KMITL
พระจอมเกล้าลาดกระบัง

KMITL GO
BEYOND THE LIMIT

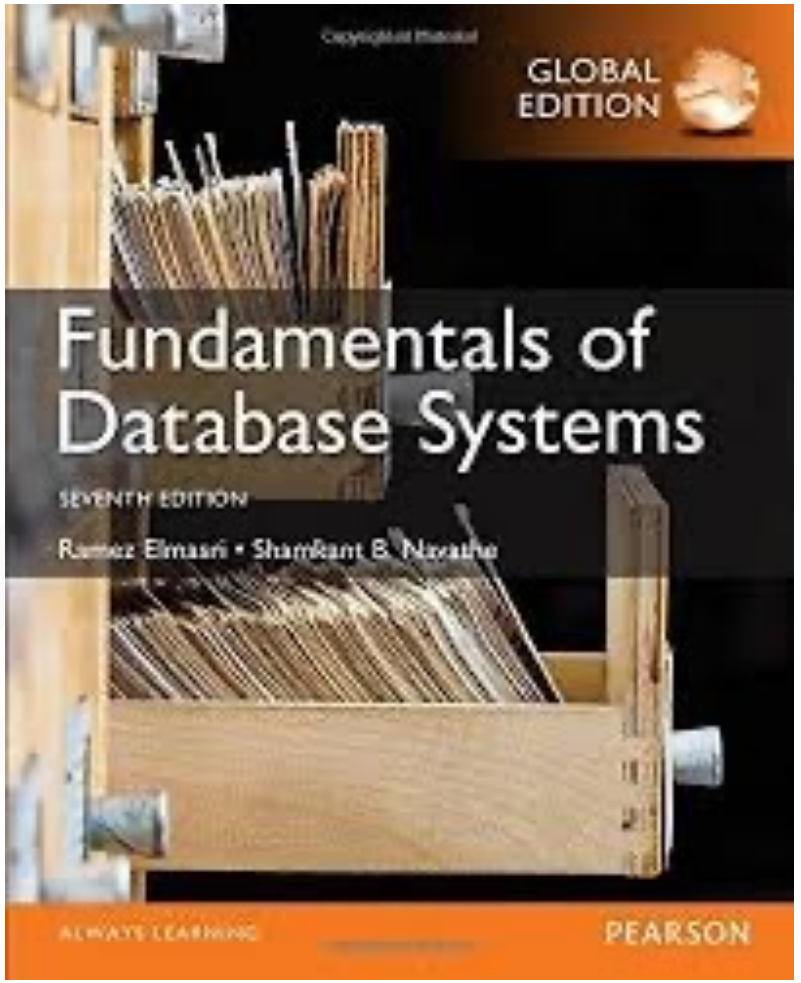
Database Systems

Program in Computer Engineering
School of Engineering

King Mongkut's Institute of Technology Ladkrabang

Text

- Ramez Elmasri and Shamkant B. Navathe. "Fundamentals of Database Systems" 7th Edition., Pearson, 2017



Chapter 14

Basics of Functional Dependencies and Normalization for Relational Database

Outline

1. Informal Design Guidelines for Relational Databases
2. Functional Dependencies (FDs)
3. Normal Forms Based on Primary Keys
4. General Normal Form Definitions for 2NF and 3NF (For Multiple Candidate Keys)
5. BCNF (Boyce-Codd Normal Form)
6. Multivalued Dependency and Fourth Normal Form
7. Join Dependencies and Fifth Normal Form

1. 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?

- Relational database design ultimately produces a set of relations.
- The implicit goals of the design activity are *information preservation* and *minimum redundancy*.

- 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)

1.1 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.

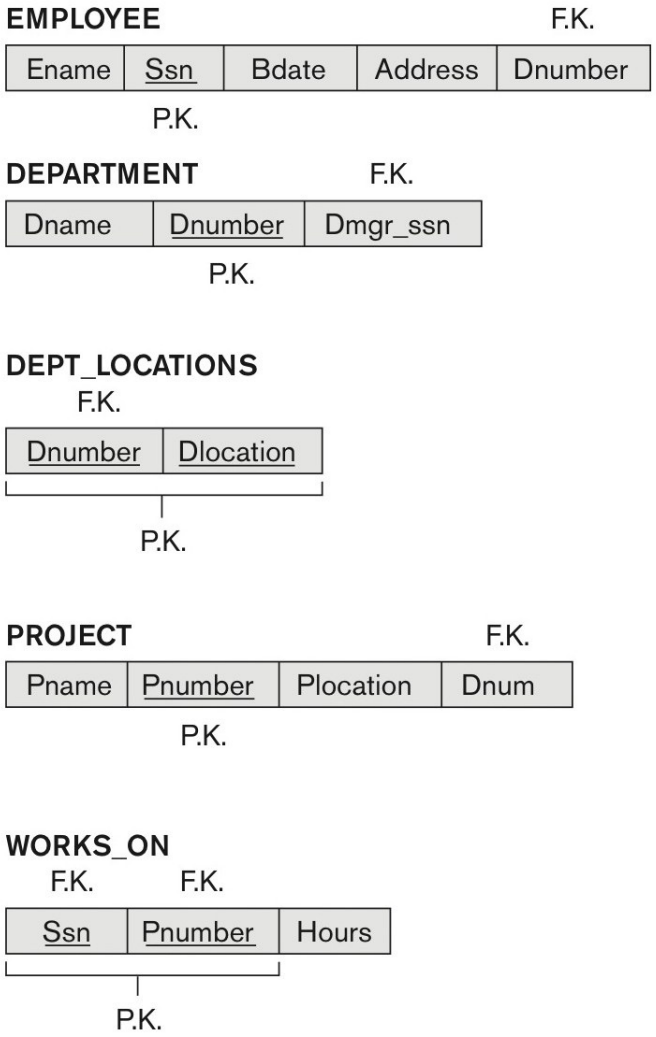


Figure 14.1 A simplified COMPANY relational database schema.

EMPLOYEE

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

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

1.2 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

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.

Example of a 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.

(a)

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

(b)

EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

FD1

FD2

FD3

Figure 14.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.

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

Figure 14.4
Sample states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 14.2. These may be stored as base relations for performance reasons.

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.

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

1.4 Generation of Spurious Tuples – avoid at any cost

- Bad designs for a relational database may result in erroneous results for certain JOIN operations

(a)

EMP_LOCS

Ename	Plocation

P.K.

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation

P.K.

(b)

EMP_LOCS

Ename	Plocation
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford
Zelaya, Alicia J.	Stafford
Jabbar, Ahmad V.	Stafford
Wallace, Jennifer S.	Stafford
Wallace, Jennifer S.	Houston
Borg, James E.	Houston

EMP_PROJ1

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	NULL	Reorganization	Houston

Figure 14.5 Particularly poor design for the EMP_PROJ relation in Figure 14.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 14.4 onto the relations EMP_LOCS and EMP_PROJ1.

	Ssn	Pnumber	Hours	Pname	Plocation	Ename
	123456789	1	32.5	ProductX	Bellaire	Smith, John B.
*	123456789	1	32.5	ProductX	Bellaire	English, Joyce A.
	123456789	2	7.5	ProductY	Sugarland	Smith, John B.
*	123456789	2	7.5	ProductY	Sugarland	English, Joyce A.
*	123456789	2	7.5	ProductY	Sugarland	Wong, Franklin T.
	666884444	3	40.0	ProductZ	Houston	Narayan, Ramesh K.
*	666884444	3	40.0	ProductZ	Houston	Wong, Franklin T.
*	453453453	1	20.0	ProductX	Bellaire	Smith, John B.
	453453453	1	20.0	ProductX	Bellaire	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Smith, John B.
	453453453	2	20.0	ProductY	Sugarland	English, Joyce A.
*	453453453	2	20.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	2	10.0	ProductY	Sugarland	Smith, John B.
*	333445555	2	10.0	ProductY	Sugarland	English, Joyce A.
	333445555	2	10.0	ProductY	Sugarland	Wong, Franklin T.
*	333445555	3	10.0	ProductZ	Houston	Narayan, Ramesh K.
	333445555	3	10.0	ProductZ	Houston	Wong, Franklin T.
	333445555	10	10.0	Computerization	Stafford	Wong, Franklin T.
*	333445555	20	10.0	Reorganization	Houston	Narayan, Ramesh K.
	333445555	20	10.0	Reorganization	Houston	Wong, Franklin T.

* * *

Figure 14.6 Result of applying NATURAL JOIN to the tuples in EMP_PROJ1 and EMP_LOCS of Figure 14.5 just for employee with Ssn = "123456789". Generated spurious tuples are marked by asterisks.

- 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.

Spurious Tuples

- 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 15).

2. 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

2.1 Defining Functional Dependencies

- X → 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 → Y in R specifies a *constraint* on all relation instances r(R)
- Written as X → 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 → ENAME
- Project number determines project name and location
  - PNUMBER → {PNAME, PLOCATION}
- Employee ssn and project number determines the hours per week that the employee works on the project
  - {SSN, PNUMBER} → 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

TEACH

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

Note that given the state of the TEACH relation, we can say that the FD: Text → Course may exist.

However, the FDs Teacher → Course, Teacher → Text and Couse → Text are ruled out.

Figure 14.8 What FDs may exist?

- A relation *R*(A, B, C, D) with its extension.
- Which FDs *may exist* in this relation?

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

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

3.1 Normalization of Relations

- **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

Normalization of Relations

- 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
- Additional properties may be needed to ensure a good relational design (lossless join, dependency preservation; see Chapter 15)

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 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

3.3 Definitions of Keys and Attributes Participating in Keys

- 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

Definitions of Keys and Attributes Participating in Keys

- 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.

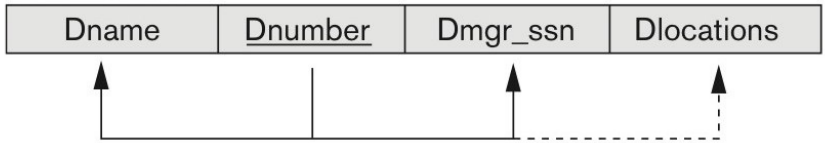
3.4 First Normal Form

- 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)

DEPARTMENT



(b)

DEPARTMENT

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

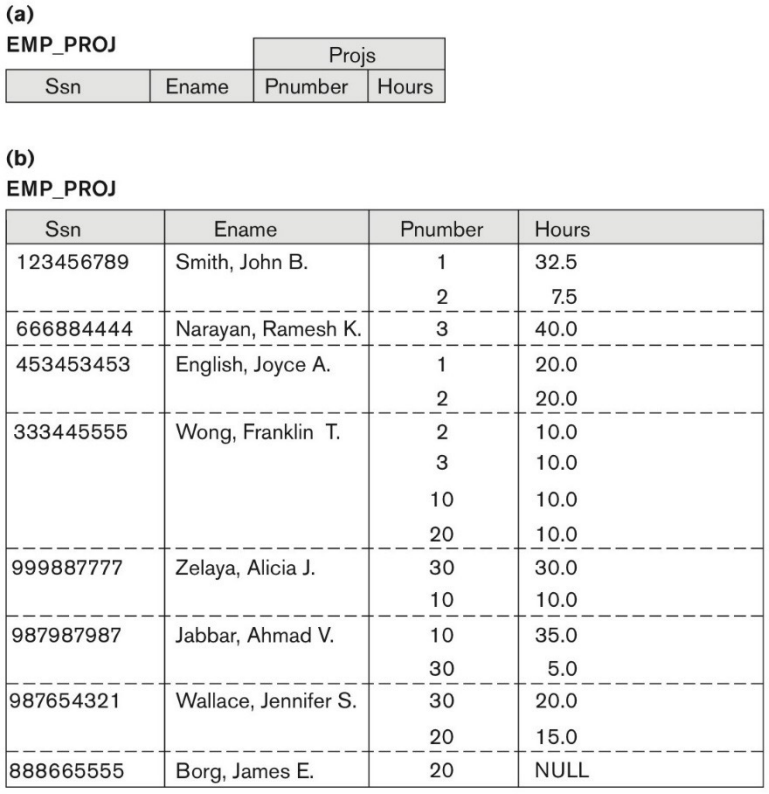
(c)

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 14.9
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.

Figure 14.10 Normalizing 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, Alicia J.	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

(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

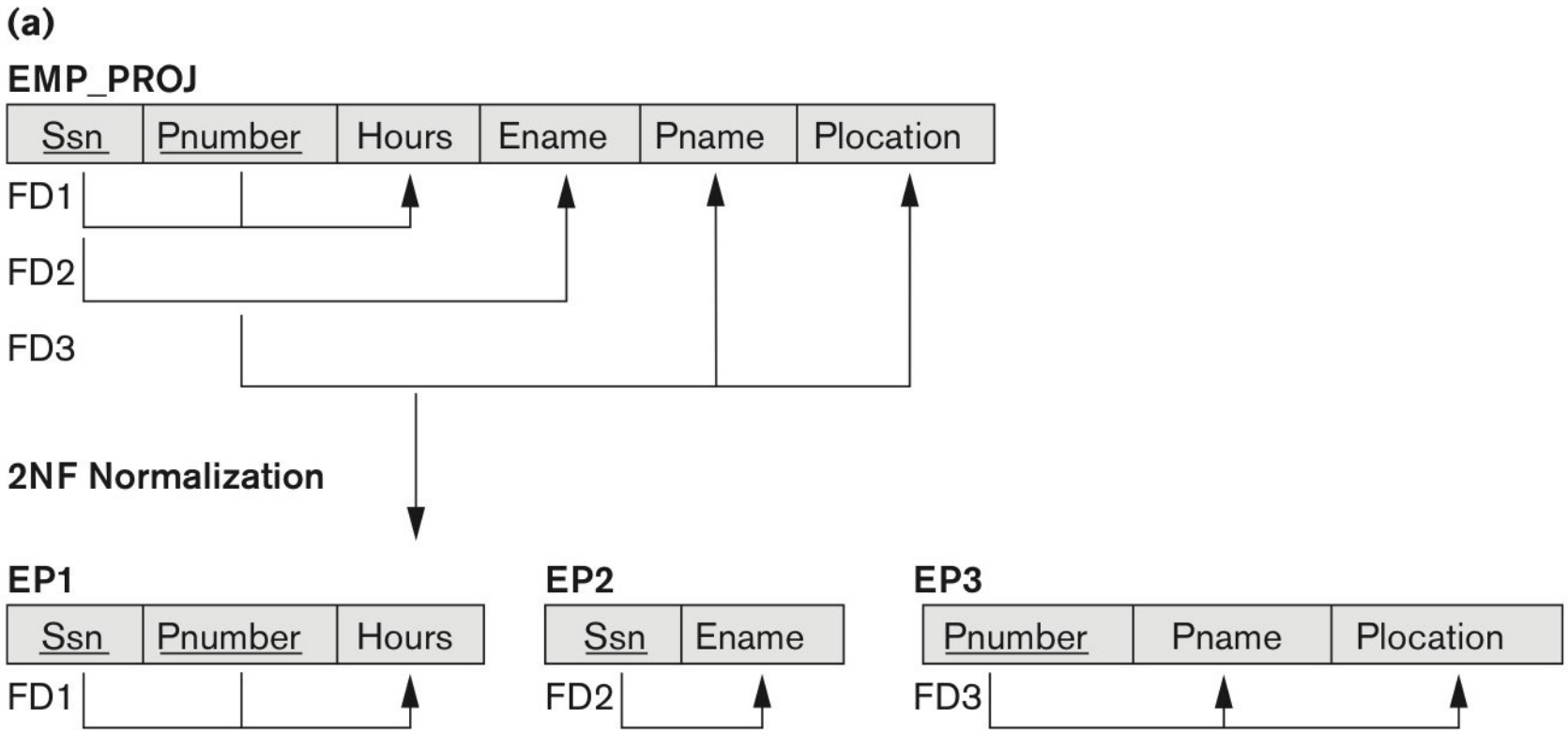
Ssn	Pnumber	Hours

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.

3.5 Second Normal Form

- 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 -> 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

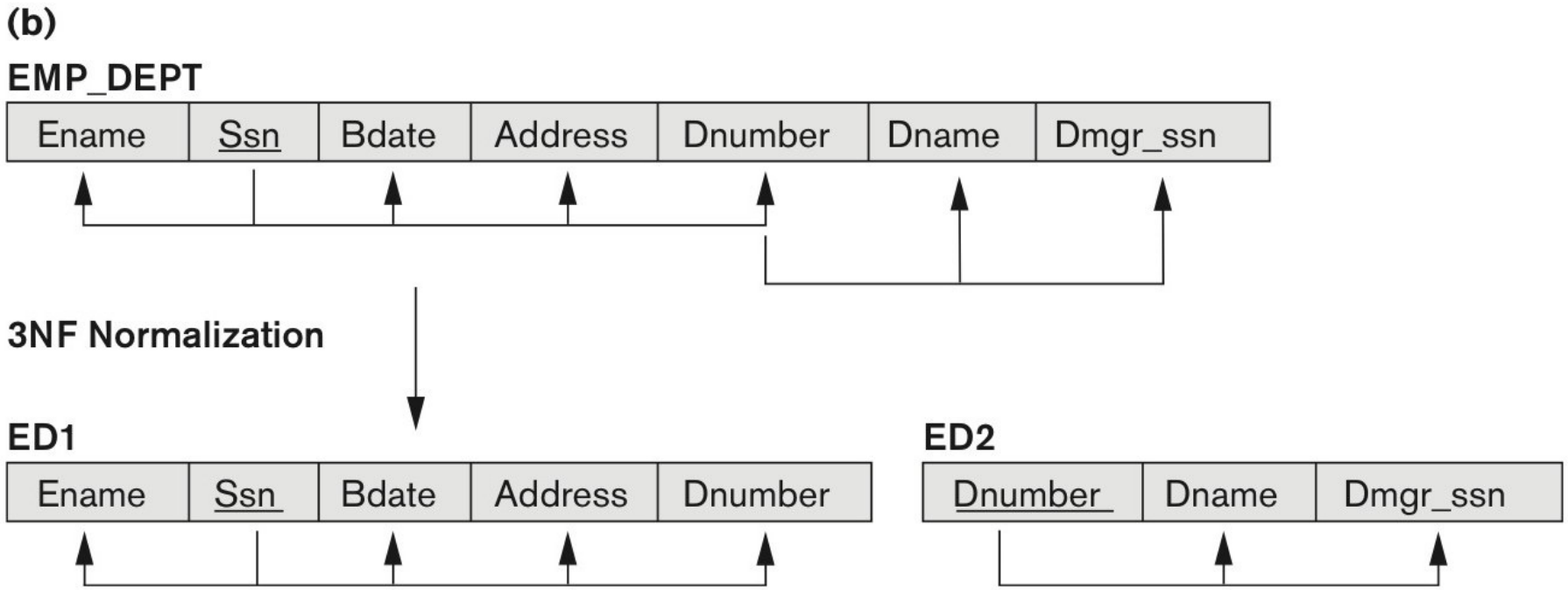
- A relation schema R is in **second normal form (2NF)** if every non-prime attribute A in R is fully functionally dependent on the primary key
- R can be decomposed into 2NF relations via the process of 2NF normalization or "second normalization"


(a)
EMP_PROJ
Ssn
Pnumber
Hours
Ename
Pname
Plocation
FD1
FD2
FD3
2NF Normalization
EP1
Ssn
Pnumber
Hours
FD1
EP2
Ssn
Ename
FD2
EP3
Pnumber
Pname
Plocation
FD3

Third Normal Form

- Definition:
  - **Transitive functional dependency**:
    a FD X -> Z that can be derived from two FDs X -> Y and Y -> Z
- Examples:
  - SSN -> DMGRSSN is a **transitive** FD
    - Since SSN -> DNUMBER and DNUMBER -> DMGRSSN hold
  - SSN -> ENAME is **non-transitive**
    - Since there is no set of attributes X where SSN -> X and X -> 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
- 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.


(b)
EMP_DEPT
Ename
Ssn
Bdate
Address
Dnumber
Dname
Dmgr_ssn
3NF Normalization
ED1
Ename
Ssn
Bdate
Address
Dnumber
ED2
Dnumber
Dname
Dmgr_ssn

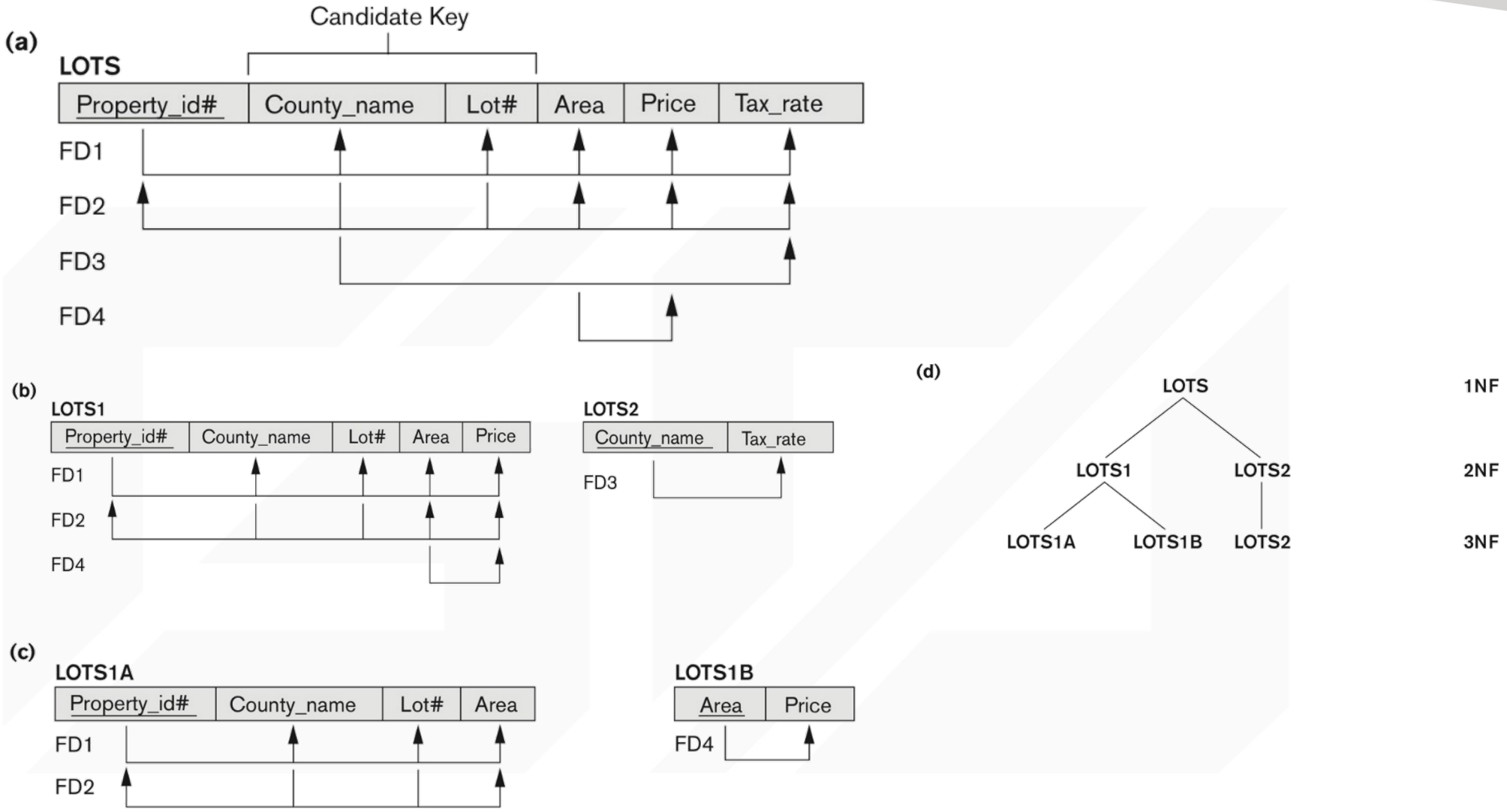


Figure 14.12
Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.

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**

4. General Normal Form Definitions (For Multiple Keys)

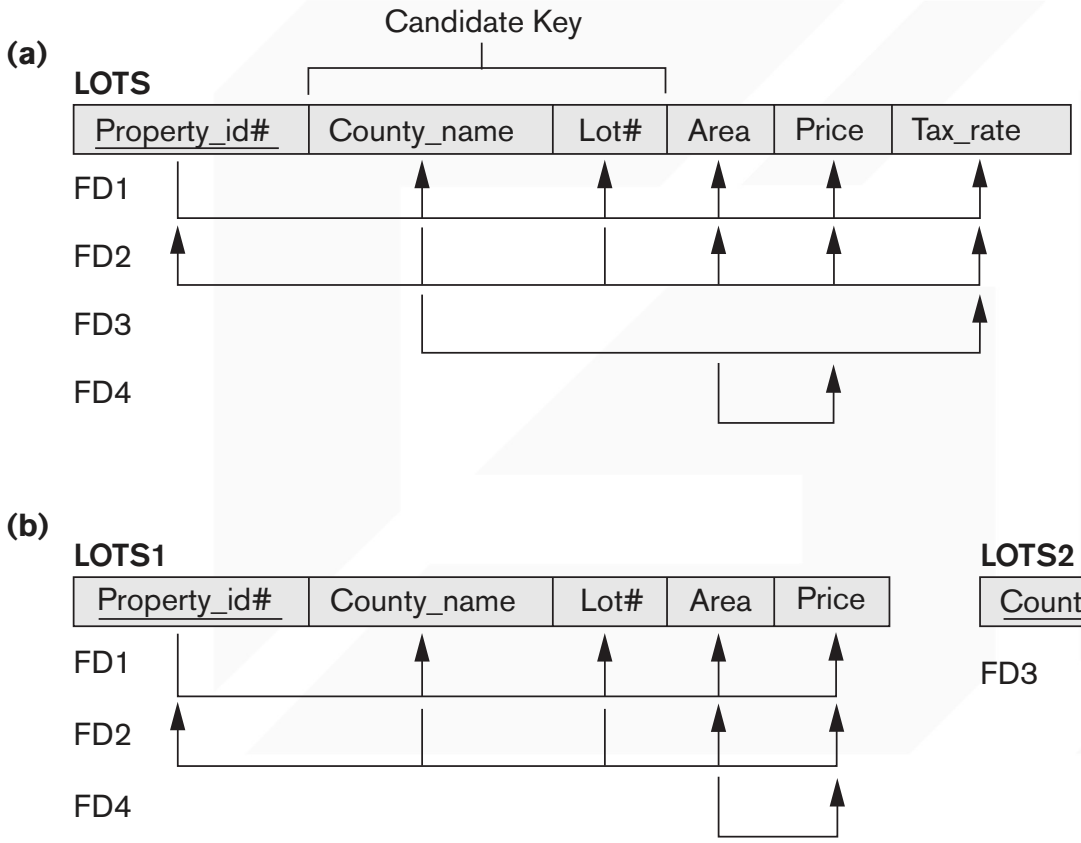
- The above definitions consider the primary key only
- The following more general definitions take into account relations with multiple candidate keys
- Any attribute involved in a candidate key is a *prime attribute*
- All other attributes are called *non-prime attributes*.

4.1 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 → Tax_rate violates 2NF.



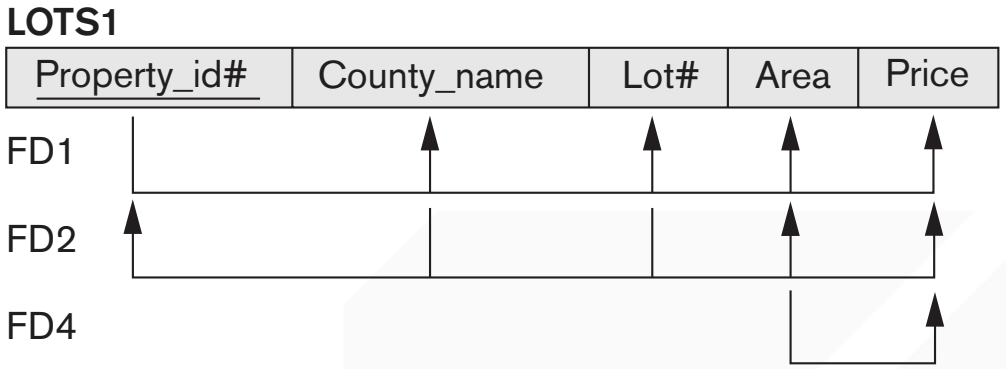
So second normalization converts LOTS into

LOTS1 (Property_id#, County_name, Lot#, Area, Price)

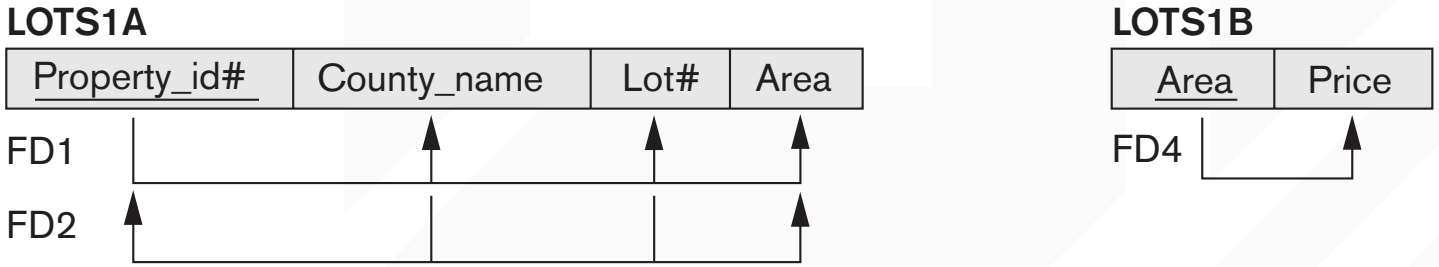
LOTS2 (County_name, Tax_rate)

4.2 General Definition of Third Normal Form

- 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



- LOTS1 relation violates 3NF because Area → Price ;
  and Area is not a superkey in LOTS1.
  (see Figure 14.12).



4.3 Interpreting the General Definition of Third Normal Form

- Consider the 2 conditions in the Definition of 3NF:

  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

- 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 previous slide takes care of the dependencies that "slip through" (are allowable to) 3NF but are "caught by" BCNF which we discuss next.

5. BCNF (Boyce-Codd Normal Form)

- A relation schema R is in **Boyce-Codd Normal Form (BCNF)** if whenever a ***nontrivial functional dependency*** **X** ⟶ **A** holds in R, then **X is a superkey of R**

The trivial dependency is a set of attributes which are called a trivial if the set of attributes are included in that attribute.

So, X -> Y is a trivial functional dependency if Y is a subset of X.

Functional dependency which also known as a nontrivial dependency occurs when A->B holds true where B is not a subset of A.

In a relationship, if attribute B is not a subset of attribute A, then it is considered as a non-trivial dependency.

- 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)

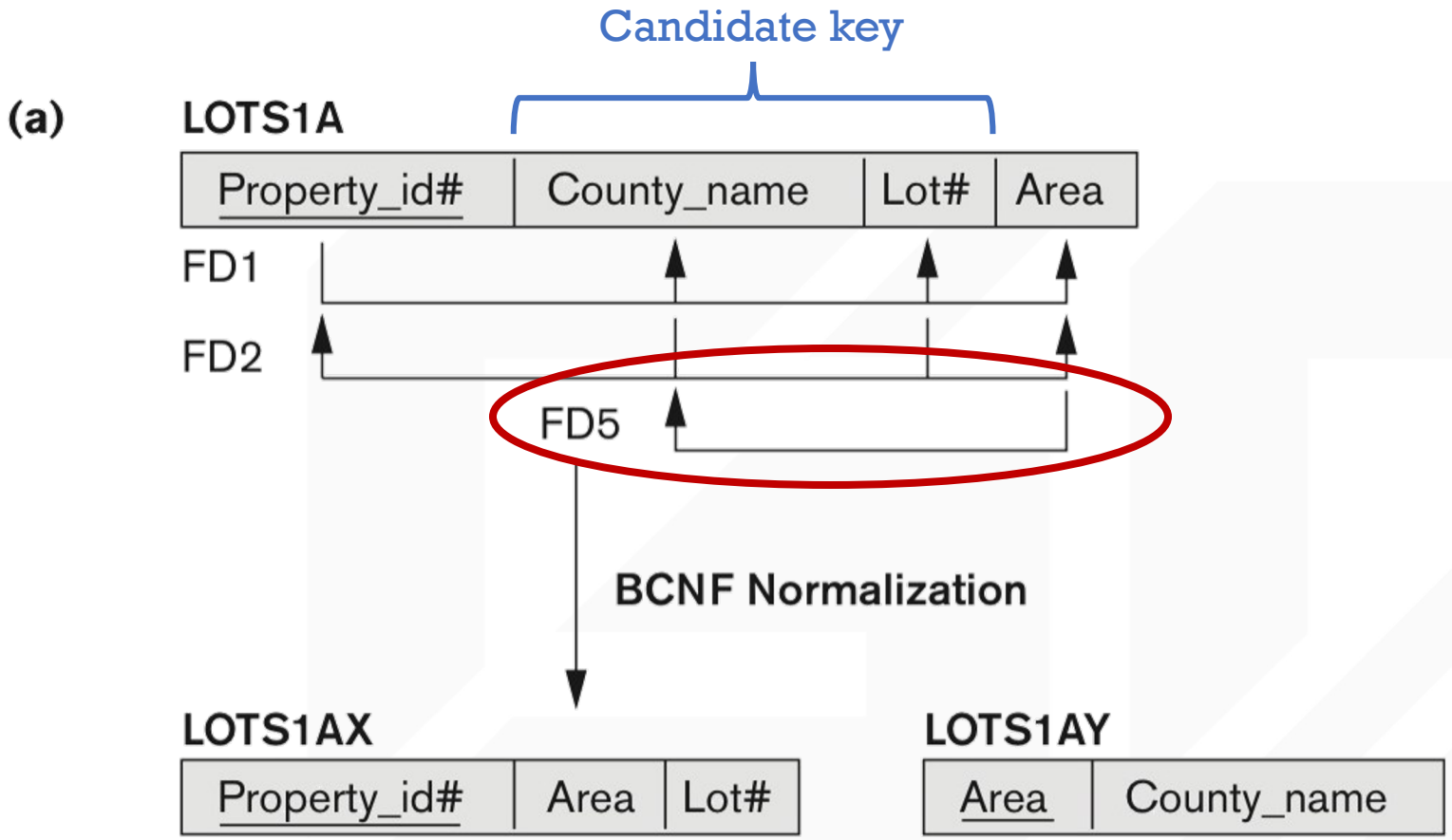
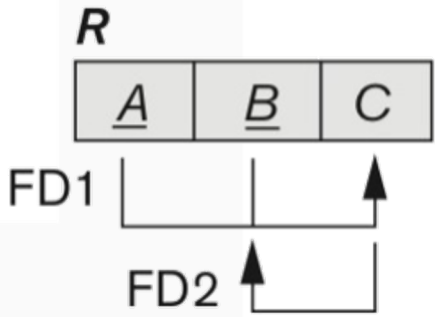
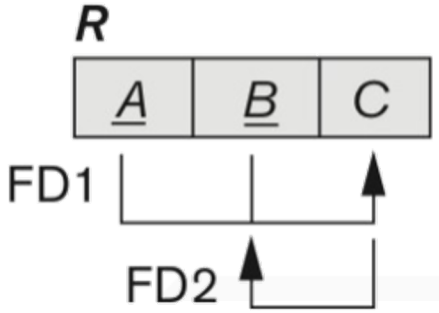


Figure 14.13
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. C → B.



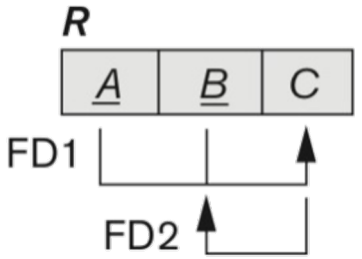


TEACH

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

Figure 14.14
A relation TEACH that is in 3NF but not BCNF.

- 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.



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

Figure 14.14 A relation TEACH that is in 3NF but not BCNF.

- 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).

- A test to determine whether a binary decomposition (decomposition into two relations) is non-additive (lossless) is discussed under Property NJB on the next slide.

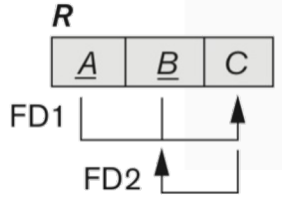
Test for checking non-additivity of Binary Relational Decompositions

- **Testing Binary Decompositions for Lossless Join (Non-additive Join) Property**
  - **Binary Decomposition:**
    Decomposition of a relation R into two relations.
  - **Property NJB (non-additive join test for binary decompositions):**
    A decomposition D = $\{R_1, R_2\}$ of R has the lossless join property with respect to a set of functional dependencies F on R *if and only if* either
    - The f.d. $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ is in F^+, or
    - The f.d. $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ is in F^+.

Note: F^+ is the (complete) set of all dependencies (functional or multivalued) that will hold in every relation state r of R that satisfies F. It is also called the **closure** of F.

- **PROPERTY NJB (non-additive join test for binary decompositions):** A decomposition $D = \{R_1, R_2\}$ of R has the lossless join property with respect to a set of functional dependencies F on R *if and only if* either
  - The f.d. $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ is in F^+, or
  - The f.d. $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ is in F^+.

- Three possible decompositions for relation TEACH
  - D1: {student, instructor} and {student, course}
  - D2: {course, instructor } and {course, student}
  - D3: {instructor, course } and {instructor, student} ✓



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

Figure 14.14 A relation TEACH that is in 3NF but not BCNF.

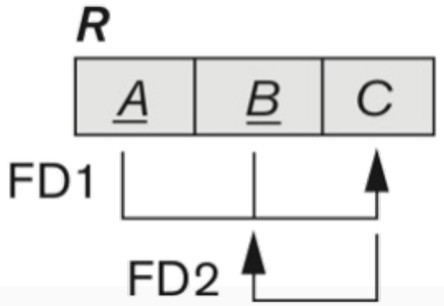
- If you apply the NJB test to the 3 decompositions of the TEACH relation:
  - D1 gives Student → Instructor or Student → Course, none of which is true.
  - D2 gives Course → Instructor or Course → Student, none of which is true.
  - However, in D3 we get Instructor → Course or Instructor → Student.
- Since Instructor → Course is indeed true, the NJB property is satisfied and D3 is determined as a non-additive (good) decomposition.

General Procedure for achieving BCNF when a relation fails BCNF

- Here we make use the algorithm from Chapter 15 (Algorithm 15.5):
  - 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.

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman



- Note that the f.d. that violated BCNF in TEACH was Instructor → Course. Hence its BCNF decomposition would be :
  - (TEACH – COURSE) and (Instructor ∪ Course), which gives
  - the relations: (Instructor, Student) and (Instructor, Course) that we obtained before in decomposition D3.

6. Multivalued Dependencies and Fourth Normal Form

(a) EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

- A tuple in this EMP relation represents the fact
  - An employee whose name is Ename works on the project whose name is Pname and has a dependent whose name is Dname.
  - An employee may work on several projects and may have several dependents, and the employee's projects and dependents are independent of one another.
- In the relation state shown in Figure 14.15(a), the employee with Ename Smith works on two projects 'X' and 'Y' and has two dependents 'John' and 'Anna', and therefore there are four tuples to represent these facts together.

(a) EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

- The relation EMP is an all-key relation (with key made up of all attributes) and therefore has no f.d.'s and as such qualifies to be a BCNF relation.
- There is an obvious redundancy in the relation EMP—the dependent information is repeated for every project and the project information is repeated for every dependent.
- The concept of multivalued dependency (MVD) was proposed and, based on this dependency, the fourth normal form was defined.

Definition

- A **multivalued dependency (MVD)** $X \twoheadrightarrow Y$ specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation state r of R:
  - If two tuples t_1 and t_2 exist in r such that $t_1[X] = t_2[X]$, then two tuples t_3 and t_4 should also exist in r with the following properties, where we use Z to denote $(R - (X \cup Y))$:
    - $t_3[X] = t_4[X] = t_1[X] = t_2[X]$.
    - $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$.
    - $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$.

- Whenever X ↠ Y holds, we say that X **multidetermines** Y.
- Because of the symmetry in the definition, whenever X ↠ Y holds in R, so does X ↠ Z. Hence, X ↠ Y implies X ↠ Z and therefore it is sometimes written as X ↠ Y | Z.
- An MVD X ↠ Y in R is called a **trivial MVD** if
  a) Y is a subset of X, or
  b) X ∪ Y = R
- For example, the relation EMP_PROJECTS has the trivial MVD Ename ↠ Pname and the relation EMP_DEPENDENTS has the trivial MVD Ename ↠ Dname.
- An MVD that satisfies neither (a) nor (b) is called a **nontrivial MVD**.

- Note that an MVD is represented by the symbol ↠ or →→.

Definition

- A relation schema R is in **4NF** with respect to a set of dependencies F (that includes functional dependencies and multivalued dependencies) if,
  for every *nontrivial* multivalued dependency $X \twoheadrightarrow Y$ in F^+,
  X is a superkey for R.

- **Note:**
  F^+ is the (complete) set of all dependencies (functional or multivalued) that will hold in every relation state r of R that satisfies F. It is also called the **closure** of F.

- We can state the following points:
  - An all-key relation is always in BCNF since it has no FDs.
  - An all-key relation such as the EMP relation in Figure 14.15(a), which has no FDs but has the MVD Ename ↠ Pname | Dname, is not in 4NF.
  - A relation that is not in 4NF due to a nontrivial MVD must be decomposed to convert it into a set of relations in 4NF.
  - The decomposition removes the redundancy caused by the MVD.

(a) EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

(b) EMP_PROJECTS

Ename	Pname
Smith	X
Smith	Y

EMP_DEPENDENTS

Ename	Dname
Smith	John
Smith	Anna

- The non-trivial MVD Ename ↠ Pname | Dname is decomposed to two trivial MVD Ename ↠ Pname and Ename ↠ Dname.

Figure 14.15
Fourth and fifth normal forms.
(a) The EMP relation with two MVDs: Ename –>> Pname and Ename –>> Dname.
(b) Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS.

7. Join Dependencies and Fifth Normal Form

- In some cases there may be no nonadditive join decomposition of *R* into two relation schemas, but there may be a nonadditive join decomposition into more than two relation schemas.
- Moreover, there may be no functional dependency in *R* that violates any normal form up to BCNF,
  and there may be no nontrivial MVD present in *R* either that violates 4NF.

Definition

- A **join dependency (JD),** denoted by JD($R_1, R_2, \ldots, R_n$), specified on relation schema R, specifies a constraint on the states r of R.

  The constraint states that every legal state r of R should have a nonadditive join decomposition into $R_1, R_2, \ldots, R_n$.

- Hence, for every such r we have

$$*\left(\pi_{R_1}(r), \pi_{R_2}(r), \ldots, \pi_{R_n}(r)\right) = r$$

Notice

- An MVD is a special case of a JD where n = 2.
  That is, a JD denoted as JD(R_1 , R_2) implies
  an MVD $(R_1 \cap R_2) \twoheadrightarrow (R_1 - R_2)$
  (or, by symmetry, $(R_1 \cap R_2) \twoheadrightarrow (R_2 - R_1)$).
- A join dependency JD(R_1 , R_2 , … , R_n), specified on relation schema R, is a trivial JD if one of the relation schemas Ri in JD(R_1 , R_2 , … , R_n) is equal to R.

Definition

- A relation schema R is in **fifth normal form (5NF)** (or project-join normal form (PJNF)) with respect to a set F of functional, multivalued, and join dependencies if, for every nontrivial join dependency JD(R_1 , R_2 , … , R_n) in F^+ (that is, implied by F), every R_i is a superkey of R.

- Discovering join dependencies in practical databases with hundreds of relations is next to impossible. Therefore, 5NF is rarely used in practice.

(c) **SUPPLY**

Sname	Part_name	Proj_name
Smith	Bolt	ProjX
Smith	Nut	ProjY
Adamsky	Bolt	ProjY
Walton	Nut	ProjZ
Adamsky	Nail	ProjX
Adamsky	Bolt	ProjX
Smith	Bolt	ProjY

(d) R_1

Sname	Part_name
Smith	Bolt
Smith	Nut
Adamsky	Bolt
Walton	Nut
Adamsky	Nail

R_2

Sname	Proj_name
Smith	ProjX
Smith	ProjY
Adamsky	ProjY
Walton	ProjZ
Adamsky	ProjX

R_3

Part_name	Proj_name
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Figure 14.15
(c) The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R1, R2, R3).
(d) Decomposing the relation SUPPLY into the 5NF relations R1, R2, R3.

- Suppose that the following additional constraint always holds:
  - Whenever a supplier *s* supplies part *p*, and a project *j* uses part *p*, and the supplier *s* supplies at least one part to project *j*, then supplier *s* will also be supplying part *p* to project *j*.
- This constraint can be restated in other ways and specifies a join dependency JD(R_1 , R_2 , R_3) among the three projections R_1 (Sname, Part_name), R_2 (Sname, Proj_name), and R_3 (Part_name, Proj_name) of SUPPLY.

(c) **SUPPLY**

Sname	Part_name	Proj_name
Smith	Bolt	ProjX
Smith	Nut	ProjY
Adamsky	Bolt	ProjY
Walton	Nut	ProjZ
Adamsky	Nail	ProjX
Adamsky	Bolt	ProjX
Smith	Bolt	ProjY

(d) R_1

Sname	Part_name
Smith	Bolt
Smith	Nut
Adamsky	Bolt
Walton	Nut
Adamsky	Nail

R_2

Sname	Proj_name
Smith	ProjX
Smith	ProjY
Adamsky	ProjY
Walton	ProjZ
Adamsky	ProjX

R_3

Part_name	Proj_name
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Figure 14.15
(c) The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R1, R2, R3).
(d) Decomposing the relation SUPPLY into the 5NF relations R1, R2, R3.

- Notice that applying a natural join to any two of these relations produces spurious tuples but applying a natural join to all three together does not.

Summary

- BCNF (Boyce-Codd Normal Form)
- Fourth and Fifth Normal Forms