Normal Forms

R&G Chapter 19

Normal Forms

- Back to the problem of schema refinement.
- A fundamental question: To eliminate data redundancy, is any refinement needed??
- Answer: determine if any refinement is needed based on the type of *normal form* that a relation is in:
    - If a relation is in a certain *normal form*, certain problems are avoided/minimized.
    - Normal forms can help decide whether decomposition (i.e., splitting tables) will be needed to eliminate data redundancy.

Normal Forms (NF)

- **Types: 1st, 2nd, 3rd, Boyce-Codd (BCNF or 3.5 NF)**
  - The higher the normal form is, the stricter constraints are put on the database
  - Order of checking normal forms: 2nd NF, 3rd NF, BCNF.

1st Normal Form

- **First Normal Form (1NF):**
  - Equivalent to the definition of relational model.
  - A relational schema is in 1NF if and only if the domains of all attributes of R are ***atomic***
    - A domain is atomic if elements of the domain are considered to be indivisible units.

Example

Course	Student
CS442	Alan Betty Carol

- **NOT 1NF form**

To make it satisfy 1NF

Course	Student
CS442	Alan
CS442	Betty
CS442	Carol

2nd Normal Form

- ***Partial dependency***
  - An FD X ->Y is said to be a *partial dependency* if there exists an FD Z ->Y such that Z $\subseteq$ X (i.e., Z is a subset of X).
- **Second Normal Form (2NF):**

A relation schema R is in 2NF if there is no partial dependency

Does the table satisfy 2NF ?

Staff

ENO	Name	Dno	DeptName	ProjNo	ProjName
E001	Somchai	D01	Physic	P01	NMR
E001	Somchai	D01	Physic	P02	Laser
E002	Sompong	D01	Physic	P03	Medical Image processing
E003	Somchay	D02	Computer Science	P05	Voice ordering
E003	Somchay	D02	Computer Science	P04	Speech Coding
E004	SomSiri	D02	Computer Science	P04	Speech coding
E004	SomSiri	D02	Computer Science	P06	Speech Synthesis

KEY = ENO + ProjNo

Answer is No. Because ProjName is dependent on ProjNo, a part of the key

Check Violation of 2NF

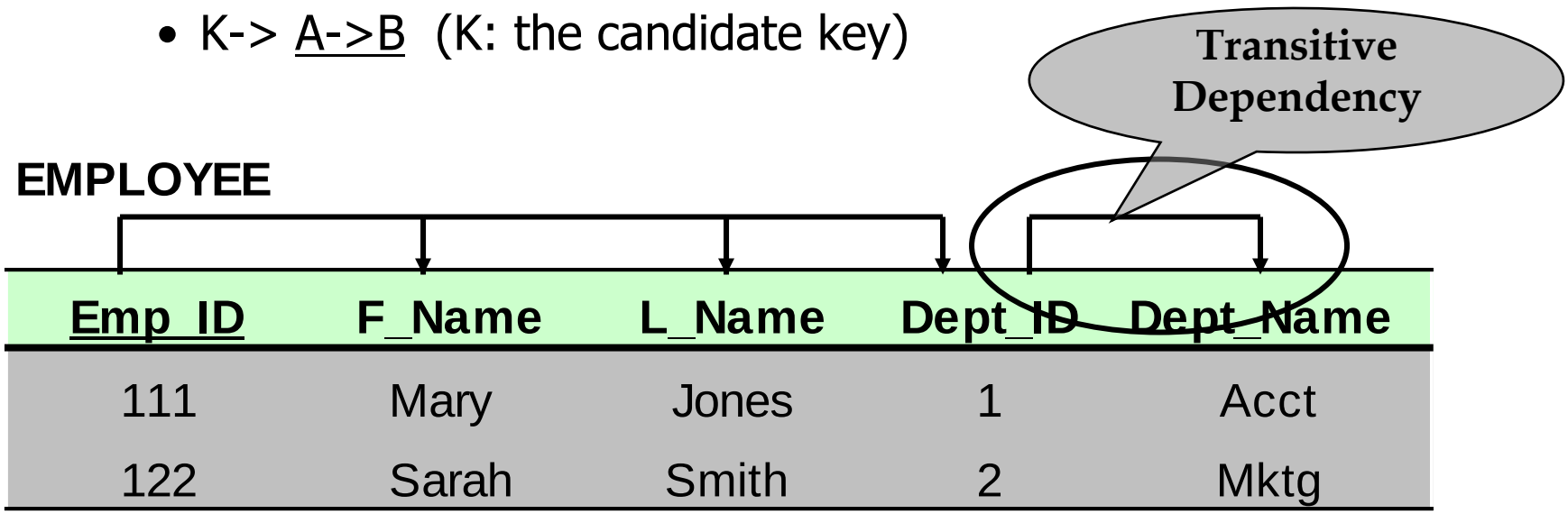
- Given relation R and its FDs F, check whether some attributes *Y partially depends on a candidate key X*
  - Check if there exists a FD X->Y such that:

    (1) X is a subset of candidate keys of R;

    (2) Y is a non-key attribute (i.e., Y does not appear in any candidate key)
  - If there is such FD, then R violates 2NF! (because X->Y is a partial dependency)

3rd Normal Form

- **Transitive dependency**
  - There exist transitive dependency when a non-key attribute A determines another non-key attribute B.
    - K-> A->B (K: the candidate key)



3rd Normal form

- **Relation R is said to be in 3rd Normal Form if:**
  - ***No partial dependency*** (i.e., the relation R is in 2NF)

  AND

  - ***No transitive dependency***: For each candidate key K of R, all non-key attributes of R are directly dependent on K.

Check Violation of 3rd Normal Form

- **There are 2 cases when a FD: X-> Y violates 3NF**

  Case 1: X is a subset of a candidate key K.

  - This is a partial dependency, as K -> Y too.

  Case 2: X is not a subset of a candidate key and Y is a non-key attribute (i.e., Y is not included in any candidate key).

  - This is a *transitive dependency*, as it has dependencies K ->X -> Y.

Shortcut Rules

- **Shortcut Rule 1**: If all candidate keys are singleton keys (i.e., only contain one attribute), then R must satisfy 2NF
- **Shortcut Rule 2**: If all attributes are part of some candidate keys, then R must be 2NF & 3NF (WHY?)

You can use these 2 rules and make the quick conclusion of normal forms without formal proof

Boyce-Codd Normal Form (BCNF)

- **Also called 3.5NF.**
- **Reln R with FDs *F* is in BCNF if, for all X -> Y in F, it satisfies the following condition:**
  - X is a superkey for R.
- **In other words: "Nothing but the key"**

3NF VS BCNF

- **Given Reln R and its FDs F, for each FD X -> Y in F,**
  - 3NF requires FD satisfies one of the two conditions:
    - X is a candidate key for R,
    - X is not a candidate key and Y is part of some candidate key (Y can be a single attribute)
  - BCNF requires FD satisfies one condition:
    - X is a superkey.

Example of 3NF VS. BCNF

- **R(ABC)**
- **R has a key: (A,B)**
- **F={C->B}**
- **Does R satisfy BCNF?**
  - No, because in C->B, C is not a key
- **Does R satisfy 3NF?**
  - Yes, because B is a part of the key

Example of 3NF VS. BCNF

- **R(ABC)**
- **F={AB->C, C->A}**
- **What normal form does R satisfy?**
  - Step 1: Find the candidate keys of R:
    - AB, BC.
  - Step 2: Does it satisfy 2NF?
    - All attributes are part of some candidate key, so it satisfies 2NF (Shortcut Rule 2).
  - Step 3: Does it satisfy 3NF?
    - AB->C is OK since AB is a candidate key; C->A is OK as A is included in a candidate key. So it satisfies 3NF.
  - Step 4: Does it satisfy BCNF?
    - In C->A, C is not a candidate key. So it does NOT satisfy BCNF.

Normal Form Summary

- **Types: 1st, 2nd, 3rd, BCNF**

Normal Form	Constraint
1NF	Atomic value
2NF	No partial dependency (i.e., there does not exist an FD X->A such that X is a subset of a candidate key and A is a non-key attr.)
3NF	No partial dependency & No transitive dependency (i.e., for each FD X->A, either X is a candidate key or A is a subset of a candidate key)
BCNF (3.5 NF)	All non-trivial FDs are key FDs (i.e., for each X->A, X is a superkey)