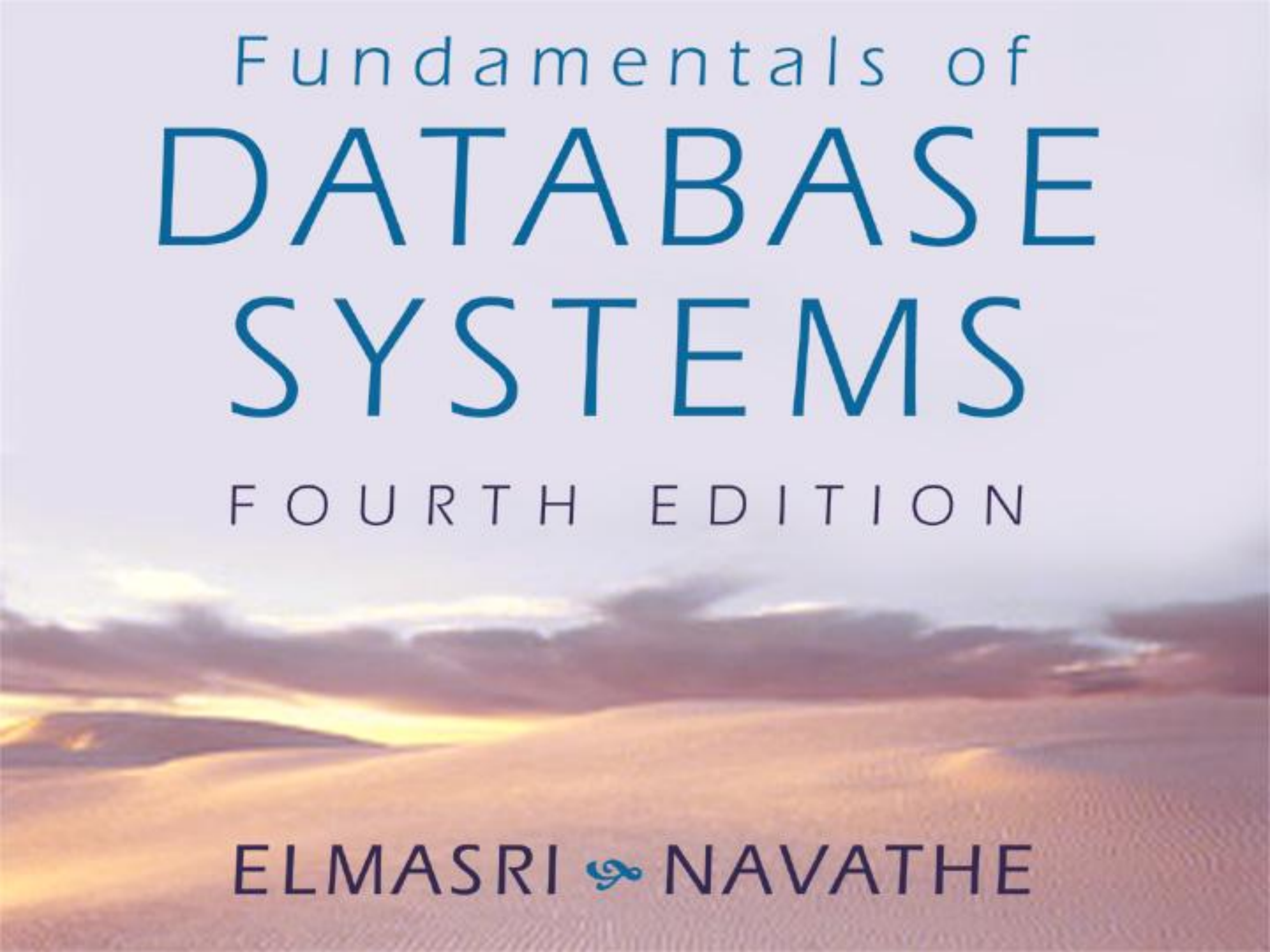


Fundamentals of DATABASE SYSTEMS

FOURTH EDITION

ELMASRI & NAVATHE

chapter 15

Basics of Functional Dependencies and Normalization for Relational Databases

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 (FDs)

- 2.1 Definition of FD
- 2.2 Inference Rules for FDs
- 2.3 Equivalence of Sets of FDs
- 2.4 Minimal Sets of FDs

Chapter Outline(contd.)

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)

1 Informal Design Guidelines for Relational Databases (1)

- What is relational database design?

  The grouping of attributes to form "good" relation schemas

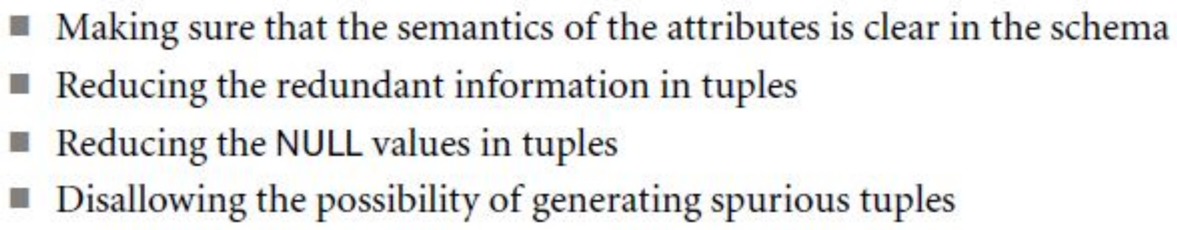
- Two levels of relation schemas
  - The logical "user view" or conceptual level
  - The implementation or physical storage "base relation" level
- Design is concerned mainly with base relations
- What are the criteria for "good" base relations?

1 Informal Design Guidelines for Relational Databases (1)

- Bottom up approach – adding up attributes to form relations – not followed usually
- Top Down – design by analysis – starts with a number of grouping of attributes to form relations
- Implicit goal of design – information preservation and minimum redundancy

Informal Design Guidelines for Relational Databases (2)

- 4 informal guidelines for good relational design



- Making sure that the semantics of the attributes is clear in the schema
- Reducing the redundant information in tuples
- Reducing the NULL values in tuples
- Disallowing the possibility of generating spurious tuples

- 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

1.1 Semantics of the Relation Attributes

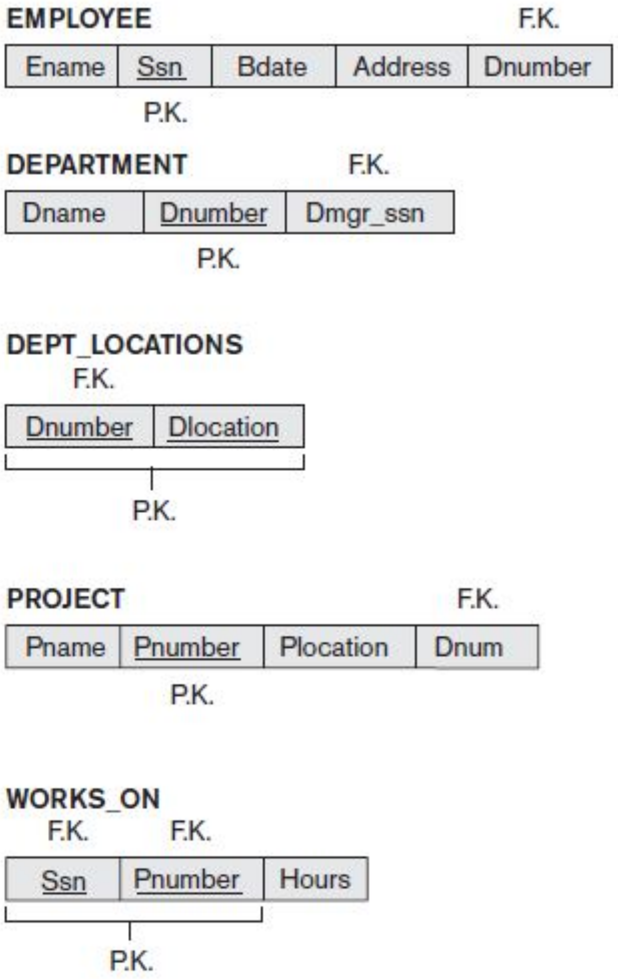
GUIDELINE 1: Informally, each tuple in a relation should represent one entity or relationship instance. (Applies to individual relations and their attributes).

- 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 15.1 A simplified COMPANY relational database schema

Figure 15.1
A simplified COMPANY relational database schema.



1.2 Redundant Information in Tuples and Update Anomalies

- Mixing attributes of multiple entities may cause problems

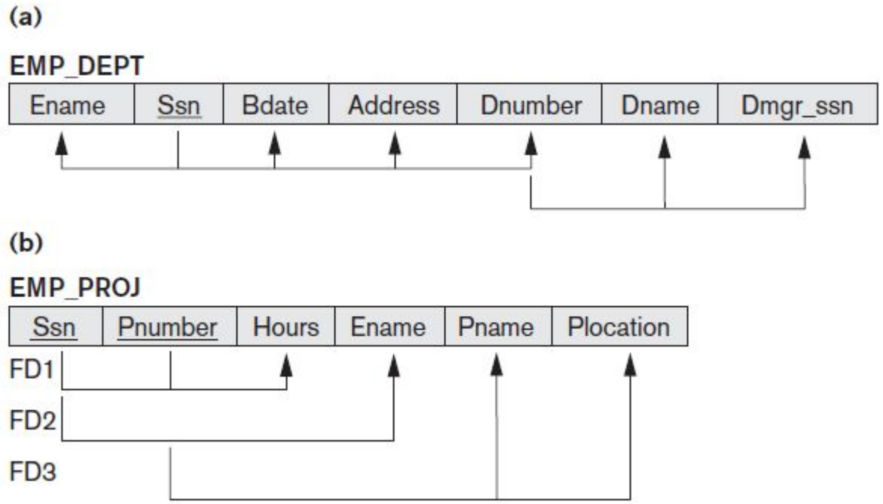


Figure 15.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.

- Information is stored redundantly wasting storage
- Problems with update anomalies
  - Insertion anomalies
  - Deletion anomalies
  - Modification anomalies

EXAMPLE OF AN UPDATE ANOMALY (1)

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 UPDATE ANOMALY (2)

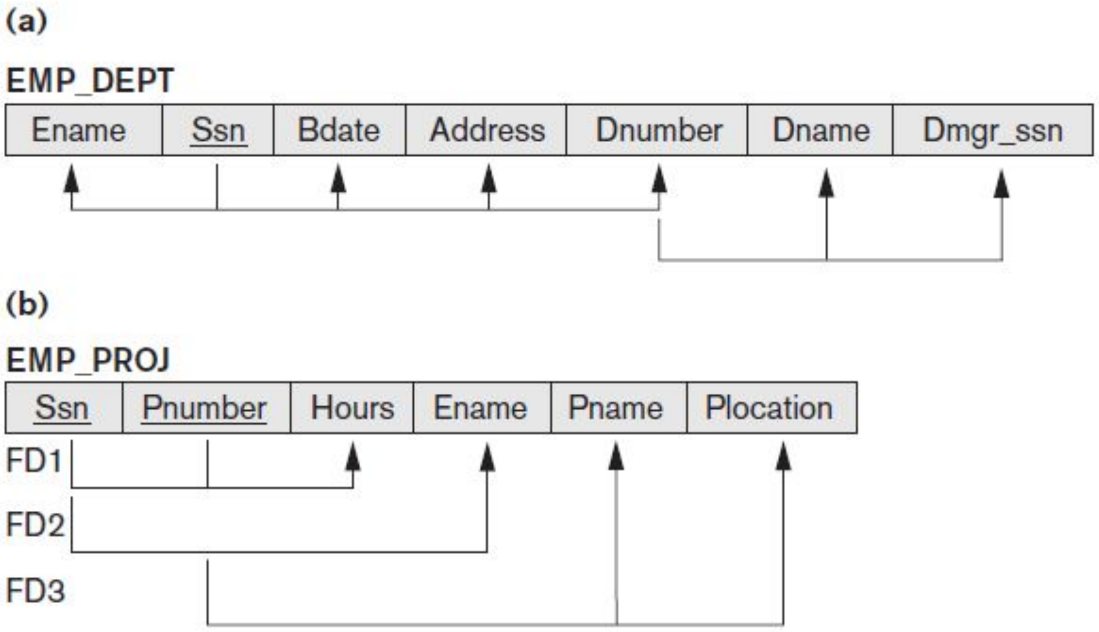
- **Insert Anomaly:** Cannot insert a project unless an employee is assigned to .

  Inversely - Cannot insert an employee unless an he/she is assigned to a project.

- **Delete Anomaly:** When a project is deleted, it will result in deleting all the employees who work *only* 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 15.3 Two relation schemas suffering from update anomalies

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



(a)

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

(b)

EMP_PROJ

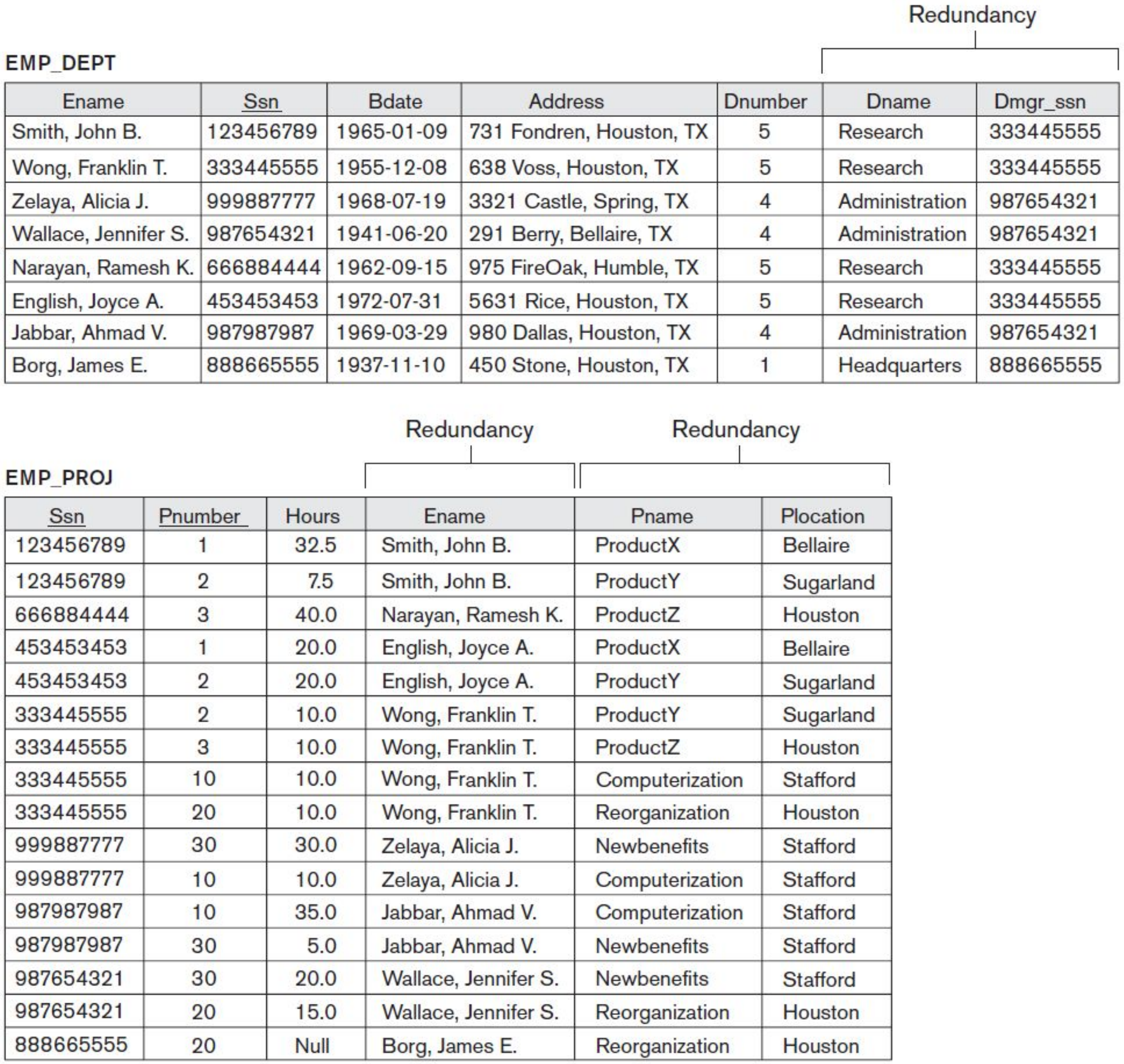
Ssn	Pnumber	Hours	Ename	Pname	Plocation

FD1

FD2

FD3

Figure 15.4 Example States for EMP_DEPT and 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

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 15.4
Sample states for EMP_DEPT and EMP_PROJ resulting from applying NATURAL JOIN to the relations in Figure 15.2. These may be stored as base relations for performance reasons.

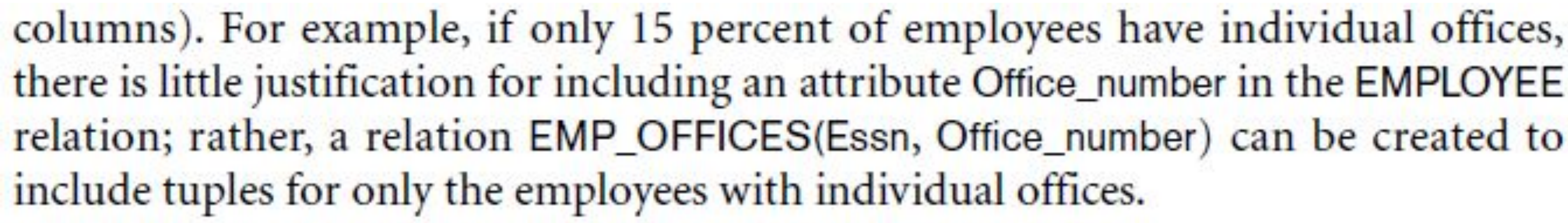
Guideline to Redundant Information in Tuples and Update Anomalies

- **GUIDELINE 2:** Design a schema that does not suffer from the insertion, deletion and update anomalies. If there are any present, then note them so that applications can be made to take them into account - a restatement of the Guideline 1.
- Best way – Use anomaly free base relations and views that use joins. Views can be updated through triggers to base relations.

1.3 Null Values in Tuples

GUIDELINE 3: Relations should be designed such that their tuples will have as few NULL values as possible (wastage of space in storage level and difficulties in JOIN operation with NULL values)

- 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)



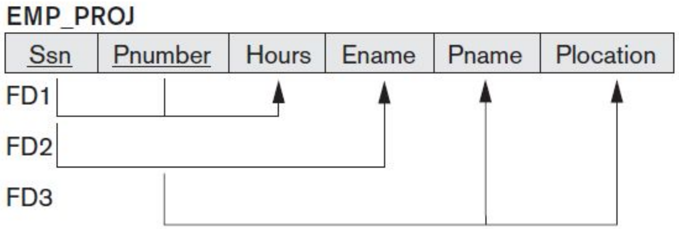
columns). For example, if only 15 percent of employees have individual offices, there is little justification for including an attribute Office_number in the EMPLOYEE relation; rather, a relation EMP_OFFICES(Essn, Office_number) can be created to include tuples for only the employees with individual offices.

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.

(b)


EMP_PROJ

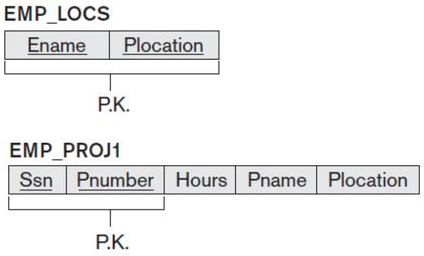
Ssn | Pnumber | Hours | Ename | Pname | Plocation

FD1

FD2

FD3

(a)


EMP_LOCS

Ename | Plocation

P.K.

EMP_PROJ1

Ssn | Pnumber | Hours | Pname | Plocation

P.K.

Figure 15.5
Particularly poor design for the EMP_PROJ relation in Figure 15.3(b). (a) The two relation schemas EMP_LOCS and EMP_PROJ1. (b) The result of projecting the extension of EMP_PROJ from Figure 15.4 onto the relations EMP_LOCS and EMP_PROJ1.

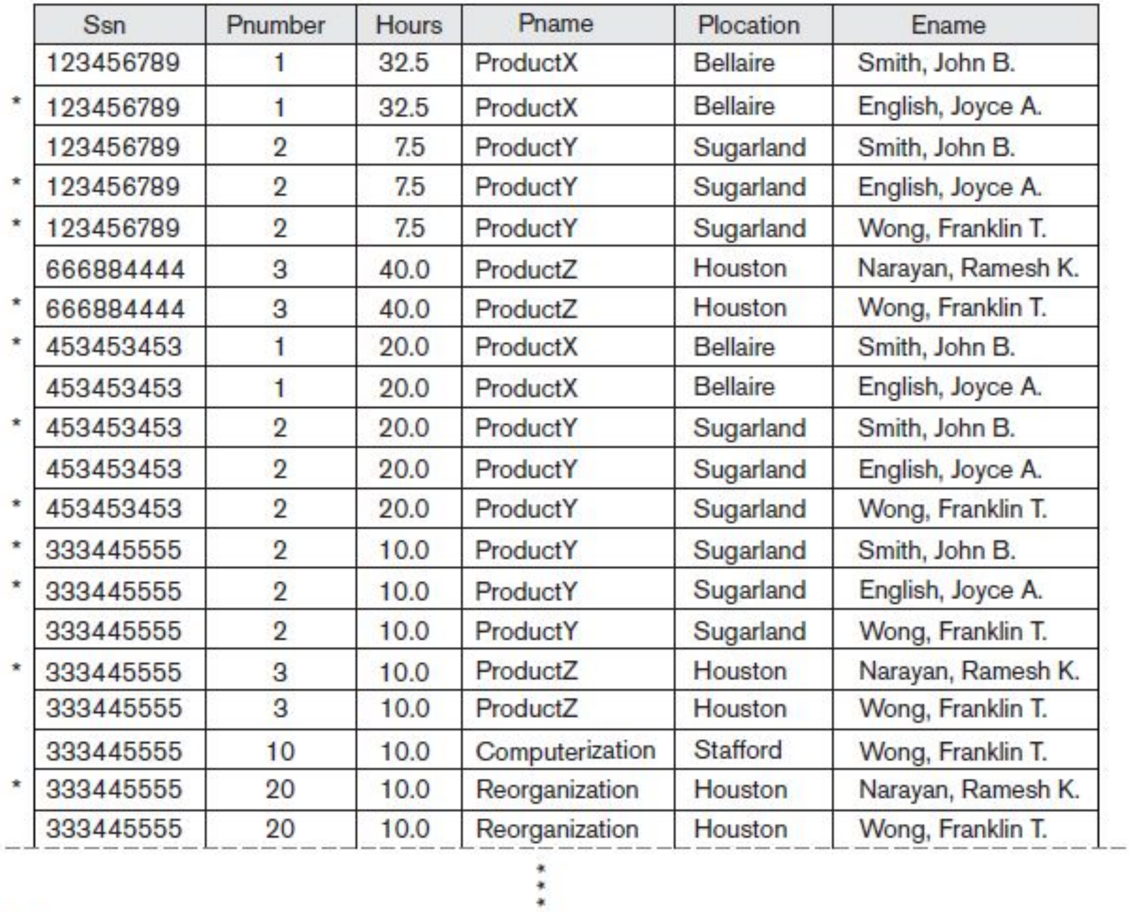
(b)

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



	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 15.6
Result of applying NATURAL JOIN to the tuples above the dashed lines in EMP_PROJ1 and EMP_LOCS of Figure 15.5. Generated spurious tuples are marked by asterisks.

Decomposing EMP_PROJ into EMP_LOCS and EMP_PROJ1 is undesirable

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.

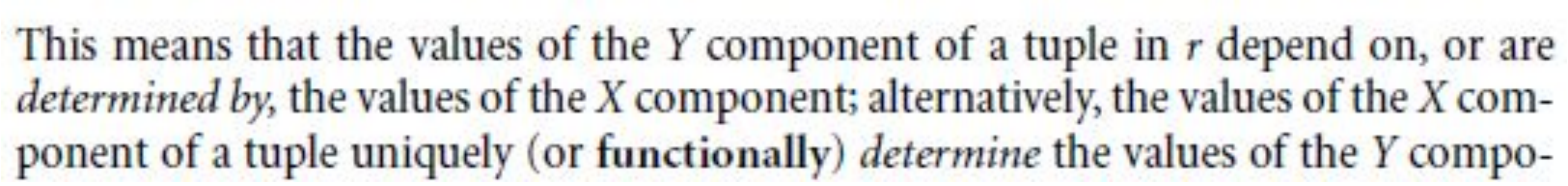
Note that property (a) is extremely important and *cannot* be sacrificed. Property (b) is less stringent and may be sacrificed. (See Chapter 11).

2.1 Functional Dependencies (1)

- Functional dependencies is a constraint between 2 sets of attributes
- Functional dependencies (FDs) are used to specify *formal measures* of the "goodness" of relational schema designs
- FDs and keys are used to define **normal forms** for relations
- FDs 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

Functional Dependencies (2)

- 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 t1 and t2 in any relation instance r(R): *If* t1[X]=t2[X], *then* t1[Y]=t2[Y]



This means that the values of the *Y* component of a tuple in *r* depend on, or are *determined by*, the values of the *X* component; alternatively, the values of the *X* component of a tuple uniquely (or **functionally**) *determine* the values of the *Y* compo-

- 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 (1)

- 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

Examples of FD constraints (2)

- An FD is a property of the attributes in the schema R, not of a particular relation state
- 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])

2.2 Inference Rules for FDs (1)

- Given a set of FDs F, we can *infer* additional FDs that hold whenever the FDs in F hold

Armstrong's inference rules:

IR1. (**Reflexive**) If Y *subset-of* X, then X **->** Y

IR2. (**Augmentation**) If X **->** Y, then XZ **->** YZ

(Notation: XZ stands for X U Z)

IR3. (**Transitive**) If X **->** Y and Y **->** Z, then X **->** Z

- IR1, IR2, IR3 form a *sound* and *complete* set of inference rules

Inference Rules for FDs (2)

Some **additional inference rules** that are useful:

(**Decomposition**) If X **->** YZ, then X **->** Y and X **->** Z

(**Union**) If X **->** Y and X **->** Z, then X **->** YZ

(**Psuedotransitivity**) If X **->** Y and WY **->** Z, then WX **->** Z

- The last three inference rules, as well as any other inference rules, can be deduced from IR1, IR2, and IR3 (completeness property)

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

Normal Forms Based on Primary Keys

Sequence of practical relational design projects

- conceptual schema design using a conceptual model - ER or EER, map the conceptual design into a set of relations
- Design the relations based on external knowledge derived from an existing implementation of files or forms or reports
- evaluate the relations for goodness and decompose them further as needed to achieve higher normal forms – Normalization

- Normalization process - first proposed by Codd (1972), takes a relation schema through a series of tests to *certify* whether it satisfies a certain **normal form**.
- The top-down process evaluates each relation against the criteria for normal forms and decomposing relations as necessary - *relational design by analysis.*
- Codd proposed 3 normal forms - first, second, and third. A stronger definition of 3NF—called Boyce-Codd normal form (BCNF) - proposed by Boyce and Codd.
- All these normal forms are based on a single analytical tool: the functional dependencies among the attributes of a relation.
- Later, a fourth normal form (4NF) and a fifth normal form (5NF) were proposed, based on the concepts of multivalued dependencies and join dependencies

3.1 Normalization of Relations (1)

- It is a process of analyzing a given relation schemas based on their FDs and PKs to achieve minimum redundancy and update anomalies

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

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, 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

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

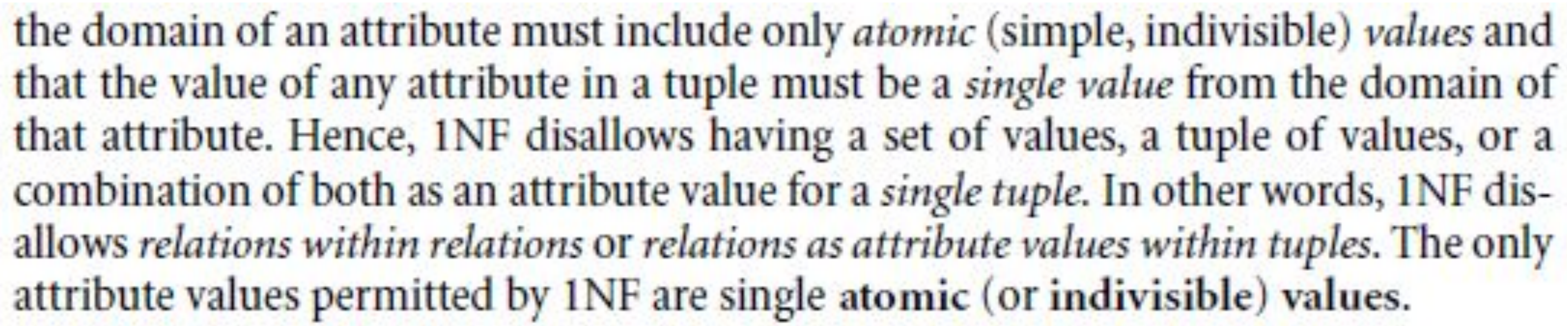
- A **superkey** of a relation schema $R = \{A_1, A_2, ..., A_n\}$ is a set of attributes S *subset-of* R with the property that **no** two tuples t_1 and t_2 in any legal relation state r of R will have $t_1[S] = t_2[S]$

- 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.

3.2 First Normal Form (1NF)

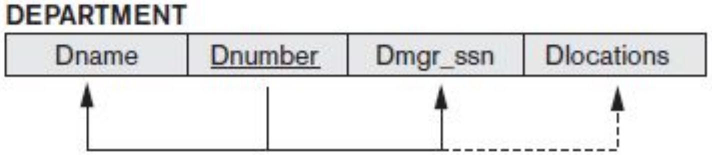


the domain of an attribute must include only *atomic* (simple, indivisible) *values* and that the value of any attribute in a tuple must be a *single value* from the domain of that attribute. Hence, 1NF disallows having a set of values, a tuple of values, or a combination of both as an attribute value for a *single tuple*. In other words, 1NF disallows *relations within relations* or *relations as attribute values within tuples*. The only attribute values permitted by 1NF are single **atomic** (or **indivisible**) **values**.

- Disallows composite attributes, multivalued attributes, and **nested relations**; attributes whose values *for an individual tuple* are non-atomic
- Considered to be part of the formal definition of a relation in the basic relational model

Figure 15.9 Normalization into 1NF

(a)



(b)

DEPARTMENT

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

(c)

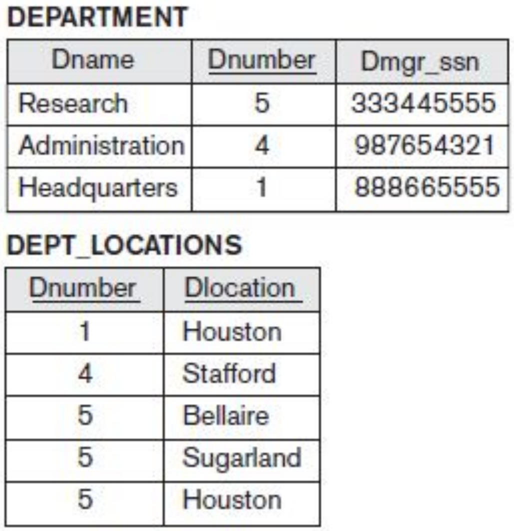
DEPARTMENT

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.

First Normal Form (1NF)

1. Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS with the primary key {Dnumber, Dlocation}, DEPT_LOCATIONS exists for *each location* of a department. This decomposes the non-1NF relation into two 1NF relations.
2. Expand the key to have Dlocation in original DEPARTMENT relation. The primary key - {Dnumber, Dlocation}. This solution introduces *redundancy*.
3. If a *max number of values for multi-valued attribute* is known (3 in this case) - replace the Dlocations attribute by three atomic attributes: Dlocation1, Dlocation2, and Dlocation3. Disadvantage – too many *NULL values,* spurious semantics about the ordering among the location values, Querying on this attribute becomes more difficult.

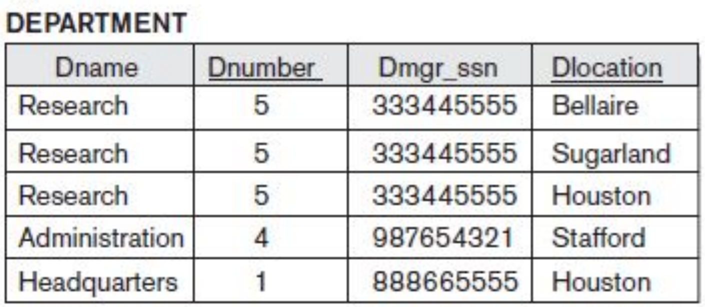


DEPARTMENT

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

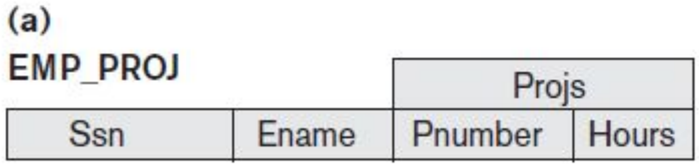


DEPARTMENT

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

Fig 15.10 Normalization nested relations into 1NF

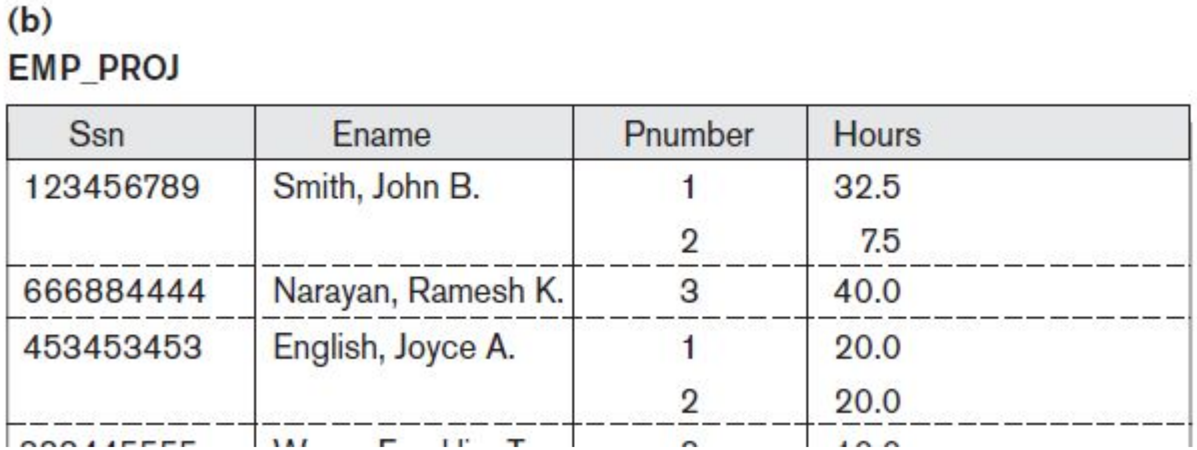
EMP_PROJ(Ssn, Ename, {PROJS(Pnumber, Hours)})



(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours



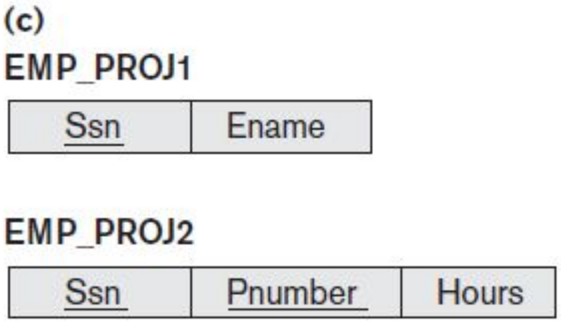
(b)

EMP_PROJ

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1 2	32.5 7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1 2	20.0 20.0

PERSON (Ss#, {Car_lic#}, {Phone#})

P1(Ss#, Car_lic#) and P2(Ss#, Phone#).



(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

3.3 Second Normal Form (1)

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

Definitions:

- **Prime attribute** - 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

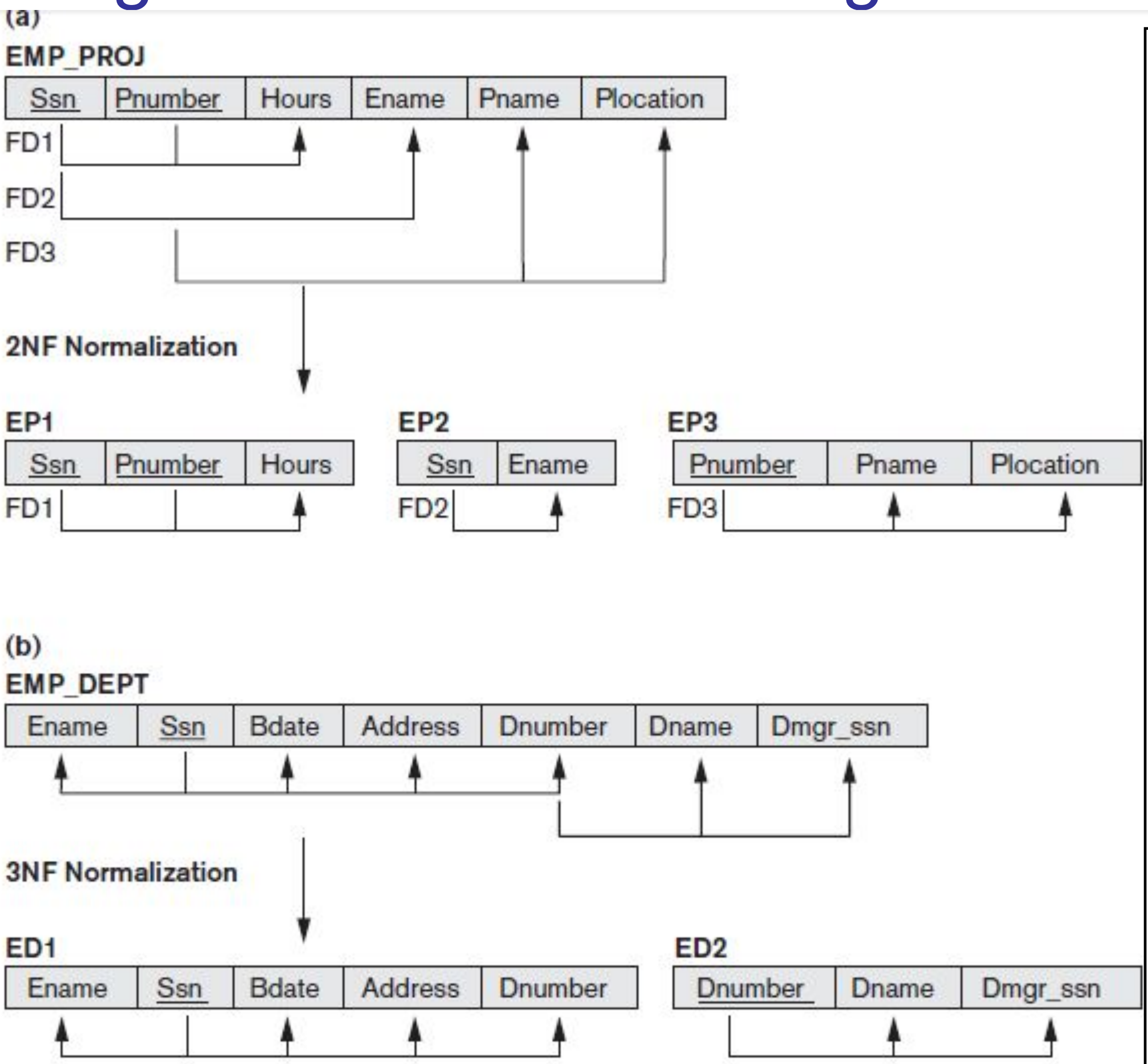
- {SSN, PNUMBER} -> ENAME is *not* a full FD (it is called a *partial dependency*) since SSN -> ENAME also holds

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

Figure 15.11 Normalizing into 2NF and 3NF



EMP_PROJ

Ssn	Pnumber	Hours	Ename	Pname	Plocation

EP1

Ssn	Pnumber	Hours

EP2

Ssn	Ename

EP3

Pnumber	Pname	Plocation

EMP_DEPT

Ename	Ssn	Bdate	Address	Dnumber	Dname	Dmgr_ssn

ED1

Ename	Ssn	Bdate	Address	Dnumber

ED2

Dnumber	Dname	Dmgr_ssn

Figure 15.11
Normalizing into 2NF and 3NF. (a) Normalizing EMP_PROJ into 2NF relations. (b) Normalizing EMP_DEPT into 3NF relations.

1. Ename violates 2NF because of FD2,
2. Pname and Plocation because of FD3 also violates 2NF.

The relation schema can be *second normalized* or *2NF normalized* into a number of 2NF relations. FD1, FD2, and FD3 lead to the decomposition of EMP_PROJ into the three relation schemas EP1, EP2, and EP3, each of which is in 2NF.

3.4 Third Normal Form (1)

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

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:

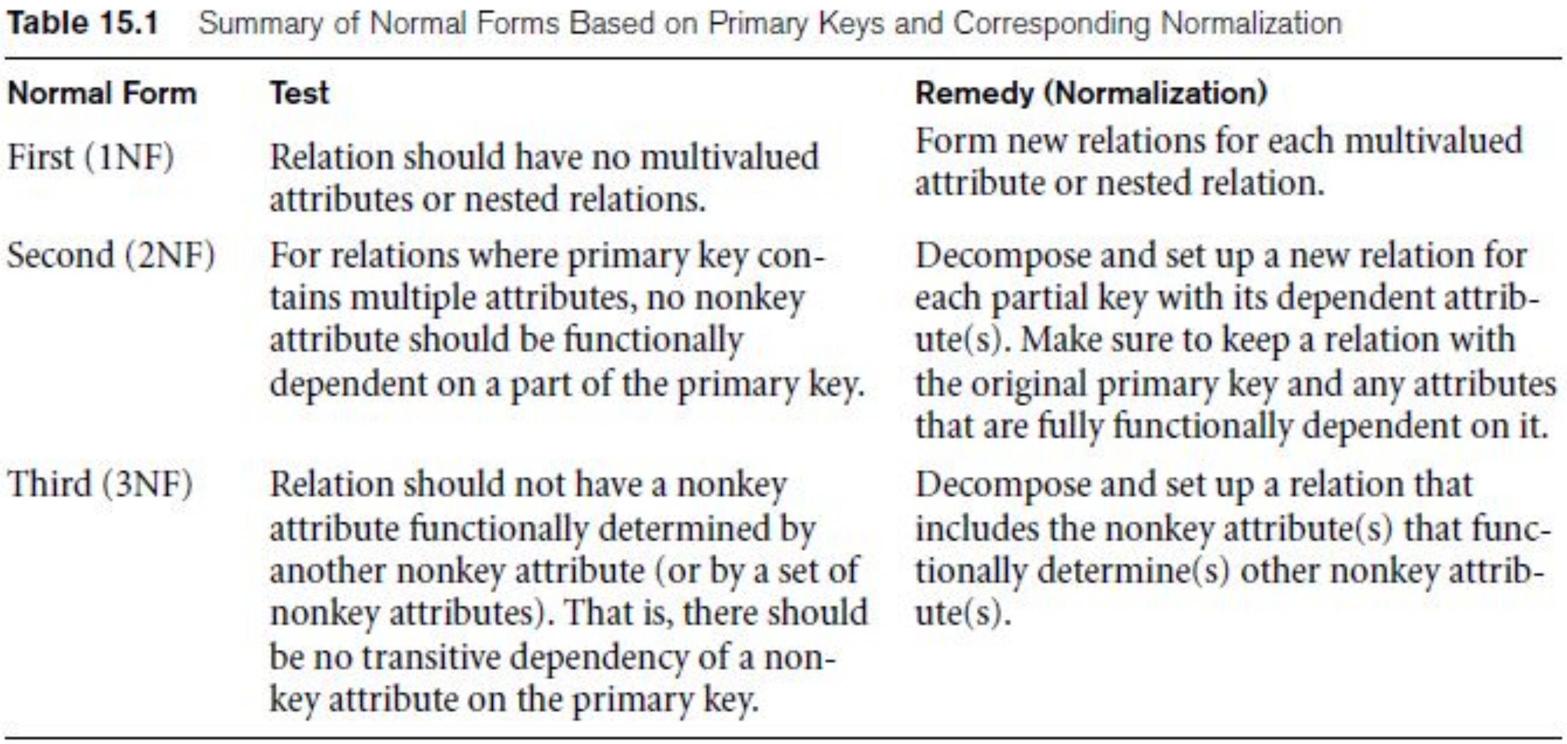
In X -> Y and Y -> Z, with X as the primary key, we consider this a problem only if Y is <u>not</u> a candidate key. When Y is a candidate key, there is no problem with the transitive dependency .

E.g., Consider EMP (SSN, Emp#, Salary).

Here, SSN -> Emp# -> Salary and Emp# is a candidate key.

General Definitions of Second and Third Normal Forms

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

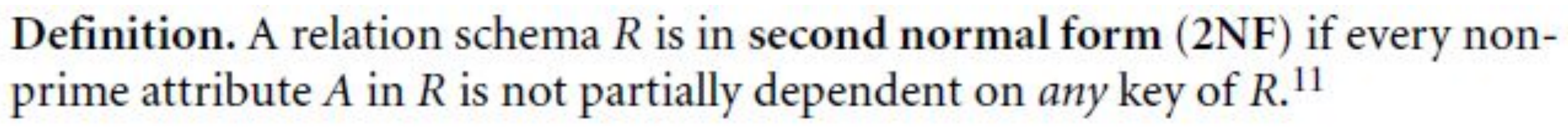


Normal Form	Test	Remedy (Normalization)
First (1NF)	Relation should have no multivalued attributes or nested relations.	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no nonkey attribute should be functionally 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).

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

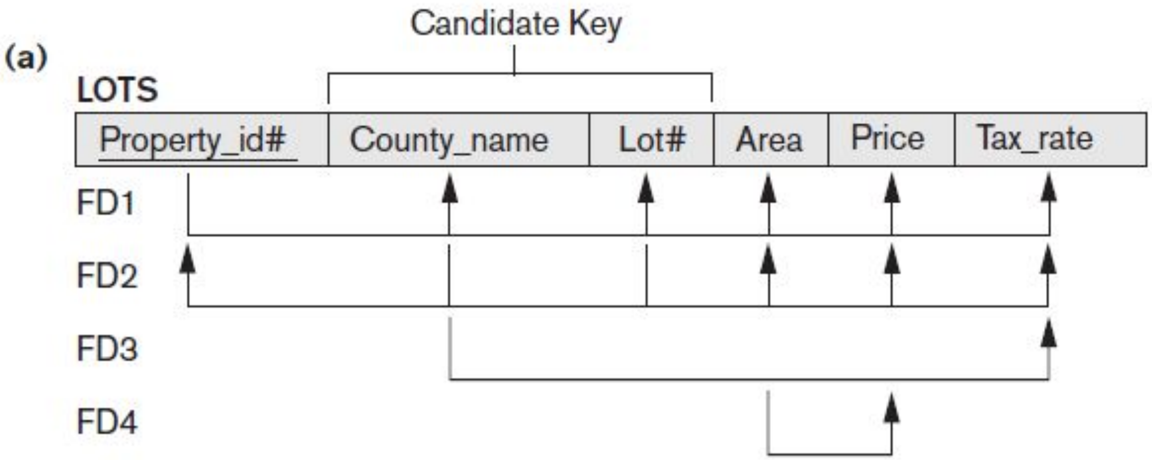
General Definition of 2NF



Definition. A relation schema *R* is in **second normal form** (2NF) if every nonprime 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.

violates the general definition of 2NF because Tax_rate is partially dependent on the candidate key {County_name, Lot#},

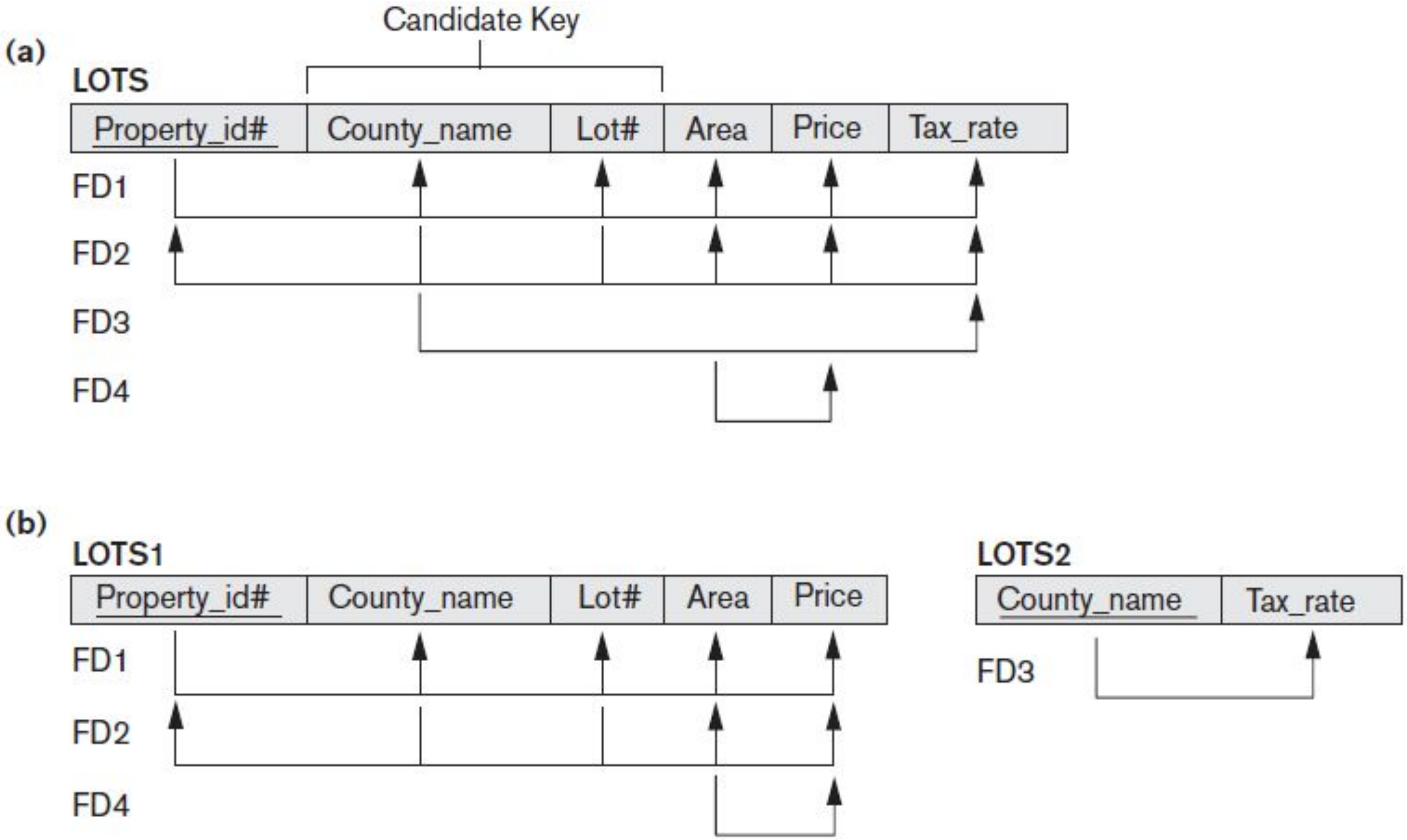


(a)

LOTS

Property_id#	County_name	Lot#	Area	Price	Tax_rate

General Definition of 2NF



Candidate Key

(a)

LOTS

Property_id#	County_name	Lot#	Area	Price	Tax_rate

FD1

FD2

FD3

FD4

(b)

LOTS1

Property_id#	County_name	Lot#	Area	Price

FD1

FD2

FD4

LOTS2

County_name	Tax_rate

FD3

General Definition of 3rd Normal Form

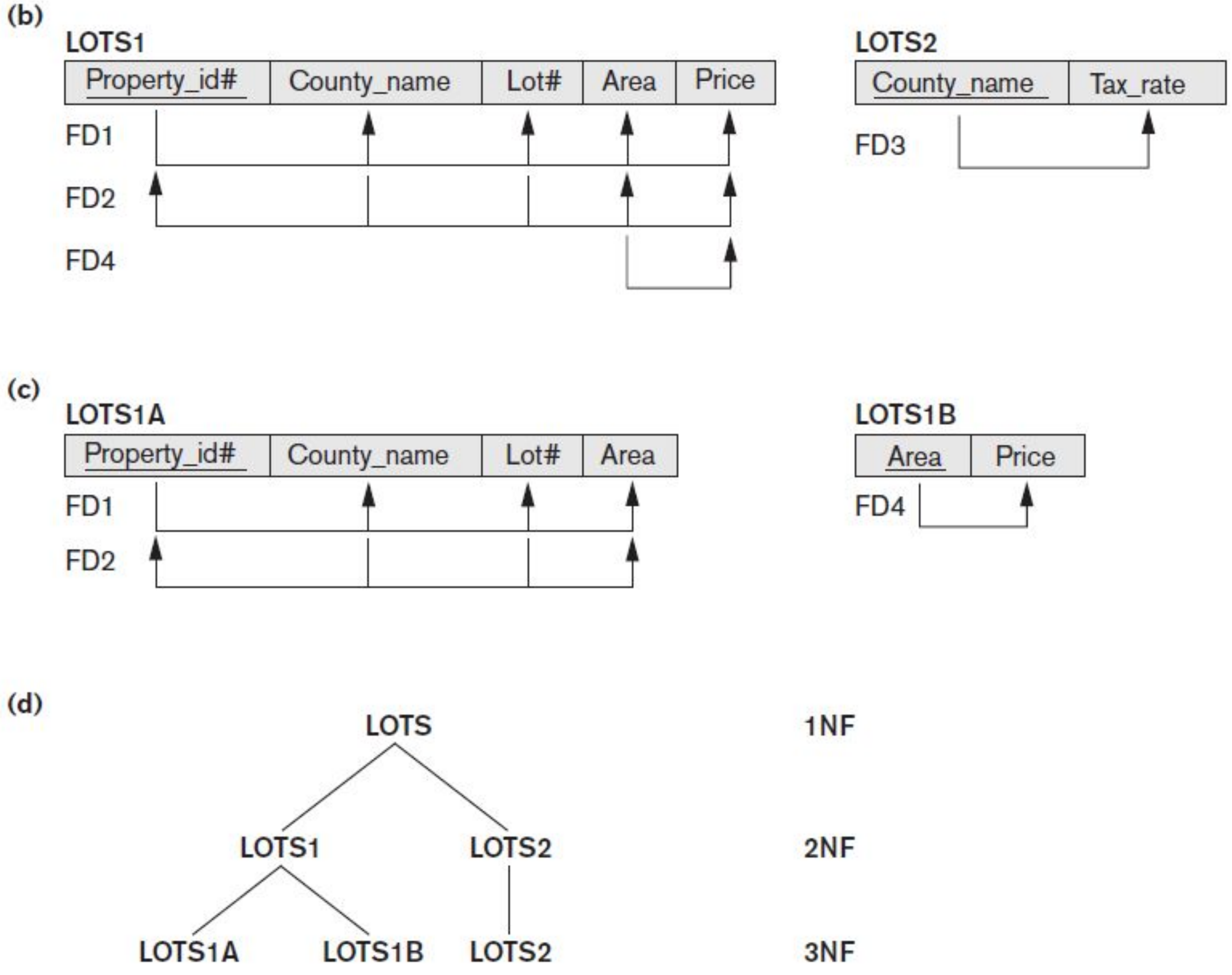
Definition:

- **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

NOTE: Boyce-Codd normal form disallows condition (b) above



(b)

LOTS1

Property_id#	County_name	Lot#	Area	Price

FD1

FD2

FD4

LOTS2

County_name	Tax_rate

FD3

(c)

LOTS1A

Property_id#	County_name	Lot#	Area

FD1

FD2

LOTS1B

Area	Price

FD4

(d)

LOTS

LOTS1

LOTS2

LOTS1A

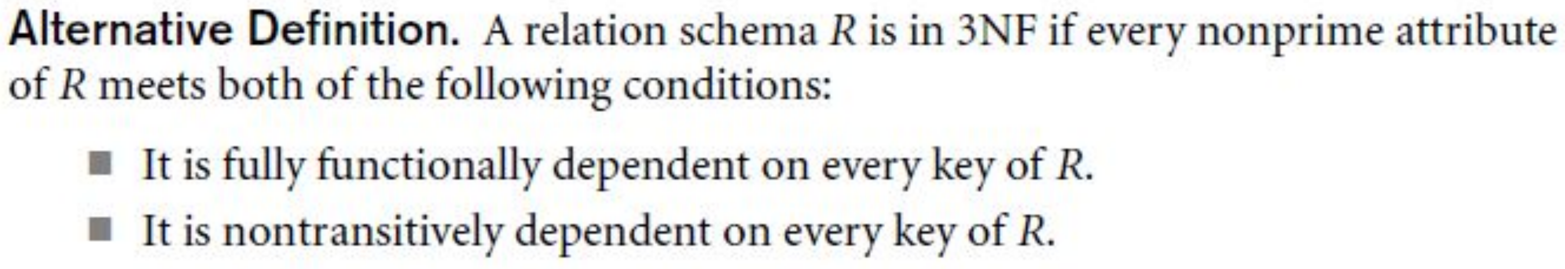
LOTS1B

LOTS2

1NF

2NF

3NF



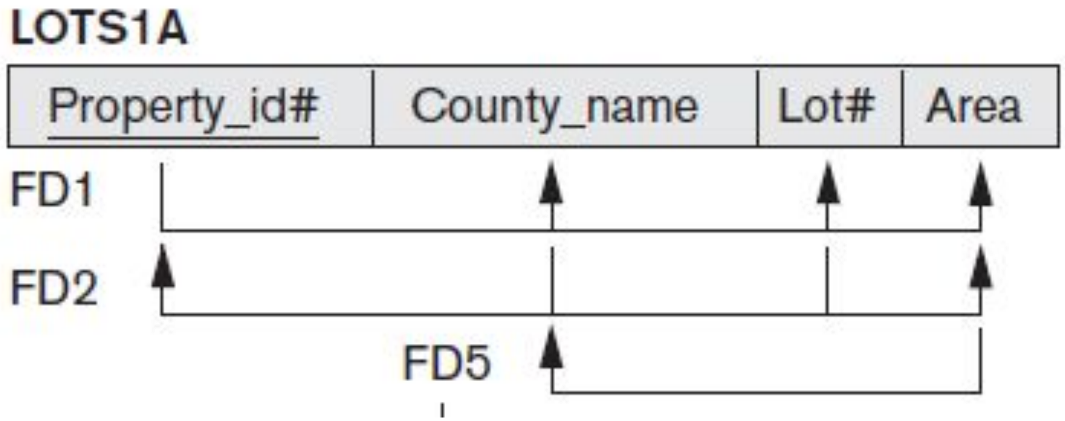
Alternative Definition. A relation schema *R* is in 3NF if every nonprime attribute of *R* meets both of the following conditions:

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

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
- 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
- Most relation schemas that are in 3NF are also in BCNF
- The goal is to have each relation in BCNF (or 3NF)

BCNF



Suppose that we have thousands of lots in the relation and lots are from only two counties: DeKalb and Fulton. Suppose also that lot sizes in DeKalb County are only 0.5 to 1.0 acres, whereas lot sizes in Fulton County are from 1.1 to 2.0 acres. This adds a functional dependency FD5: ***Area->County_name***.

LOTS1A still is in 3NF because ***County_name*** is a prime attribute, but not in BCNF as ***Area*** is not a superkey of LOTS1A

Figure 15.3 Boyce-Codd normal form

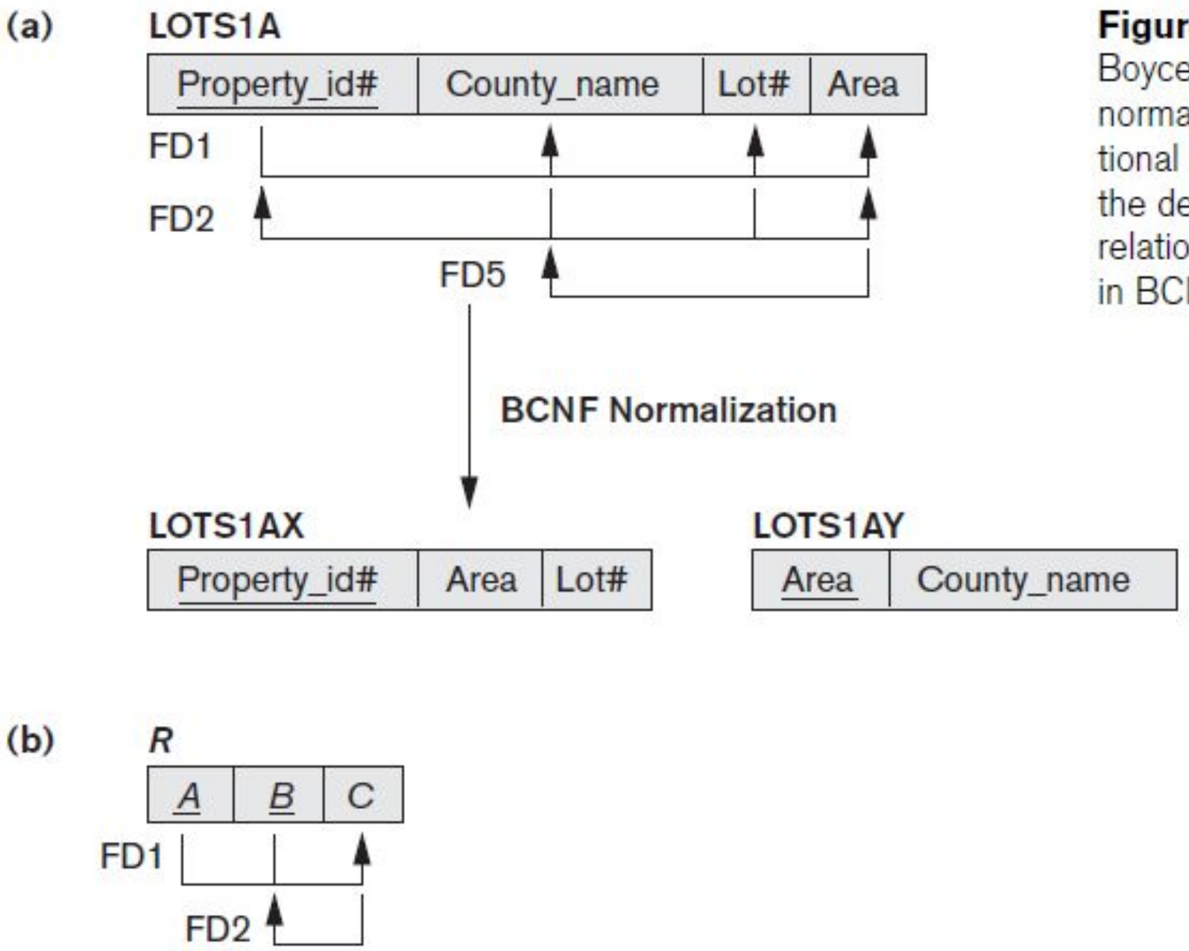


Figure 15.13
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF.

Figure 15.4 a relation TEACH that is in 3NF but not in BCNF

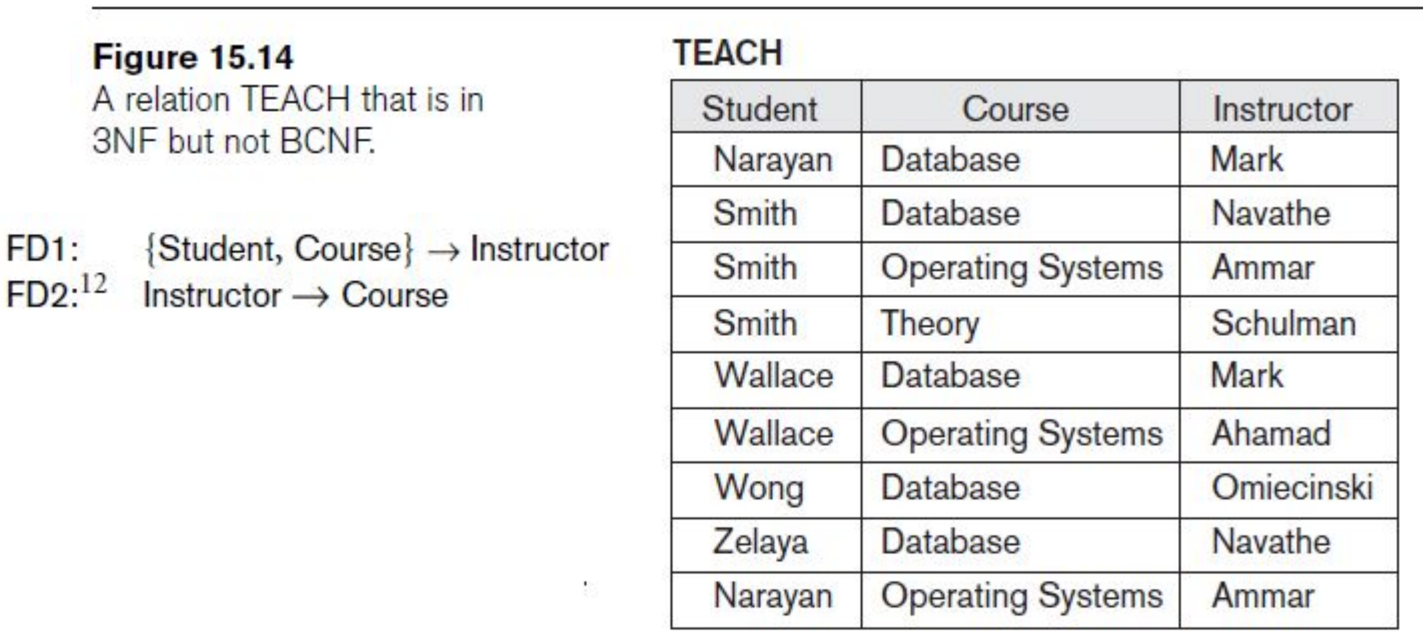


Figure 15.14
A relation TEACH that is in 3NF but not BCNF.

FD1: {Student, Course} → Instructor
FD2:[12] Instructor → Course

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

Achieving the BCNF by Decomposition (1)

- 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 10.12 (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. (See Algorithm 11.3)

Achieving the BCNF by Decomposition (2)

- Three possible decompositions for relation TEACH
  1. {student, instructor} and {student, course}
  2. {course, instructor } and {course, student}
  3. {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 nonadditive (lossless) is discussed in section 11.1.4 under Property LJ1. Verify that the third decomposition above meets the property.

Summary

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