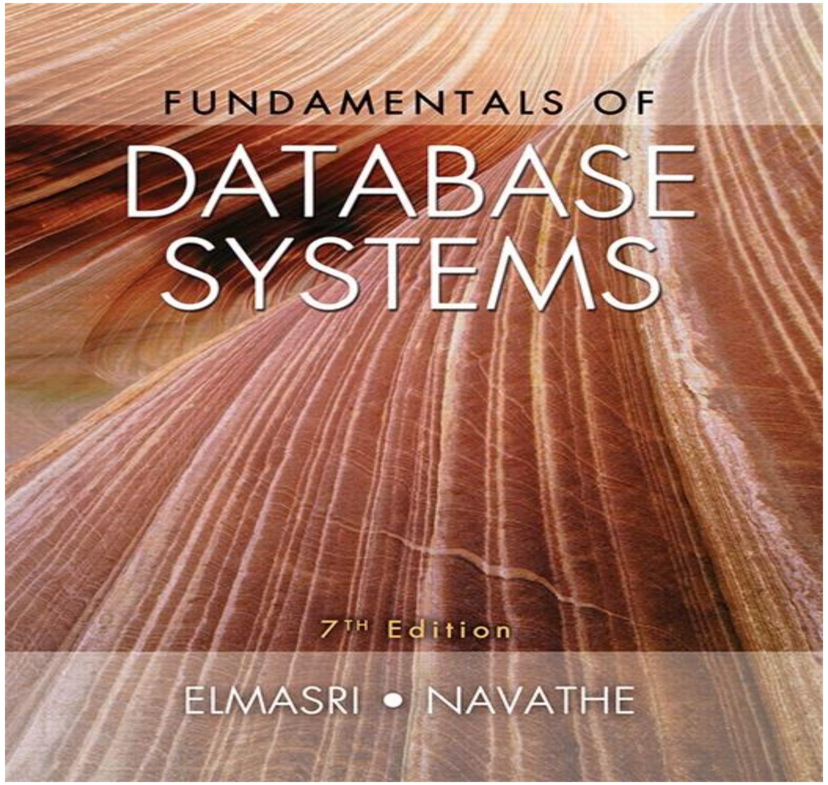
Comp-3150: Database Management Systems

- Ramez Elmasri , Shamkant B. Navathe(2016) Fundamentals of Database Systems (7th Edition), Pearson, isbn 10: 0-13-397077-9; isbn-13:978-0-13-397077-7.

Chapter 14:
Basics of Functional Dependencies and Normalization for Relational Databases



Chapter 14: Basics of Functional Dependencies and Normalization for Relational Databases: 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 14 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)

1 Informal Design Guidelines for Relational Databases

- How do we analyze database design? How do we analyze the grouping of the attributes into relations for a mini world?
- How do we measure the goodness of relation schemas.
- Relational DB design produces a set of relations with the implicit goals of information preservation and minimum redundancy.
- What are the criteria for "good" base relations?

1 Informal Design Guidelines for Relational Databases

1. Informal Design guidelines
    a. Four information guidelines used to measure the quality of relation schema design are
        i. Attributes in the schema have clear semantics
        ii. Redundant information in tuples are reduced.
        iii. NULL values in tuples are reduced.
        iv. Possibility of generating spurious tuples are disallowed.
    b. Fig 14.1 shows a simplified form of the company schema with clear meaning

Guideline 1: Design a relation with clear meaning and which does not combine attributes from multiple entity types such as Employee and Department.

Figure 14.1 A simplified COMPANY relational database schema

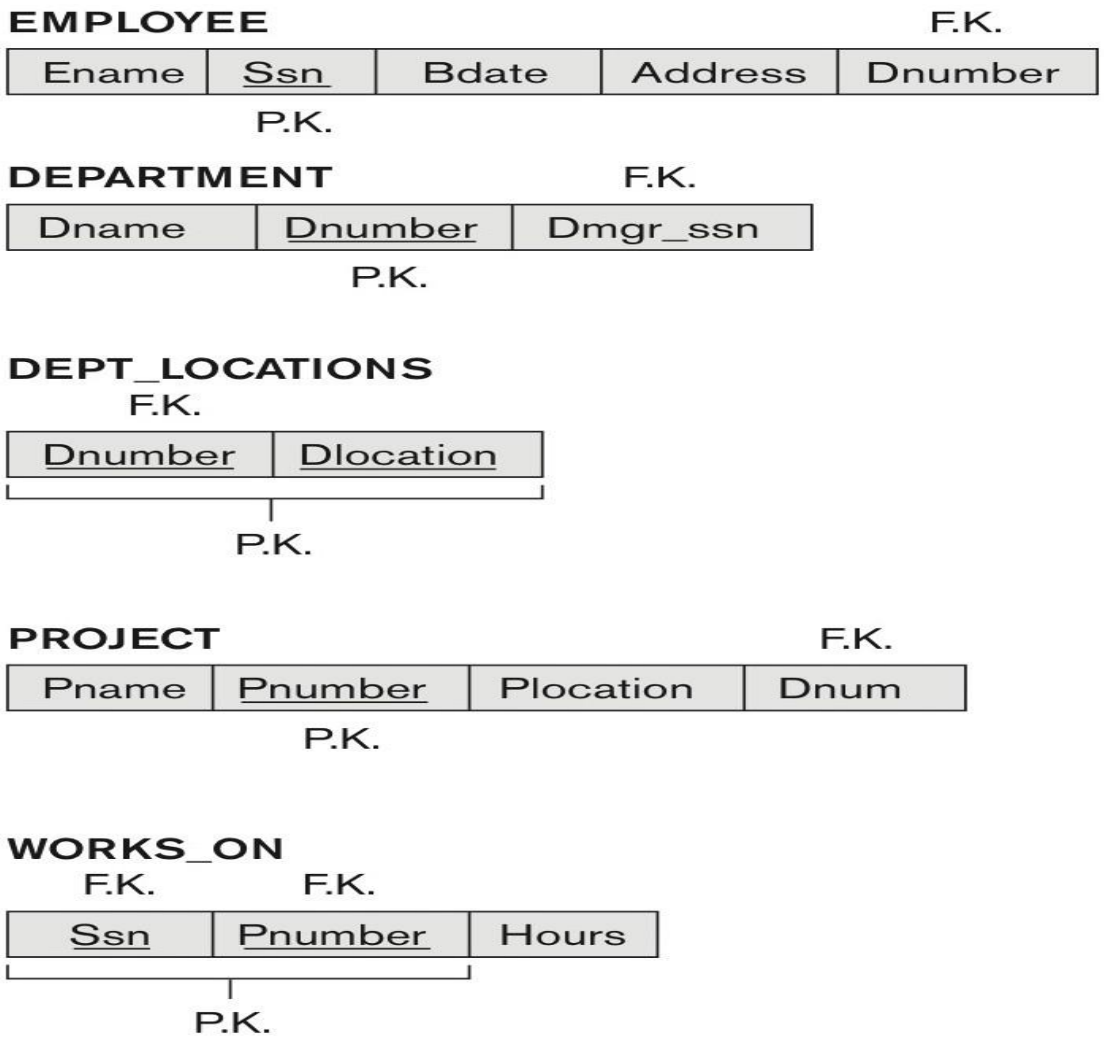


Figure 14.1 A simplified COMPANY relational database schema.

1.2 Redundant information in tuples and update anomalies

- Note that fig 14.4 is the result of applying natural join operation on Employee and Department of fig 14.2 (a database state of the schema of Fig. 14.1).
- Storing natural joins of base relations leads to update anomalies which are insertion, deletion and modification anomalies.
- Anomalies are indications of presence of redundancy in DB design.

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(<u>Emp#, Proj#, </u>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(<u>Emp#, Proj#, </u>Ename, Pname, No_hours)
- Insert Anomaly:
  - Cannot insert a project unless an employee is assigned to it.
- Conversely
  - Cannot insert an employee unless he/she is assigned to a project.

EXAMPLE OF A DELETE ANOMALY

- Consider the relation:
  - EMP_PROJ(<u>Emp#, Proj#,</u> Ename, Pname, No_hours)
- Delete Anomaly:
  - When a project is deleted, it may result in deleting all the employees who work on that project.
  - Alternatively, 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

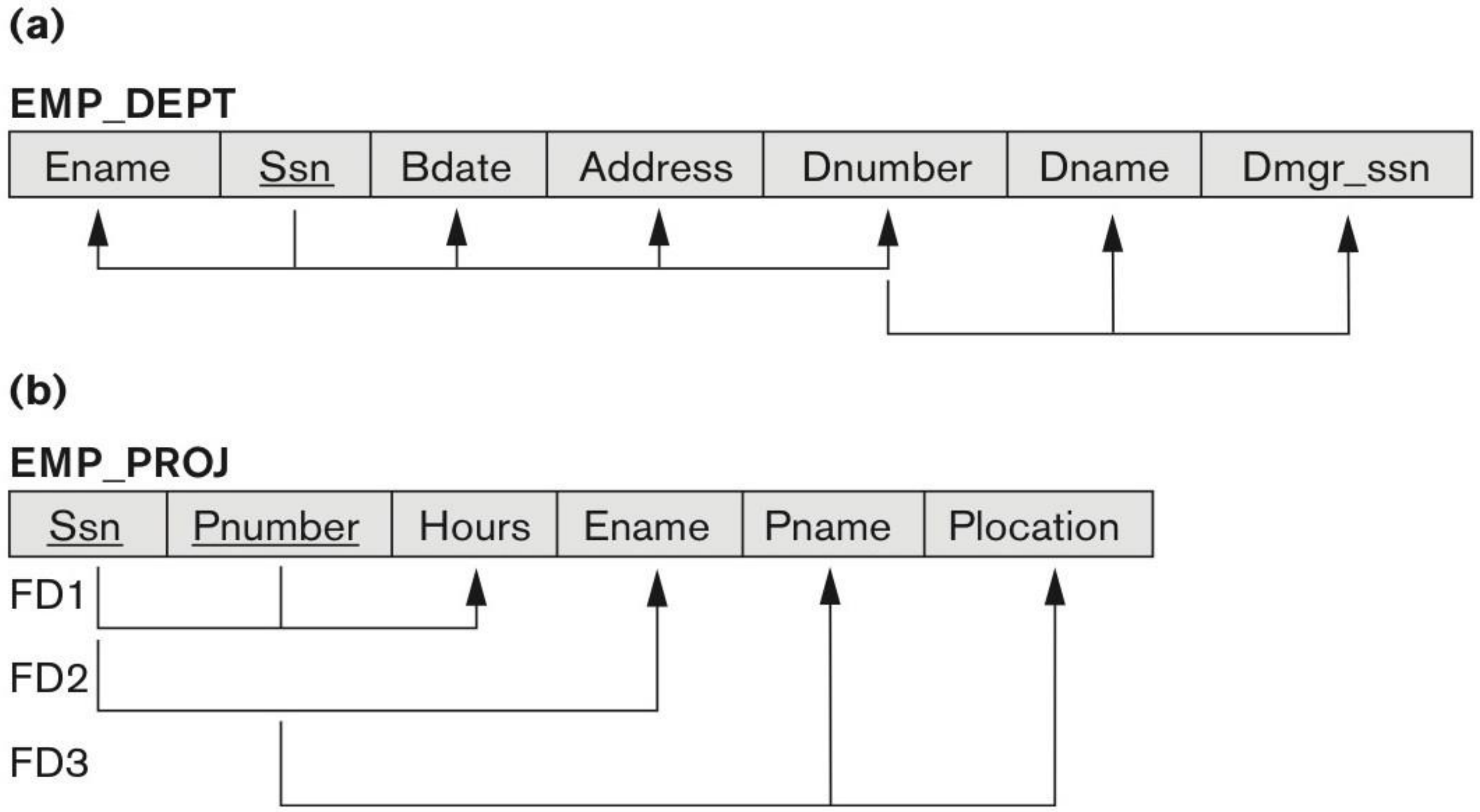


Figure 14.3
Two relation schemas suffering from update anomalies. (a) EMP_DEPT and (b) EMP_PROJ.

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 (Redundancy: Dname, Dmgr_ssn)

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 (Redundancy: Ename; Redundancy: Pname, Plocation)

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 (not in the original relation) should be generated by doing a natural-join of any decomposed relations.
  - Design relation schemas that can be joined with equality conditions on attributes that are (primary key, foreign key) pairs

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)
  - FDs are used to specify *formal measures* of the "goodness" of relational 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

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 t1 and t2 in any relation instance r(R): If t1[X]=t2[X], *then* t1[Y]=t2[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 (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
- 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 14.8 What FDs may exist?

- A relation *R*(A, B, C, D) with its extension.
- Which FDs *may exist* in this relation?
- The following FDs may hold B->C, C->B
- The following FDs do not hold A->B, B->A, D->C
- We denote by F the set of functional dependencies specified on relation schema R.

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

3.1 Normalization of Relations (1)

- **Normalization:**
  - Is 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 = {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] because it contains the key K.

- 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

- 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 (1NF)

- INF states that the domain of an attribute must include only atomic (simple, indivisible) values.
- Thus, 1NF disallows
  - composite attributes
  - multivalued attributes
  - **nested relations**; attributes whose values for an *individual tuple* are non-atomic
- 1NF is 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

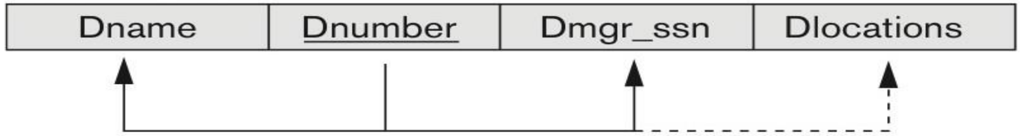
3.4 First Normal Form (1NF)

- For example, Fig 14.9b is not in 1NF
- To put in 1NF, break the fig 14.9b table into two tables:
  - Department(Dname, Dnumber, Dmgr-ssn) and
  - DEPT_LOCATIONS(Dnumber, DLocations) as in Fig 14.2 (a db state of Fig 14.1).
  - This decomposition is done by removing the attribute Dlocations that violates 1NF and placing it in a separate relation along with the primary key Dnumber of DEPARTMENT to maintain connection.
- Fig 14.10(b) has the schema EMP_PROJ(ssn, Ename {PROJS(Pnumner, Hours) })
- This relation EMP_PROJ is not in 1NF as it has nested relations as value of attribute PROJS identified in { } as multivalued.

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

Ssn	Ename	Projs	
		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 (2NF)

- 2NF is based on the concepts of **FDs, and primary key**
- Definitions
  - **Prime attribute:** An attribute that is member of the candidate key K
  - **Full functional dependency:** a FD Y -> Z where removal of any attribute from Y means the FD does not hold any more (or if no proper subset of Y also determines Z).
- 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 2NF if every nonprime attribute (not a member of primary or candidate key) A in R is FFD (fully functionally Dependent) on the primary or candidate key of R.

3.5 Second Normal Form (1)

- A Relation schema R is in 2NF if every nonprime attribute A in R is FFD(fully functionally Dependent) on the primary/candidate key of R.
- The Test for 2NF involves testing for FD's whose left hand side attributes are part of the primary key.
  - Eg. The EMP_PROJ relation of Fig 14.3 copied to Fig 14.11 is not in 2NF
  - Emp_Proj(<u>Ssn</u>, <u>Pnumber</u>, Hours, Ename, Pname, Plocation) has the following FDs
  - FD1: (Ssn,Pnumber)->Hours
  - FD2: Ssn-> Ename
  - FD3: Pnumber-> Pname, Plocation

3.5 Second Normal Form (2NF)

- EMP_PROJ is not in 2NF because from FD2 and FD3:
  - the nonprime (non key) attributes Ename, Pname and Plocation are determined by a subset of the primary key of EMP_PROJ(Ssn, Pnumber) thus violating the 2NF test stating that all non key attributes should be FFD on the key.
- To place in 2NF, break into a number of relations in which non key attributes are associated only with the part of the primary key on which they are FFD on.
- Thus, decomposition of EMP_PROJ into Fig 14.11(a) which are EP1, Ep2 and Ep3 in 2NF.
- EP1(<u>Ssn</u>, <u>pnumber</u>, Hours)
- Ep2(<u>Ssn</u>, Ename)
- EP3(<u>Pnumber</u>, Pname, Plocation)

Figure 14.11 Normalizing into 2NF and 3NF

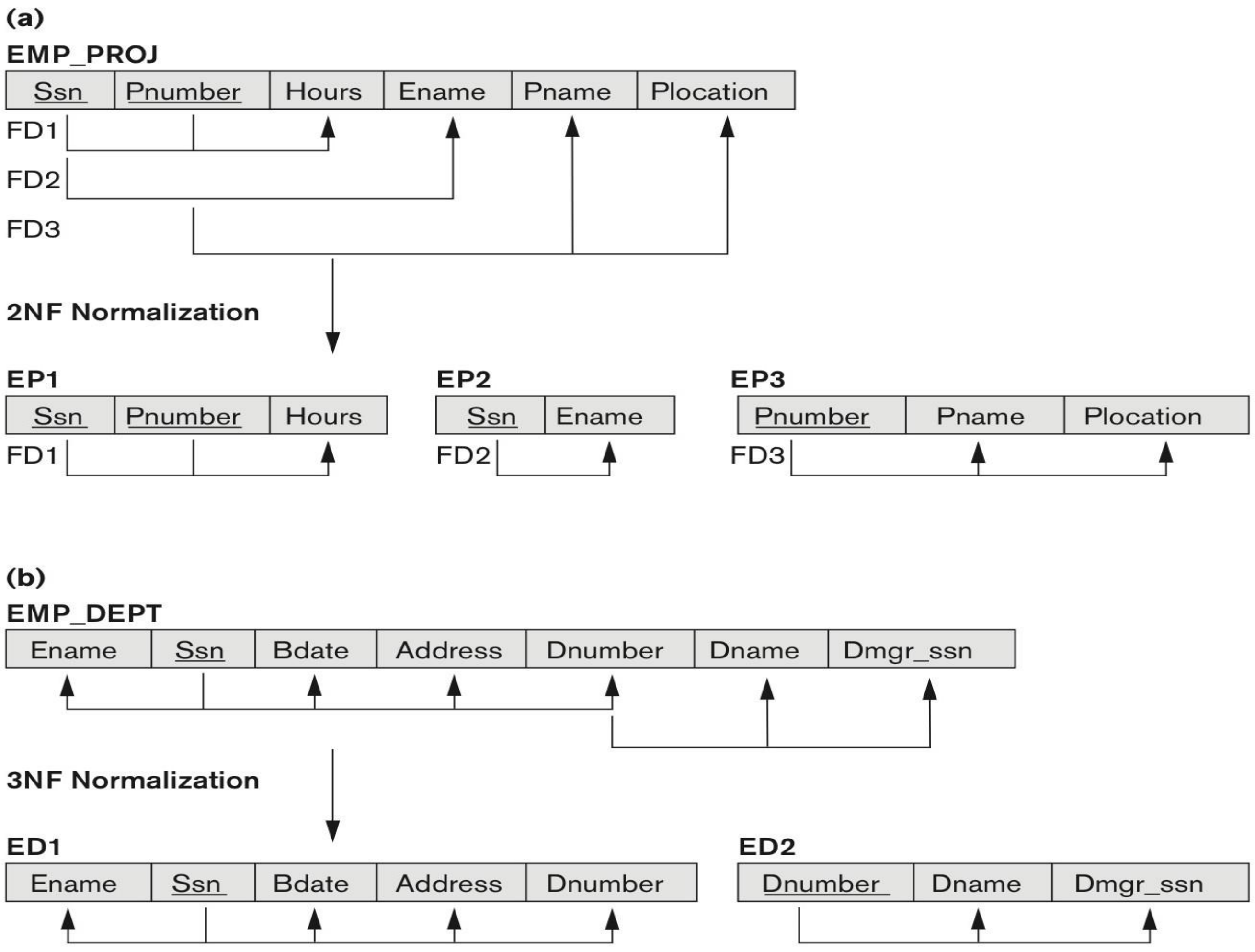


Figure 14.11
Normalizing into 2NF and 3NF. (a) Normalizing EMP_PROJ into 2NF relations. (b) Normalizing EMP_DEPT into 3NF relations.

3.6 Third Normal Form (3NF)

- Definition:
  - **Transitive functional dependency: is** 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 (R) is in 3NF if it is in 2NF and no non prime attribute of R is transitively dependent on the primary/candidate key. That is, when both of these conditions hold:
  - (a) R is fully functionally dependent on every key of R
  - (b) R is non-transitively dependent on every key of R
- 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

Normal Forms Defined Informally

- 1st normal form
  - All attributes depend on **the key** and are single valued**.**
  - 2nd normal form
  - All attributes depend on **the whole key**
    (ie, all attributes are fully functional dependent on the whole key).
- 3rd normal form
  - All attributes depend on **nothing but the key**
    (ie, no attribute should depend on the key through another attribute: no transitive dependency)

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 $X \rightarrow A$ 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 proper subset of a key 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.

5. BCNF (Boyce-Codd Normal Form)

- A relation schema R is in **Boyce-Codd Normal Form (BCNF)** if whenever an **FD X** $\rightarrow$ **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
- Hence BCNF is considered a stronger form of 3NF
- The goal is to have each relation in BCNF (or 3NF)
- If there exists some **FD X** $\rightarrow$ **A** that holds in a relation schema, R where X is not a superkey (ie, a non-key attribute) and A is a prime attribute (part of the key), R will be in 3Nf but not in BCNF.

Figure 14.13 Boyce-Codd normal form

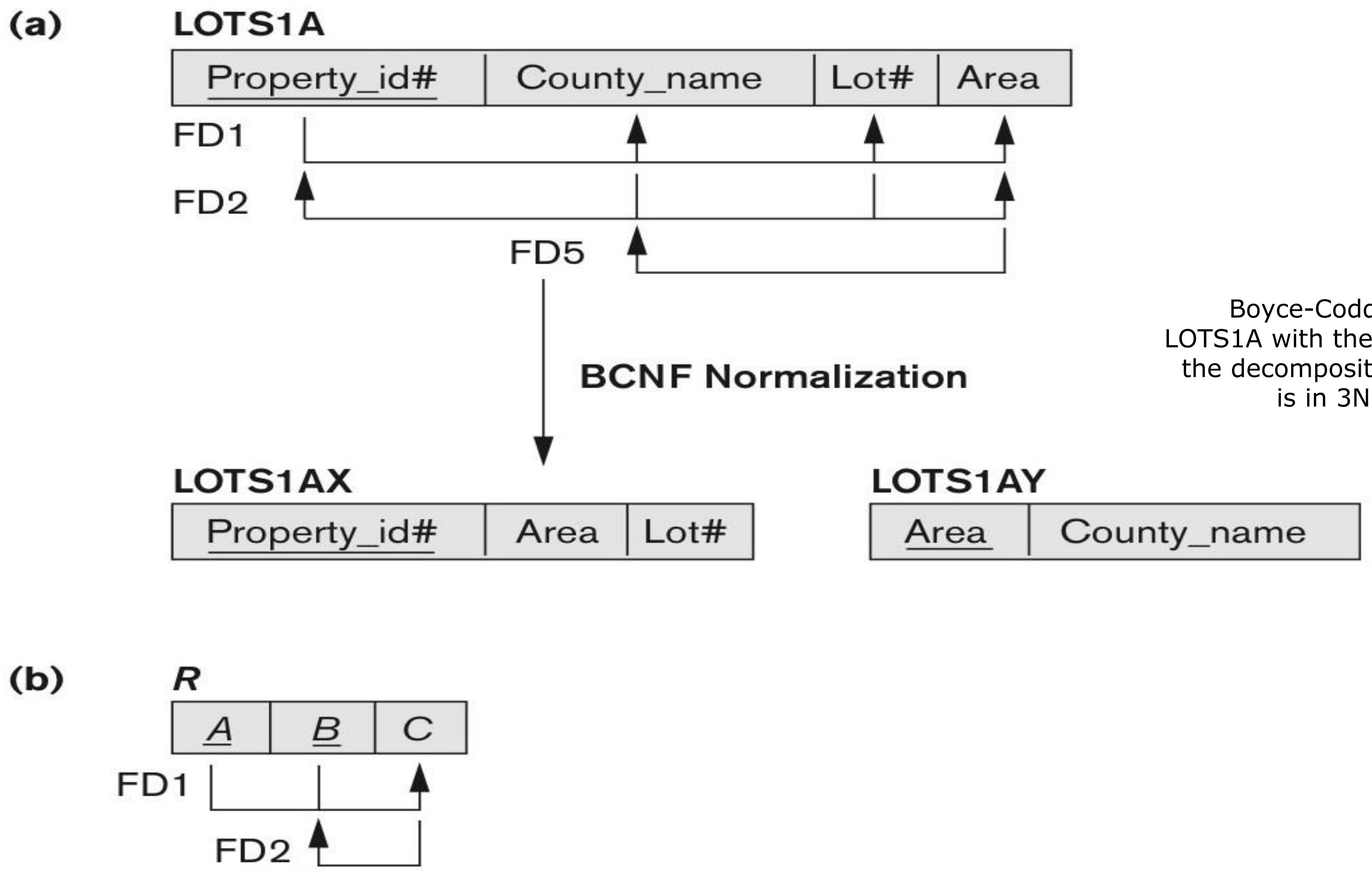


Figure 14.13
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. $C \rightarrow B$.

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

TEACH

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

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

Achieving the BCNF by Decomposition (2)

- 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.
- Three possible decompositions for relation TEACH
  - D1: {<u>student, instructor</u>} and {<u>student, course</u>}
  - D2: {course, <u>instructor</u> } and {<u>course, student</u>}
  - D3: {<u>instructor</u>, course } and {<u>instructor, student</u>} ✓

Figure 14.2 Sample database state for the relational database schema in Figure

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	291 Berry, 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