Database Design and Normalization

Chapter 10

(Week 11)
Computing Closure $F^+$

Example: List all FDs with:
- a single attribute on left
- two attributes on left
- three attributes on left

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a1</td>
<td>b1</td>
<td>c1</td>
<td>d1</td>
</tr>
<tr>
<td>2</td>
<td>a1</td>
<td>b1</td>
<td>c2</td>
<td>d2</td>
</tr>
<tr>
<td>3</td>
<td>a2</td>
<td>b1</td>
<td>c1</td>
<td>d3</td>
</tr>
<tr>
<td>4</td>
<td>a2</td>
<td>b1</td>
<td>c3</td>
<td>d4</td>
</tr>
</tbody>
</table>
Computing Closure $F^+$

Example: Let $F = \{ A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow E, E \rightarrow F, F \rightarrow G, G \rightarrow H \}$

How many Fds in $F^+$ and how to find them?
Reasoning About FDs (Contd.)

- Computing the closure of a set of FDs can be expensive. (Size of closure is exponential in # attrs!)
- Typically, we just want to check if a given FD \( X \rightarrow Y \) is in the closure of a set of FDs \( F \). An efficient check:
  - Compute attribute closure of \( X \) (denoted \( X^+ \)) wrt \( F \):
    - Set of all attributes \( A \) such that \( X \rightarrow A \) is in \( F^+ \)
    - There is a linear time algorithm to compute this.
  - Check if \( Y \) is in \( X^+ \)
- Does \( F = \{ A \rightarrow B, \ B \rightarrow C, \ C \rightarrow D \rightarrow E \} \) imply \( A \rightarrow E \)?
  - i.e., is \( A \rightarrow E \) in the closure \( F^+ \)? Equivalently, is \( E \) in \( A^+ \)?
Computing Closure (see algorithm in section 10.5)

**Algorithm rule:** If $X \rightarrow YZ$ and $Z \rightarrow W$ then $X \rightarrow YZW$

**Example:** $A \rightarrow B$, $A \rightarrow GC$

$GD \rightarrow F$

$A \rightarrow ABGC$

**Example:** $A \rightarrow B$, $A \rightarrow GC$, $GD \rightarrow FGB \rightarrow E$, $BC \rightarrow D$

Compute $A^+$

$A \rightarrow ACDFBGE$
Minimal Cover for a Set of FDs

- **Cover** $G$ is a set of FDs covered by another set of FDs $F$ of $G$ can be derived from $F$, that is, $F^+ \supseteq G$

- **Minimal cover** $G$ for a set of FDs $F$:
  - Closure of $F = \text{closure of } G$.
  - Right hand side of each FD in $G$ is a single attribute.
  - If we modify $G$ by deleting an FD or by deleting attributes from an FD in $G$, the closure changes.

- Intuitively, every FD in $G$ is needed, and "as small as possible" in order to get the same closure as $F$.

- e.g., $A \rightarrow B$, $ABCD \rightarrow E$, $EF \rightarrow GH$, $ACDF \rightarrow EG$ has the following minimal cover:
  - $A \rightarrow B$, $ACD \rightarrow E$, $EF \rightarrow G$ and $EF \rightarrow H$
Computing Minimal Cover

- **Step 1** From set $F$ create an equivalent set $H$ of FDs with only single attributes on the R.H.S.
- **Step 2** From $H$, successively remove individual FDs that are inessential in $H$. In other words, after removal, the new set remains equivalent to $H$.
- **Step 3** Now successively replace individual FDs with FDs that have a smaller number of attributes on the LHS, in a manner that does not change $H^+$. (You may need to repeat Step 2)
- **Step 4** From the remaining set of FDs, gather all FDs with equal left-hand sides and use the union rule to create an equivalent FDs where all left-hand sides are unique. (You may not get singleton on RHS)
Computing Minimal Cover

Example: \( ABD \rightarrow AC, \ C \rightarrow BE, \ AD \rightarrow BF, \ B \rightarrow E \)

Step 1: \( H = \{ ABD \rightarrow A, \ ABD \rightarrow C, \ C \rightarrow B, \ C \rightarrow E, \ AD \rightarrow B, \ AD \rightarrow F, \ B \rightarrow E \} \)

Step 2:
- \( ABD \rightarrow A \) is not essential.
- How about \( ABD \rightarrow C \)? It cannot be derived from other FDs using set closure rule.
- How about \( C \rightarrow B \)? Can it be implied by other FDs? Compute \( C^+ = \{ CE \} \). Since, \( C^* \) does not contain B, \( C \rightarrow B \) is essential.
- \( C \rightarrow E \) is inessential
- Compute \( (AD)^+ = (ADF) \), so \( AD \rightarrow B \) is essential. How about \( AD \rightarrow F \)? No FD has F on left hand side. So this is essential.
- Also, \( B \rightarrow E \) is essential.

\( H = \{ ABD \rightarrow C, \ C \rightarrow B, \ C \rightarrow E, \ AD \rightarrow B, \ AD \rightarrow F, \ B \rightarrow E \} \)
Computing Minimal Cover

- **Step 3** Can we drop A in ABD \(\rightarrow\) C?

The new set

\[ J = \{BD \rightarrow C, C \rightarrow B, C \rightarrow E, AD \rightarrow B, AD \rightarrow F, B \rightarrow E\} \]

Is \( J = H \)? Yes iff \((BD)^+\) under \( H \) = \( BD^+\) under \( J \).

- **Step 4**

\[ H = \{AD \rightarrow C, C \rightarrow B, AD \rightarrow F, B \rightarrow E\} \]

- **Step 4**

\[ H = \{AD \rightarrow CF, C \rightarrow B, B \rightarrow E\} \]
Computing Minimal Cover

Example: \( F = \{ AB \rightarrow C, \ C \rightarrow A, BC \rightarrow D, \ ACD \rightarrow B, \ D \rightarrow G, \ BE \rightarrow C, CG \rightarrow BD, CE \rightarrow AG \} \)

Minimal covers: \( H = \{ AB \rightarrow C, \ C \rightarrow A, BC \rightarrow D, \ CD \rightarrow B, \ D \rightarrow E, D \rightarrow G, \ BE \rightarrow C, CG \rightarrow D, CE \rightarrow G \} \)

\( H = \{ AB \rightarrow C, \ C \rightarrow A, BC \rightarrow D, D \rightarrow E, D \rightarrow G, \ BE \rightarrow C, CG \rightarrow B, CE \rightarrow G \} \)
Normal Forms

- Returning to the issue of schema refinement, the first question to ask is whether any refinement is needed!
- If a relation is in a certain normal form (BCNF, 3NF etc.), it is known that certain kinds of problems are avoided/minimized. This can be used to help us decide whether decomposing the relation will help.
- Role of FDs in detecting redundancy:
  - Consider a relation R with 3 attributes, A BC.
    - No FDs hold: There is no redundancy here.
    - Given A → B: Several tuples could have the same A value, and if so, they’ll all have the same B value!
Normalization (Chapter 11)

Universe of relations (normalized and unnormalized)

* 1NF relations
* 2NF relations
* 3NF relations
* BCNF relations
* 4NF relations
* PJ/NF (5NF) relation
Moving Towards 3NF Form

Objective 3NF
R is in 3NF iff its non-key attributes (if any)

- are mutually independent i.e. no FD’s among them.
- Full dependence on the P. Key
### 1NF

**FIRST**

Sample tabulation of FIRST

<table>
<thead>
<tr>
<th>S#</th>
<th>Status</th>
<th>City</th>
<th>P#</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P3</td>
<td>400</td>
</tr>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P4</td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P5</td>
<td>100</td>
</tr>
<tr>
<td>S1</td>
<td>20</td>
<td>London</td>
<td>P6</td>
<td>100</td>
</tr>
<tr>
<td>S2</td>
<td>10</td>
<td>Paris</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S2</td>
<td>10</td>
<td>Paris</td>
<td>P2</td>
<td>400</td>
</tr>
<tr>
<td>S3</td>
<td>10</td>
<td>Paris</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>20</td>
<td>London</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S4</td>
<td>20</td>
<td>London</td>
<td>P4</td>
<td>300</td>
</tr>
<tr>
<td>S4</td>
<td>20</td>
<td>London</td>
<td>P5</td>
<td>400</td>
</tr>
</tbody>
</table>
A relation R is in 1NF iff all underlying simple domains contain atomic values.
Problems with 1NF Example

Insert: We cannot insert a supplier. Why?

Delete: Problem?

Update: Problem?
2N Form

Convert 1NF → 2NF (using Projection)
R is in 1NF. Must has a composite P.K.
Project into two relations R1 and R2

**Def:** R is in 2NF if it is 1NF and every non-key attribute is fully dependent on the P.K.
Procedure $1NF \rightarrow 2NF$

If \( R(A,B,C,D) \)
  
  \( P.Key \) \( (A,B) \)
  
  with \( A \rightarrow D \)

Then generate \( R2 \) and \( R2 \) (lossless)

\( R1(A,D) \) \hspace{1cm} \( R2(A,B,C) \)

\( P.Key = A \) \hspace{1cm} \( P.Key = (A,B) \)

\( F.Key = (A) \) reference \( R1 \)