movie*
- *r* satisface DF *director*  $\rightarrow$  *composer*

Acest lucru nu înseamnă că *director* $\rightarrow$ *composer* e respectat de *Movie*!

# Problema implicației

Putem deduce că o DF  $f$  e respectată de  $R$  pe baza unei mulțimi de DF  $F$  ?

**Exemplu:** în *MovieList*, avem

$$F = \{ \begin{array}{l} \text{Title} \rightarrow \text{Director} \\ \text{Cinema} \rightarrow \text{Phone} \\ \text{Cinema, Time} \rightarrow \text{Title} \end{array} \}$$

- $\text{Time} \rightarrow \text{Director}$  este respectată?
- Dar  $\text{Cinema, Time} \rightarrow \text{Director}$ ?



$F$  implică logic pe  $f$

notat prin

$$F \Rightarrow f$$

daca fiecare instanță  $r$  a relației  $R$   
ce satisface  $F$   
satisfice și  $f$

$F$  &  $G$  : mulțimi de dependențe funcționale  
 $f$  : dependența funcțională

$F$  implică logic  $G$

notat prin

$$\mathbf{F \Rightarrow G}$$

dacă  $F \Rightarrow g$  pentru fiecare  $g \in G$

## Închiderea lui $F$

(notată prin  $F^+$ )

este mulțimea tuturor DF implicate de  $F$

$$F^+ = \{f \mid F \Rightarrow f\}$$

$F$  și  $G$  sunt **echivalente**

(notat prin  $F \equiv G$ )

dacă

$$F^+ = G^+$$

(adică  $F \Rightarrow G$  și  $G \Rightarrow F$ )

# Axiomele lui Armstrong

Fie  $\alpha, \beta, \gamma \subseteq R$

**Reflexivitate:** Dacă  $\beta \subseteq \alpha$ , atunci  $\alpha \rightarrow \beta$

**Augmentare:** Dacă  $\alpha \rightarrow \beta$ , atunci  $\alpha\gamma \rightarrow \beta\gamma$

**Tranzitivitate:** Dacă  $\alpha \rightarrow \beta$  și  $\beta \rightarrow \gamma$ , atunci  $\alpha \rightarrow \gamma$

Sistemul axiomelor lui Armstrong  
este

**Corect**

(Orice FD derivată este implicată de F)

&

**Complet**

(Toate DF din  $F^+$  pot fi derivate)

*Exemplu:* Fie  $R(A, B, C, D, E)$  cu mulțimea

$$F = \{A \rightarrow C; B \rightarrow C; CD \rightarrow E\}.$$

Arătați că  $F \Rightarrow AD \rightarrow E$

*Soluție:*

1.  $A \rightarrow C$  (dat)
2.  $AD \rightarrow CD$  (augmentare cu (1))
3.  $CD \rightarrow E$  (dat)
4.  $AD \rightarrow E$  (tranzitivitate cu (2) și (3))

# Reguli de inferență adiționale

## Reuniunea:

Dacă  $\alpha \rightarrow \beta$  și  $\alpha \rightarrow \gamma$ , atunci  $\alpha \rightarrow \beta\gamma$

## Descompunerea:

Dacă  $\alpha \rightarrow \beta$ , atunci  $\alpha \rightarrow \beta'$  pentru orice  $\beta' \subseteq \beta$



*Exemplu:* Aratati ca  $\{A \rightarrow BCD\} \equiv \{A \rightarrow B; A \rightarrow C; A \rightarrow D\}$

Fie  $F = \{A \rightarrow BCD\}$

Fie  $G = \{A \rightarrow B; A \rightarrow C; A \rightarrow D\}$

Prin regula de descompunere avem

$$F \Rightarrow A \rightarrow B,$$

$$F \Rightarrow A \rightarrow C, \text{ si}$$

$$F \Rightarrow A \rightarrow D$$

Prin urmare  $F \Rightarrow G$

Din regula reuniunii avem

$$\{A \rightarrow B; A \rightarrow C\} \Rightarrow A \rightarrow BC \text{ si}$$

$$\{A \rightarrow BC; A \rightarrow D\} \Rightarrow A \rightarrow BCD$$

Prin urmare  $G \Rightarrow F$ , deci  $F \equiv G$

# Superchei, chei & attribute prime

- O mulțime de attribute  $\alpha$  reprezintă o **supercheie** a relației  $R$  (având mulțimea de DF  $F$ ) dacă

$$F \Rightarrow \alpha \rightarrow R.$$

- O mulțime de attribute  $\alpha$  e o **cheie** a relației  $R$  dacă
  - (1)  $\alpha$  este o supercheie, și
  - (2) nici o submulțime a lui  $\alpha$  nu e supercheie (adică, pentru fiecare  $\beta \subset \alpha$ ,  $\beta \rightarrow R \notin F^+$ )
- Un atribut  $A \in R$  se numește atribut **prim** dacă  $A$  face parte dintr-o cheie a lui  $R$ ; în caz contrar,  $A$  se numește atribut **neprim**.

- Considerăm din nou relația

*MovieList* (Title, Director, Cinema, Phone, Time)

cu DF

- (1) Cinema, Time  $\rightarrow$  Title
- (2) Cinema  $\rightarrow$  Phone
- (3) Title  $\rightarrow$  Director

- {Cinema, Time} este singura **cheie** a relației *MovieList*.
- Cinema și Time sunt singurele attribute **prime** din *MovieList*.
- Orice mulțime ce include {Cinema; Time} e **supercheie**  
a *MovieList*.

# Închiderea atributelor

Fie  $\alpha \subseteq R$  și  $F$  o mulțime de DF satisfăcute pe  $R$

■ **Închiderea lui  $\alpha$**  (cu respectarea mulțimii  $F$  de DF), notată cu  $\alpha^+$ , este mulțimea de attribute ce sunt determinate funcțional din  $\alpha$  pe baza dependențelor funcționale din  $F$ ; adică

$$\alpha^+ = \{A \in R \mid F \Rightarrow \alpha \rightarrow A\}$$

■ Se observă că  $F \Rightarrow \alpha \rightarrow \beta$  dacă și numai dacă  $\beta \subseteq \alpha^+$  (cu respectarea DF din  $F$ )

# Algorithm pt determinarea închiderii atributelor

**Input:**  $\alpha, F$

**Output:**  $\alpha^+$  (w.r.t.  $F$ )

Compute a sequence of sets of attrs  $\alpha_0, \alpha_1, \dots, \alpha_k, \alpha_{k+1}$  as follows:

$$\alpha_0 = \alpha$$

$$\alpha_{i+1} = \alpha_i \cup \gamma \text{ such that there is some FD } \beta \rightarrow \gamma \in F \text{ and } \beta \subseteq \alpha_i$$

Terminate the computation once

$$\alpha_{k+1} = \alpha_k \text{ for some } k$$

Return  $\alpha_k$

**Input:**  $\alpha, F$

**Output:**  $\alpha^+$  (w.r.t.  $F$ )

Compute a sequence of sets of attrs  $\alpha_0, \alpha_1, \dots, \alpha_k, \alpha_{k+1}$  as follows:

$$\alpha_0 = \alpha$$

$$\alpha_{i+1} = \alpha_i \cup \gamma \text{ such that there is some FD } \beta \rightarrow \gamma \in F \text{ and } \beta \subseteq \alpha_i$$

Terminate the computation once  $\alpha_{k+1} = \alpha_k$  for some  $k$

Return  $\alpha_k$

Exemple: Fie  $F = \{A \rightarrow C; B \rightarrow C; CD \rightarrow E\}$ , aratati ca  $F \Rightarrow AD \rightarrow E$

<i>i</i>	$\alpha_i$	<i>FD folosit</i>
0	AD	dat
1	ACD	$A \rightarrow C$
2	ACDE	$CD \rightarrow E$
3	ACDE	-

Deci  $AD^+ = ACDE$ . Deoarece  $E \in AD^+$ , rezulta ca  $F \Rightarrow AD \rightarrow E$

# Descompunerea relațiilor

**Descompunerea** unei relații  $R$   
este o mulțime de (sub)relații

$$\{R_1, R_2, \dots, R_n\}$$

astfel încât fiecare  $R_i \subseteq R$  și  $R = \cup R_i$

Dacă  $r$  este o instanță din  $R$ ,  
atunci  $r$  se descompune în

$$\{r_1, r_2, \dots, r_n\},$$

unde fiecare  $r_i = \pi_{R_i}(r)$

# Descompunerea relațiilor

$$\begin{aligned} &\{ M_1 = (\text{Cinema}, \text{Time}) \\ &\quad M_2 = (\text{Time}, \text{Title}), \\ &\quad M_3 = (\text{Title}, \text{Director}), \\ &\quad M_4 = (\text{Cinema}, \text{Phone}) \} \end{aligned}$$

e o descompunere a:

*MovieList*(Title, Director, Cinema, Phone, Time)



# Proprietățile descompunerii relațiilor

## 1. Descompunerea trebuie să **păstreze informațiile**

- Datele din relația originală  $\equiv$  Datele din relațiile descompunerii
- Crucial pentru păstrarea consistenței datelor!

## 2. Descompunerea trebuie să **respecte toate DF**

- Dependențele funcționale din relația originală  $\equiv$  reuniunea dependențelor funcționale din relațiile descompunerii
- Facilitează verificarea violărilor DF

# 1. Descompunerea trebuie să păstreze informațiile

Cu alte cuvinte:  
putem reconstrui  $r$   
prin jonctiunea proiecțiilor sale  
 $\{r_1, r_2, \dots, r_n\}$

Observatie: daca  $\{R_1, R_2, \dots, R_n\}$  e o descompunere a  $R$ ,  
atunci pentru orice instanta  $r$  din  $R$ , avem

$$r \subseteq \pi_{R_1}(r) \otimes \pi_{R_2}(r) \otimes \dots \otimes \pi_{R_n}(r)$$

# *MovieList*(Title, Director, Cinema, Phone, Time)

***M1***

<i>Cinema</i>	<i>Time</i>
Florin Piersic	11:30
Florin Piersic	14:30
Victoria	11:30
Victoria	14:00
Victoria	16:30

***M2***

<i>Time</i>	<i>Title</i>
11:30	The Hobbit
14:30	The Lord of the Rings 3
11:30	Adventures of Tintin
14:00	The Lord of the Rings 3
16:30	War Horse

***M3***

<i>Title</i>	<i>Director</i>
The Hobbit	Jackson
The Lord of the Rings 3	Jackson
Adventures of Tintin	Spielberg
War Horse	Spielberg

***M4***

<i>Cinema</i>	<i>Phone</i>
Florin Piersic	441111
Victoria	442222

# $M1 \otimes M2 \otimes M3 \otimes M4$

<i>Title</i>	<i>Director</i>	<i>Cinema</i>	<i>Phone</i>	<i>Time</i>
The Hobbit	Jackson	Florin Piersic	441111	11:30
The Hobbit	Jackson	Victoria	442222	11:30
The Lord of the Rings 3	Jackson	Florin Piersic	441111	14:30
Adventures of Tintin	Spielberg	Florin Piersic	441111	11:30
Adventures of Tintin	Spielberg	Victoria	442222	11:30
The Lord of the Rings 3	Jackson	Victoria	442222	14:00
War Horse	Spielberg	Victoria	442222	16:30

# Descompunere cu joncțiune fără pierderi (Lossless - Join Decomposition)

O descompunere a  $R$  (având DF  $F$ ) în  
 $\{R_1, R_2, \dots, R_n\}$

este o

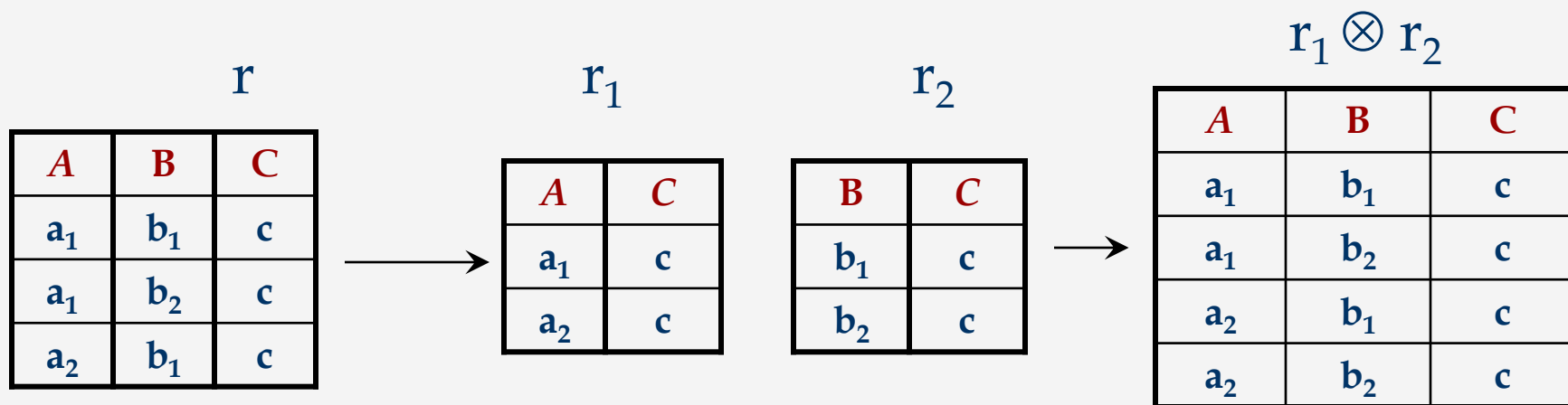
descompunere cu joncțiuni fără pierderi  
cu respectarea mulțimii  $F$

dacă

$$\pi_{R_1}(r) \otimes \pi_{R_2}(r) \otimes \dots \otimes \pi_{R_n}(r) = r$$

pentru orice instanță  $r$  din  $R$  ce satisface  $F$ .

Fie descompunere lui  $R(A,B,C)$   
in  $\{R_1(AC), R_2(BC)\}$



- Deoarece  $r \subset r_1 \otimes r_2$ , descompunerea **nu**  
este cu joncțiuni fără pierderi  
(*lossy decomposition*)

## Întrebarea 1

*Cum determinăm dacă  $\{R_1, R_2\}$  este o descompunere cu joncțiuni fără pierderi a lui  $R$ ?*

## Întrebarea 2

*Cum descompunem  $R$  în  $\{R_1, R_2\}$  astfel încât aceasta e cu joncțiuni fără pierderi?*

# Întrebarea 1

*Cum determinăm dacă  $\{R_1, R_2\}$  este o descompunere cu joncțiuni fără pierderi a lui  $R$ ?*

**Teorema:** Descompunerea lui  $R$  (cu mulțimea  $F$  de DF) în  $\{R_1, R_2\}$  este cu joncțiuni fără pierderi cu respectarea mulțimii  $F$  dacă și numai dacă :

$$F \Rightarrow R_1 \cap R_2 \rightarrow R_1$$

sau

$$F \Rightarrow R_1 \cap R_2 \rightarrow R_2$$



## Întrebarea 2

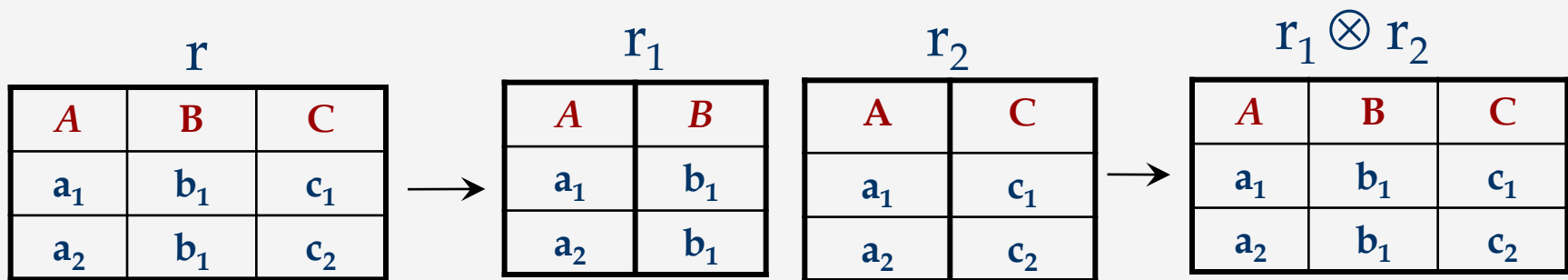
*Cum descompunem  $R$  în  $\{R_1, R_2\}$  astfel încât aceasta e cu joncțiuni fără pierderi?*

**Corolar:** Dacă  $\alpha \rightarrow \beta$  este satisfăcută pe  $R$  și  $\alpha \cap \beta = \emptyset$ , atunci descompunerea lui  $R$  în  $\{R - \beta, \alpha\beta\}$  este o descompunere cu joncțiuni fără pierderi.

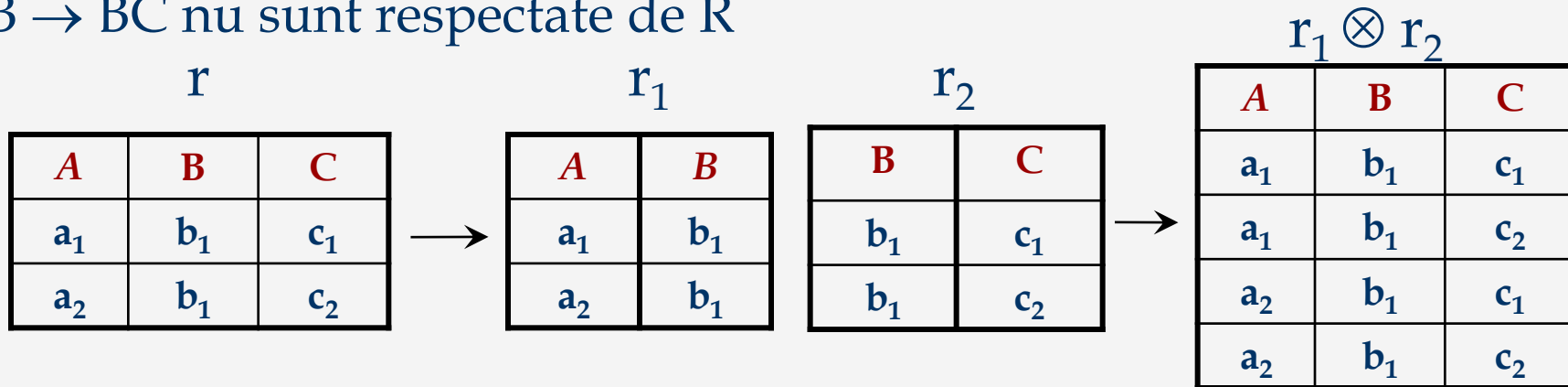
# Exemplu

- Fie  $R(A,B,C)$  cu dependențele funcționale  $F = \{ A \rightarrow B \}$
- Descompunerea  $\{AB, AC\}$  e cu joncțiuni fara pierderi deoarece

$$AB \cap AC = A \quad \text{și} \quad A \rightarrow AB$$



- Descompunere  $\{AB, BC\}$  nu e cu joncțiuni fara pierderi deoarece  $AB \cap BC = B$  nici una din urmatoarele dependențe:  $B \rightarrow AB$  sau  $B \rightarrow BC$  nu sunt respectate de  $R$



## Teorema

**Dacă**

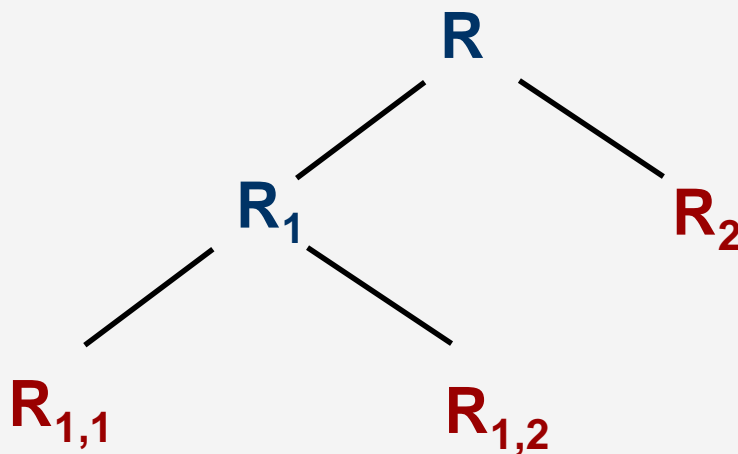
$\{R_1, R_2\}$  este o descompunere cu joncțiuni fără pierderi a lui  $R$ ,

**și dacă**

$\{R_{1,1}, R_{1,2}\}$  e o descompunere cu joncțiuni fără pierderi a lui  $R_1$ ,

**atunci**

$\{R_{1,1}, R_{1,2}, R_2\}$  e o descompunere cu joncțiuni fără pierderi a  $R$ :



## *MovieList*

<i>Title</i>	<i>Director</i>	<i>Cinema</i>	<i>Phone</i>	<i>Time</i>
The Hobbit	Jackson	Florin Piersic	441111	11:30
The Lord of the Rings 3	Jackson	Florin Piersic	441111	14:30
Adventures of Tintin	Spielberg	Victoria	442222	11:30
War Horse	Spielberg	Victoria	442222	14:00
The Lord of the Rings 3	Jackson	Victoria	442222	16:30

## *Cinema-Screens*

### *Movie*

<i>Title</i>	<i>Director</i>
The Hobbit	Jackson
The Lord of the Rings 3	Jackson
Adventures of Tintin	Spielberg
War Horse	Spielberg

<i>Cinema</i>	<i>Phone</i>	<i>Time</i>	<i>Title</i>
F. Piersic	441111	11:30	The Hobbit
F. Piersic	441111	14:30	The Lord of the Rings 3
Victoria	442222	11:30	Adventures of Tintin
Victoria	442222	14:00	War Horse
Victoria	442222	16:30	The Lord of the Rings 3

*Movie*

<i>Title</i>	<i>Director</i>
The Hobbit	Jackson
The Lord of the Rings 3	Jackson
Adventures of Tintin	Spielberg
War Horse	Spielberg

*Cinema-Screens*

<i>Cinema</i>	<i>Phone</i>	<i>Time</i>	<i>Title</i>
F. Piersic	441111	11:30	The Hobbit
F. Piersic	441111	14:30	The Lord of the Rings 3
Victoria	442222	11:30	Adventures of Tintin
Victoria	442222	14:00	War Horse
Victoria	442222	16:30	The Lord of the Rings 3

*Cinema*

<i>Cinema</i>	<i>Phone</i>
F. Piersic	441111
Victoria	442222

*Screens*

<i>Cinema</i>	<i>Time</i>	<i>Title</i>
F. Piersic	11:30	The Hobbit
F. Piersic	14:30	Saving Private Ryan
Victoria	11:30	Adventures of Tintin
Victoria	14:00	War Horse
Victoria	16:30	Saving Private Ryan

# Proiecția dependențelor funcționale

- Proiecția mulțimii  $F$  pe  $\alpha$  (notată prin  $F_\alpha$ ) este mulțimea acelor dependențe din  $F^+$  care implică doar attribute din  $\alpha$ , adică:

$$F_\alpha = \{ \beta \rightarrow \gamma \in F^+ \mid \beta\gamma \subseteq \alpha \}$$

- Algoritm pentru determinare proiecției DF:

**Input:**  $\alpha, F$

**Output:**  $F_\alpha$

result =  $\emptyset$ ;

for each  $\beta \subseteq \alpha$  do

$T = \beta^+$  (w.r.t.  $F$ )

    result = result  $\cup \{ \beta \rightarrow T \cap \alpha \}$

return result

Complexitatea  
e exponențială

# Descompunere cu păstrarea dependențelor

Descompunerea  $\{R_1, R_2, \dots, R_n\}$  a relației  $R$  e cu **păstrarea dependențelor** dacă  $(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})$  și  $F$  sunt echivalente, adică:

$$(F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n}) \Rightarrow F \text{ și}$$
$$F \Rightarrow (F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})$$