Database Management Systems (CSE 220)

Vikas Bajpai

Revisiting Functional Dependency

Functional Dependency:

- Functional dependency (FD) is a set of constraints between two attributes in a relation.
- Functional dependency says that if two tuples have same values for attributes A1, A2,..., An, then those two tuples must have to have same values for attributes B1, B2, ..., Bn.

Functional Dependency:

- Functional dependency is represented by an arrow sign (→) that is, X→Y, where X functionally determines Y. The left-hand side attributes determine the values of attributes on the right-hand side.

Functional Dependency:

A relation R with attributes A, B, and C, satisfies the FDs:

$$A \rightarrow B \text{ and } B \rightarrow C.$$

What other FDs does it satisfy?

$$A \rightarrow C$$

- What is the key for R ?
  - A, because $A \rightarrow B$ and $A \rightarrow C$

Set of Functional Dependencies F*

- Formal Definition of F*, the **cover** of F:

if F is a set of FD's, then $F^* \equiv \{ X \rightarrow Y \mid F \vdash X \rightarrow Y \}$

- Informal Definitions
  - F* is the set of all FD's logically implied by F (entailed)

- ... usually F* is much too large even to enumerate!

F^*

- Usually F^* is much too large even to enumerate!

$$A \longrightarrow B \longrightarrow C$$

- Example (3 attributes, 2 FD's and 43 entailed dependencies)

Attr(R)=ABC and F ={ $A \rightarrow B$, $B \rightarrow C$ } then F^* is

$A \rightarrow S$ for all [subset of ABC]	8 FDs
$B \rightarrow BC$, $B \rightarrow B$, $B \rightarrow C$, $B \rightarrow \varnothing$	4 FDs
$C \rightarrow C$, $C \rightarrow \varnothing$, $\varnothing \rightarrow \varnothing$	3 FDs
$AB \rightarrow S$ for all subsets S of ABC	8 FDs
$AC \rightarrow S$ for all subsets S of ABC	8 FDs
$BC \rightarrow BC$, $BC \rightarrow B$, $BC \rightarrow C$, $BC \rightarrow \varnothing$	4 FDs
$ABC \rightarrow S$ for all subsets S of ABC	8 FDs

Closure of FD sets

- Given a relation schema R and set S of FDs
  - is the FD *F* logically implied by S?
- Example
  - R = {A,B,C,G,H,I}
  - S = A → B, A → C, CG → H, CG → I, B → H
  - would A → H be logically implied?
  - yes (you can prove this, using the definition of FD)
- Closure of S: S^+= all FDs logically implied by S
- How to compute S^+ ?
  - we can use <u>Armstrong's axioms</u>

Armstrong's Axioms

- Reflexivity rule
  - $A_1\ A_2 \ldots A_n \rightarrow$ *a subset of* $A_1\ A_2 \ldots A_n$
- Augmentation rule
  - $A_1\ A_2 \ldots A_n \rightarrow B_1\ B_2 \ldots B_m$

    then

    $A_1\ A_2 \ldots A_n\ \ C_1\ C_2 \ldots C_k \rightarrow B_1\ B_2 \ldots B_m\ C_1\ C_2 \ldots C_k$
- Transitivity rule
  - $A_1\ A_2 \ldots A_n \rightarrow B_1\ B_2 \ldots B_m$ and

    $B_1\ B_2 \ldots B_m \rightarrow C_1\ C_2 \ldots C_k$

    then

    $A_1\ A_2 \ldots A_n \rightarrow C_1\ C_2 \ldots C_k$

Inferring S^+ using Armstrong's Axioms

- S^+ = S
- Loop
  - For each *F* in S, apply reflexivity and augmentation rules
  - add the new FDs to S^+
  - For each pair of FDs in S, apply the transitivity rule
  - add the new FD to S^+
- Until S^+ does not change any further

Additional Rules:

- Union rule
  - X → Y and X → Z, then X → YZ
  - (X, Y, Z are sets of attributes)

- Decomposition rule
  - X → YZ, then X → Y and X → Z

- Pseudo-transitivity rule
  - X → Y and YZ → U, then XZ → U

- These rules can be inferred from Armstrong's axioms

Example:

- $R = (A, B, C, G, H, I)$
  $F = \{\ A \rightarrow B \quad A \rightarrow C \quad CG \rightarrow H \quad CG \rightarrow I \quad B \rightarrow H\}$
- some members of F^+
  - $A \rightarrow H$
    - by transitivity from $A \rightarrow B$ *and* $B \rightarrow H$
  - $AG \rightarrow I$
    - by augmenting $A \rightarrow C$ with G, to get $AG \rightarrow CG$
      and then transitivity with $CG \rightarrow I$
  - $CG \rightarrow HI$
    - from $CG \rightarrow H$ *and* $CG \rightarrow I$: "union rule" can be inferred from
      - definition of functional dependencies, or
      - Augmentation of $CG \rightarrow I$ to infer $CG \rightarrow CGI$, augmentation of $CG \rightarrow H$ to infer $CGI \rightarrow HI$, and then transitivity

Closures of Attributes:

Suppose a relation with attributes A, B, C, D, E, and F satisfies the FDs

$$AB \rightarrow C \quad BC \rightarrow AD \quad D \rightarrow E, \quad CF \rightarrow B$$

Given these FDs,

- what is the set X of attributes such that $AB \rightarrow X$ is true?
  $X = \{A, B, C, D, E\}$, i.e., $AB \rightarrow ABCDE$.

- what is the set Y of attributes such that $BCF \rightarrow Y$ is true?
  $Y = \{A, B, C, D, E, F\}$, i.e., $BCF \rightarrow ABCDEF$

- $\{B, C, F\}$ is a superkey.

Closures of Attributes: Definition

Given

- a set of attributes $\{A_1, A_2, \ldots, A_n\}$ and
- a set of FDs S,

the *closure* of $\{A_1, A_2, \ldots, A_n\}$ under the FDs in S is

- the set of attributes $\{B_1, B_2, \ldots, B_m\}$ such that for $1 \leq i \leq m$, the FD $A_1A_2\ldots A_n \rightarrow B_i$ follows from S.
- the closure is denoted by $\{A_1, A_2, \ldots, A_n\}^+$.

- Which attributes must $\{A_1, A_2, \ldots, A_n\}^+$ contain at a minimum? $\{A_1, A_2, \ldots, A_n\}$. Why?

$A_1A_2\ldots A_n \rightarrow A_i$ is a trivial FD.

Closures of Attributes: Algorithm

Given

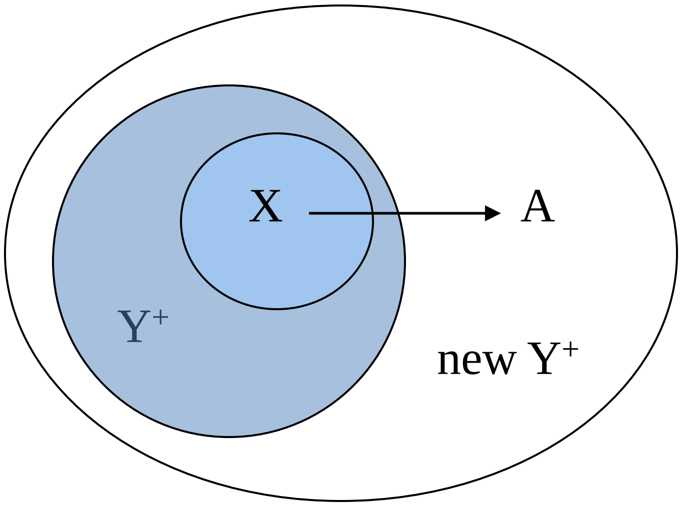
- a set of attributes $\{A_1, A_2, \ldots, A_n\}$ and
- a set of FDs S,
- compute $X = \{A_1, A_2, \ldots, A_n\}^+$.

1. Set $X \leftarrow \{A_1, A_2, \ldots, A_n\}$.
2. Find an FD $B_1 B_2 \ldots B_k \rightarrow C$ in S such that $\{B_1, B_2, \ldots B_k\} \subseteq X$ but $C \notin X$.
3. Add C to X.
4. Repeat the last two steps until you cannot find such an attribute C.
5. The final value of X is the desired closure.

Closures of Attributes: Algorithm

- Basis: $Y^+ = Y$
- Induction: Look for an FD's left side X that is a subset of the current Y^+
  - If the FD is $X \to A$, add A to Y^+

Diagramatically:



Why is the Concept of Closures Useful?

- Closures allow us to prove correctness of rules for manipulating FDs.
  - *Transitive rule*: if
    $A_1A_2\ldots A_n \rightarrow B_1B_2\ldots B_m$
    and
    $B_1B_2\ldots B_m \rightarrow C_1C_2\ldots C_n$
    then
    $A_1A_2\ldots A_n \rightarrow C_1C_2\ldots C_n$.
  - To prove this rule, simply check if
    $\{C_1, C_2, \ldots, C_n\} \subseteq \{A_1, A_2, \ldots, A_n\}^+$.
- Closures allow us to procedurally define keys. A set of attributes X is a key for a relation R if and only if
  - $\{X\}^+$ is the set of all attributes of R and
  - for no attribute $A \in X$ is $\{X - \{A\}\}^+$ the set of all attributes of R.

Uses of Attribute Closure

There are several uses of the attribute closure algorithm:

- Testing for superkey:
  - To test if α is a superkey, we compute $\alpha^{+,}$ and check if α^{+} contains all attributes of *R*.
- Testing functional dependencies
  - To check if a functional dependency $\alpha \rightarrow \beta$ holds (or, in other words, is in F^{+}), just check if $\beta \subseteq \alpha^{+}$.
  - That is, we compute α^{+} by using attribute closure, and then check if it contains β.
  - Is a simple and cheap test, and very useful
- Computing closure of F
  - For each $\gamma \subseteq R$, we find the closure γ^{+}, and for each $S \subseteq \gamma^{+}$, we output a functional dependency $\gamma \rightarrow S$.

Example of Attribute Set Closure

- $R = (A, B, C, G, H, I)$
- $F = \{A \rightarrow B \quad A \rightarrow C \quad CG \rightarrow H \quad CG \rightarrow I \quad B \rightarrow H\}$
- $(AG)^+$
  1. *result* $= AG$
  2. $(A \rightarrow C$ and $A \rightarrow B)$ *result* $= ABCG$
  3. $(CG \rightarrow H$ and $CG \subseteq AGBC)$ *result* $= ABCGH$
  4. $(CG \rightarrow I$ and $CG \subseteq AGBCH)$ *result* $= ABCGHI$

- Is AG a super key?
- Is AG a key?
  1. Does $A^+ \rightarrow R$?
  2. Does $G^+ \rightarrow R$?

Example of Closure Computation

- Consider the "bad" relation `Students(Id, Name, AdvisorId, AdvisorName, FavouriteAdvisorId)`.
- What are the FDs that hold in this relation?

  $\texttt{Id} \longrightarrow \texttt{Name}$

  $\texttt{Id} \longrightarrow \texttt{FavouriteAdvisorId}$

  $\texttt{AdvisorId} \longrightarrow \texttt{AdvisorName}$

- To compute the key for this relation,
  1. Compute the closures for all sets of attributes.
  2. Find the minimal set of attributes whose closure is the set of all attributes.