CSC 261/461 – Database Systems
Lecture 9

Spring 2017
MW 3:25 pm – 4:40 pm
January 18 – May 3
Dewey 1101

Announcement

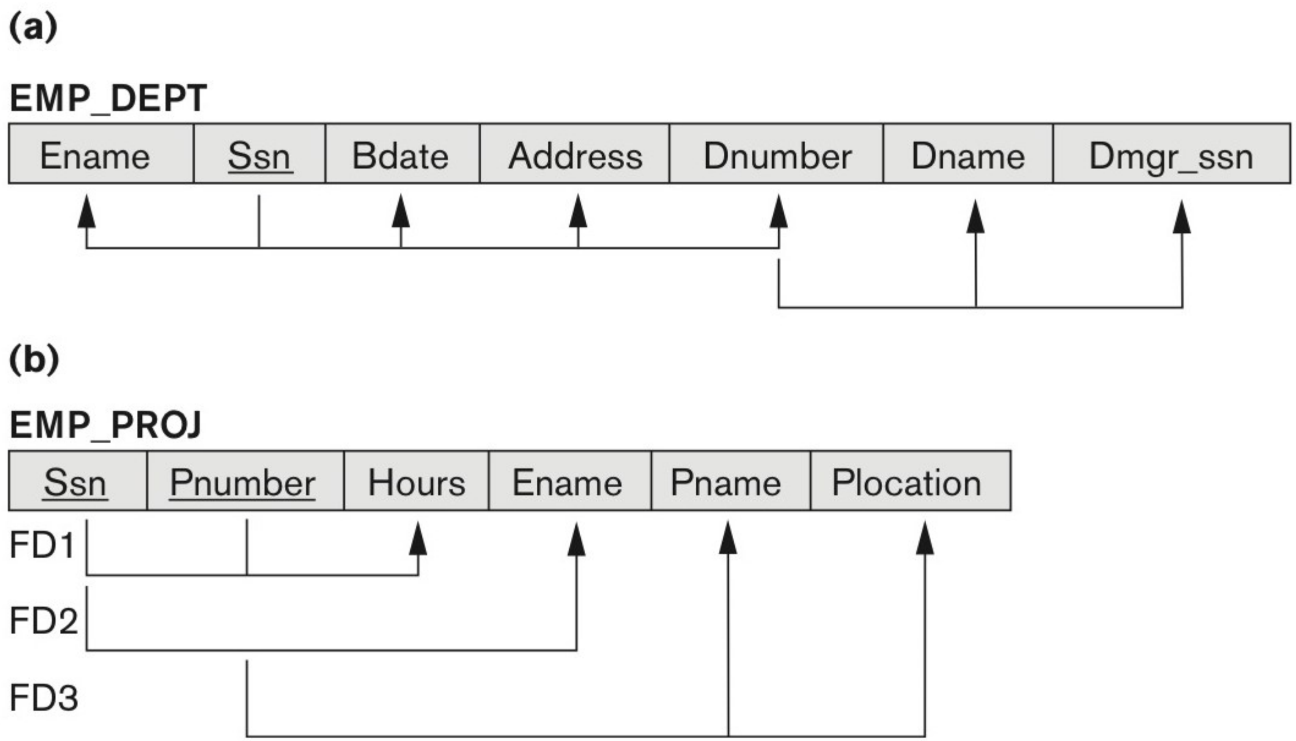
- Project 1 Milestone II is out.
  - Read Chapter 3 and 9 before your submit it.
- Project II part 1 will be released on Monday
- Read your textbook!
  - Chapter 8:
    - Will cover later; But self-study the chapter
    - Everything except Section 8.4
  - Chapter 14:
    - Section 14.1 – 14.5
  - Chapter 15:
    - Section 15.1 – 15.4
    - Will finish on Monday

Today's Lecture

1. 2NF, 3NF and Boyce-Codd Normal Form

2. Decompositions (Next Lecture)

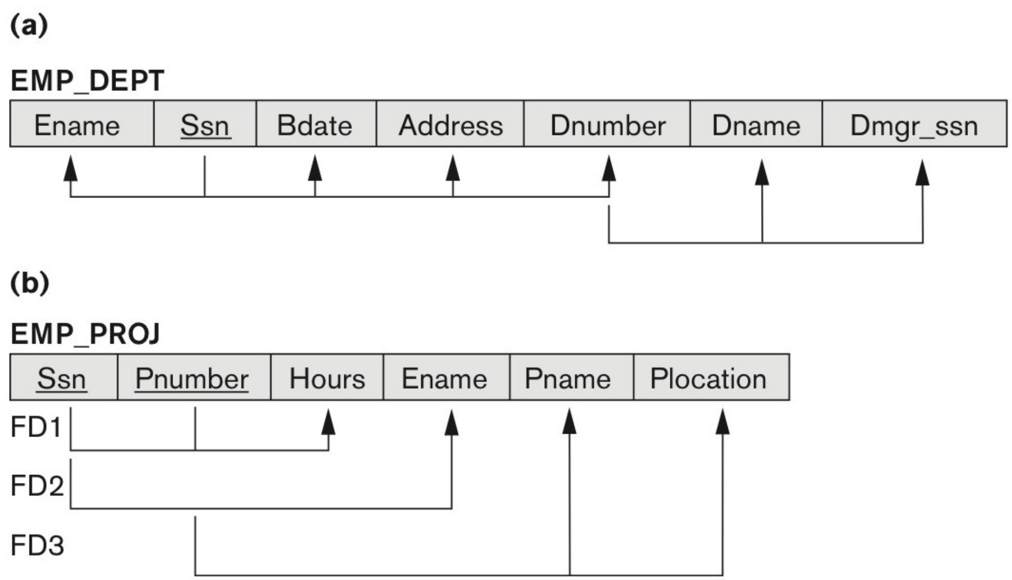
Functional Dependencies (Graphical Representation)



Prime and Non-prime attributes

- 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.



Back to Conceptual Design

Now that we know how to find FDs, it's a straight-forward process:

1. Search for "bad" FDs

2. If there are any, then *keep decomposing the table into sub-tables* until no more bad FDs

3. When done, the database schema is *normalized*

Boyce-Codd Normal Form (BCNF)

- Main idea is that we define "good" and "bad" FDs as follows:
  - X → A is a "*good FD" if X is a (super)key*
    - In other words, if A is the set of all attributes
  - X → A is a *"bad FD"* otherwise
- We will try to eliminate the "bad" FDs!
  - Via normalization

Second Normal Form (1)

- Uses the concepts of **FDs, primary key**
- Definitions
  - **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 or "second normalization"

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 -> Dmgr_ssn is a **transitive** FD
    - Since Ssn -> Dnumber and Dnumber -> Dmgr_ssn 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

Normalizing into 2NF and 3NF

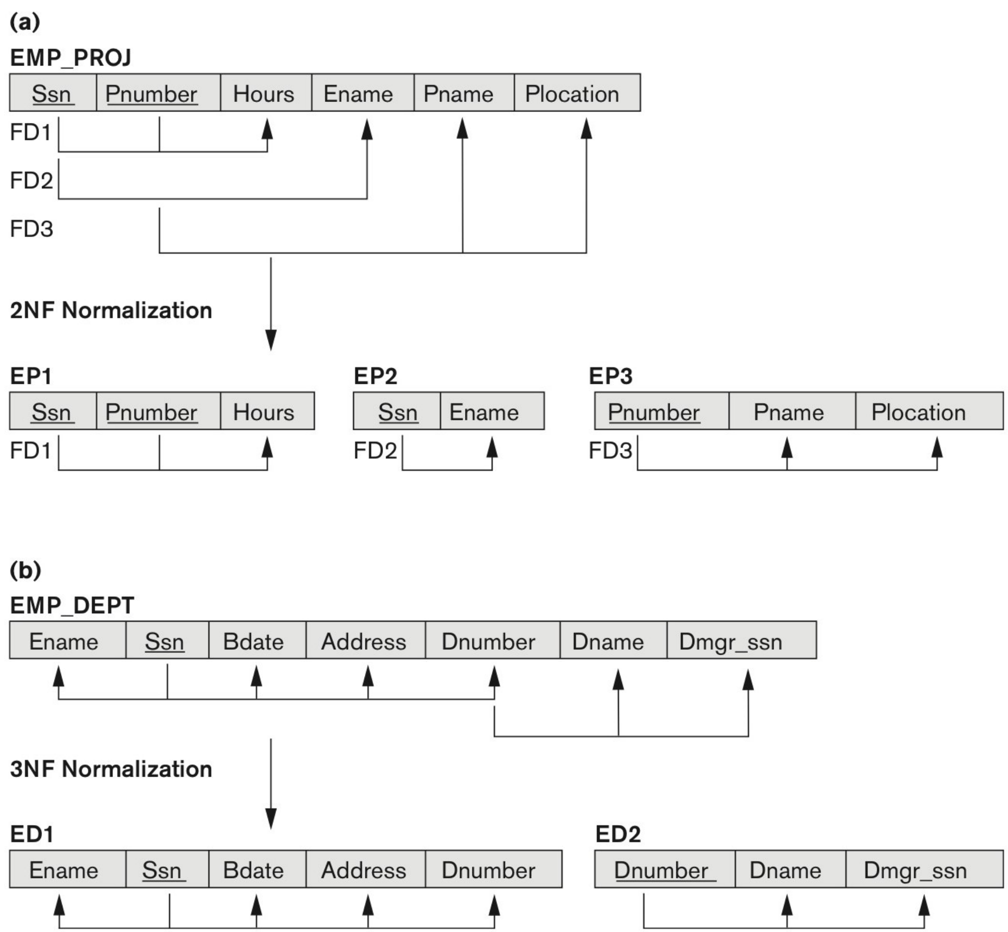
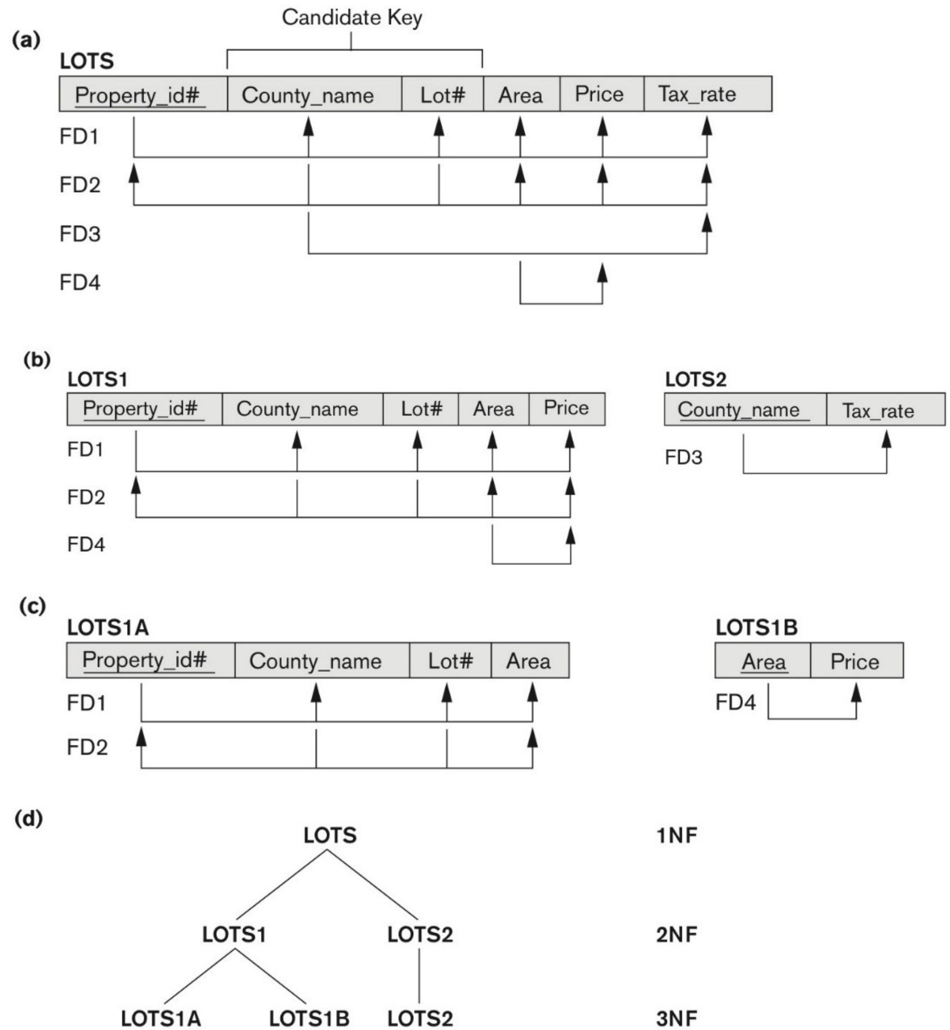


Figure 14.12 Normalization into 2NF and 3NF

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.



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**

General Definition of 2NF and 3NF (For Multiple Candidate Keys)

- A relation schema R is in **second normal form (2NF)** if every non-prime attribute A in R is fully functionally dependent on ***every*** key of R

- 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 ***any*** key of R

4.3 Interpreting the General Definition of Third Normal Form (2)

- **ALTERNATIVE DEFINITION of 3NF: We can restate the definition as:**

  A relation schema R is in **third normal form** (3NF) if, whenever a nontrivial FD X→A holds in R, either

  a) X is a superkey of R or

  b) A is a prime attribute of R

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

1. BOYCE-CODD NORMAL FORM

What you will learn about in this section

1. Conceptual Design

2. Boyce-Codd Normal Form

3. The BCNF Decomposition Algorithm

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

Figure 14.13 Boyce-Codd normal form

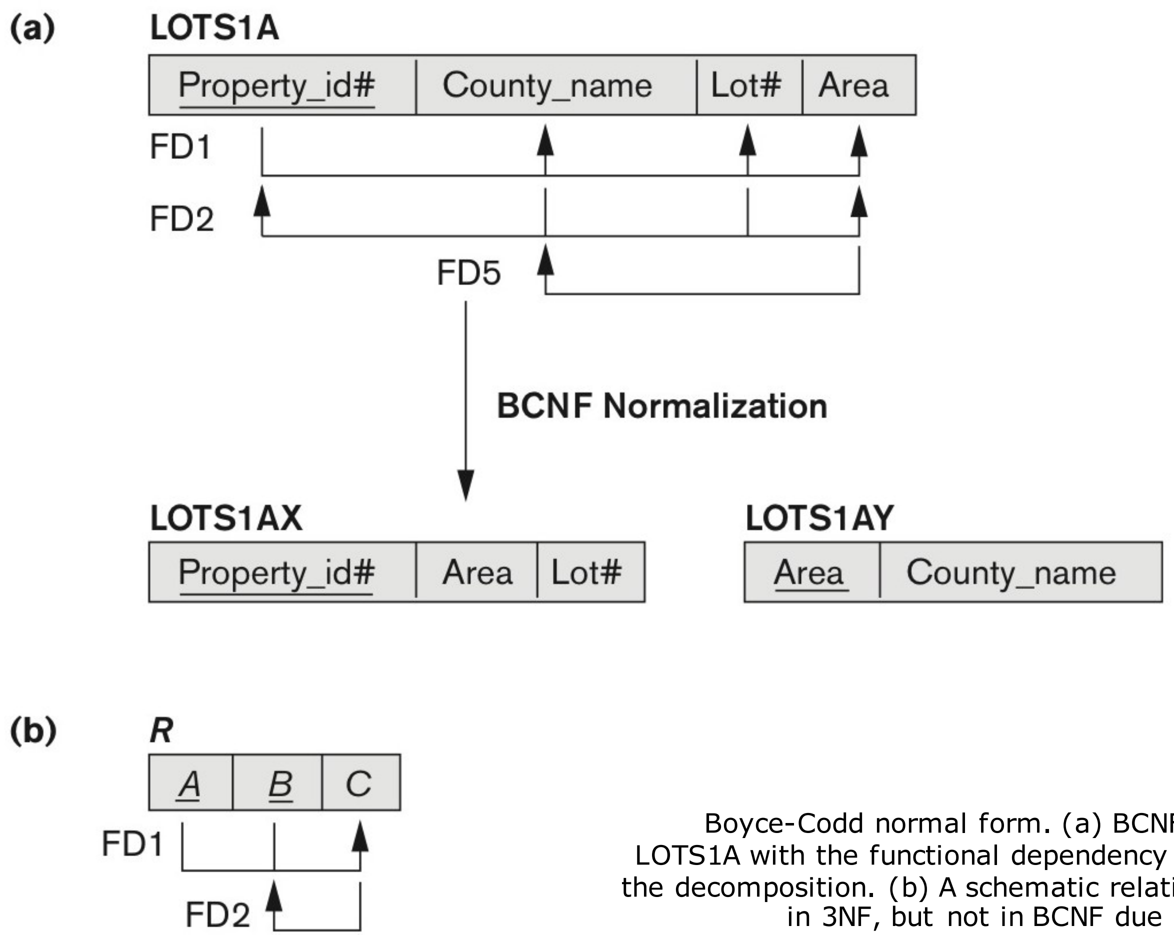


Figure 14.13
Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. $C \rightarrow B$.

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

- Two FDs exist in the relation TEACH:
  - fd1: { student, course} -> instructor
  - fd2: instructor -> course
- {student, course} is a candidate key for this relation
- 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.

Achieving the BCNF by Decomposition (2)

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

Acknowledgement

- Some of the slides in this presentation are taken from the slides provided by the authors.
- Many of these slides are taken from cs145 course offered by Stanford University.
- Thanks to YouTube, especially to Dr. Daniel Soper for his useful videos.