Database Management System – 32 Database design – General Normal Form Definition (2NF, 3NF and BCNF)

Ajay James
Asst. Prof in CSE
Government Engineering College Thrissur

Outline

- General Normal Form
- General definition of 2NF
- General definition of 3NF
- BCNF

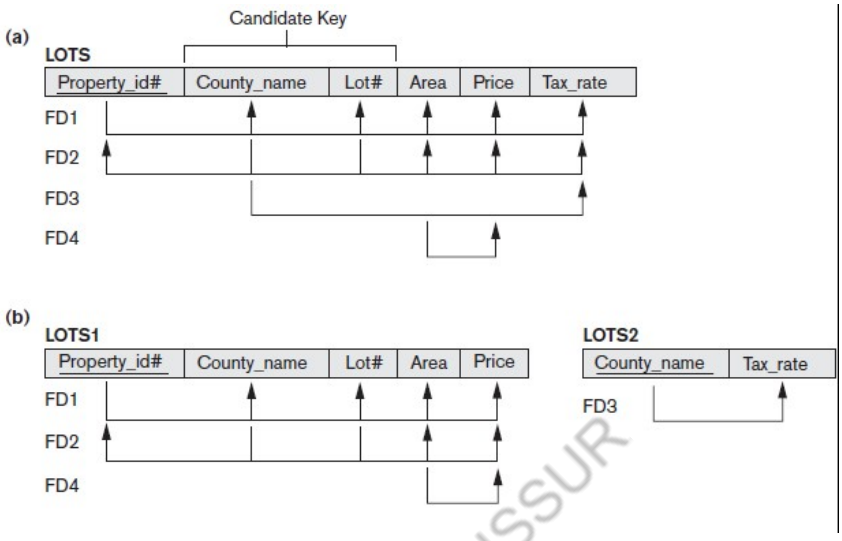
General Normal Form Definitions (For Multiple Keys)

- Previous definitions consider the primary key only
- General definitions - relations with ***multiple candidate keys***
- Any attribute involved in a **candidate key** is a ***prime attribute***
- All other attributes are called ***non-prime attributes***

General Definition of 2NF

- 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
- Every key means all candidate keys

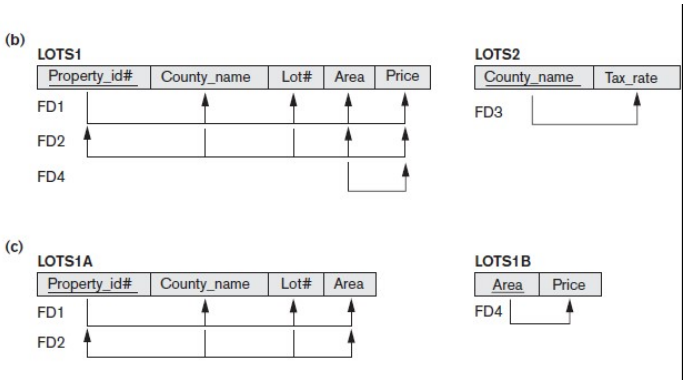
2NF Normalization example



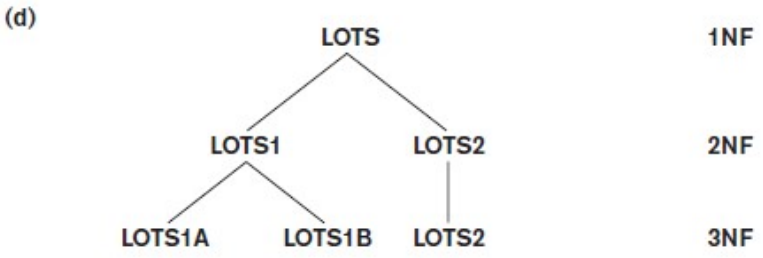
General Definition of 3 NF

- 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

3 NF example



Normalization example



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 → 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 prime attribute functionally determines a non-prime attribute
    - **catches 2NF violations** due to non-full functional dependencies
  - second, where a non-prime attribute functionally determines a non-prime attribute.
    - **catches 3NF violations** due to a transitive dependency.

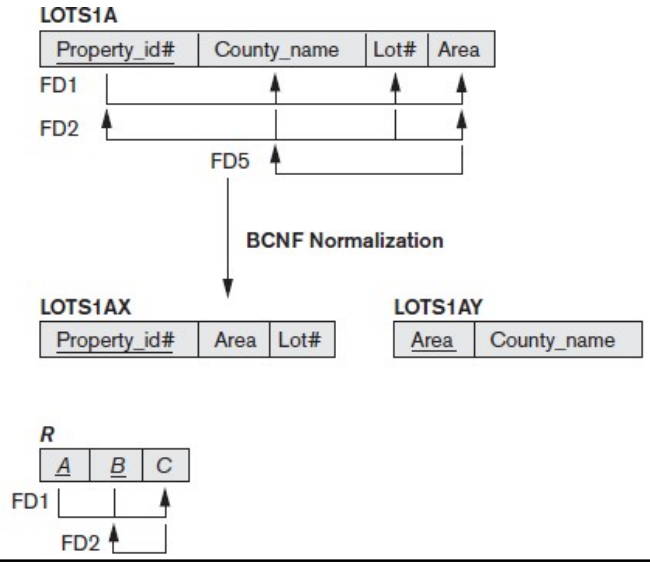
ALTERNATIVE DEFINITION of 3NF

- A relation schema R is in third normal form (3NF)if **every non-prime attribute** in R meets both of these conditions:
  - It is **fully functionally dependent** on **every key** of R
  - It is **non-transitively dependent** on **every key** of R

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
- Hence BCNF is considered a stronger form of 3NF
- The goal is to have each relation in BCNF (or 3NF)

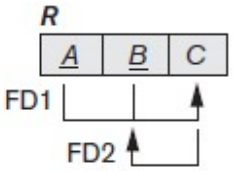
BCNF Example



BCNF Example

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



- FD1: {Student, Course} → Instructor
- FD2:Instructor → Course

1. R1 (Student, Instructor) and R2(Student, Course)
2. R1 (Course, Instructor) and R2(Course, Student)
3. R1 (Instructor, Course) and R2(Instructor, Student)

Exercise 7

A	B	C	TUPLE#
10	b1	c1	1
10	b2	c2	2
11	b4	c1	3
12	b3	c4	4
13	b1	c1	5
14	b3	c4	6

1. Which of the following dependencies may hold in the above relation? If the dependency cannot hold, explain why by specifying the tuples that cause the violation.

i. A → B, ii. B → C, iii. C → B, iv. B → A, v. C → A

Exercise 7

2. Consider the relation R, which has attributes that hold schedules of courses and sections at a university;

R = {Course_no, Sec_no, Offering_dept, Credit_hours, Course_level, Instructor_ssn, Semester, Year, Days_hours, Room_no, No_of_students}.

Suppose that the following functional dependencies hold on R:

1. {Course_no} → {Offering_dept, Credit_hours, Course_level}
2. {Course_no, Sec_no, Semester, Year} → {Days_hours, Room_no, No_of_students, Instructor_ssn}
3. {Room_no, Days_hours, Semester, Year} → {Instructor_ssn, Course_no, Sec_no}

Try to determine which sets of attributes form keys of R. How would you normalize this relation?

Exercise 7

3. Consider the following relation:

CAR_SALE(Car#, Date_sold, Salesperson#, Commission%, Discount_amt)

Assume that a car may be sold by multiple salespeople, and hence {Car#, Salesperson#} is the primary key.

Additional dependencies are

Date_sold → Discount_amt and

Salesperson# → Commission%

Based on the given primary key, is this relation in 1NF, 2NF, or 3NF? Why or why not? How would you successively normalize it completely?

Reference

- Elmasri R. and S. Navathe, Database Systems: Models, Languages, Design and Application Programming, Pearson Education 6th edition and 7th edition

Thank you