MULTIVALUED DEPENDENCIES AND 4NF

Relational Database Design _Part5

Outline

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

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 were to combine these schemas to get

- *inst_info*(*ID*, *child_name*, *phone_number*)
- Example data:
  (99999, David, 512-555-1234)
  (99999, David, 512-555-4321)
  (99999, William, 512-555-1234)
  (99999, William, 512-555-4321)

This relation is in BCNF

- Why?

Multivalued Dependencies (MVDs)

Let R be a relation schema and let $\alpha \subseteq R$ and $\beta \subseteq R$. The **multivalued dependency**

$$\alpha \rightarrow\rightarrow \beta$$

holds on R if in any legal relation $r(R)$, for all pairs for tuples t_1 and t_2 in r such that $t_1[\alpha] = t_2[\alpha]$, there exist tuples t_3 and t_4 in r such that:

$$\begin{aligned}
t_1[\alpha] &= t_2[\alpha] = t_3[\alpha] = t_4[\alpha] \\
t_3[\beta] &= t_1[\beta] \\
t_3[R - \beta] &= t_2[R - \beta] \\
t_4[\beta] &= t_2[\beta] \\
t_4[R - \beta] &= t_1[R - \beta]
\end{aligned}$$

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

MVD (Cont.)

Tabular representation of $\alpha \rightarrow\rightarrow \beta$

	α	β	$R - \alpha - \beta$
t_1	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$a_{j+1} \ldots a_n$
t_2	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$b_{j+1} \ldots b_n$
t_3	$a_1 \ldots a_i$	$a_{i+1} \ldots a_j$	$b_{j+1} \ldots b_n$
t_4	$a_1 \ldots a_i$	$b_{i+1} \ldots b_j$	$a_{j+1} \ldots a_n$

Example

Let R be a relation schema with a set of attributes that are partitioned into 3 nonempty subsets.

$$Y, Z, W$$

We say that $Y \rightarrow\rightarrow Z$ (Y **multi-determines** 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

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

Note that since the behavior of Z and W are identical it follows that

$Y \rightarrow\rightarrow Z$ if $Y \rightarrow\rightarrow W$

Example (Cont.)

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.

Note:

- If $Y \rightarrow Z$ then $Y \rightarrow\rightarrow Z$
- Indeed we have (in above notation) $Z_1 = Z_2$
  The claim follows.

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.

If a relation *r* fails to satisfy a given multivalued dependency, we can construct a relations r' that does satisfy the multivalued dependency by adding tuples to *r.*

Theory of MVDs

From the definition of multivalued dependency, we can derive the following rule:

- If $\alpha \rightarrow \beta$, then $\alpha \twoheadrightarrow \beta$

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

$\twoheadrightarrow$

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

$\twoheadrightarrow$

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^+

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 - β) $\cup$ (α, β);
end
else *done*:= true;

Note: each R_i is in 4NF, and decomposition is lossless-join

$\twoheadrightarrow$

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

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

Overall Database Design Process

We have assumed schema *R* is given

- *R* could have been generated when converting E-R diagram to a set of tables.
- *R* could have been a single relation containing *all* attributes that are of interest (called **universal relation**).
- Normalization breaks *R* into smaller relations.
- *R* could have been the result of some ad hoc design of relations, which we then test/convert to normal form.

Overall Database Design Process

We have assumed schema *R* is given

- *R* could have been generated when converting E-R diagram to a set of tables.
- *R* could have been a single relation containing *all* attributes that are of interest (called **universal relation**).
- Normalization breaks *R* into smaller relations.
- *R* could have been the result of some ad hoc design of relations, which we then test/convert to normal form.