CSI T5300: Advanced Database Systems

L05: Functional Dependencies

Dr. Kenneth LEUNG

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Hong Kong SAR, China
Let R be a relation schema and X, Y be sets of attributes in R.

A **functional dependency** from X to Y exists if and only if:
- For every instance |R| of R, if two tuples in |R| agree on the values of the attributes in X, then they agree on the values of the attributes in Y.

We write $X \rightarrow Y$ and say that X **determines** Y.

**Example on PGStudent (sid, name, supervisor_id, specialization):**
- $\{\text{supervisor_id}\} \rightarrow \{\text{specialization}\}$ means
  - If two student records have the **same** supervisor (e.g., Stavros), then their specialization (e.g., Databases) must be the **same**.
  - On the other hand, if the supervisors of 2 students are **different**, we do not care about their specializations (they may be the **same or different**).

Sometimes, we omit the brackets for simplicity:
- $\text{supervisor_id} \rightarrow \text{specialization}$
A functional dependency $X \rightarrow Y$ is trivial if $Y$ is a subset of $X$:
- $\{\text{name}, \text{supervisor}\_id\} \rightarrow \{\text{name}\}$
  - If two records have the same values on both the name and supervisor\_id attributes, then they obviously have the same name.
  - Trivial dependencies hold for all relation instances.

A functional dependency $X \rightarrow Y$ is non-trivial if $Y \cap X = \emptyset$:
- $\{\text{supervisor}\_id\} \rightarrow \{\text{specialization}\}$
  - Non-trivial FDs are given in the form of constraints when designing a database.
    - For instance, the specialization of a student must be the same as that of the supervisor.
  - They constrain the set of legal relation instances. For instance, if I try to insert two students under the same supervisor with different specializations, the insertion will be rejected by the DBMS.

Some FDs are neither trivial nor non-trivial.
• A FD is a generalization of the notion of a key

• For PGStudent (sid, name, supervisor_id, specialization), if we write \{sid\} → \{name, supervisor_id, specialization\}
  - The sid determines all attributes (i.e., the entire record)
  - If two tuples in the relation student have the same sid, then they must have the same values on all attributes
  - In other words, they must be the same tuple (since the relational model does not allow duplicate records)
A set of attributes that determines the entire tuple is a **superkey**
- \{sid, name\} is a superkey for the PGStudent table.
- Also \{sid, name, supervisor_id\} etc.

A **minimal** set of attributes that determines the entire tuple is a **candidate key**
- \{sid, name\} is not a candidate key because I can remove the name.
- sid is a candidate key, and so is HKID (provided that it is stored in the table).

If there are multiple candidate keys, the DB designer designates one as the **primary key**.
• Given a set of functional dependencies $F$, there are certain other functional dependencies that are logically implied by $F$
• The set of all functional dependencies logically implied by $F$ is the closure of $F$
• We denote the closure of $F$ by $F^+$
• We can find all of $F^+$ by applying Armstrong’s Axioms:
  - if $Y \subseteq X$, then $X \rightarrow Y$ (reflexivity)
  - if $X \rightarrow Y$, then $ZX \rightarrow ZY$ (augmentation)
  - if $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$ (transitivity)

  these rules are sound and complete.
Examples of Armstrong’s Axioms

• if \( Y \subseteq X \), then \( X \rightarrow Y \) (*reflexivity* generates trivial FDs)
  \[ \text{name} \rightarrow \text{name} \]
  \[ \text{name}, \text{supervisor\_id} \rightarrow \text{name} \]
  \[ \text{name}, \text{supervisor\_id} \rightarrow \text{supervisor\_id} \]

• if \( X \rightarrow Y \), then \( ZX \rightarrow ZY \) (*augmentation*)
  \[ \text{sid} \rightarrow \text{name} \quad \text{(given)} \]
  \[ \text{supervisor\_id}, \text{sid} \rightarrow \text{supervisor\_id}, \text{name} \]

• if \( X \rightarrow Y \) and \( Y \rightarrow Z \), then \( X \rightarrow Z \) (*transitivity*)
  \[ \text{sid} \rightarrow \text{supervisor\_id} \quad \text{(given)} \]
  \[ \text{supervisor\_id} \rightarrow \text{specialization} \quad \text{(given)} \]
  \[ \text{sid} \rightarrow \text{specialization} \]
• We can further simplify computation of $F^+$ by using the following additional rules

  - If $X \rightarrow Y$ holds and $X \rightarrow Z$ holds, then $X \rightarrow YZ$ holds (union)
  - If $X \rightarrow YZ$ holds, then $X \rightarrow Y$ holds and $X \rightarrow Z$ holds (decomposition)
  - If $X \rightarrow Y$ holds and $ZY \rightarrow W$ holds, then $ZX \rightarrow W$ holds (pseudotransitivity)

• The above rules can be inferred from Armstrong’s axioms. (How?)
  E.g., pseudotransitivity
  
  \[
  X \rightarrow Y, \quad ZY \rightarrow W \quad \text{(given)}
  
  \[
  ZX \rightarrow ZY \quad \text{(by augmentation)}
  
  \[
  ZX \rightarrow W \quad \text{(by transitivity)}
  \]
Example of FDs in the closure $F^+$


- $F = \{ A \rightarrow B, \\
  A \rightarrow C, \\
  CG \rightarrow H, \\
  CG \rightarrow I, \\
  B \rightarrow H \}$

- some members of $F^+$
  
  - $A \rightarrow H$ \quad (A \rightarrow B; B \rightarrow H)$
  - $AG \rightarrow I$ \quad (A \rightarrow C; AG \rightarrow CG; CG \rightarrow I)$
  - $CG \rightarrow HI$ \quad (CG \rightarrow H; CG \rightarrow I)$
The closure of $X$ under $F$ (denoted by $X^+$) is the set of attributes that are functionally determined by $X$ under $F$:

$$X \rightarrow Y \text{ is in } F^+ \iff Y \subseteq X^+$$

- If $sid \rightarrow name$ is in $F^+$ then name is part of $sid^+$
  i.e., $sid^+ = \{sid, name, \ldots\}$

- If $sid \rightarrow supervisor_id$ is in $F^+$ then supervisor_id is part of $sid^+$
  i.e., $sid^+ = \{sid, name, supervisor_id, \ldots\}$

...
Algorithm for Computing Attribute Closure

- **Input:**
  - \( R \): a relation schema
  - \( F \): a set of functional dependencies
  - \( X \subset R \): the set of attributes for which we want to compute the closure

- **Output:**
  - \( X^+ \): the closure of \( X \) w.r.t. \( F \)

\[
X^{(0)} := X \\
i = 0 \\
\text{repeat} \\
i = i + 1 \\
X^{(i)} := X^{(i-1)} \cup Z, \text{ where } Z \text{ is the set of attributes such that there exists } Y \rightarrow Z \text{ in } F, \text{ and } Y \subset X^{(i)} \\
\text{until } X^{(i)} := X^{(i-1)} \\
\text{return } X^{(i)}
\]
\[ R = \{A, B, C, D, E, G\} \]

\[ F = \{ \{A, B\} \rightarrow \{C\}, \{C\} \rightarrow \{A\}, \{B, C\} \rightarrow \{D\}, \{A, C, D\} \rightarrow \{B\}, \{D\} \rightarrow \{E, G\}, \{B, E\} \rightarrow \{C\}, \{C, G\} \rightarrow \{B, D\}, \{C, E\} \rightarrow \{A, G\}\} \]

\[ X = \{B, D\} \]

\[ X^{(0)} = \{B, D\} \]
\[ \{D\} \rightarrow \{E, G\}, \]

\[ X^{(1)} = \{B, D, E, G\}, \]
\[ \{B, E\} \rightarrow \{C\} \]

\[ X^{(2)} = \{B, C, D, E, G\}, \]
\[ \{C\} \rightarrow \{A\} \]

\[ X^{(3)} = \{A, B, C, D, E, G\} \]

\[ X^{(4)} = X^{(3)} \]
Uses of Attribute Closure

• **Testing for superkey**
  - To test if $X$ is a superkey, we compute $X^+$, and check if $X^+$ contains all attributes of $R$.

• **Testing functional dependencies**
  - To check if a functional dependency $X \rightarrow Y$ holds (or, in other words, $X \rightarrow Y$ is in $F^+$), just check if $Y \subseteq X^+$.

• **Computing the closure of $F$, i.e., $F^+$**
  - For each subset $X \subseteq R$, we find the closure $X^+$, and for each $Y \subseteq X^+$, we output a functional dependency $X \rightarrow Y$.

• **Computing if two sets of functional dependencies $F$ and $G$ are equivalent, i.e., $F^+ = G^+$**
  - For each functional dependency $Y \rightarrow Z$ in $F$
    • Compute $Y^+$ with respect to $G$
    • If $Z \subseteq Y^+$ then $Y \rightarrow Z$ is in $G^+$
  - And vice versa
Sets of functional dependencies may have redundant dependencies that can be inferred from the others

- **Example:**
  \{A\} \rightarrow \{C\} is redundant in: \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}, \{A\} \rightarrow \{C\}
  
  **Because:** \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}: \{A\} \rightarrow \{C\} (transitivity)

There may be extraneous/redundant attributes on the LHS of a dependency

- Let \( \alpha \rightarrow \beta \) be a functional dependency in \( F \). Attribute \( A \) is extraneous in \( \alpha \) if \( F \) logically implies \( F - \{ \alpha \rightarrow \beta \} \cup \{ (\alpha \setminus A) \rightarrow \beta \} \)

- **Example:**
  \( F = \{ \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}, \{A,C\} \rightarrow \{D\} \} \) can be simplified to
  \{\{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}, \{A\} \rightarrow \{D\}\}
  
  **Because:** \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}: \{A\} \rightarrow \{C\} (transitivity)
  
  \{A\} \rightarrow \{C\} : \{A\} \rightarrow \{A,C\} (augmentation)
  
  \{A\} \rightarrow \{A,C\}, \{A,C\} \rightarrow \{D\}: \{A\} \rightarrow \{D\} (transitivity)
• There may be extraneous/redundant attributes on the RHS of a dependency
  - Let $\alpha \rightarrow \beta$ be a functional dependency in $F$. Attribute $A$ is extraneous in $\beta$ if 
    \[ (F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - A)\} \] logically implies $F$
  - Example:
    $F = \{\{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}, \{A\} \rightarrow \{C,D\}\}$ can be simplified to 
    \{\{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}, \{A\} \rightarrow \{D\}\}
    Because: \{A\} \rightarrow \{B\}, \{B\} \rightarrow \{C\}: \{A\} \rightarrow \{C\} (transitivity)
    \{A\} \rightarrow \{C\}, \{A\} \rightarrow \{D\}: \{A\} \rightarrow \{C,D\} (union)
A canonical cover for $F$ is a set of dependencies $F_c$ such that
- $F$ and $F_c$ are equivalent
- $F_c$ contains no redundancy
- Each left side of functional dependency in $F_c$ is unique
  For instance, if we have two FD, $X \rightarrow Y$ and $X \rightarrow Z$, we convert them to $X \rightarrow YZ$.

**Algorithm for canonical cover of $F$:**

```plaintext
repeat
  Use the union rule to replace any dependencies in $F$
  $X_1 \rightarrow Y_1$ and $X_1 \rightarrow Y_2$ with $X_1 \rightarrow Y_1 Y_2$
  Find a functional dependency $X \rightarrow Y$ with an extraneous attribute either in $X$ or in $Y$
  If an extraneous attribute is found, delete it from $X \rightarrow Y$
until $F$ does not change
return $F$
```

**Note:** The union rule may become applicable after some extraneous attributes have been deleted, so it has to be re-applied.
Example for Computing the Canonical Cover

- $R = (A, B, C)$
  $F = \{ A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C \}$

- Combine $A \rightarrow BC$ and $A \rightarrow B$ into $A \rightarrow BC$
  - Set is now $F = \{ A \rightarrow BC, B \rightarrow C, AB \rightarrow C \}$

- $A$ is extraneous in $AB \rightarrow C$,
  because: $B \rightarrow C$ is already in $F$
  - Set is now $\{ A \rightarrow BC, B \rightarrow C \}$

- $C$ is extraneous in $A \rightarrow BC$,
  because: $A \rightarrow B, B \rightarrow C$: $A \rightarrow C$ (transitivity)
  $A \rightarrow B, A \rightarrow C$: $A \rightarrow BC$ (union)

- The canonical cover is:
  $F_c = \{ A \rightarrow B, B \rightarrow C \}$