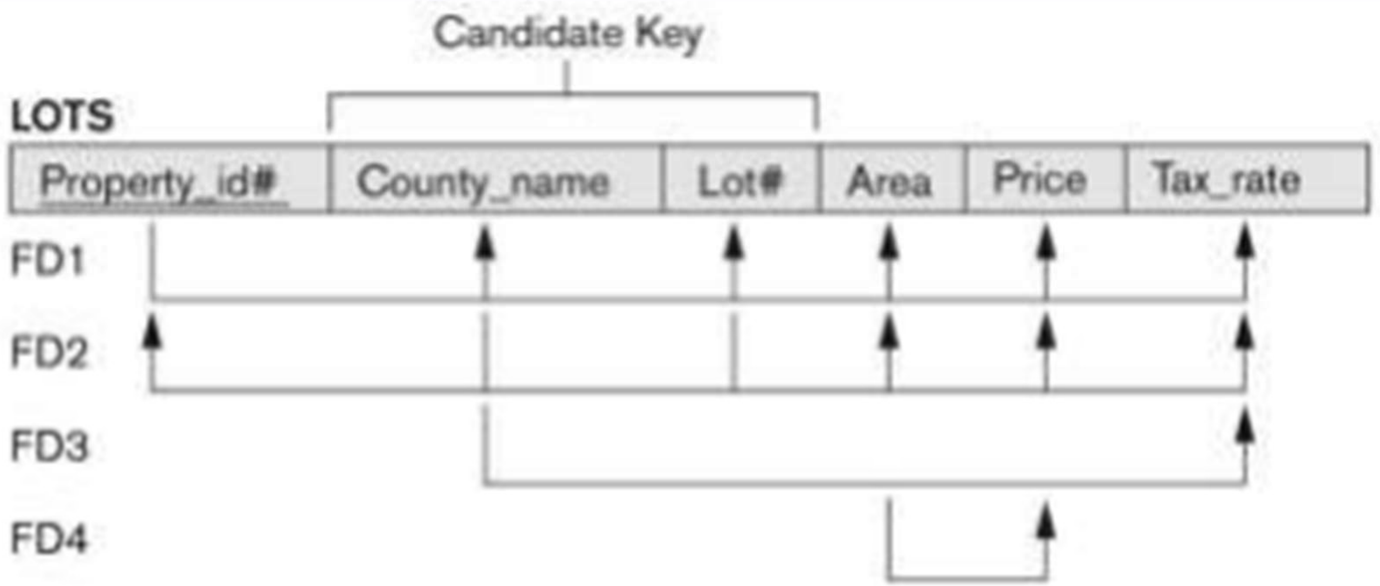
Unit-3

Cont…

Example for Normalization



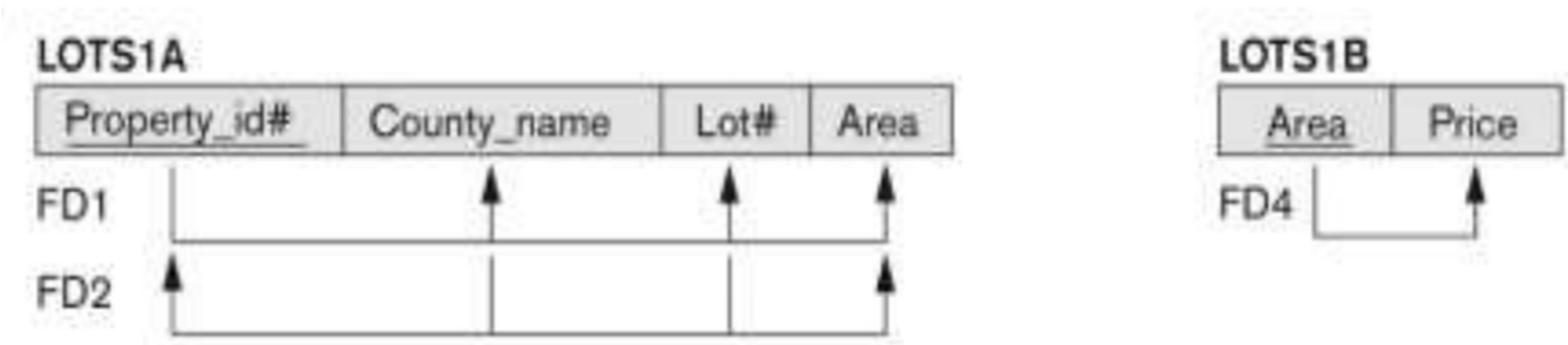
- The above relation describes parcels of land for sale in various county of a state. Suppose there are 2 candidate keys as:
- 1. Property_ID
- 2. County_Name and Lot_No

2 NF Conversion

- Remove partial functional dependency
- County_Name -> Tax_Rate

3 NF Conversion of relation LOT1

- Here remove transitive dependency
- Area -> Price   and
- Property_Id -> Area



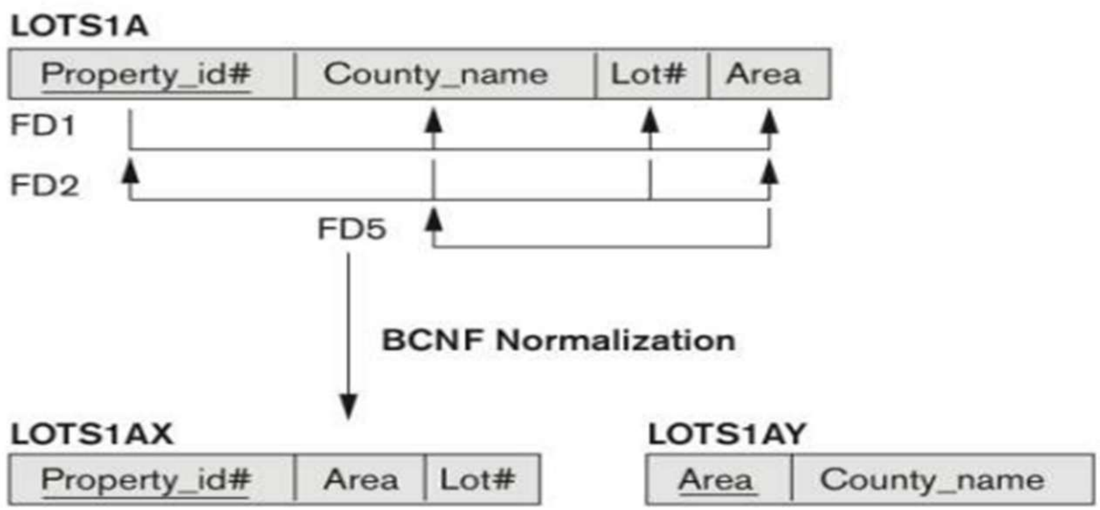
BCNF – Boyce Codd Normal Form

- BCNF is the advance version of 3NF. It is stricter than 3NF.
- A table is in BCNF if every functional dependency X → Y,
    - **X is the super key of the table**.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

- General definition of 3NF is:
    - For any FD X->Y:
        I. X is a super Key of R OR
        II. Y is a Prime Attribute of R

BCNF

- In the above relation LOTs1A if we assume that the Lot size/ area is fix per county and they are from 2 county only
- So in that case we can add one more FD i.e. Area -> County_Name
- So, according to 3NF this is allowed FD as: Area is not a super key according to rule (i) but as rule (ii) says County_Name is a prime attribute.
- But according to BCNF only one condition is allowed for FD is that the determinant must be a super key. So, Need to convert the relation LOTs1A into BCNF as shown below:



BCNF Example

- Candidate Key:
- {Student,Course}

- FDs:
- {Student,Course} -> Instructor
- Instructor -> Course

- Decomposition Rule:

$((R1 \cap R2) \rightarrow (R1 - R2))$ is in F^+, or
$((R1 \cap R2) \rightarrow (R2 - R1))$ is in F^+.

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

Conversion into BCNF

D1: {student, instructor} and {student, course}

D2: {course, instructor } and {course, student}

D3: {instructor, course } and {instructor, student} ✓

- Out of all above 3 decomposition
- Only D3 follows the decomposition rule for BCNF
- So D3 is the only valid decomposition

Examples for Normalization

R(StudentID, StudentName, CourseID, CourseName, InstructorID, InstructorName, Dept, Grade)

Given functional dependencies (FDs)

FD1 : StudentID → StudentName

FD2 : CourseID → CourseName, InstructorID

FD3 : InstructorID → InstructorName, Dept

FD4 : {StudentID, CourseID} → Grade

Explanation

- **Key:** {StudentID, CourseID}
- After 2NF:

  R1: Student (StudentID , StudentName)

  R2: Course (CourseID, CourseName, InstructorID, InstructorName, Dept)

  R3: Enrollment (StudentID, CourseID, Grade)
- After 3NF:

  R1 and R3 will remain as it is and remove transiti FD from R2

  R2a : Instructor(InstructorID PK, InstructorName, Dept)

  R2b : Course(CourseID PK, CourseName, InstructorID)

Examples for Normalization

R(StuID, StuName, Dept, CourseID, CourseName, Instructor, RoomNo)

StuID → StuName, Dept(Student ID uniquely identifies student info)

CourseID → CourseName, Instructor(Each course has a fixed name and instructor)

Instructor → RoomNo(An instructor is assigned a specific room for lectures)

(StuID, CourseID) → (all other attributes)

(Student enrollment in a course is uniquely identified by StuID+CourseID)

Decomposition

- R = (StuID, StuName, Dept, CourseID, CourseName, Instructor, RoomNo) already in 1NF
- Primary key = (StuID, CourseID) (composite).

- In 2NF, no non-prime attribute should depend on part of composite key.
- Check partial dependencies:
- StuID → StuName, Dept → partial dependency (depends only on StuID). ❌
- CourseID → CourseName, Instructor → partial dependency (depends only on CourseID). ❌

- So 2NF decompose:
- R1(StuID, StuName, Dept)
- R2(CourseID, CourseName, Instructor, RoomNo)
- R3(StuID, CourseID)

3NF

Check transitive dependencies:

- In R2: CourseID → Instructor and Instructor → RoomNo

→ CourseID → RoomNo (transitive dependency). ❌

Decompose R2:

- **R2a(CourseID, CourseName, Instructor)**
- **R2b(Instructor, RoomNo)**

Now all transitive dependencies are removed.

So after **3NF decomposition**:

1. R1(StuID, StuName, Dept)
2. R2a(CourseID, CourseName, Instructor)
3. R2b(Instructor, RoomNo)
4. R3(StuID, CourseID)