



CHAPTER 14

Basics of Functional Dependencies and Normalization for Relational Databases

Chapter Outline

- 1 Informal Design Guidelines for Relational Databases
 - 1.1 Semantics 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 (FDs)
 - 2.1 Definition of Functional Dependency

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 2NF and 3NF (For Multiple Candidate Keys)
- 5 BCNF (Boyce-Codd Normal Form)

Chapter Outline

- 6 Multivalued Dependency and Fourth Normal Form
- 7 Join Dependencies and Fifth Normal Form

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?

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 15

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

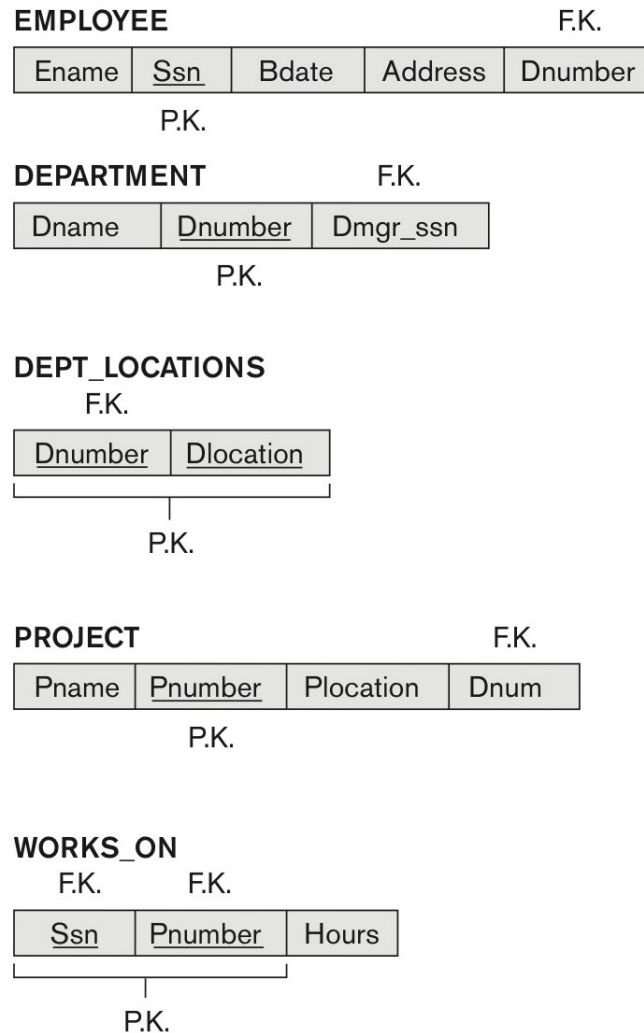


Figure 14.1 A simplified COMPANY relational database schema.

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.

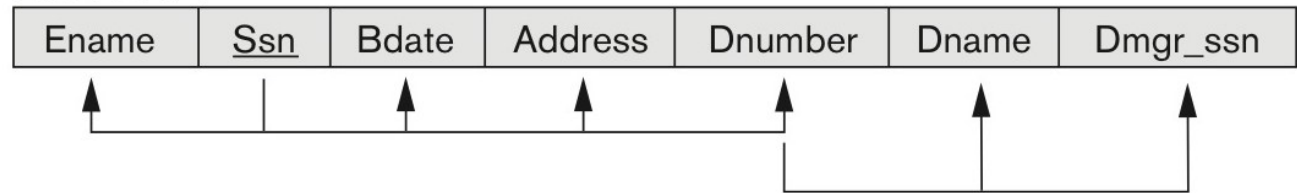
Figure 14.3 Two relation schemas suffering from update anomalies

Figure 14.3

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

(a)

EMP_DEPT



(b)

EMP_PROJ

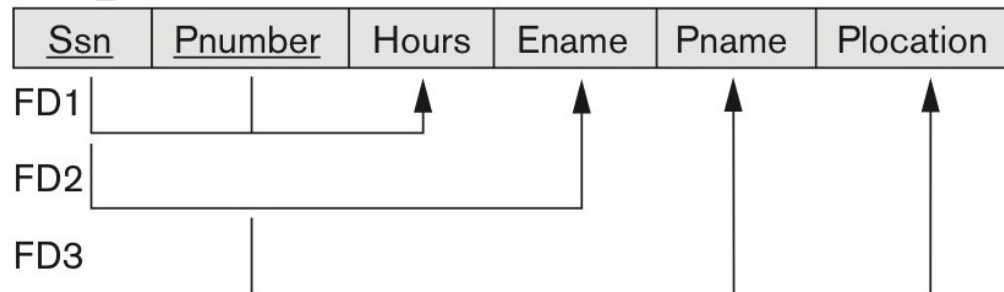


Figure 14.4 Sample states for EMP_DEPT and EMP_PROJ

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.

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

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

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
- 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 (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 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

- $X \rightarrow Y$ holds if whenever two tuples have the same value for X , they *must have* the same value for Y
 - For any two tuples $t1$ and $t2$ in any relation instance $r(R)$: If $t1[X]=t2[X]$, *then* $t1[Y]=t2[Y]$
- $X \rightarrow Y$ 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 (1)

- Social security number determines employee name
 - $SSN \rightarrow ENAME$
- Project number determines project name and location
 - $PNUMBER \rightarrow \{PNAME, PLOCATION\}$
- Employee ssn and project number determines the hours per week that the employee works on the project
 - $\{SSN, PNUMBER\} \rightarrow HOURS$

Examples of FD constraints (2)

- 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 $t1[K]=t2[K]$)

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

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 (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

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
- 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 (1)

- 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.

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.

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

Dname	<u>Dnumber</u>	Dmgr_ssn	Dlocations




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.

(b)

DEPARTMENT

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

(c)

DEPARTMENT

Dname	<u>Dnumber</u>	Dmgr_ssn	<u>Dlocation</u>
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)

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 (1)

- 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

Second Normal Form (2)

- 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”

Figure 14.11 Normalizing into 2NF and 3NF

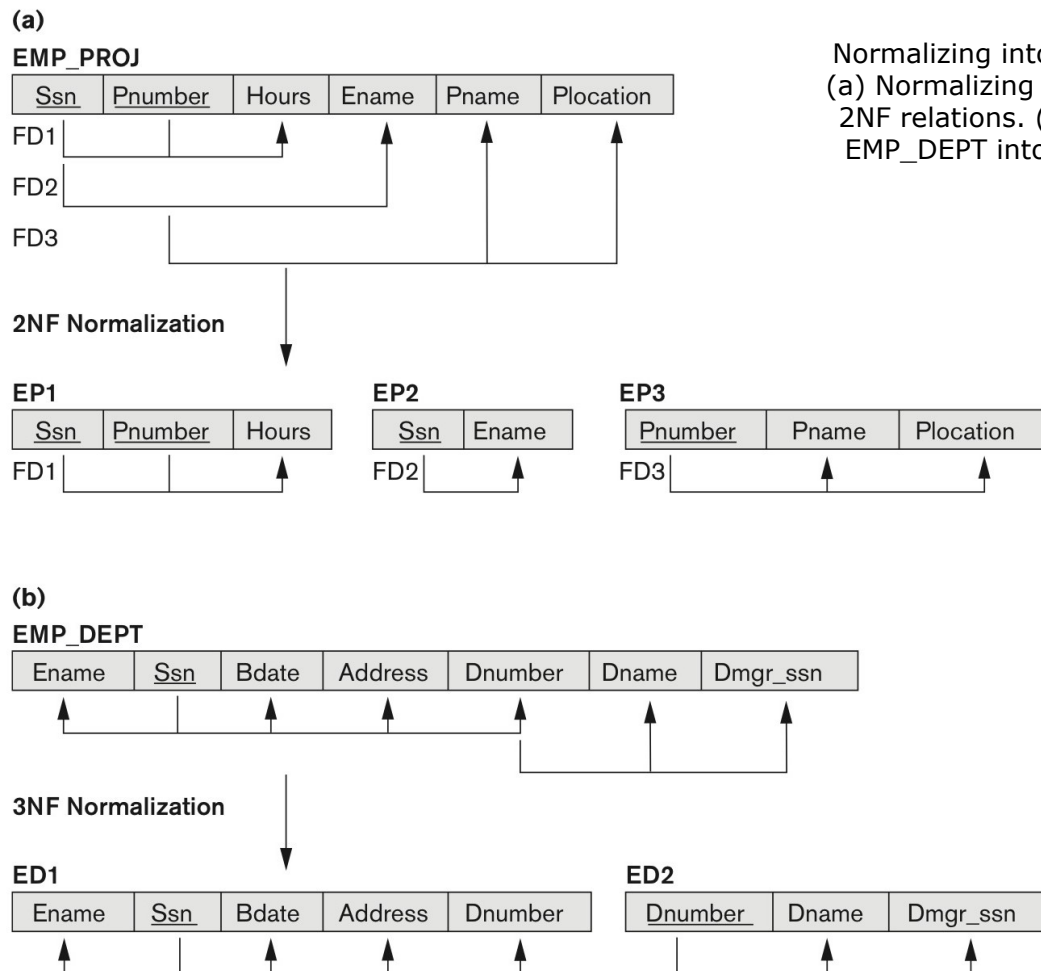
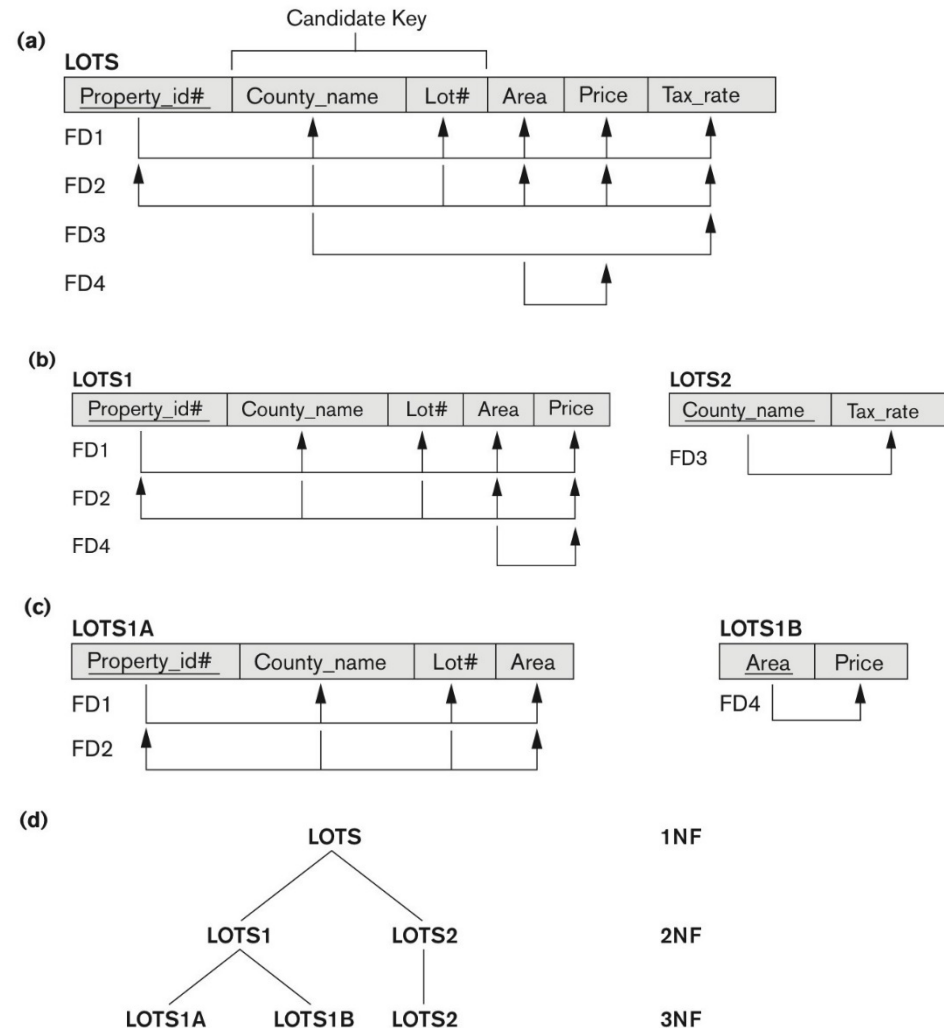


Figure 14.11
Normalizing into 2NF and 3NF.
(a) Normalizing EMP_PROJ into 2NF relations. (b) Normalizing EMP_DEPT into 3NF relations.

Figure 14.12 Normalization into 2NF and 3NF

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.



3.6 Third Normal Form (1)

- 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 $X \rightarrow ENAME$

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 $X \rightarrow Y$ 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.

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**

5. Multivalued Dependencies and Fourth Normal Form (1)

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 $t_4[Z] = t_1[Z]$.
- An MVD $X \twoheadrightarrow Y$ in R is called a **trivial MVD** if (a) Y is a subset of X , or (b) $X \cup Y = R$.

Multivalued Dependencies and Fourth Normal Form (3)

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 .

Figure 14.15 Fourth and fifth normal forms.

(a) EMP

<u>Ename</u>	<u>Pname</u>	<u>Dname</u>
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

(c) SUPPLY

<u>Sname</u>	<u>Part_name</u>	<u>Proj_name</u>
Smith	Bolt	ProjX
Smith	Nut	ProjY
Adamsky	Bolt	ProjY
Walton	Nut	ProjZ
Adamsky	Nail	ProjX
Adamsky	Bolt	ProjX
Smith	Bolt	ProjY

(b) EMP_PROJECTS

<u>Ename</u>	<u>Pname</u>
Smith	X
Smith	Y

EMP_DEPENDENTS

<u>Ename</u>	<u>Dname</u>
Smith	John
Smith	Anna

(d) R_1

<u>Sname</u>	<u>Part_name</u>
Smith	Bolt
Smith	Nut
Adamsky	Bolt
Walton	Nut
Adamsky	Nail

R_2

<u>Sname</u>	<u>Proj_name</u>
Smith	ProjX
Smith	ProjY
Adamsky	ProjY
Walton	ProjZ
Adamsky	ProjX

R_3

<u>Part_name</u>	<u>Proj_name</u>
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Figure 14.15

Fourth and fifth normal forms. (a) The EMP relation with two MVDs: Ename \twoheadrightarrow Pname and Ename \twoheadrightarrow Dname. (b) Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS. (c) The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R_1, R_2, R_3). (d) Decomposing the relation SUPPLY into the 5NF relations R_1, R_2, R_3 .

6. Join Dependencies and Fifth Normal Form (1)

Definition:

- A **join dependency (JD)**, denoted by $JD(R_1, R_2, \dots, R_n)$, 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 non-additive join decomposition into R_1, R_2, \dots, R_n ; that is, for every such r we have
- $$* (\pi_{R_1}(r), \pi_{R_2}(r), \dots, \pi_{R_n}(r)) = r$$

Note: an MVD is a special case of a JD where $n = 2$.

- A join dependency $JD(R_1, R_2, \dots, R_n)$, specified on relation schema R , is a **trivial JD** if one of the relation schemas R_i in $JD(R_1, R_2, \dots, R_n)$ is equal to R .

Join Dependencies and Fifth Normal Form (2)

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 $JD(R_1, R_2, \dots, R_n)$ 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.

Chapter Summary

- Informal Design Guidelines for Relational Databases
- Functional Dependencies (FDs)
- Normal Forms (1NF, 2NF, 3NF) Based on Primary Keys
- General Normal Form Definitions of 2NF and 3NF (For Multiple Keys)
- BCNF (Boyce-Codd Normal Form)
- Fourth and Fifth Normal Forms