Relational Database Design
(Normalization part 2)

Chapter 8: Relational Database Design

- Features of Good Relational Design
- Atomic Domains and First Normal Form
- Decomposition Using Functional Dependencies
- Functional Dependency Theory
- Algorithms for Functional Dependencies
- Decomposition Using Multivalued Dependencies
- More Normal Form
- Database-Design Process

Multivalued Dependencies

- Suppose we record names of children, and phone numbers for instructors:
  - *inst_child*(*ID*, *child_name*)
  - *inst_phone*(*ID*, *phone_number*)
- If we combine these schemas we get
  - *inst_info*(*ID*, *child_name*, *phone_number*)

  - Example data:

(99999, David, 512-555-1234)
(99999, William, 512-555-4321)
(99999, David, 512-555-4321)
(99999, William, 512-555-1234)

- This relation is in BCNF
  - Why?

Multivalued Dependencies (MVDs)

- Let R be a relation schema with attributes partitioned into 3 nonempty subsets.

$$Y, Z, W$$

- We say that $Y \rightarrow\rightarrow Z$ (Y **multidetermines** Z) if and only if for all possible relations $r(R)$

$$< y_1, z_1, w_1 > \in r \text{ and } < y_1, z_2, w_2 > \in r$$

then

$$< y_1, z_1, w_2 > \in r \text{ and } < y_1, z_2, w_1 > \in r$$

y_1	z_1	w_1
y_1	z_2	w_2
y_1	z_1	w_2
y_1	z_2	w_1

(99999, David, 512-555-1234)
(99999, William, 512-555-4321)
(99999, David, 512-555-4321)
(99999, William, 512-555-1234)

- Note that since the behavior of Z and W are identical it follows that $Y \rightarrow\rightarrow Z$ if $Y \rightarrow\rightarrow W$

Example

- In our example:

$$ID \rightarrow\rightarrow child\_name$$
$$ID \rightarrow\rightarrow phone\_number$$

- The above formal definition is supposed to formalize the notion that given a particular value of Y (ID) it has associated with it a set of values of Z $(child\_name)$ and a set of values of W $(phone\_number)$, and these two sets are in some sense independent of each other.

(99999, David, 512-555-1234)
(99999, William, 512-555-4321)
(99999, David, 512-555-4321)
(99999, William, 512-555-1234)

- Note:
  - If $Y \rightarrow Z$ then $Y \rightarrow\rightarrow Z$

Use of Multivalued Dependencies

- We use multivalued dependencies in two ways:
  1. To test relations to **determine** whether they are **legal** under a given set of functional and multivalued dependencies
  2. To **specify constraints** on the set of legal relations. We shall thus concern ourselves *only* with relations that satisfy a given set of functional and multivalued dependencies.

Theory of MVDs

- From the definition of multivalued dependency, we can derive the following rule:
  - If $\alpha \rightarrow \beta$, then $\alpha \rightarrow\rightarrow \beta$

  That is, every functional dependency is also a multivalued dependency

- The **closure** D^+ of *D* is the set of all functional and multivalued dependencies logically implied by *D*.
  - We can compute D^+ from *D*, using the formal definitions of functional dependencies and multivalued dependencies.
  - We can manage with such reasoning for very simple multivalued dependencies, which seem to be most common in practice
  - For complex dependencies, it is better to reason about sets of dependencies using a system of inference rules

$\rightarrow\!\!\!\rightarrow$

Fourth Normal Form

- A relation schema R is in **4NF** with respect to a set D of functional and multivalued dependencies if for all multivalued dependencies in D^+ of the form $\alpha \rightarrow\rightarrow \beta$, where $\alpha \subseteq R$ and $\beta \subseteq R$, at least one of the following hold:
  - $\alpha \rightarrow\rightarrow \beta$ is trivial (i.e., $\beta \subseteq \alpha$ or $\alpha \cup \beta = R$)
  - α is a superkey for schema R
- If a relation is in 4NF it is in BCNF

compare BCNF

Restriction of Multivalued Dependencies

- The restriction of D to R_i is the set D_i consisting of
  - All functional dependencies in D^+ that include only attributes of R_i
  - All multivalued dependencies of the form

    $\alpha \rightarrow\rightarrow (\beta \cap R_i)$

    where $\alpha \subseteq R_i$ and $\alpha \rightarrow\rightarrow \beta$ is in D^+

As restriction of Functional dependencies

4NF Decomposition Algorithm

result: = $\{R\}$;
done := false;
compute D^+;
Let D_i denote the restriction of D^+ to R_i
while (**not** *done*)
  if (there is a schema $\mathbf{R}_i$ in *result* that is not in 4NF) **then**
    begin
      let $\alpha \rightarrow\rightarrow \beta$ be a nontrivial multivalued dependency that holds
       on R_i such that $\alpha \rightarrow R_i$ is not in D_i, and $\alpha \cap \beta = \phi$;
      result := $(result - R_i) \cup (R_i - \beta) \cup (\alpha, \beta)$;
    end
  else *done*:= true;
Note: each R_i is in 4NF, and decomposition is lossless-join

Similar to BCNF algorithm (except for $\rightarrow\rightarrow$)

$\twoheadrightarrow$

Example

- $R = (A, B, C, G, H, I)$
  $F = \{ A \rightarrow\rightarrow B$
  $B \rightarrow\rightarrow HI$
  $CG \rightarrow\rightarrow H \}$
- R is not in 4NF since $A \rightarrow\rightarrow B$ and A is not a superkey for R
- Decomposition

$R = (A, B, C, G, H, I)$	(R is not in 4NF, decompose into R_1 and R_2)
a) $R_1 = (A, B)$	(R_1 is in 4NF,)
b) $R_2 = (A, C, G, H, I)$	(R_2 is not in 4NF, decompose into R_3 and R_4)
c) $R_3 = (C, G, H)$	(R_3 is in 4NF)
d) $R_4 = (A, C, G, I)$	(R_4 is not in 4NF, decompose into R_5 and R_6)

– $A \rightarrow\rightarrow B$ and $B \rightarrow\rightarrow HI$ ➔ $A \rightarrow\rightarrow HI$, (MVD transitivity), and

– and hence $A \rightarrow\rightarrow I$ *(MVD restriction to R_4)*

e) $R_5 = (A, I)$	(R_5 is in 4NF)
f) $R_6 = (A, C, G)$	(R_6 is in 4NF)

Further Normal Forms

- **Join dependencies** generalize multivalued dependencies
  - lead to **project-join normal form (PJNF)** (also called **fifth normal form**)
- A class of even more general constraints, leads to a normal form called **domain-key normal form**.
- Problem with these generalized constraints: are hard to reason with, and no set of sound and complete set of inference rules exists.
- Hence rarely used