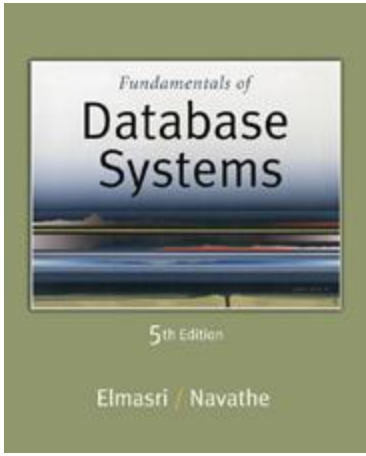
Chapter 10

Functional Dependencies and Normalization for Relational Databases



Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Chapter Outline

- 1 Informal Design Guidelines for Relational Databases
  - 1.1Semantics of the Relation Attributes
  - 1.2 Redundant Information in Tuples and Update Anomalies
  - 1.3 Null Values in Tuples
  - 1.4 Spurious Tuples
- 2. Functional Dependencies (*skip*)

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Chapter Outline

- 3. Normal Forms Based on Primary Keys
  - 3.1 Normalization of Relations
  - 3.2 Practical Use of Normal Forms
  - 3.3 Definitions of Keys and Attributes Participating in Keys
  - 3.4 First Normal Form
  - 3.5 Second Normal Form
  - 3.6 Third Normal Form
- 4. General Normal Form Definitions (For Multiple Keys)
- 5. BCNF (Boyce-Codd Normal Form)

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Informal Design Guidelines for Relational Databases (2)

- 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)
- Additional types of dependencies, further normal forms, relational design algorithms by synthesis are discussed in Chapter 11

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

1 Informal Design Guidelines for Relational Databases (1)

- 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?

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

1.1 Semantics of the Relation Attributes

- **GUIDELINE 1**:

  Informally, each tuple in a relation should represent *one* entity or relationship instance.

  - 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.*

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.1 A simplified COMPANY relational database schema

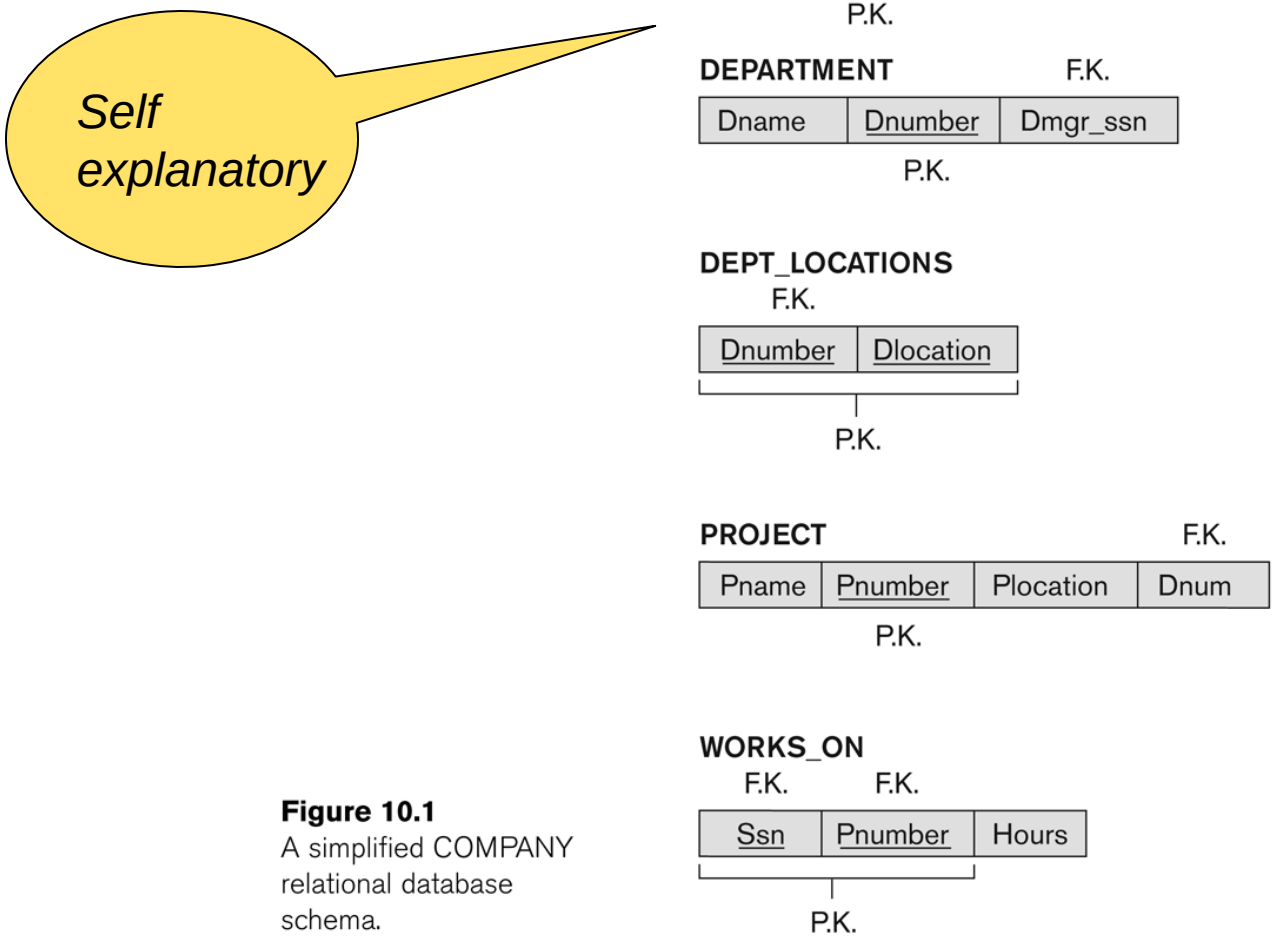


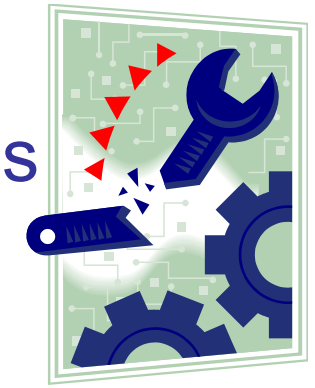
Figure 10.1
A simplified COMPANY relational database schema.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

1.2 Redundant Information in Tuples and Update Anomalies

- **Big (and common) DB Problem:**

  In a ‘poorly designed’ DB information is stored redundantly



Consequences:

- Wastes storage
- Causes problems with update anomalies
  - Insertion anomalies
  - Deletion anomalies
  - Modification anomalies

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

EXAMPLE OF AN INSERT ANOMALY

- Consider the relation:

  EMP_PROJ (Emp#, Proj#, Ename, Pname, No_hours)

- **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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

EXAMPLE OF AN DELETE ANOMALY

- Consider the relation:

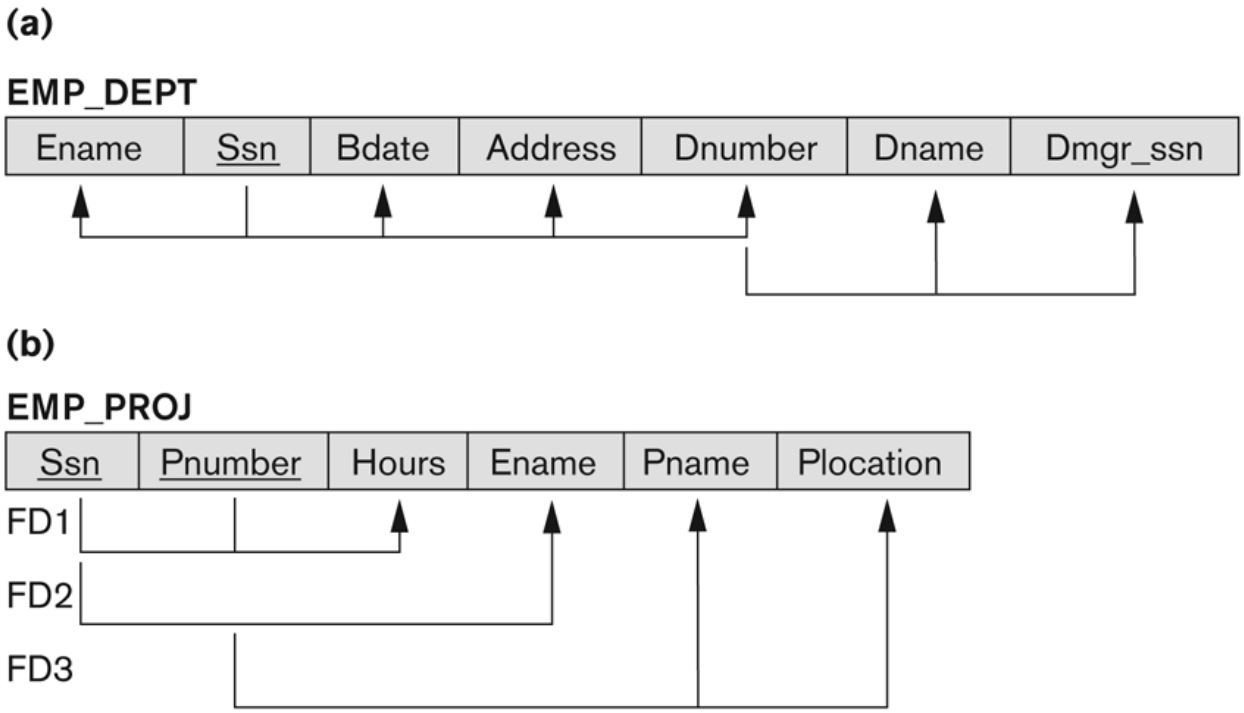
  EMP_PROJ (Emp#, Proj#, Ename, Pname, No_hours)

- **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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.3 Two relation schemas suffering from update anomalies

Figure 10.3
Two relation schemas suffering from update anomalies.
(a) EMP_DEPT and
(b) EMP_PROJ.



Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.4 Example States for EMP_DEPT and EMP_PROJ

Figure 10.4

Example states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 10.2. These may be stored as base relations for performance reasons.

Redundancy

EMP_DEPT

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

Redundancy Redundancy

EMP_PROJ

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

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Guideline to 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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

1.3 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

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

1.4 Spurious Tuples

- 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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

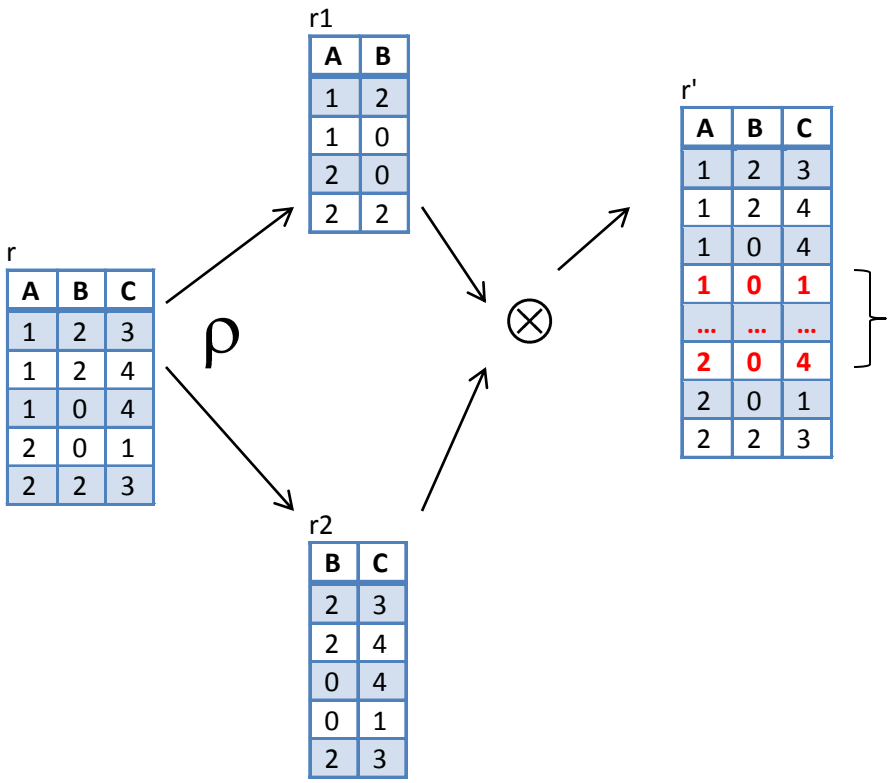
Spurious Tuples (2)

- 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. (See Chapter 11).

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Spurious Tuples (3)



Consider relation r(ABCD) and its projections r1(AB) and r2(BC).

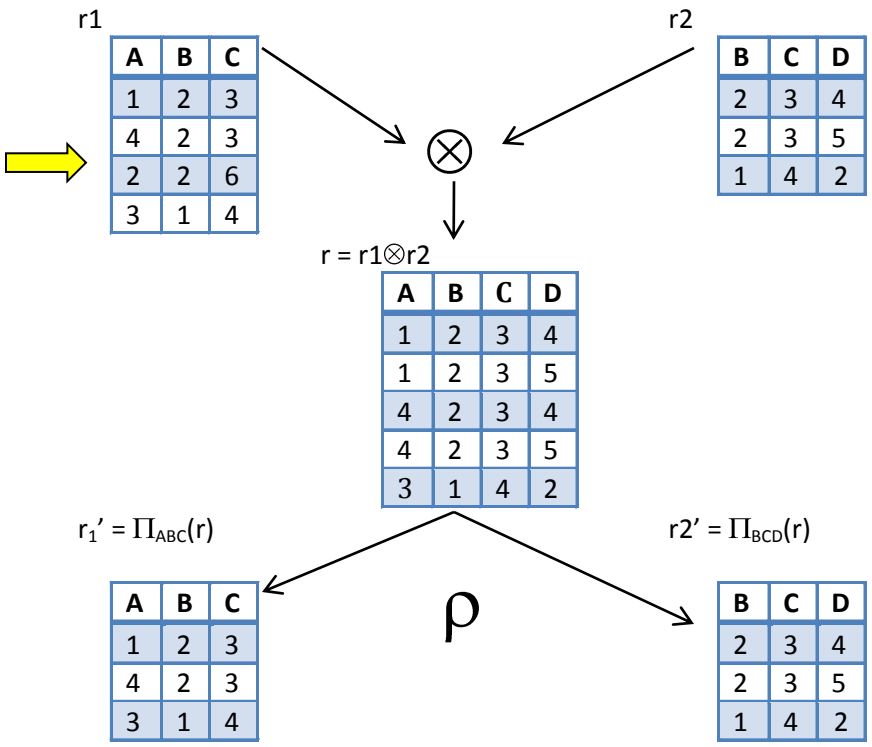
Phantom records

Observation

Not all decompositions of a table can be combined using *natural join* to reproduce the original table.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Spurious Tuples (4)



Consider the following two relations $r_1(ABC)$ and $r_2(BCD)$.

Compute natural join $r = r_1 \otimes r_2$

Evaluate projections $r_1' = \Pi_{ABC}(r)$ and $r_2' = \Pi_{BCD}(r)$

Observation

Tables r_2 and r_2' are the same however tuple $<2,2,6> \in r_1$ but not present in r_1'

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3 Normal Forms Based on Primary Keys

- 3.1 Normalization of Relations
- 3.2 Practical Use of Normal Forms
- 3.3 Definitions of Keys and Attributes Participating in Keys
- 3.4 First Normal Form
- 3.5 Second Normal Form
- 3.6 Third Normal Form

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.1 Normalization of Relations (1)

- **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

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Normalization of Relations (2)

- **2NF, 3NF, BCNF**
  - based on keys and FDs of a relation schema
- **4NF**
  - based on keys, multi-valued dependencies : MVDs; 5NF based on keys,

    Join dependencies : JDs (Chapter 11)
- Additional properties may be needed to ensure a good relational design (*lossless join, dependency preservation*; Chapter 11)

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.2 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 is 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, BCNF or 4NF)
- **Denormalization**:
  - The process of storing the join of higher normal form relations as a base relation—which is in a lower normal form

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.3 Definitions of Keys and Attributes Participating in Keys (1)

- A **superkey** of a relation schema R = {A1, A2, ..., An} is a set of attributes S *subset-of* R with the property that no two tuples t1 and t2 in any legal relation state r of R will have t1[S] = t2[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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Definitions of Keys and Attributes Participating in Keys (2)

- 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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.2 First Normal Form 1NF

- A relation scheme R is in first normal form (1NF) if the values in *dom* (*A*) are atomic for every attribute A in R.

- **Disallows**
  - composite attributes
  - Set-valued attributes
  - **nested relations**; a cell of an *individual tuple* is a complex relation

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.8 Normalization into 1NF

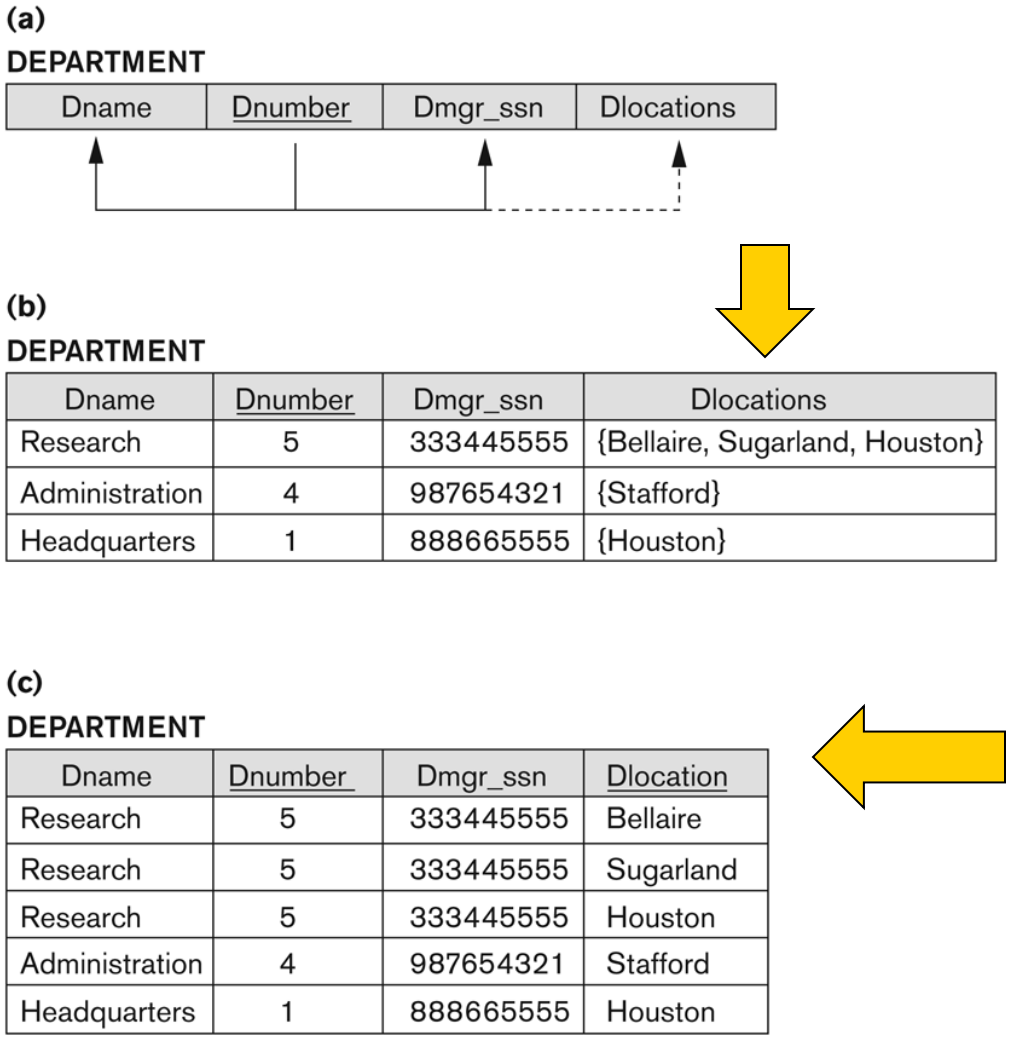
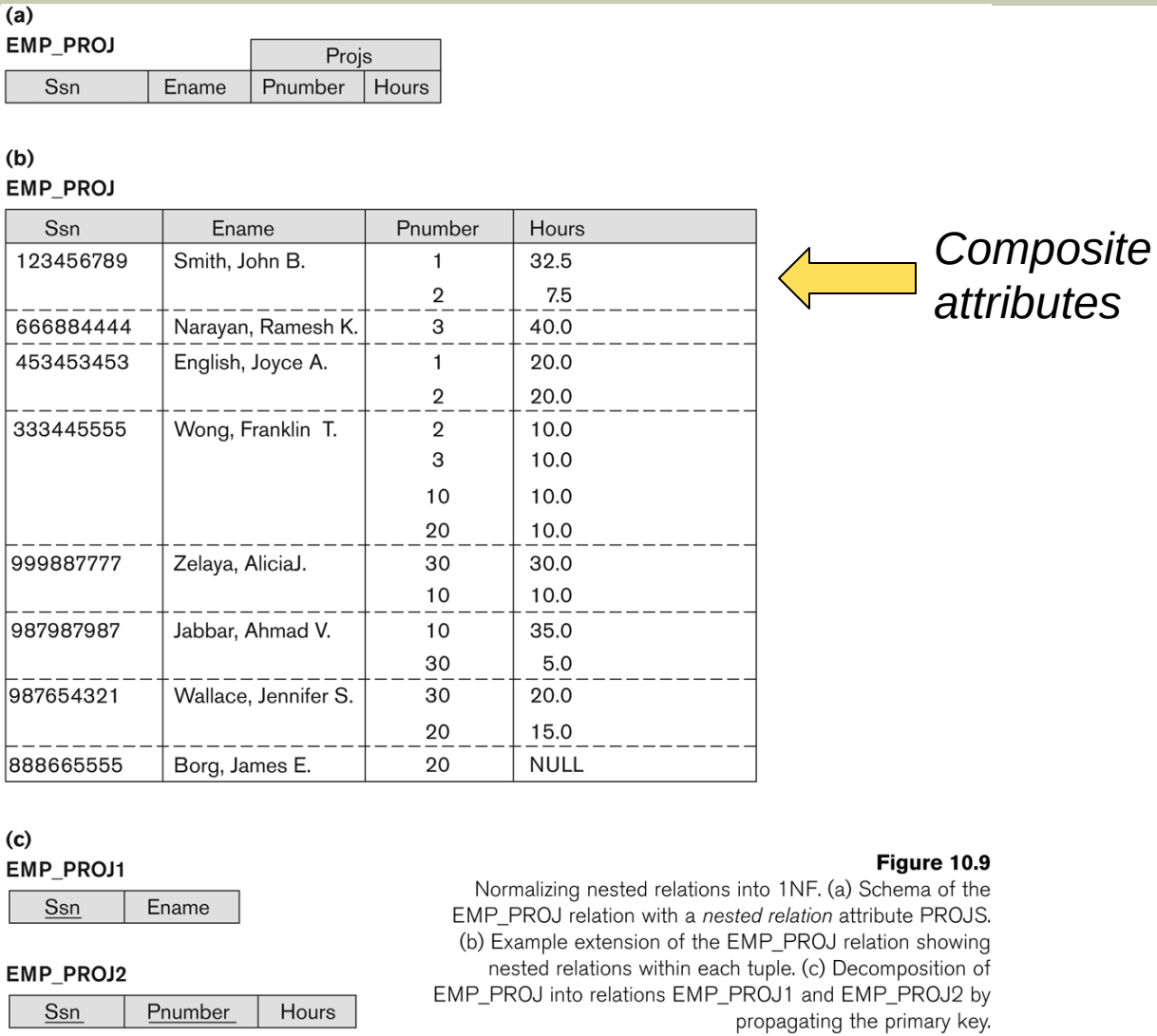


Figure 10.8
Normalization into 1NF. (a) A relation schema that is not in 1NF. (b) Example state of relation DEPARTMENT. (c) 1NF version of the same relation with redundancy.

Remember $Pack_A(r)$ and $Unpack_A(r)$ operators?

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.9 Normalization nested relations into 1NF



(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

(b)

EMP_PROJ

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, AliciaJ.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	NULL

Composite attributes

(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

Figure 10.9
Normalizing nested relations into 1NF. (a) Schema of the EMP_PROJ relation with a *nested relation* attribute PROJS. (b) Example 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.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.3 Second Normal Form (1)

- Uses the concepts of **FDs, primary key**

- **Definitions**
  - **Prime attribute:** An attribute that is member of the primary key K
  - **Left-Reduced or Full functional dependency:** a FD Y → Z where removal of any attribute from Y means the FD does not hold any more

- **Examples:**
  - {SSN, PNUMBER} → HOURS is a full FD since neither SSN → HOURS nor PNUMBER → HOURS hold
  - {SSN, PNUMBER} → ENAME is not a full FD (it is called a partial dependency) since SSN → ENAME also holds

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Second Normal Form (2)

- **A relation scheme R is in second normal form (2NF) with respect to a set of FDs F if it is in 1NF and every nonprime attribute is fully dependent on every key of R.**

- R can be decomposed into 2NF relations via the process of 2NF normalization

Example

Let R=ABCD and F = { AB → C, B → D }. Here AB is a key. C and D are non-prime. C is fully dependent on the entire key AB, however D functionally depends on just *part* of the key (B →D). This is called a *partial dependency.*

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Example (Krocnke pg. 291)
ACTIVITIES: KEY(SID, Activity)

Figure 10.10 Normalizing into 2NF and 3NF

Example

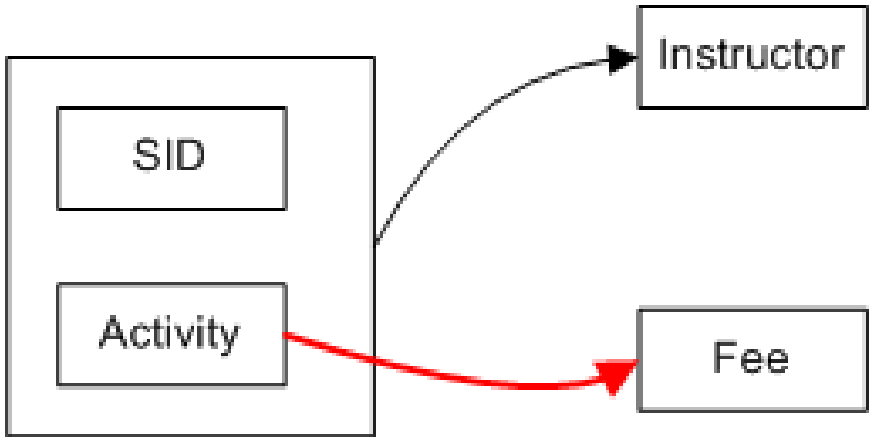
We deduce from the data sample

activity →fee,
sid activity →instructor

SID	Activity	Fee	Instructor
100	Basket Ball	200	Lebron
100	Golf	65	Arnold
200	Golf	65	Jack
300	Golf	65	Lebron

Key: *sid activity*. Non-key attributes: { Fee, Instructor }

There is a partial dependency
therefore the schema is not 2NF



Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.10 Normalizing into 2NF and 3NF

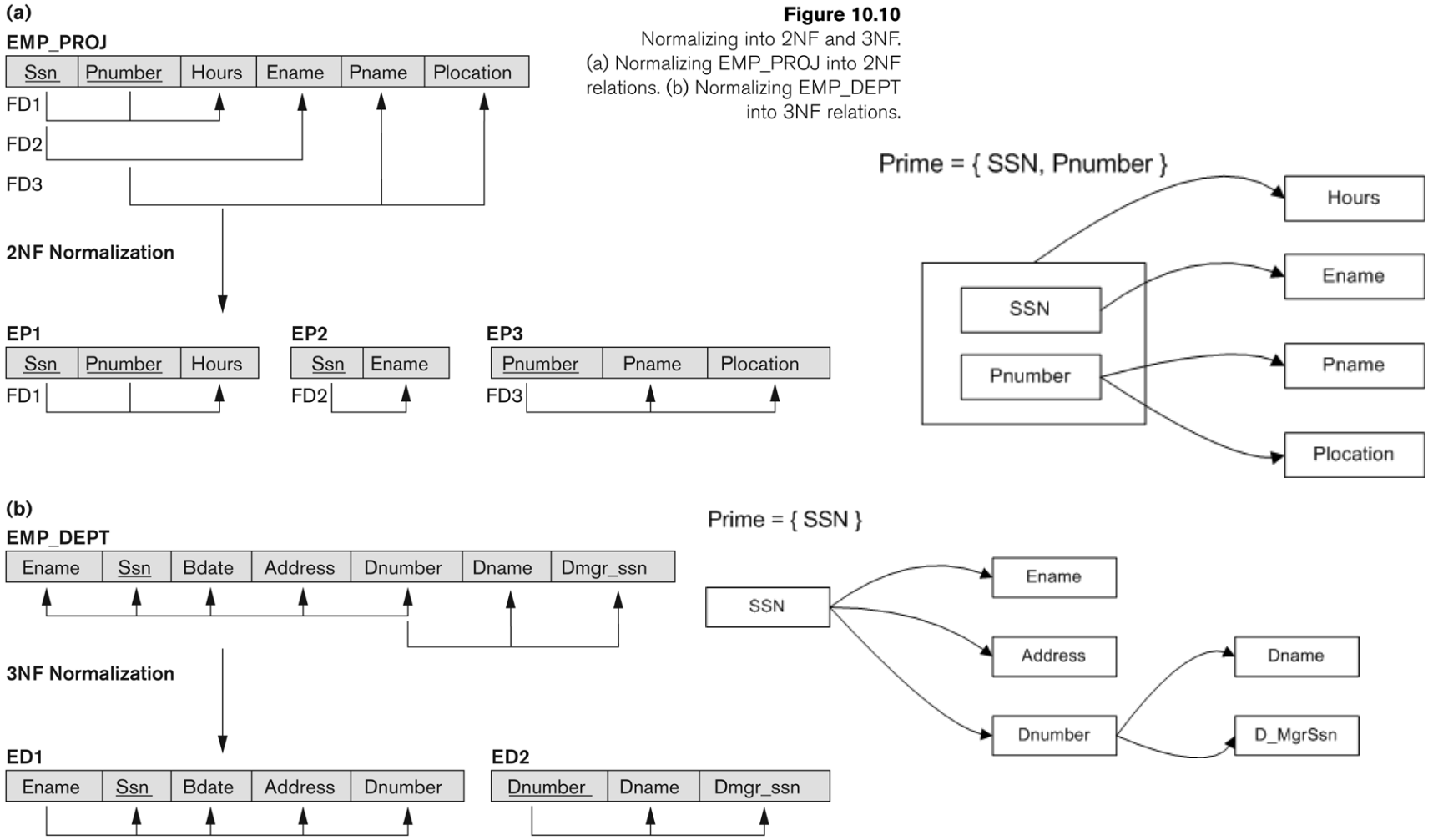


Figure 10.10
Normalizing into 2NF and 3NF. (a) Normalizing EMP_PROJ into 2NF relations. (b) Normalizing EMP_DEPT into 3NF relations.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.11 Normalization into 2NF and 3NF

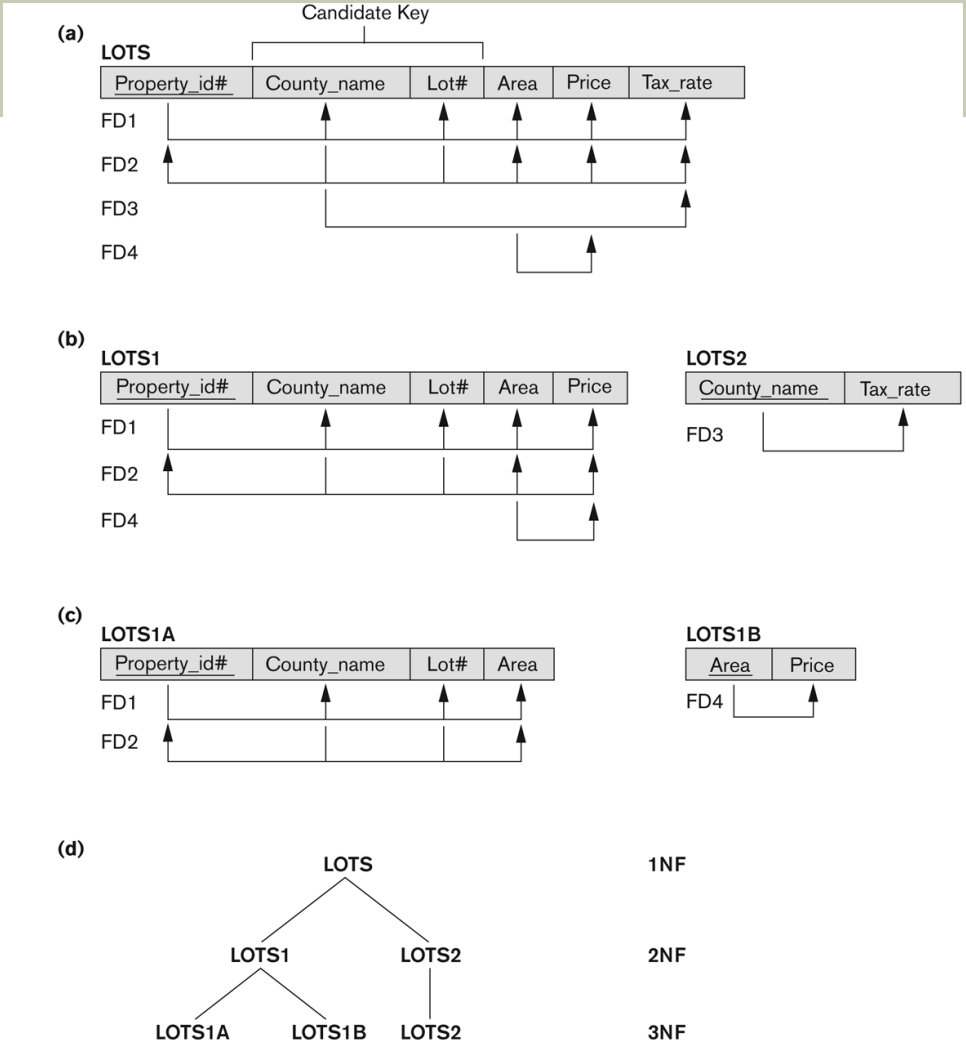


Figure 10.11
Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Summary of the progressive normalization of LOTS.

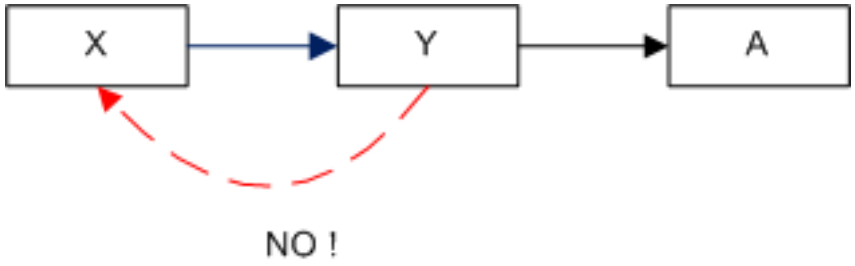
Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

3.4 Third Normal Form (1)

- **Definition**:

  Given a relation scheme R, a subset X of R, an attribute A in R, and a set of FDs F, A is **transitively dependent** upon X in R if there is a subset Y of R with:

  $X \rightarrow Y$, $Y \not\rightarrow X$ and $Y \rightarrow A$ under F and $A \notin XY$.



Examples:

Schema (ABCD) and F = {A →B, B →AC, C →D }

D is transitively dependent on A(and B) via C, however

C is not transitively dependent on A via B (B is prime).

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Third Normal Form (2)

- 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

- R can be decomposed into 3NF relations via the process of 3NF normalization

- **NOTE**:
  - In X → Y and Y → 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 → Emp# → Salary and Emp# is a candidate key.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Normal Forms Defined Informally

- 1st normal form
  - All attributes depend on **the key**
- 2nd normal form
  - All attributes depend on **the whole key**
- 3rd normal form
  - All attributes depend on **nothing but the key**

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

4 General Normal Form Definitions (For Multiple Keys) (1)

- The above definitions consider the primary key only
- The following more general definitions take into account relations with 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

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

4 General Normal Form Definitions (For Multiple Keys) (1)

Example Consider the schema

SUPPLIER(sname, saddress, item, iname, price) and FDs

F= { sname → saddress, item →iname, sname item →price }

1. *sname item* is the primary key, all other attributes are non-prime.
2. Observe that *saddress* depends on part of the key (*sname*).
3. Likewise iname depends on part of the key (item)
4. Therefore SUPPLIER is not in 2NF.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

General Normal Form Definitions (2)

- 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 → A holds in R, then either:
  - (a) X is a superkey of R, or
  - (b) A is a prime attribute of R

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

General Normal Form Definitions (2)

Example

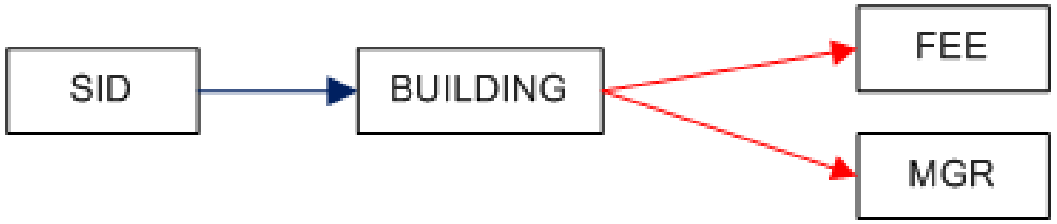
SID	Building	Fee	Manager
100	Fenn	300	Mr. T
300	ΔΠ	400	Ali
200	Holiday Inn	400	Tyson

Key: { SID }

SID→Building

Building →Fee

Building →Mgr



Fee (and *Manager*) transitively depend on *SID* via the non-prime attribute *Building*. Therefore the relation is not in 3NF.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

5 BCNF (Boyce-Codd Normal Form)

- A relation schema R is in **Boyce-Codd Normal Form (BCNF)** if whenever an **FD X→A** holds in R, then **X is a superkey** of R

Example

Keys: { Sid Major, Sid Fname }
Sid Major → Fname
Sid Fname →Major
Fname →Major

SID	MAJOR	FNAME
100	MATH	CAUCHY
100	PHYL	PLATO
200	MATH	CAUCHY
300	PHYS	NEWTON
400	PHYS	EINSTEIN

The relation is in 3NF but not in BCNF. Observe that Fname →Major is valid, but Fname is not a superkey.

Problem: *Student 300 drops PHYS.* We lose information that says NEWTON is a PHYS advisor.

A solution: (SID, FNAME), (FNAME, MAJOR)

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

5 BCNF (Boyce-Codd Normal Form)

- 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
- The goal is to have each relation in BCNF (or 3NF)

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.12 Boyce-Codd normal form

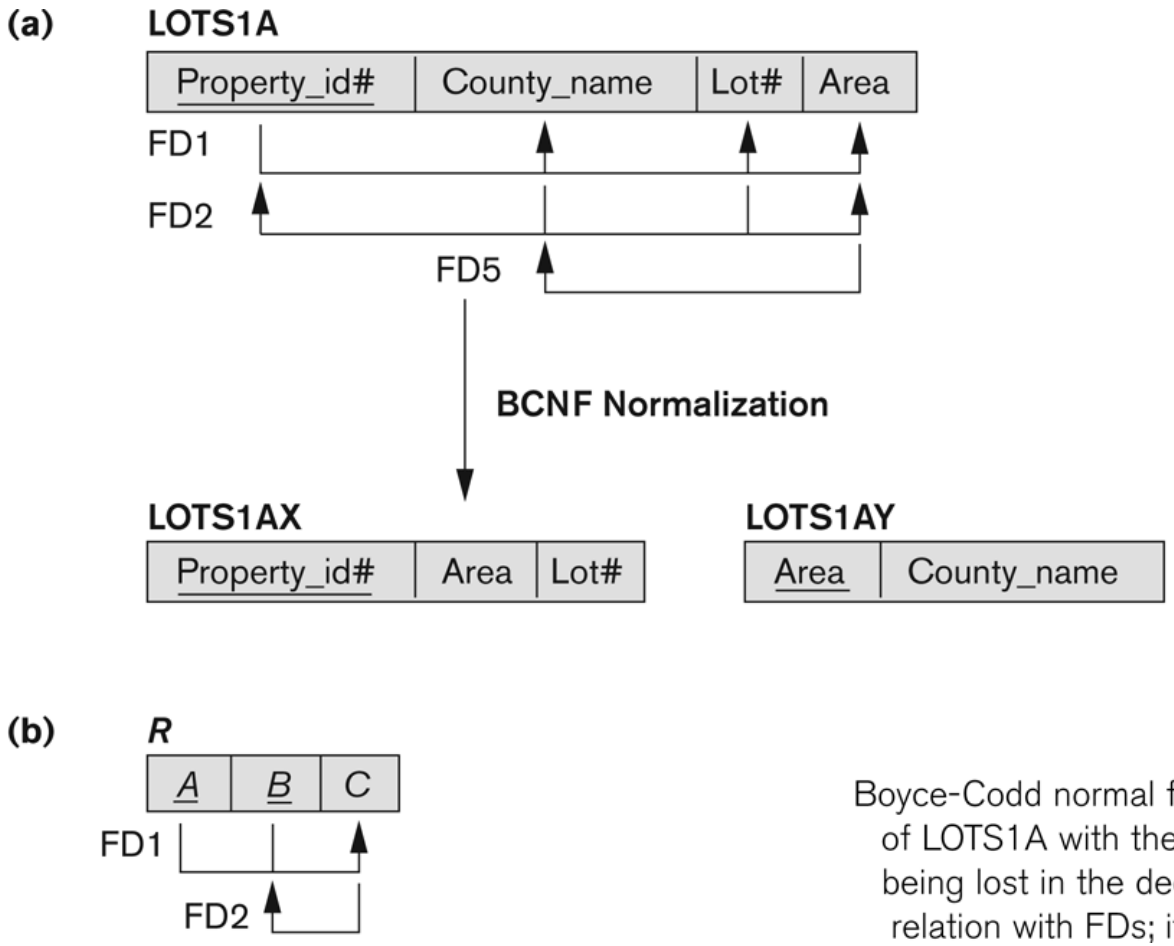


Figure 10.12
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Figure 10.13 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

Key: Student Course
Student Instructor

Dependencies
Stud Course → Instructor
Stud Instructor→Course
Instructor→Course

Figure 10.13
A relation TEACH that is in 3NF but not BCNF.

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

Achieving the BCNF by Decomposition (2)

- Three possible decompositions for relation TEACH

  {student, instructor} and {student, course}

  {course, instructor } and {course, student}

  {instructor, course } and {instructor, student}

- All three decompositions will lose FD { student, course} → instructor
  - 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 – to be discussed later) .

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe

What comes next?

In the next section we will study Decomposition & Synthesis algorithms as well as other type of dependencies

Copyright © 2007 Ramez Elmasri and Shamkant B. Navathe