Chapter 8: Relational Database Design

Database System Concepts, 6th Ed.

Edited by Radhika Sukapuram

Testing whether a set α is a superkey

- An attribute B is functionally determined by a set of attributes α if

  $\alpha \rightarrow B$
- To test whether α is a superkey
  - Compute a set of attributes functionally determined by α
- For this,
  1. Compute F+
  2. Take all FDs with α as the LHS
  3. Take the union of the RHS of each such FD

  Expensive, as F+ can be large.

Closure of Attribute Sets

- Given a set of attributes α, define the ***closure*** of α **under** F (denoted by α^+) as the set of attributes that are functionally determined by α under F

- Algorithm to compute α^+, the closure of α under F

    result := α;
    repeat
        for each $\beta \rightarrow \gamma$ **in** F **do**
            begin
                if $\beta \subseteq$ *result* **then** *result* := *result* $\cup\ \gamma$
            end
    until (no change to *result*)

Example of Attribute Set Closure

- $R = (A, B, C, G, H, I)$
- $F = \{CG \rightarrow H$
  $CG \rightarrow I$
  $B \rightarrow H$
  $A \rightarrow B$
  $A \rightarrow C\}$
- **Compute $(AG)^+$**
- Is AG a candidate key?
  1. Is AG a super key?
     1. Does $AG \rightarrow R$? == Is $(AG)^+ \supseteq R$
  2. Is any subset of AG a superkey?
     1. Does $A \rightarrow R$? == Is $(A)^+ \supseteq R$
     2. Does $G \rightarrow R$? == Is $(G)^+ \supseteq R$

Example of Attribute Set Closure

- $R = (A, B, C, G, H, I)$
- $F = \{A \rightarrow B$
  $A \rightarrow C$
  $CG \rightarrow H$
  $CG \rightarrow I$
  $B \rightarrow H\}$
- $(AG)^+$
  1. *result* = *AG*
  2. *result* = *ABCG* $(A \rightarrow C$ and $A \rightarrow B)$
  3. *result* = *ABCGH* $(CG \rightarrow H$ and $CG \subseteq ABCG)$
  4. *result* = *ABCGHI* $(CG \rightarrow I$ and $CG \subseteq AGBCH)$
- Is *AG* a candidate key?
  1. Is AG a super key?
     1. Does $AG \rightarrow R$? == Is $(AG)^+ \supseteq R$
  2. Is any subset of AG a superkey?
     1. Does $A \rightarrow R$? == Is $(A)^+ \supseteq R$
     2. Does $G \rightarrow R$? == Is $(G)^+ \supseteq R$

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 $\gamma \rightarrow S$ such that $\gamma \cap S = \emptyset$

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

A+ = A, B+ =BD, C+ =C, D+ =D

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

A+ = A, B+ =BD, C+ =C, D+ =D

AB+ = ABCD, AC+ = AC, AD+=ABCD, BC+ = BCD, BD+=BD, CD+=CD

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

A+ = A, B+ =BD, C+ =C, D+ =D

AB+ = ABCD, AC+ = AC, AD+=ABCD, BC+ = BCD, BD+=BD, CD+=CD

ABC+ =ABD+ = ACD+ = ABCD, BCD+ = BCD

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

A+ = A, B+ =BD, C+ =C, D+ =D

AB+ = ABCD, AC+ = AC, AD+=ABCD, BC+ = BCD, BD+=BD, CD+=CD

ABC+ =ABD+ = ACD+ = ABCD, BCD+ = BCD

2. For each $S \subseteq \gamma^+$,output a functional dependency $\gamma \to S$ such that $\gamma \cap S = \emptyset$:

Computing F+ using attribute closure

F= { AB -> C, AD -> B, B -> D }

Compute F+ using the method of attribute closure.

1. For each $\gamma \subseteq R$, find the closure γ^+:

A+ = A, B+ =BD, C+ =C, D+ =D

AB+ = ABCD, AC+ = AC, AD+=ABCD, BC+ = BCD, BD+=BD, CD+=CD

ABC+ =ABD+ = ACD+ = ABCD, BCD+ = BCD

2. For each $S \subseteq \gamma^+$,output a functional dependency $\gamma \rightarrow S$ such that $\gamma \cap S = \emptyset$:

AD->BC, AB->CD, ABC->D, ABD->C, ACD->B

The rest are trivial or already present

Proof of algorithm for attribute closure of α under F

```
result := α;
while (changes to result) do
        for each β → γ in F do
                begin
                        if β ⊆ result then  result := result ∪ γ
                end
```

Proof of algorithm for attribute closure of α under F

```
1 result := α;
2 while (changes to result) do
      3 for each β → γ in F do
            4 begin
                  5 if β ⊆ result then  result := result ∪ γ
            6 end
```

- Step 1- Correct since $\alpha \rightarrow \alpha$ holds. Why ?

  $\alpha \rightarrow$ *result* is trivially true in the beginning

- We can add γ to *result* only if $\beta \subseteq$ *result* and $\beta \rightarrow \gamma$ (steps 3 and 5)
  $\beta \subseteq$ *result* implies *result* -> β by reflexivity

  $\alpha \rightarrow$ *result*, *result* -> β . Therefore $\alpha \rightarrow \beta$

  Now $\alpha \rightarrow \beta$ **,** $\beta \rightarrow \gamma$ **.** Therefore $\alpha \rightarrow \gamma$

  $\alpha \rightarrow$ *result* , $\alpha \rightarrow \gamma$. By the union rule, $\alpha \rightarrow$ *result* $\cup\ \gamma$

- α functionally determines any new *result* generated in the while loop

- Any attribute returned by the algorithm is in $\alpha+$

The algorithm finds all of $\alpha+$

- Assume there is an attribute that is $\alpha+$ that is not yet in *result* at any point during execution
- There must be an FD $\beta \rightarrow \gamma$ for which $\beta \subseteq$ *result* and at least one attribute in γ not in result
- When the algorithm terminates, all such FDs have been processed and the attributes in γ added to *result*
- Thus all attributes in γ are in *result*

Extraneous Attributes

- Consider a set F of functional dependencies and the functional dependency $\alpha \rightarrow \beta$ in F.
  - Attribute A is **extraneous** in α if $A \in \alpha$ and F logically implies $(F - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - A) \rightarrow \beta\}$.
  - Attribute A is **extraneous** in β if $A \in \beta$ and the set of functional dependencies $(F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - A)\}$ logically implies F.
- *Note:* implication in the opposite direction is trivial in each of the cases above, since a "stronger" functional dependency always implies a weaker one
  - E.g. : $(F - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - A) \rightarrow \beta\}$ always implies F

Testing if an Attribute is Extraneous

- Consider a set F of functional dependencies and the functional dependency $\alpha \rightarrow \beta$ in F.
- To test if attribute $A \in \alpha$ is extraneous in α
  1. compute $(\{\alpha\} - A)^+$ using the dependencies in F
  2. check that $(\{\alpha\} - A)^+$ contains β; if it does, A is extraneous in α

- Example: Given $F = \{A \rightarrow C, AB \rightarrow C\}$
  - B is extraneous in $AB \rightarrow C$ because

  $\{A \rightarrow C, AB \rightarrow C\}$ logically implies $A \rightarrow C$ (I.e. the result of dropping B from $AB \rightarrow C$).

Testing if an Attribute is Extraneous

- Consider a set F of functional dependencies and the functional dependency $\alpha \rightarrow \beta$ in F.

- To test if attribute $A \in \beta$ is extraneous in β

  1. compute α^+ using only the dependencies in
     $F' = (F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - A)\}$,
  2. check that α^+ contains A; if it does, A is extraneous in β

- Example: Given $F = \{A \rightarrow C, AB \rightarrow CD\}$
  - C is extraneous in $AB \rightarrow CD$ since $AB \rightarrow C$ can be inferred even after deleting C from CD