# Relational Design Theory

Carla Teixeira Lopes

Bases de Dados Mestrado Integrado em Engenharia Informática e Computação, FEUP

Based on Jennifer Widom and Christopher Ré slides

# Does BCNF guarantee a good decomposition?

Remove anomalies?

Yes

Can logically reconstruct original relation?

$$R_1 \bowtie R_2 = R$$
?

Too few or too many tuples?

R		
Α	В	O
1	2	3
4	2	5

$R_1$						
Α	В					
1	2					
4	2					

$R_2$							
В	C						
2	3						
2	5						

$$R_1 \bowtie R_2 =$$
123
125
423
425

## Does BCNF guarantee a good decomposition?

R		
A	В	С
1	2	3
4	2	5

$R_1$	
Α	В
1	2
4	2

$H_2$							
В	O						
2	3						
2	5						

$$R_1 \bowtie R_2 = 123$$
125
423
425

What happened?

Not a BCNF decomposition R1 and R2 would demand B  $\rightarrow$  A or B  $\rightarrow$  C and none hold

BCNF always lossless

BCNF decomposition is standard practice - very powerful & widely used!

DFs A->B; A->C; CD->A

Does this decomposition ensure lossless joins?

S (A, B, C, D) decomposed in S1 (A, D), S2 (A,C) and S3 (B, C, D) A->B; A->C; CD->A

#### Build the tableau

One line per decomposed relation A letter per each attribute in the decomposed relation Subscript the letter with i, if the attribute is not in  $S_i$ 

АВ	СС	)	Α	В	С	D	Α	В	С	D	
			a			d	a	b <sub>1</sub>	C <sub>1</sub>	d	S1 (A, D)
			a		С		a	$b_2$	С	$d_2$	S2 (A, C)
				b	С	d	<b>a</b> <sub>3</sub>	b	С	d	S3 (B, C, D)

S (A, B, C, D) decomposed in S1 (A, D), S2 (A,C) and S3 (B, C, D) A->B; B->C; CD->A

Α	В	С	D
a	b <sub>1</sub>	C <sub>1</sub>	d
a	b <sub>2</sub>	С	$d_2$
$a_3$	b	С	d

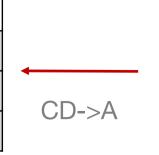
#### **Iterations**

A->B tells us that the first two rows must agree in the B attribute, that is,  $b_1=b_2$  From B->C, we know that  $c_1=c$  From CD->A, we know that  $a=a_3$ 

Α	В	C	D
a	b <sub>1</sub>	C <sub>1</sub>	d
a	b <sub>1</sub>	С	$d_2$
$a_3$	b	С	d

B->C

Α	В	С	D
a	b <sub>1</sub>	C	d
a	b <sub>1</sub>	С	$d_2$
а	b	С	d



A	В	C	D
a	b <sub>1</sub>	С	d
a	b <sub>1</sub>	С	$d_2$
$a_3$	Ь	С	d

S (A, B, C, D) decomposed in S1 (A, D), S2 (A,C) and S3 (B, C, D) A->B; B->C; CD->A

#### Conclusion

If the final table has a line without subscripts, it is a lossless join decomposition

А	В	С	D
a	b <sub>1</sub>	C	d
a	b <sub>1</sub>	C	$d_2$
a	b	O	d

## Consider the following relation and FDs

Movie (title, year, studioName, president, presAddr)

title, year -> studioName

studioName -> president

president -> presAddr

## Test if the following decomposition is lossless:

S1 (studioName, president)

S3 (studioName, presAddr)

S4 (studioName, title, year)

Movie (title, year, studioName, president, presAddr) decomposed in S1 (studioName, president); S3 (studioName, presAddr); S4 (studioName, title, year)

title, year -> studioName studioName -> president president -> presAddr

#### Build the tableau

A (title)	B (year)	C (studioName)	D (president)	E (presAddr)
a <sub>1</sub>	b <sub>1</sub>	С	d	e <sub>1</sub>
$a_2$	b <sub>2</sub>	С	$d_2$	е
а	b	С	$d_3$	$e_3$

**S1** 

**S3** 

**S4** 

Movie (title, year, studioName, president, presAddr) decomposed in

S1 (studioName, president); S3 (studioName, presAddr); S4 (studioName, title, year) title, year -> studioName; studioName -> president; president -> presAddr

studioName->president

title

a<sub>1</sub>

 $a_2$ 

a

vear

b

ba

b

title	year	studio Name	presi dent	pres Addr
$a_1$	b <sub>1</sub>	С	d	e <sub>1</sub>
$a_2$	b <sub>2</sub>	С	d	е
а	b	С	d	$e_3$

studio

Name

C

C

presi

dent

d

da

 $d_3$ 

pres

Addr

e<sub>1</sub>

е

 $e_3$ 

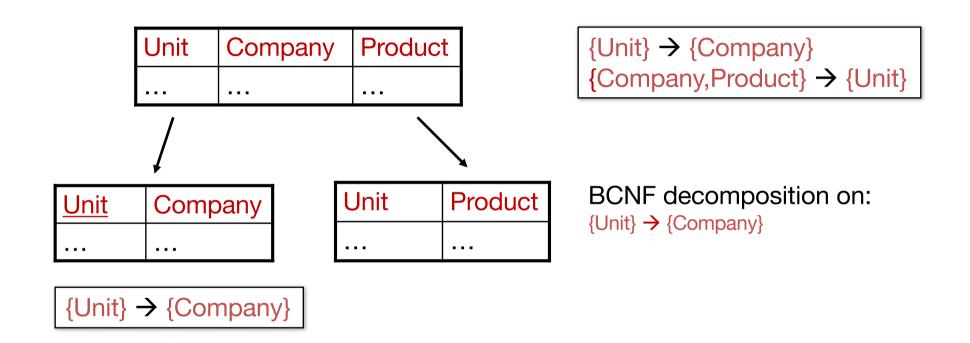
title studio presi year pres Name dent Addr b<sub>1</sub> d е a<sub>1</sub> C lossless join decomposition  $b_2$ d C е a b d a C e

president->presAddr

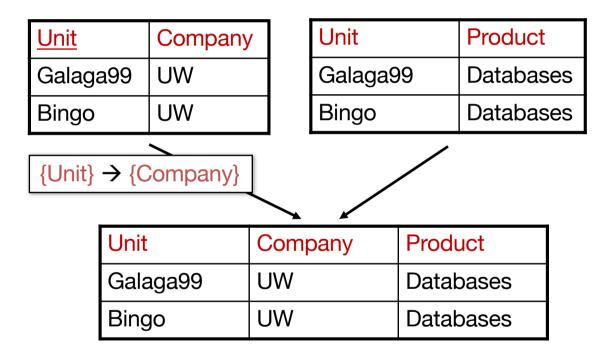
```
Apply(SSN, cName, date, major)
    Can apply to each college once and for one major
    Colleges have non-overlapping application dates
FDs: SSN, cName -> date, major; date ->cName
Keys: {SSN, cName}
BCNF?
    No.
    A1 (date, cName)
    A2 (date, SSN, major)
Good design? Not necessarily.
    Student's application separated from the college
    Checking the first DF would require a join
```

We might just prefer to keep everything together

Apply is in 3NF



We lose the FD {Company, Product} -> {Unit}



No problem so far. All *local* FDs are satisfied.

Let's put all the data back into a single table again

Violates the FD {Company, Product} -> {Unit}

College (cName, state)
CollegeSize (cName, enrollment)
CollegeScores (cName, avgSAT)
CollegeGrades (cName, avgGPA)

#### Too decomposed

We could capture all of the information in one relation or a couple and still be in BCNF

## **BCNF** shortcomings

Dependency preservation is not guaranteed

No guarantee that all original dependencies can be checked on decomposed relations

This may require joins of those relations in order to check them

Various ways to handle so that decompositions are all lossless / no FDs lost For example 3NF

Usually a tradeoff between redundancy / data anomalies and FD preservation

BCNF still most common
With additional steps to keep track of lost FDs

## 3NF Decomposition Algorithm

Input: relation R + set F of FDs for R

Output: decomposition of R into 3NF relations with "lossless join" and "dependency preservation"

#### 1. Find a minimal basis for F, say G

Right sides with only 1 attribute

No redundant FDs

For each DF  $\bar{X} \to \bar{A}$ , compute  $\bar{X}$  using the other DFs. If  $A \subseteq \bar{X}$ , the DF  $\bar{X} \to \bar{A}$  is redundant

No redundant attributes on the left sides

Remove 1 attribute from left side and compute closure of the remaining attributes with the **original** DFs. If closure includes the right side, the attribute can be removed

## 3NF Decomposition Algorithm

Input: relation R + set F of FDs for R

Output: decomposition of R into 3NF relations with "lossless join" and "dependency preservation"

- 2. For each DF  $\bar{X} \to \bar{A}$  in G, create a relation R'  $(\bar{X}, \bar{A})$  Previously, merge DFs with equal left sides
- 3. If none of the relations of step 2 is a superkey for R, add another relation for a key for R

## 3NF Decomposition Example

R (A, B, C, D, E); AB->C, C->B and A->D Minimal base

1. Find a minimal basis for DFs

Right sides with only 1 attribute?

No redundant DFs?

 $\{A, B\}^+=\{A, B, D\}$   $\longrightarrow$  It does not contain C thus the DF is essential

 $\{C\}^+ = \{C\}$  It does not contain B thus the DF is essential

 $\{A\}^+ = \{A\}$  It does not contain D thus the DF is essential

No redundant attributes on left side?

On AB-> C, remove A, getting B->C. {B}+={B}. Since it does not contain C, the attribute A is essential

On AB->C, remove B, getting A->C. {A}+={AD}. Since it does not contain C, the attribute B is essential

## 3NF Decomposition Example

2. For each DF  $\bar{X} \to \bar{A}$  in G, create a relation R' (X, A)

 $R_1$  (A, B, C)

 $R_2$  (C, B)

 $R_3$  (A, D)

3. If none of the relations of step 2 is a superkey for R, add another relation for a key for R

Keys: {A, B, E}, {A, C, E}

R<sub>4</sub> should be one of them

## Consider the following relation and FDs

Movie (title, year, studioName, president, presAddr) title, year -> studioName studioName -> president president -> presAddr

Decompose into 3NF relations.

Any advantages over the BCNF decomposition?

## BCNF and 3NF decomposition

#### **BCNF** decomposition

Assures lossless joins
Dependency preservation is not always possible

#### 3NF decomposition

Assures lossless joins and dependency preservation

## Summary

## Designing a database schema

Usually many designs possible Some are (much) better than others! How do we choose?

## Very nice theory for relational database design

Normal forms - "good" relations

Design by decomposition

Usually intuitive and works well

Some shortcomings

Dependency enforcement

Query workload

Over-decomposition

## Kahoot time!

Any doubts?

## Readings

# Jeffrey Ullman, Jennifer Widom, A first course in Database Systems 3<sup>rd</sup> Edition

Section 3.1 – Functional Dependencies

Section 3.2 – Rules About Functional Dependencies

Section 3.3 – Design of Relational Database Schemas

Section 3.4 – Decomposition: The Good, Bad, and Ugly

Section 3.5 – Third Normal Form