


# Functional Dependencies and Normalization for Relational Databases

Week 5


COS60009: Data Management for the Big Data Age



1

## Learning Objectives

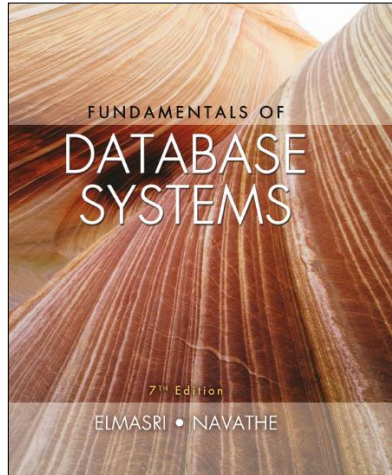
- Informal Design Guidelines for Relational Databases
- Functional Dependencies (FDs)
- Normal Forms Based on Primary Keys
  - Normalization of Relations
  - Practical Use of Normal Forms
  - Definitions of Keys and Attributes Participating in Keys
- First Normal Form
- Second Normal Form
- Third Normal Form
- BCNF (Boyce-Codd Normal Form)



2

# Fundamentals of Database Systems

Seventh Edition



## Chapter 14

Basics of Functional Dependencies and Normalization for Relational Databases



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

3

## Informal Design Guidelines for Relational Databases

- What is relational database design?
  - The grouping of attributes to form “good” relation schemas
- Two levels of relation schemas
  - The logical “user view” level
  - The storage “base relation” level
- Design is concerned mainly with base relations
- What are the criteria for “good” base relations?
- We first discuss informal guidelines for good relational design
- Then we discuss formal concepts of functional dependencies and normal forms
  - 1NF (First Normal Form)
  - 2NF (Second Normal Form)
  - 3NF (Third Normal Form)
  - BCNF (Boyce-Codd Normal Form)



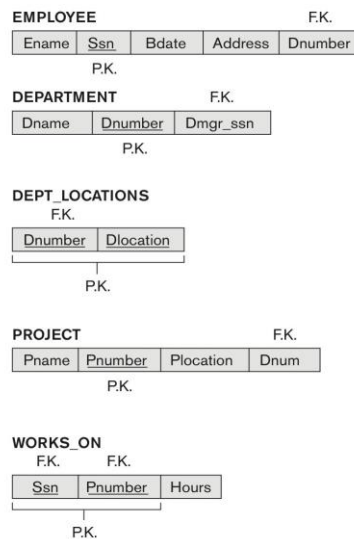
Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

4

## Semantics of the Relational Attributes Must Be Clear

- **Guideline 1:** Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).
  - Attributes of different entities (EMPLOYEEs, DEPARTMENTs, PROJECTs) should not be mixed in the same relation
  - Only foreign keys should be used to refer to other entities
  - Entity and relationship attributes should be kept apart as much as possible.
- **Bottom Line:** Design a schema that can be explained easily relation by relation. The semantics of attributes should be easy to interpret.

## Figure 14.1 A Simplified COMPANY Relational Database Schema



## Redundant Information in Tuples and Update Anomalies

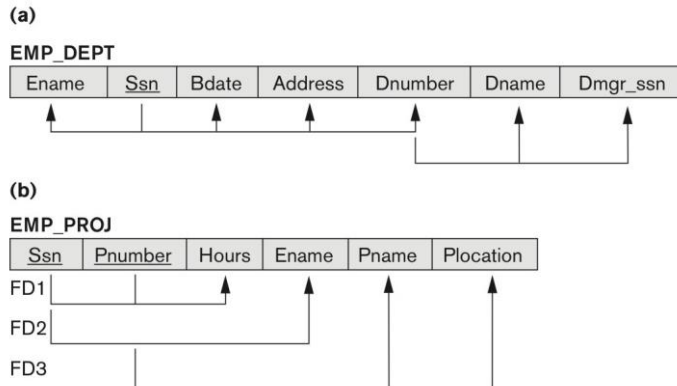
- Information is stored redundantly
  - Wastes storage
  - Causes problems with update anomalies
    - Insertion anomalies
    - Deletion anomalies
    - Modification anomalies

## Example of an Update Anomaly

- Consider the relation:
  - EMP\_PROJ(Emp#, Proj#, Ename, Pname, No\_hours)
- Update Anomaly:
  - Changing the name of project number P1 from “Billing” to “Customer-Accounting” may cause this update to be made for all 100 employees working on project P1.
- Insert Anomaly:
  - Cannot insert a project unless an employee is assigned to it.
  - Conversely, cannot insert an employee unless an he/she is assigned to a project.
- Delete Anomaly:
  - When a project is deleted, it will result in deleting all the employees who work on that project.
  - Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.

## Figure 14.3 Two Relation Schemas Suffering from Update Anomalies

(a) EMP\_DEPT and (b) EMP\_PROJ



## Figure 14.4 Sample States For EMP\_DEPT and EMP\_PROJ

EMP_DEPT							Redundancy	
Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn		
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5	Research	333445555		
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5	Research	333445555		
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321		
Wallace, Jennifer S.	987654321	1941-06-20	291 Berry, Bellaire, TX	4	Administration	987654321		
Narayan, Ramesh K.	666884444	1962-09-15	975 FireOak, Humble, TX	5	Research	333445555		
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5	Research	333445555		
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4	Administration	987654321		
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1	Headquarters	888665555		

EMP_PROJ						Redundancy	
Ssn	Pnumber	Hours	Ename	Pname	Plocation		
123456789	1	32.5	Smith, John B.	ProductX	Bellaire		
123456789	2	7.5	Smith, John B.	ProductY	Sugarland		
666884444	3	40.0	Narayan, Ramesh K.	ProductZ	Houston		
453453453	1	20.0	English, Joyce A.	ProductX	Bellaire		
453453453	2	20.0	English, Joyce A.	ProductY	Sugarland		
333445555	2	10.0	Wong, Franklin T.	ProductY	Sugarland		
333445555	3	10.0	Wong, Franklin T.	ProductZ	Houston		
333445555	10	10.0	Wong, Franklin T.	Computerization	Stafford		
333445555	20	10.0	Wong, Franklin T.	Reorganization	Houston		
999887777	30	30.0	Zelaya, Alicia J.	Newbenefits	Stafford		
999887777	10	10.0	Zelaya, Alicia J.	Computerization	Stafford		
987987987	10	35.0	Jabbar, Ahmad V.	Computerization	Stafford		
987987987	30	5.0	Jabbar, Ahmad V.	Newbenefits	Stafford		
987654321	30	20.0	Wallace, Jennifer S.	Newbenefits	Stafford		
987654321	20	15.0	Wallace, Jennifer S.	Reorganization	Houston		
888665555	20	Null	Borg, James E.	Reorganization	Houston		

## Guideline for Redundant Information in Tuples and Update Anomalies

- **Guideline 2:**

- Design a schema that does not suffer from the insertion, deletion and update anomalies.
- If there are any anomalies present, then note them so that applications can be made to take them into account.

## Null Values in Tuples

- **Guideline 3:**

- Relations should be designed such that their tuples will have as few NULL values as possible
- Attributes that are NULL frequently could be placed in separate relations (with the primary key)

- Reasons for nulls:

- Attribute not applicable or invalid
- Attribute value unknown (may exist)
- Value known to exist, but unavailable

## Generation of Spurious Tuples – Avoid at Any Cost

- Bad designs for a relational database may result in erroneous results for certain JOIN operations
- The “lossless join” property is used to guarantee meaningful results for join operations
- **Guideline 4:**
  - The relations should be designed to satisfy the lossless join condition.
  - No spurious tuples should be generated by doing a natural-join of any relations.
- There are two important properties of decompositions:
  - a) Non-additive or losslessness of the corresponding join
  - b) Preservation of the functional dependencies.
- Note that:
  - Property (a) is extremely important and **cannot** be sacrificed.
  - Property (b) is less stringent and may be sacrificed.

## Functional Dependencies

- Functional dependencies (FDs)
  - Are used to specify **formal measures** of the “goodness” of relational designs
  - And keys are used to define **normal forms** for relations
  - Are **constraints** that are derived from the **meaning** and **interrelationships** of the data attributes
- A set of attributes **X functionally determines** a set of attributes **Y** if the value of **X** determines a unique value for **Y**

## Defining Functional Dependencies

- $X \rightarrow Y$  holds if whenever two tuples have the same value for  $X$ , they **must have** the same value for  $Y$ 
  - For any two tuples  $t_1$  and  $t_2$  in any relation instance  
 $r(R)$ : If  $t_1[X] = t_2[X]$ , **then**  $t_1[Y] = t_2[Y]$
- $X \rightarrow Y$  in  $R$  specifies a **constraint** on all relation instances  $r(R)$
- Written as  $X \rightarrow Y$  can be displayed graphically on a relation schema as in Figures. (denoted by the arrow:).
- FDs are derived from the real-world constraints on the attributes

## Examples of FD Constraints

- Social security number determines employee name
  - $SSN \rightarrow ENAME$
- Project number determines project name and location
  - $PNUMBER \rightarrow \{PNAME, PLOCATION\}$
- Employee s s n and project number determines the hours per week that the employee works on the project
  - $\{SSN, PNUMBER\} \rightarrow HOURS$
- An FD is a property of the attributes in the schema  $R$
- The constraint must hold on **every** relation instance  $r(R)$
- If  $K$  is a key of  $R$ , then  $K$  functionally determines all attributes in  $R$ 
  - (since we never have two distinct tuples with  $t_1[K] = t_2[K]$ )



## Defining FDs From Instances

- Note that in order to define the FDs, we need to understand the meaning of the attributes involved and the relationship between them.
- An FD is a property of the attributes in the schema  $R$
- Given the instance (population) of a relation, all we can conclude is that an FD **may exist** between certain attributes.
- What we can **definitely** conclude is – that certain FDs **do not exist** because there are tuples that show a violation of those dependencies.

## Figure 14.7 Ruling Out FDs

Note that given the state of the TEACH relation, we can say that the FD: Text  $\rightarrow$  Course may exist. However, the FDs Teacher  $\rightarrow$  Course, Teacher  $\rightarrow$  Text and Course  $\rightarrow$  Text are ruled out.

### TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

## Normalization of Relations and Normal Forms

- **Normalization:**
  - The process of decomposing unsatisfactory “bad” relations by breaking up their attributes into smaller relations
- **Normal form:**
  - Condition using keys and FDs of a relation to certify whether a relation schema is in a particular normal form
- 2NF, 3NF, BCNF
  - based on keys and FDs of a relation schema
- 4NF (rarely used)
  - based on keys, multi-valued dependencies: MVDs
- 5NF (rarely used)
  - based on keys, join dependencies: JDs

## Practical Use of Normal Forms

- **Normalization** is carried out in practice so that the resulting designs are of high quality and meet the desirable properties
- The practical utility of these normal forms becomes questionable when the constraints on which they are based are **hard to understand** or to **detect**
- The database designers **need not** normalize to the highest possible normal form
  - (usually up to 3NF and BCNF. 4NF rarely used in practice.)
- **Denormalization:**
  - The process of storing the join of higher normal form relations as a base relation—which is in a lower normal form

## Definitions of Keys and Attributes Participating in Keys

- A **superkey** of a relation schema  $R = \{A_1, A_2, \dots, A_n\}$  is a set of attributes  $S$  **subset-of**  $R$  with the property that no two tuples  $t_1$  and  $t_2$  in any legal relation state  $r$  of  $R$  will have  $t_1[S] = t_2[S]$
- A **key**  $K$  is a **superkey** with the **additional property** that removal of any attribute from  $K$  will cause  $K$  not to be a superkey any more.
- If a relation schema has more than one key, each is called a **candidate** key.
  - One of the candidate keys is **arbitrarily** designated to be the **primary key**, and the others are called **secondary keys**.
- A **Prime attribute** must be a member of **some** candidate key
- A **Nonprime attribute** is not a prime attribute—that is, it is not a member of any candidate key.

## First Normal Form (1NF)

- Disallows
  - composite attributes
  - multivalued attributes
  - **nested relations**; attributes whose values for an **individual tuple** are non-atomic
- Considered to be part of the definition of a relation
- Most RDBMSs allow only those relations to be defined that are in First Normal Form

## Figure 14.9 Normalization into 1NF

(a) A relation schema that is not in 1NF.

(b) Sample state of relation DEPARTMENT.

(c) 1NF version of the same relation with redundancy.

(a)

DEPARTMENT			
Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT			
Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT			
Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

## Figure 14.10 Normalizing Nested Relations into 1NF

(a) Schema of the EMP\_PROJ relation with a nested relation attribute PROJS.

(b) Sample extension of the EMP\_PROJ relation showing nested relations within each tuple.

(c) Decomposition of EMP\_PROJ into relations EMP\_PROJ1 and EMP\_PROJ2 by propagating the primary key.

(a)

EMP_PROJ		Projs	
Ssn	Ename	Pnumber	Hours

(b)

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
987654321	Wallace, Jennifer S.	30	5.0
		20	20.0
888665555	Borg, James E.	20	15.0
888665555	Borg, James E.	20	NULL

(c)

EMP_PROJ1	
Ssn	Ename

EMP_PROJ2		
Ssn	Pnumber	Hours

## Second Normal Form (2NF)

- Uses the concepts of **FDs, primary key**
- Definitions
  - **Prime attribute:** An attribute that is member of the primary key  $K$
  - **Full functional dependency:** a FD  $Y \rightarrow Z$  where removal of any attribute from  $Y$  means the FD does not hold any more
- Examples:
  - $\{SSN, PNUMBER\} \rightarrow HOURS$  is a full FD since neither  $SSN \rightarrow HOURS$  nor  $PNUMBER \rightarrow HOURS$  hold
  - $\{SSN, PNUMBER\} \rightarrow ENAME$  is not a full FD (it is called a partial dependency ) since  $SSN \rightarrow ENAME$  also holds
- A relation schema  $R$  is in **second normal form (2NF)** if every non-prime attribute  $A$  in  $R$  is fully functionally dependent on the primary key
- $R$  can be decomposed into 2NF relations via the process of 2NF normalization or “second normalization”
- Informally, all attributes depend on the whole key



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

25

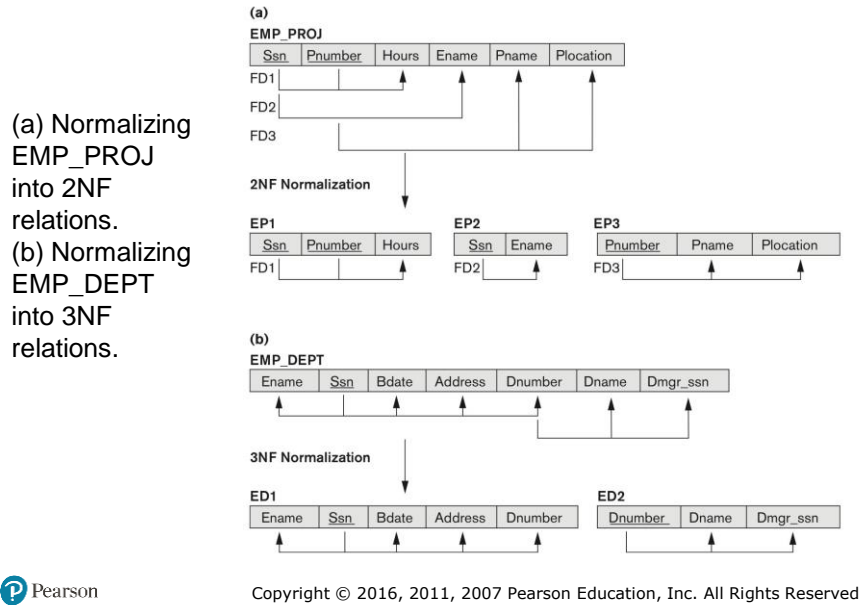
## Third Normal Form (3NF)

- Definition:
  - **Transitive functional dependency:** a FD  $X \rightarrow Z$  that can be derived from two FDs  $X \rightarrow Y$  and  $Y \rightarrow Z$
- Examples:
  - $SSN \rightarrow DMGRSSN$  is a **transitive** FD
    - Since  $SSN \rightarrow DNUMBER$  and  $DNUMBER \rightarrow DMGRSSN$  hold
  - $SSN \rightarrow ENAME$  is **non-transitive**
    - Since there is no set of attributes  $X$  where  $SSN \rightarrow X$  and  $X \rightarrow ENAME$
- A relation schema  $R$  is in **third normal form (3NF)** if it is in 2NF **and** no non-prime attribute  $A$  in  $R$  is transitively dependent on the primary key
- Informally, all attributes depend on nothing but the key
- $R$  can be decomposed into 3NF relations via the process of 3NF normalization
- NOTE:
  - In  $X \rightarrow Y$  and  $Y \rightarrow Z$ , with  $X$  as the primary key, we consider this a problem only if  $Y$  is not a candidate key.
  - When  $Y$  is a candidate key, there is no problem with the transitive dependency .
  - E.g., Consider EMP (SSN, Emp#, Salary ).
    - Here,  $SSN \rightarrow Emp\# \rightarrow Salary$  and  $Emp\#$  is a candidate key.

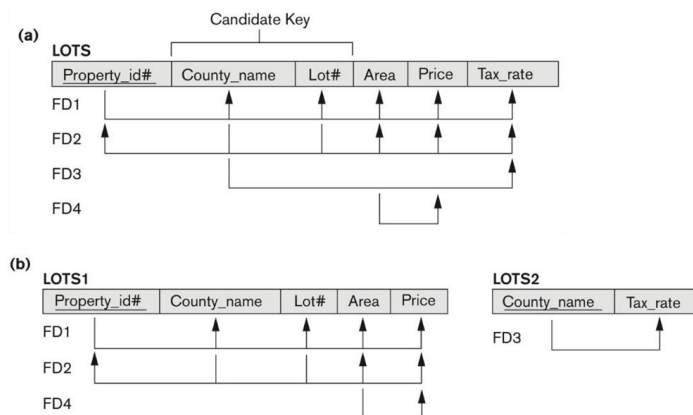


Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

26

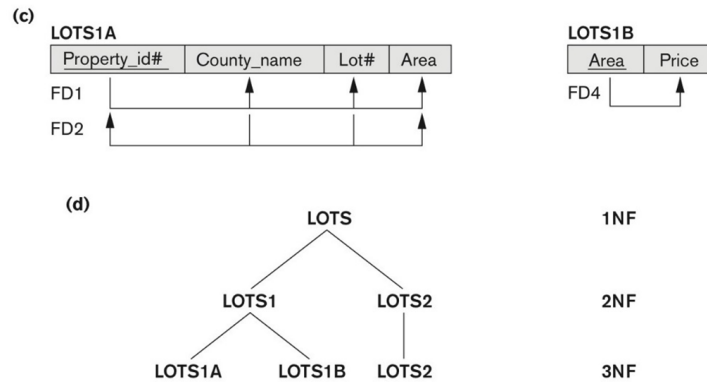
**Figure 14.11 Normalizing into 2NF and 3NF**

27

**Figure 14.12 Normalization into 2NF and 3NF (1 of 2)**

28

## Figure 14.12 Normalization into 2NF and 3NF (2 of 2)



(c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.

## General Normal Form Definitions (For Multiple Keys)

- The above definitions consider the primary key only
- The following more general definitions take into account relations with multiple candidate keys
- Any attribute involved in a candidate key is a **prime attribute**
- All other attributes are called **non-prime attributes**.

## General Definition of 2NF (For Multiple Candidate Keys)

- A relation schema  $R$  is in **second normal form (2NF)** if every non-prime attribute  $A$  in  $R$  is fully functionally dependent on **every** key of  $R$
- In Figure 14.12 the FD  
County\_name  $\rightarrow$  Tax\_rate violates 2NF.

So second normalization converts LOTS into

LOTS1(Property\_id#, County\_name, Lot#, Area, Price)  
LOTS2 (County\_name, Tax\_rate)



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

31

## General Definition of 3NF

- Definition:
  - **Superkey** of relation schema  $R$  - a set of attributes  $S$  of  $R$  that contains a key of  $R$
  - A relation schema  $R$  is in **third normal form (3NF)** if whenever a FD  $X \rightarrow A$  holds in  $R$ , then either:
    - (a)  $X$  is a superkey of  $R$ , or
    - (b)  $A$  is a prime attribute of  $R$
- LOTS1 relation violates 3NF because Area  $\rightarrow$  Price ; and Area is not a superkey in LOTS1. (see Figure 14.12).



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

32



## Interpreting the General Definition of 3NF

- Condition (a) catches two types of violations :
  - one where a prime attribute functionally determines a non-prime attribute. This catches 2NF violations due to non-full functional dependencies.
  - second, where a non-prime attribute functionally determines a non-prime attribute. This catches 3NF violations due to a transitive dependency.
- **Alternative Definition of 3NF: We can restate the definition as:**
  - A relation schema  $R$  is in **third normal form (3NF)** if every non-prime attribute in  $R$  meets both of these conditions:
    - It is fully functionally dependent on every key of  $R$
    - It is non-transitively dependent on every key of  $R$
  - Note that stated this way, a relation in 3NF also meets the requirements for 2NF.
- The condition (b) from the last slide takes care of the dependencies that “**slip through**” (**are allowable to**) 3NF but are “caught by” BCNF which we discuss next.



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

33

## BCNF (Boyce-Codd Normal Form)

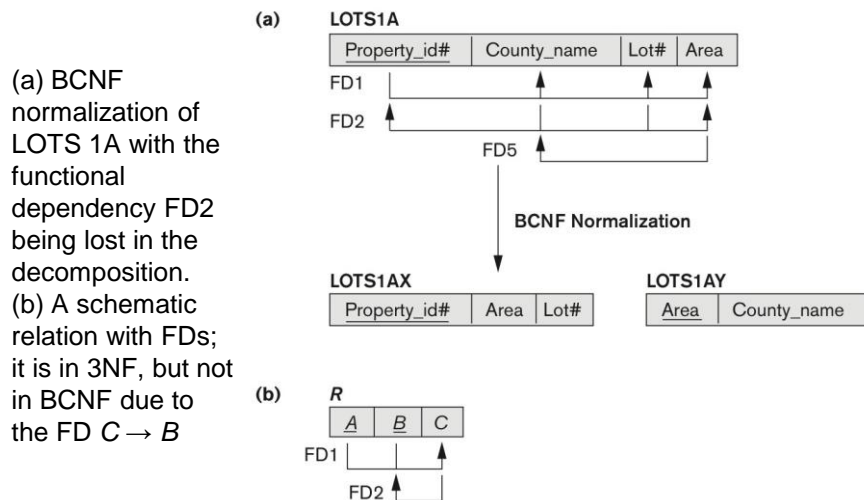
- A relation schema  $R$  is in **Boyce-Codd Normal Form (BCNF)** if whenever an **FD**  $X \rightarrow A$  holds in  $R$ , then  **$X$  is a superkey of  $R$**
- Each normal form is strictly stronger than the previous one
  - Every 2NF relation is in 1NF
  - Every 3NF relation is in 2NF
  - Every BCNF relation is in 3NF
- There exist relations that are in 3NF but not in BCNF
- Hence BCNF is considered a **stronger form of 3NF**
- The goal is to have each relation in BCNF (or 3NF)



Copyright © 2016, 2011, 2007 Pearson Education, Inc. All Rights Reserved

34

## Figure 14.13 Boyce-Codd Normal Form



## Figure 14.14 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

## Achieving the BCNF by Decomposition (1 of 2)

- Two FDs exist in the relation TEACH:
  - FD1: { student, course} → instructor
  - FD2: instructor → course
- {student, course} is a candidate key for this relation and that the dependencies shown follow the pattern in Figure 14.13 (b).
  - So this relation is in 3NF **but not in BCNF**
- A relation **NOT** in BCNF should be decomposed so as to meet this property, while possibly forgoing the preservation of all functional dependencies in the decomposed relations.

## Achieving the BCNF by Decomposition (2 of 2)

- Three possible decompositions for relation TEACH
  - D1: {**student, instructor**} and {**student, course**}
  - D2: {course, **instructor**} and {**course, student**}
  - D3: {**instructor, course**} and {**instructor, student**} ✓
- All three decompositions will lose FD1.
  - We have to settle for sacrificing the functional dependency preservation. But we **cannot** sacrifice the non-additivity property after decomposition.
- Out of the above three, only the 3rd decomposition will not generate spurious tuples after join. (and hence has the non-additivity property).

## General Procedure for Achieving BCNF When a Relation Fails BCNF

- Let  $R$  be the relation not in BCNF, let  $X$  be a subset-of  $R$ , and let  $X \rightarrow A$  be the FD that causes a violation of BCNF. Then  $R$  may be decomposed into two relations:
  - (i)  $R - A$  and (ii)  $X \cup A$ .
  - If either  $R - A$  or  $X \cup A$  is not in BCNF, repeat the process.

Note that the f.d. that violated BCNF in TEACH was  $\text{Instructor} \rightarrow \text{Course}$ . Hence its BCNF decomposition would be :

(TEACH – COURSE) and (Instructor  $\cup$  Course), which gives the relations: (Instructor, Student) and (Instructor, Course) that we obtained before in decomposition D3.

## Copyright

