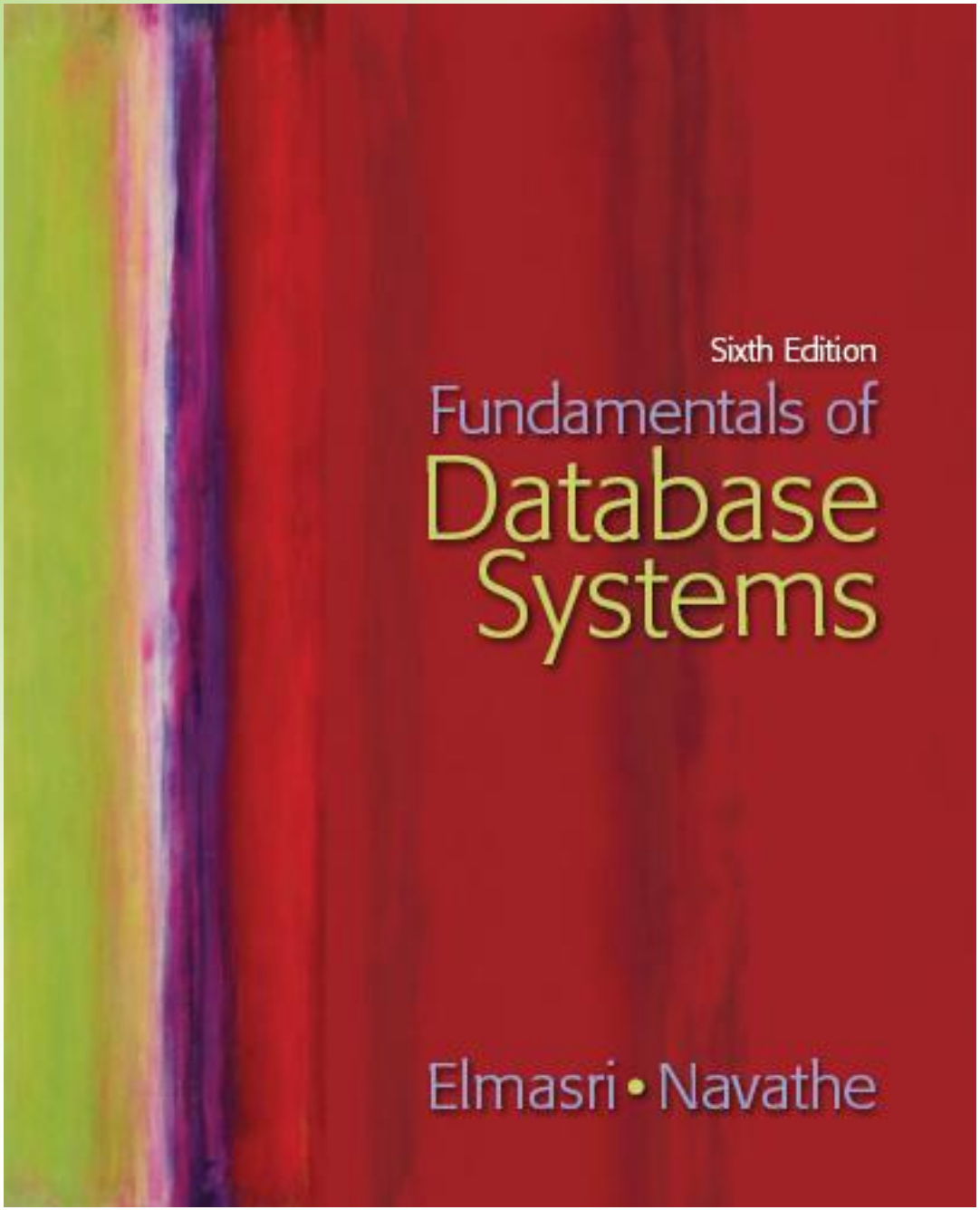
Chapter 15

Basics of Functional Dependencies and Normalization for Relational Databases



Addison-Wesley
is an imprint of
PEARSON

Chapter 15 Outline

- Informal Design Guidelines for Relation Schemas
- Functional Dependencies
- Normal Forms Based on Primary Keys
- General Definitions of Second and Third Normal Forms
- Boyce-Codd Normal Form

Chapter 15 Outline (cont'd.)

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

Introduction

- Levels at which we can discuss *goodness* of relation schemas
  - Logical (or conceptual) level
  - Implementation (or physical storage) level
- Approaches to database design:
  - Bottom-up or top-down

Informal Design Guidelines for Relation Schemas

- Measures of quality
  - Making sure attribute semantics are clear
  - Reducing redundant information in tuples
  - Reducing NULL values in tuples
  - Disallowing possibility of generating spurious tuples

Imparting Clear Semantics to Attributes in Relations

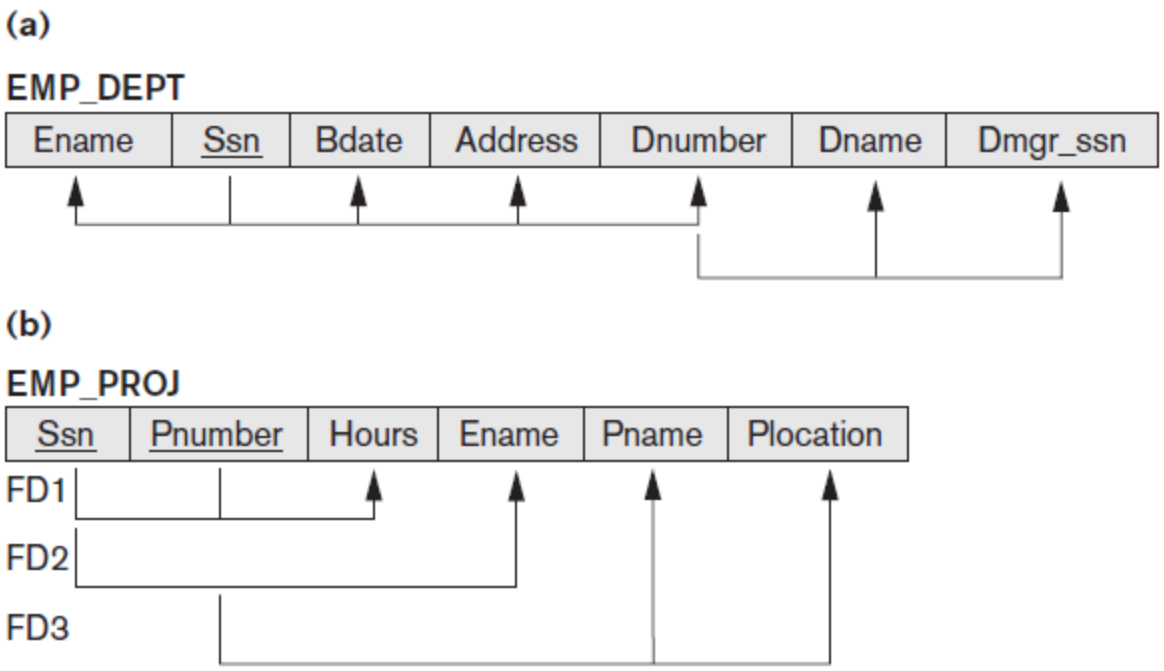
- Semantics of a relation
  - Meaning resulting from interpretation of attribute values in a tuple
- Easier to explain semantics of relation
  - Indicates better schema design

Guideline 1

- Design relation schema so that it is easy to explain its meaning
- Do not combine attributes from multiple entity types and relationship types into a single relation
- Example of violating Guideline 1: Figure 15.3

Guideline 1 (cont'd.)

Figure 15.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.



Redundant Information in Tuples and Update Anomalies

- Grouping attributes into relation schemas
  - Significant effect on storage space
- Storing natural joins of base relations leads to **update anomalies**
- Types of update anomalies:
  - Insertion
  - Deletion
  - Modification

Guideline 2

- Design base relation schemas so that no update anomalies are present in the relations
- If any anomalies are present:
  - Note them clearly
  - Make sure that the programs that update the database will operate correctly

NULL Values in Tuples

- May group many attributes together into a "fat" relation
  - Can end up with many NULLs
- Problems with NULLs
  - Wasted storage space
  - Problems understanding meaning

Guideline 3

- Avoid placing attributes in a base relation whose values may frequently be NULL
- If NULLs are unavoidable:
  - Make sure that they apply in exceptional cases only, not to a majority of tuples

Generation of Spurious Tuples

- Figure 15.5(a)
  - Relation schemas EMP_LOCS and EMP_PROJ1
- NATURAL JOIN
  - Result produces many more tuples than the original set of tuples in EMP_PROJ
  - Called **spurious tuples**
  - Represent spurious information that is not valid

Guideline 4

- Design relation schemas to be joined with equality conditions on attributes that are appropriately related
  - Guarantees that no spurious tuples are generated
- Avoid relations that contain matching attributes that are not (foreign key, primary key) combinations

Summary and Discussion of Design Guidelines

- Anomalies cause redundant work to be done
- Waste of storage space due to NULLs
- Difficulty of performing operations and joins due to NULL values
- Generation of invalid and spurious data during joins

Functional Dependencies

- Formal tool for analysis of relational schemas
- Enables us to detect and describe some of the above-mentioned problems in precise terms
- Theory of functional dependency

Definition of Functional Dependency

- Constraint between two sets of attributes from the database

Definition. A **functional dependency**, denoted by $X \rightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a *constraint* on the possible tuples that can form a relation state r of R. The constraint is that, for any two tuples t_1 and t_2 in r that have $t_1[X] = t_2[X]$, they must also have $t_1[Y] = t_2[Y]$.

- Property of semantics or meaning of the attributes
- **Legal relation states**
  - Satisfy the functional dependency constraints

Definition of Functional Dependency (cont'd.)

- Given a populated relation
  - Cannot determine which FDs hold and which do not
  - Unless meaning of and relationships among attributes known
  - Can state that FD does not hold if there are tuples that show violation of such an FD

Normal Forms Based on Primary Keys

- Normalization process
- Approaches for relational schema design
  - Perform a conceptual schema design using a conceptual model then map conceptual design into a set of relations
  - Design relations based on external knowledge derived from existing implementation of files or forms or reports

Normalization of Relations

- Takes a relation schema through a series of tests
  - Certify whether it satisfies a certain normal form
  - Proceeds in a top-down fashion
- **Normal form tests**

Definition. The **normal form** of a relation refers to the highest normal form condition that it meets, and hence indicates the degree to which it has been normalized.

Normalization of Relations (cont'd.)

- Properties that the relational schemas should have:
  - **Nonadditive join property**
    - Extremely critical
  - **Dependency preservation property**
    - Desirable but sometimes sacrificed for other factors

Practical Use of Normal Forms

- Normalization carried out in practice
  - Resulting designs are of high quality and meet the desirable properties stated previously
  - Pays particular attention to normalization only up to 3NF, BCNF, or at most 4NF
- Do not need to normalize to the highest possible normal form

Definition. Denormalization is 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

- Definition of **superkey** and **key**
- **Candidate key**
  - If more than one key in a relation schema
    - One is **primary key**
    - Others are **secondary keys**

Definition. An attribute of relation schema R is called a **prime attribute** of R if it is a member of *some candidate key* of R. An attribute is called **nonprime** if it is not a prime attribute—that is, if it is not a member of any candidate key.

First Normal Form

- Part of the formal definition of a relation in the basic (flat) relational model
- Only attribute values permitted are single **atomic (or indivisible) values**
- Techniques to achieve first normal form
  - Remove attribute and place in separate relation
  - Expand the key
  - Use several atomic attributes

First Normal Form (cont'd.)

- Does not allow **nested relations**
  - Each tuple can have a relation within it
- To change to 1NF:
  - Remove nested relation attributes into a new relation
  - Propagate the primary key into it
  - **Unnest** relation into a set of 1NF relations

(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 15.9
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.

Second Normal Form

- Based on concept of **full functional dependency**
  - Versus **partial dependency**

Definition. A relation schema R is in 2NF if every nonprime attribute A in R is *fully functionally dependent* on the primary key of R.

- Second normalize into a number of 2NF relations
  - Nonprime attributes are associated only with part of primary key on which they are fully functionally dependent

Third Normal Form

- Based on concept of transitive dependency

Definition. According to Codd's original definition, a relation schema R is in **3NF** if it satisfies 2NF *and* no nonprime attribute of R is transitively dependent on the primary key.

- Problematic FD
  - Left-hand side is part of primary key
  - Left-hand side is a nonkey attribute

General Definitions of Second and Third Normal Forms

Table 15.1 Summary of Normal Forms Based on Primary Keys and Corresponding Normalization

Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations.	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally dependent on a part of the primary key.	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it.
Third (3NF)	Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

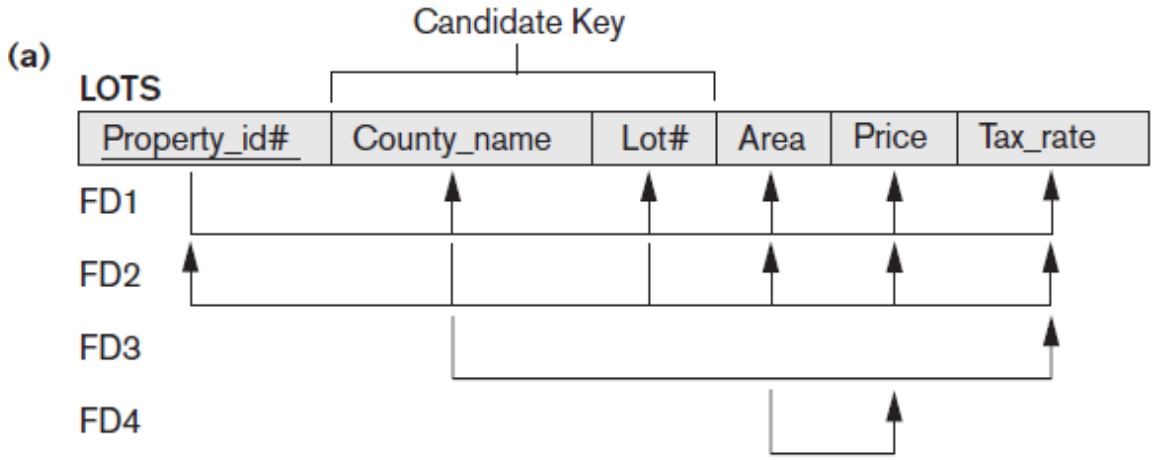
General Definitions of Second and Third Normal Forms (cont'd.)

- **Prime attribute**
  - Part of any candidate key will be considered as prime
- Consider partial, full functional, and transitive dependencies with respect to all candidate keys of a relation

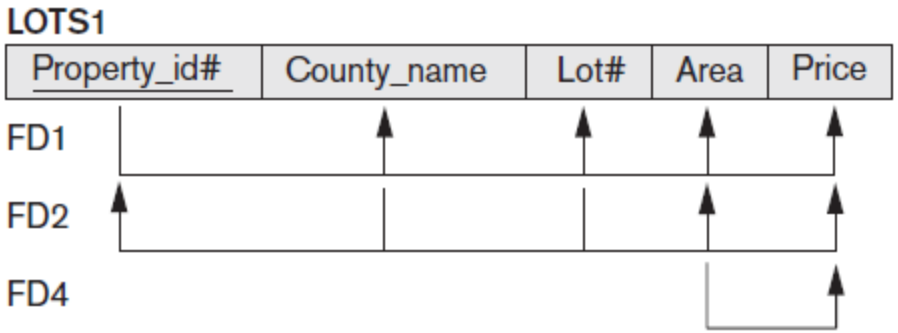
General Definition of Second Normal Form

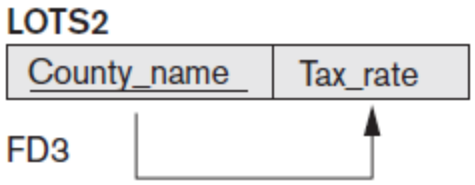
Definition. A relation schema R is in **second normal form** (2NF) if every non-prime attribute A in R is not partially dependent on *any* key of R.[11]

Figure 15.12
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) Summary of the progressive normalization of LOTS.

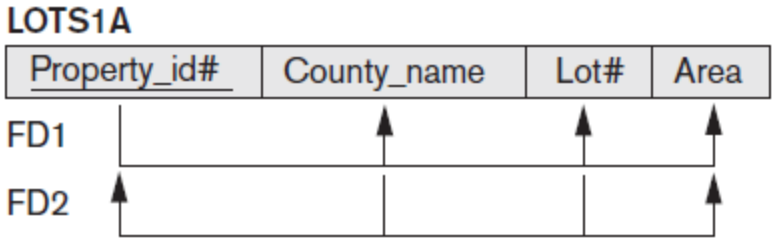


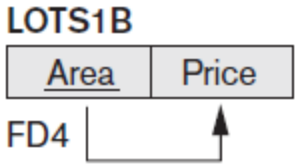
(b)



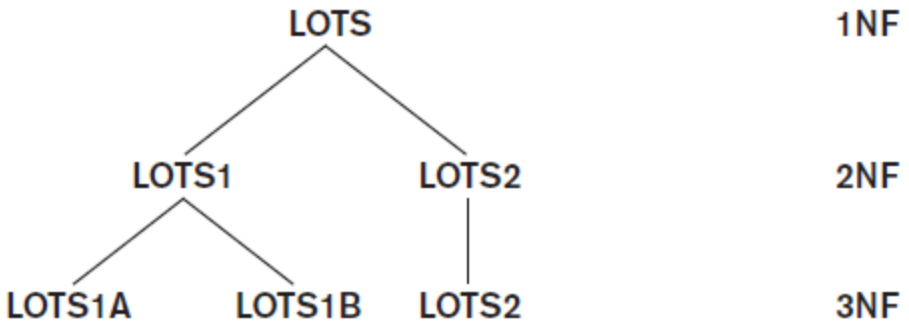


(c)





(d)



General Definition of Third Normal Form

Definition. A relation schema R is in **third normal form** (3NF) if, whenever a *nontrivial* functional dependency $X \rightarrow A$ holds in R, either (a) X is a superkey of R, or (b) A is a prime attribute of R.

Alternative Definition. A relation schema R is in 3NF if every nonprime attribute of R meets both of the following conditions:

- It is fully functionally dependent on every key of R.
- It is nontransitively dependent on every key of R.

Boyce-Codd Normal Form

- Every relation in BCNF is also in 3NF
  - Relation in 3NF is not necessarily in BCNF

Definition. A relation schema R is in **BCNF** if whenever a *nontrivial* functional dependency $X \rightarrow A$ holds in R, then X is a superkey of R.

- Difference:
  - Condition which allows A to be prime is absent from BCNF
- Most relation schemas that are in 3NF are also in BCNF

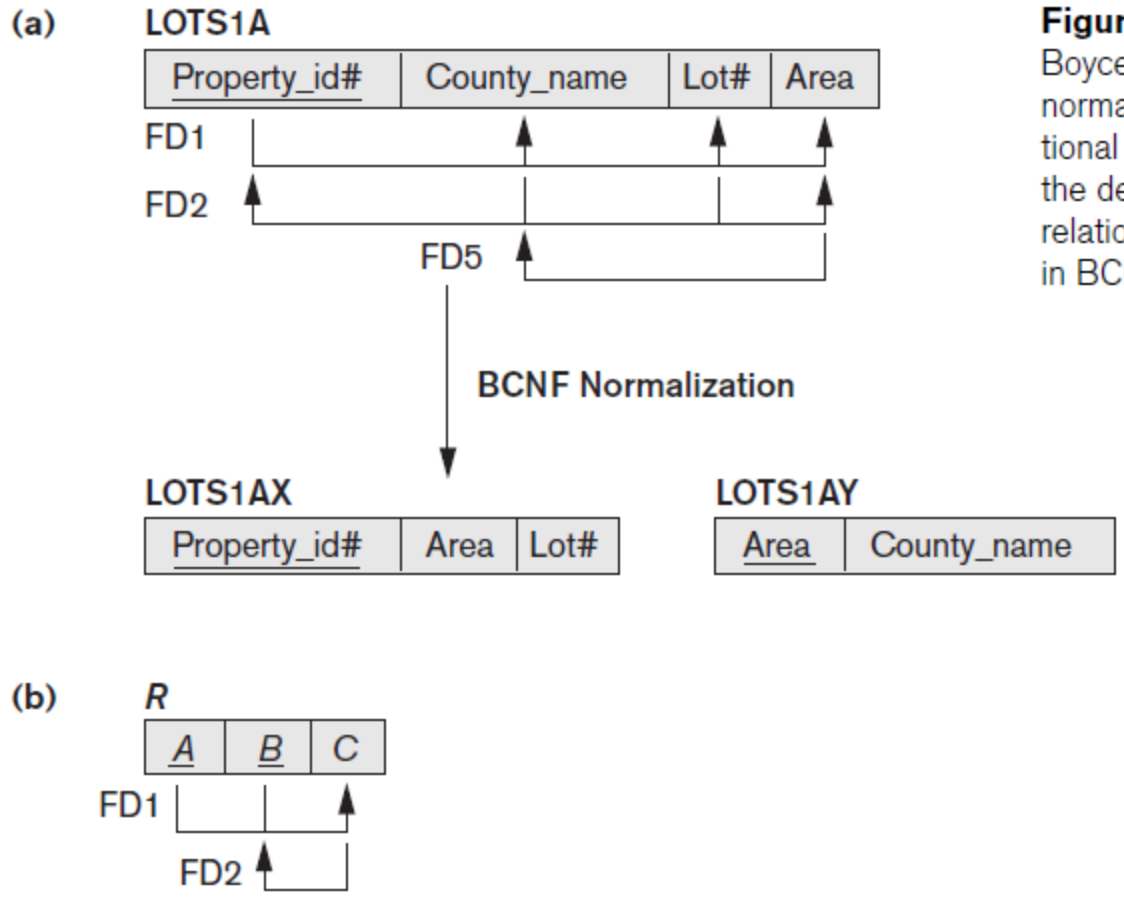


Figure 15.13
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF.

Multivalued Dependency and Fourth Normal Form

- Multivalued dependency (MVD)
  - Consequence of first normal form (1NF)

Definition. A multivalued dependency $X \twoheadrightarrow Y$ 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,[15] where we use Z to denote $(R - (X \cup Y))$:[16]

- $t_3[X] = t_4[X] = t_1[X] = t_2[X]$.
- $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$.
- $t_3[Z] = t_2[Z]$ and $t_4[Z] = t_1[Z]$.

Multivalued Dependency and Fourth Normal Form (cont'd.)

- Relations containing nontrivial MVDs
  - **All-key relations**
- **Fourth normal form (4NF)**
  - Violated when a relation has undesirable multivalued dependencies

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^{+17} X is a superkey for R.

Join Dependencies and Fifth Normal Form

- **Join dependency**
- Multiway decomposition into fifth normal form (5NF)
- Very peculiar semantic constraint
  - Normalization into 5NF is very rarely done in practice

Join Dependencies and Fifth Normal Form (cont'd.)

Definition. A **join dependency** (**JD**), denoted by JD(R_1, R_2, ..., 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, ..., R_n. Hence, for every such r we have

* $(\pi_{R_1}(r), \pi_{R_2}(r), ..., \pi_{R_n}(r)) = r$

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),[18] every R_i is a superkey of R.

Summary

- Informal guidelines for good design
- Functional dependency
  - Basic tool for analyzing relational schemas
- Normalization:
  - 1NF, 2NF, 3NF, BCNF, 4NF, 5NF

15.6 Multivalued Dependency and Fourth Normal Form

unnormalized relation into 1NF. If we have two or more multivalued *independent* attributes in the same relation schema, we get into a problem of having to repeat every value of one of the attributes with every value of the other attribute to keep the relation state consistent and to maintain the independence among the attributes involved. This constraint is specified by a multivalued dependency.

15.6 Multivalued Dependency and Fourth Normal Form

For example, consider the relation EMP shown in Figure 15.15(a). A tuple in this EMP relation represents the fact that 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.[13] To keep the relation state consistent, and to avoid any spurious relationship between the two independent attributes, we must have a separate tuple to represent every combination of an employee's dependent and an employee's project. This constraint is specified as a multivalued dependency on the EMP relation.

(a) EMP

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

15.6 Multivalued Dependency and Fourth Normal Form

Ename $\twoheadrightarrow$ Pname | Dname

(a) EMP

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

Smith works on **5** projects

Smith Has 4 Dependents

Smith Has 5 * 4 Tuples

(b) EMP_PROJECTS

Ename	Pname
Smith	X
Smith	Y

EMP_DEPENDENTS

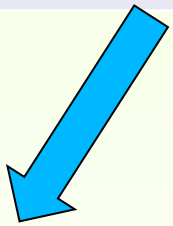
Ename	Dname
Smith	John
Smith	Anna

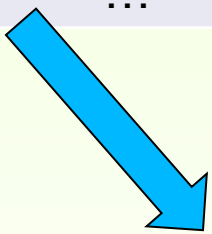
Student	Course	Hoppy
Ahmed	CS 2300	Tennis
Ahmed	CS 2300	Football
Ahmed	IS 2510	Tennis
Ahmed	IS 2510	Football
….	….	…
….	…..	…

Ahmed reg on **4** courses

Ahmed Has 3 hoppies

Ahmed Has 4 * 3 Tuples





Student	Course
A	CS 230
A	IS 2510

Student	Hoppy
A	Tennis
A	Football