Database Design: Boyce-Code Normal Form

Abdu Alawini

University of Illinois at Urbana-Champaign

CS411: Database Systems

Normal Forms

First Normal Form = all attributes are atomic

Second Normal Form (2NF) = old and obsolete

Boyce Codd Normal Form (BCNF)

Third Normal Form (3NF)

Others...

Learning Objectives

After this lecture, you should be able to:

- Decompose a database schema into a set of relations obeying BCNF

Eliminating Anomalies

Main idea:

- X → A is OK, if X is a (super)key
- X → A is NOT OK, otherwise
  - Need to decompose the table, but how?

Boyce-Codd Normal Form (BCNF)

Boyce-Codd Normal Form

Definition. A relation R is in BCNF if and only if:

Whenever there is a nontrivial FD: $A_1A_2\ldots A_n \rightarrow B$, then $A_1A_2\ldots A_n$ is a superkey for R.

There are no "bad" FDs: whenever there is a nontrivial FD, its left side must be a superkey

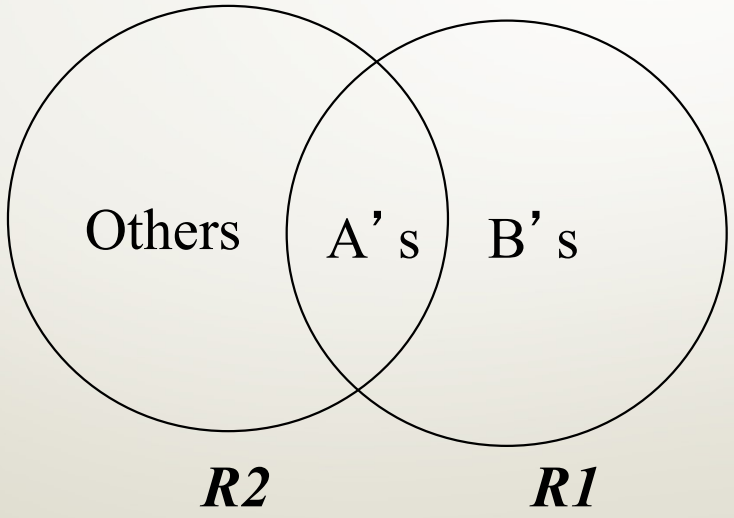
BCNF Decomposition

Find a dependency that violates the BCNF condition:

$$A_1, A_2, \ldots A_n \longrightarrow B_1, B_2, \ldots B_m$$

Heuristic : choose $B_1, B_2, \ldots B_m$ "as large as possible"

Decompose:



Continue until there are no BCNF violations left.

Example Decomposition

Person:

Name	SSN	Age	EyeColor	Phone

Functional dependencies:

SSN ➔ Name, Age, Eye Color

BCNF: Person1(SSN, Name, Age, EyeColor),
Person2(SSN, Phone)

BCNF Decomposition: The Algorithm

Input: relation R, set S of FDs over R

1) Check if R is in BCNF, if not:

   a) pick a violation FD f: A → B

   b) compute A+

   c) create R1 = A+, R2 = A union (R – A+)

   d) compute all FDs over R1, using R and S. Repeat similarly for R2. (**See Algorithm 3.12**)

   e) Repeat Step 1 for R1 and R2

2) Stop when all relations are BCNF, or are two-attributes

(Two attribute relations are always in BCNF, see E.g. 3.17 (pg. 89) for proof and examples)

Another Example

- Person (Name, SSN, Age, EyeColor, Phone, HairColor)
- FD 1: SSN → Name, Age, EyeColor
- FD 2: Age → HairColor

FD 1 and 2 imply: SSN → Name, Age, EyeColor, HairColor

Iteration 1: Split based on SSN → Name, Age, EyeColor, HairColor

- Person(SSN, Name, Age, EyeColor, HairColor)
- Phone(SSN, Phone)

Iteration 2: Split based on Age → HairColor

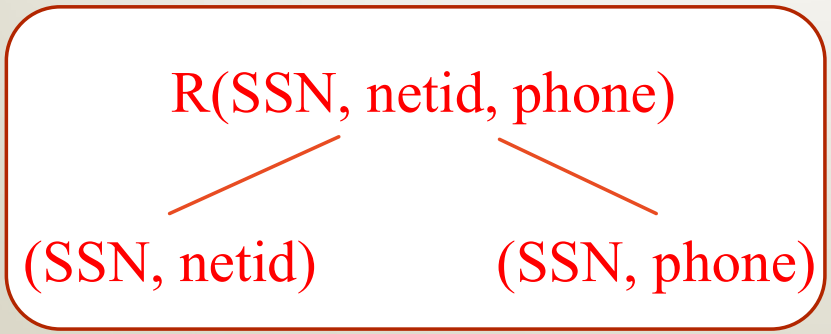
- Person(SSN, Name, Age, EyeColor)
- Hair(Age, HairColor)
- Phone(SSN, Phone)

Q: Is BCNF Decomposition unique?

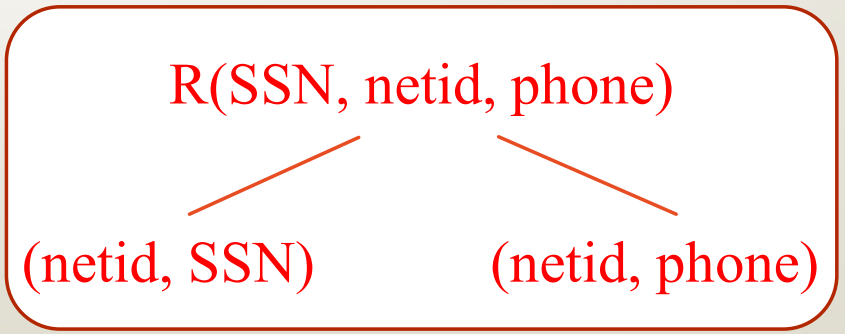
- R(SSN, netid, phone).
  - FD1: SSN -> netid
  - FD2: netid -> SSN
- Each of these two FDs violates BCNF.

Can you tell me two different BCNF decomp for R?

Pick FD1



Pick FD2



Properties of BCNF

- BCNF removes certain types of redundancies
  - All redundancies based on FDs are removed.
- BCNF Decomposition avoids information loss
  - You can construct the original relation instance from the decomposed relations' instances.

Desirable Properties of Schema Refinement

- ✓ 1) minimize redundancy
- ✓ 2) avoid info loss
- 3) preserve dependency
- 4) ensure good query performance