

# CSC 261/461 – Database Systems

## Lecture 12

Fall 2017

# Today's Lecture

1. 3NF vs Boyce-Codd Normal Form (Already covered)
2. Decompositions

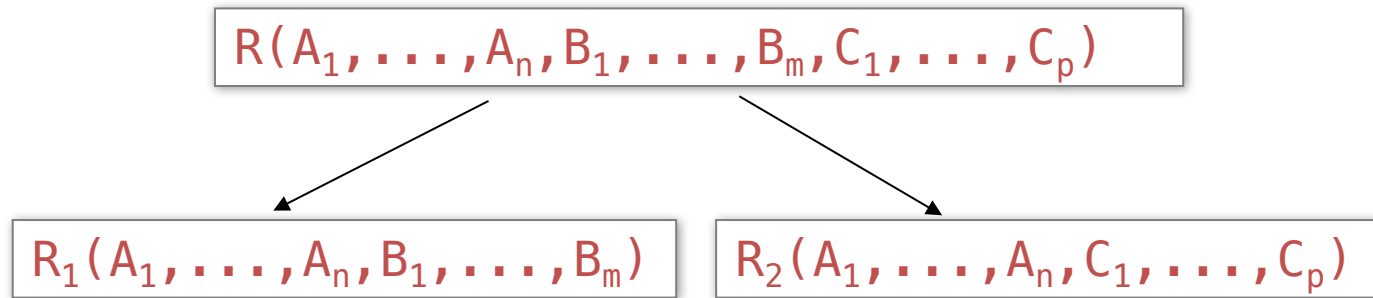
## **2. DECOMPOSITIONS**

# Recap: Decompose to remove redundancies

1. We saw that **redundancies** in the data (“bad FDs”) can lead to data anomalies
2. We developed mechanisms to **detect and remove redundancies by decomposing tables into BCNF**
  1. BCNF decomposition is *standard practice*- very powerful & widely used!
3. However, sometimes decompositions can lead to **more subtle unwanted effects...**

When does this happen?

# Decompositions in General



$R_1$  = the *projection* of  $R$  on  $A_1, \dots, A_n, B_1, \dots, B_m$


$R_2$  = the *projection* of  $R$  on  $A_1, \dots, A_n, C_1, \dots, C_p$

# Theory of Decomposition


Name	Price	Category
Gizmo	19.99	Gadget
OneClick	24.99	Camera
Gizmo	19.99	Camera

Sometimes a decomposition is “correct”

I.e. it is a **Lossless decomposition**



Name	Price
Gizmo	19.99
OneClick	24.99
<del>Gizmo</del>	<del>19.99</del>



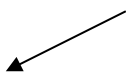
Name	Category
Gizmo	Gadget
OneClick	Camera
Gizmo	Camera

# Lossy Decomposition


Name	Price	Category
Gizmo	19.99	Gadget
OneClick	24.99	Camera
Gizmo	19.99	Camera

*However  
sometimes it isn't*

What's wrong  
here?

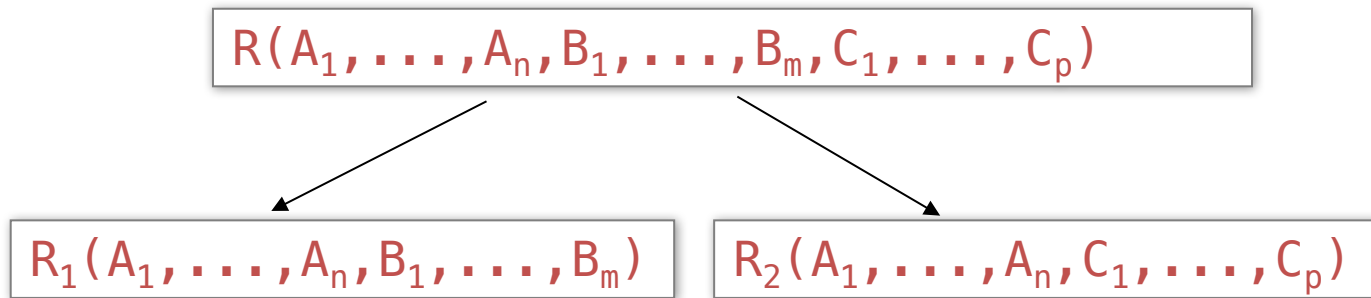


Name	Category
Gizmo	Gadget
OneClick	Camera
Gizmo	Camera



Price	Category
19.99	Gadget
24.99	Camera
19.99	Camera

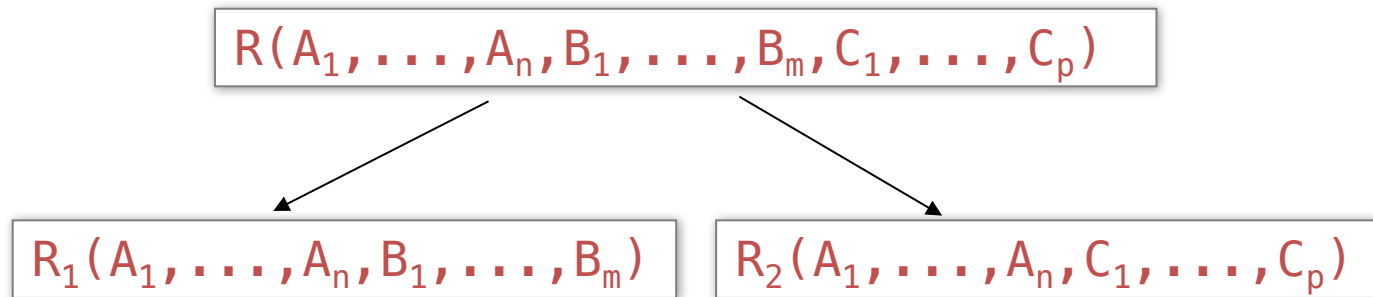
# Lossless Decompositions



A decomposition  $R$  to  $(R_1, R_2)$  is **lossless** if  $R = R_1 \text{ Join } R_2$



# Lossless Decompositions



If  $\{A_1, \dots, A_n\} \rightarrow \{B_1, \dots, B_m\}$   
Then the decomposition is lossless

Note: don't need  
 $\{A_1, \dots, A_n\} \rightarrow \{C_1, \dots, C_p\}$

BCNF decomposition is always lossless. Why?

# A relation TEACH that is in 3NF but not in BCNF

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

- Two FDs exist in the relation TEACH:

X

→ A

- {student, course} → instructor
- instructor → course

- {student, course} is a candidate key for this relation
- So this relation is in 3NF *but not in* BCNF
- A relation **NOT** in BCNF should be decomposed

# Achieving the BCNF by Decomposition (2)

- Three possible decompositions for relation TEACH
  - D<sub>1</sub>: {student, instructor} and {student, course}
  - D<sub>2</sub>: {course, instructor} and {course, student}
  - ✓ D<sub>3</sub>: {instructor, course} and {instructor, student}

## A problem with BCNF

Problem: To enforce a FD, must reconstruct original relation—*on each insert!*

# A Problem with BCNF

Unit	Company	Product
...	...	...

$\{\text{Unit}\} \rightarrow \{\text{Company}\}$   
 $\{\text{Company}, \text{Product}\} \rightarrow \{\text{Unit}\}$

↙ ↘

<u>Unit</u>	Company
...	...

Unit	Product
...	...

$\{\text{Unit}\} \rightarrow \{\text{Company}\}$

We do a BCNF decomposition  
on a “bad” FD:  
 $\{\text{Unit}\}^+ = \{\text{Unit}, \text{Company}\}$

We lose the FD  $\{\text{Company}, \text{Product}\} \rightarrow \{\text{Unit}\}!!$

## So Why is that a Problem?

<u>Unit</u>	Company
Galaga99	UW
Bingo	UW

Unit	Product
Galaga99	Databases
Bingo	Databases

No problem so far.  
All *local* FD's are satisfied.

$\{\text{Unit}\} \rightarrow \{\text{Company}\}$

Unit	Company	Product
Galaga99	UW	Databases
Bingo	UW	Databases

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

Violates the FD  $\{\text{Company, Product}\} \rightarrow \{\text{Unit}\}!!$

# The Problem

- We started with a table  $R$  and FDs  $F$
- We decomposed  $R$  into BCNF tables  $R_1, R_2, \dots$  with their own FDs  $F_1, F_2, \dots$
- We insert some tuples into each of the relations—which satisfy their local FDs but when reconstruct it violates some FD across tables!

Practical Problem: To enforce FD, must reconstruct  $R$ —*on each insert!*

# Possible Solutions

- Various ways to handle so that decompositions are all lossless / no FDs lost
  - For example 3NF- stop short of full BCNF decompositions.
- Usually a tradeoff between redundancy / data anomalies and FD preservation...

BCNF still most common- with additional steps to keep track of lost FDs...



# Summary

- Constraints allow one to reason about **redundancy** in the data
- Normal forms describe how to **remove** this redundancy by **decomposing** relations
  - Elegant—by representing data appropriately certain errors are essentially impossible
  - For FDs, BCNF is the normal form.
- A tradeoff for insert performance: 3NF

# Acknowledgement

- Some of the slides in this presentation are taken from the slides provided by the authors.
- Many of these slides are taken from cs145 course offered by Stanford University.
- Thanks to YouTube, especially to [Dr. Daniel Soper](#) for his useful videos.