Functional Dependencies: Part 2

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

In designing a database, for the purpose of minimizing redundancy, we need to collect a set F of functional dependencies (FD) that reflect the constraints of the underlying application.

Ideally, we do not want to miss any FD, i.e., we want to obtain an F that is as large as possible. However, in practice, FD collection is a difficult process. No one can guarantee always discovering all FDs.

In practice, it is often the case that some FDs are easier to see, while others are more subtle and harder to observe. Some of those subtle FDs, fortunately, can be derived from the easy ones. In other words, the derivation permits us to "rescue" some FDs that have skipped our attention.

How about the subtle FDs that cannot be derived from the easy ones? There is nothing we can do about them, unfortunately, and will have to continue the design without them. This is why even an experienced database professional may not always be able to come up with a perfect design!

Let us identify an important special type of FDs:

Definition

A functional dependency $X \rightarrow Y$ is regular if Y contains only a single attribute.

For example, $AB \rightarrow C$ is regular, but $AB \rightarrow CD$ is not, where A, B, C, and D are attributes.

There is an equivalence that explains why we can capture all irregular FDs by considering only regular ones:

- An irregular FD $X \rightarrow A_1 A_2 ... A_t$ (where X is an attribute set, and each A_i $(1 \leq i \leq t)$ is an attribute) is equivalent to:

$$\begin{aligned} X &\rightarrow A_1 \\ X &\rightarrow A_2 \\ &\dots \\ X &\rightarrow A_t \end{aligned}$$

Example: $AB \rightarrow CD$ if and only if $AB \rightarrow C$ and $AB \rightarrow D$.

Again, let F be the set of FDs we have collected. Then:

Definition

The closure of F, denoted as F^+, is the set of all regular FDs that can be derived from F.

Do not confuse the closure of F with the closure of an attribute set.

Example. Assume that there are 4 attributes A, B, C, D, and that $F = \{A \to B, B \to C\}$. Then, F^+ includes all the following FDs:

$A \to A$, $A \to B$, $A \to C$, $B \to B$, $B \to C$, $C \to C$, $D \to D$, $AB \to A$, $AB \to B$, $AB \to C$, $AC \to A$, $AC \to B$, $AC \to C$, $AD \to A$, $AD \to B$, $AD \to C$, $AD \to D$, $BC \to B$, $BC \to C$, $BD \to B$, $BD \to C$, $BD \to D$, $CD \to C$, $CD \to D$, $ABC \to A$, $ABC \to B$, $ABC \to C$, $ABD \to A$, $ABD \to B$, $ABD \to C$, $ABD \to D$, $BCD \to B$, $BCD \to C$, $BCD \to D$, $ABCD \to A$, $ABCD \to B$, $ABCD \to C$, $ABCD \to D$.

Finding the Closure of a Set of FDs

algorithm (F)
/* F is a set of FDs */
1. $F^+ = \emptyset$
2. **for** each possible attribute set X
3. compute the closure X^+ of X on F
4. **for** each attribute $A \in X^+$
5. add to F^+ the FD: $X \rightarrow A$
5. **return** F^+

Example. Assume that there are 4 attributes A, B, C, D, and that $F = \{A \rightarrow B, B \rightarrow C\}$. To compute F^+, we first get:

- $A^+ = AB^+ = AC^+ = ABC^+ = \{A, B, C\}$
- $B^+ = BC^+ = \{B, C\}$
- $C^+ = \{C\}$
- $D^+ = \{D\}$
- $AD^+ = \{A, D\}$
- $BC^+ = \{B, C\}$
- $BD^+ = BCD^+ = \{B, C, D\}$
- $ABD^+ = ABCD^+ = \{A, B, C, D\}$
- $ACD^+ = \{A, C, D\}$

It is easy to generate the FDs in F^+ from the closures of the above attribute sets.

Candidate Key Revisited

In creating a table, it may seem that so far we have been specifying candidate keys based on our preferences. This illusion is created because we did not understand FDs. In fact, candidate keys are not up to us at all. Instead, they are uniquely determined by the set F of functional dependencies from the underlying application. See the next slide.

Candidate Key Revisited

Let F be a set of FDs, and R a relation.

Definition

A candidate key is a set X of attributes in R such that

- X^+ includes all the attributes in R.
- There is no proper subset Y of X such that Y^+ includes all the attributes in R.

Note: A proper subset Y is a subset of X such that $Y \neq X$ (i.e., X has at least one element not in Y).

Example. Consider a table $R(A, B, C, D)$, and that $F = \{A \to B, B \to C\}$.

- A is not a candidate key, because $A^+ = \{A, B, C\}$ which does not include D.
- ABD is not a candidate key even though $ABD^+ = \{A, B, C, D\}$. This is because $AD^+ = \{A, B, C, D\}$, namely, there is a proper subset AD of ABD such that AD^+ includes all the attributes.
- AD is a candidate key.