Normalization Highlights

Need to know from book:

- definition of trivial FD
- definition of partial FD
- definition of transitive FD
- definitions of 1NF, 2NF, 3NF, and BCNF and be able to understand when schemas are in which normal form.

Goal: BCNF, lossless, DP

If cannot achieve goal, settle for 3NF, lossless, DP

Assume R is decomposed into $R_1, R_2$. The decomposition is lossless if either:

- $R_1 \cap R_2 \rightarrow R_1$ or
- $R_1 \cap R_2 \rightarrow R_2$

Def: $F^+ = \text{closure of } F = \text{set of all logically implied FDs}$

**Algorithm for determining closure of attr set } \alpha{**:

resultAttr = $\alpha$
resultFD = {}
while (changes to either result set)
{

\forall \text{ } \forall FD \beta \rightarrow \gamma \in F

if $\beta \subseteq \text{resultAttr}$
then {
resultAttr = resultAttr $\cup \gamma$
resultFd = resultFD $\cup (\beta \rightarrow \gamma)$
}
}
Example1:

Assume $R = \{A, B, C, D, E, F, G\}$ and $F = \{B \rightarrow D, A \rightarrow B, CE \rightarrow F, CE \rightarrow G, A \rightarrow C\}$

Calculate $A^+$:

<table>
<thead>
<tr>
<th>iteration</th>
<th>resultAttr</th>
<th>resultFD</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>A, B</td>
<td>$A \rightarrow B$</td>
</tr>
<tr>
<td>1</td>
<td>A, B, C</td>
<td>$A \rightarrow B, A \rightarrow C$</td>
</tr>
<tr>
<td>2</td>
<td>A, B, C, D</td>
<td>$A \rightarrow B, A \rightarrow C, B \rightarrow D$</td>
</tr>
<tr>
<td>3</td>
<td>no change</td>
<td>no change</td>
</tr>
</tbody>
</table>

thus, $A^+ = \{A, B, C, D\}\{A \rightarrow B, A \rightarrow C, B \rightarrow D\}$.

Example2:

$F = \{B \rightarrow D, A \rightarrow B, CE \rightarrow F, CE \rightarrow G, A \rightarrow C\}$

Calculate $AE^+$:

<table>
<thead>
<tr>
<th>iteration</th>
<th>resultAttr</th>
<th>resultFD</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>A, E</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>A, B, E</td>
<td>$A \rightarrow B$</td>
</tr>
<tr>
<td>1</td>
<td>A, B, C, E</td>
<td>$A \rightarrow B, A \rightarrow C$</td>
</tr>
<tr>
<td>2</td>
<td>A, B, C, D, E</td>
<td>$A \rightarrow B, A \rightarrow C, B \rightarrow D$</td>
</tr>
<tr>
<td>2</td>
<td>A, B, C, D, E, F</td>
<td>$A \rightarrow B, A \rightarrow C, B \rightarrow D, CE \rightarrow F$</td>
</tr>
<tr>
<td>2</td>
<td>A, B, C, D, E, F, G</td>
<td>$A \rightarrow B, A \rightarrow C, B \rightarrow D, CE \rightarrow F, CE \rightarrow G$</td>
</tr>
<tr>
<td>3</td>
<td>no change</td>
<td>no change</td>
</tr>
</tbody>
</table>

thus, $AE$ is a candidate key
Minimal Cover

Before defining minimal cover, we need to define extraneous attribute.

Def: Attr A is extraneous in \( \alpha, \alpha \rightarrow \beta \), if \( A \in \alpha \) and \( [(F - (\alpha \rightarrow \beta)) \cup ((\alpha - A) \rightarrow \beta)]^+ \equiv F^+ \)

Def: Attr A is extraneous in \( \beta, \alpha \rightarrow \beta \), if \( A \in \beta \) and \( [(F - (\alpha \rightarrow \beta)) \cup (\alpha \rightarrow (\beta - A))]^+ \equiv F^+ \)

Def: \( F_c \) = minimal cover = set of FD such that:

- \( F_c^+ \equiv F^+ \)
- no FD \( \in F_c \) contains extraneous attributes
- every FD \( \in F_c \) has a single value on the right hand side
- can not remove a FD from \( F_c \) and stil have equivalence to \( F^+ \)

Minimal Cover Construction Algorithm:

1. Set \( G = F \)

2. replace each FD, \( X \rightarrow A_1, A_2, \ldots A_n \in G \) by FDs \( X \rightarrow A_1, X \rightarrow A_2, \ldots X \rightarrow A_n \)

3. \( \forall \) FD \( X \rightarrow A \in G \):
   \( \forall \) attribute \( B \) that is an element of \( X \):
   if \( ((G - \{X \rightarrow A\}) \cup \{(X - B) \rightarrow A\}) \) is equivalent to \( G \), then replace \( X \rightarrow A \) with \( (X - B) \rightarrow A \) in \( G \).

4. \( \forall \) FD \( X \rightarrow A \in G \), compute \( X^+ \) with respect to \( \{G - (X \rightarrow A)\} \). If \( A \in X^+ \), remove \( X \rightarrow A \) from \( G \).

Note, you must do the steps in order. As your book shows, reversal of 3 and 4 may result in something that is NOT a minimal cover.

Example:

Let \( F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\} \). In step 2 we rewrite \( G \) as \( \{A \rightarrow B, A \rightarrow C, B \rightarrow C, A \rightarrow B, AB \rightarrow C\} \). There are no extraneous attributes to remove in step 3. In step 4 we remove the first \( A \rightarrow B \), then remove \( A \rightarrow C \), then finally remove \( AB \rightarrow C \). Thus, \( F_c = \{A \rightarrow B, B \rightarrow C\} \).
Dependency Preservation

Consider the decomposition $R \Rightarrow R_1, R_2, \ldots R_N$.

Def: $F_i =$ restriction of $F$ to $F_i =$ set of FDS that include only attributes of $R_i$.

Let $F' = F_1 \cup F_2 \cup \ldots F_N$.

If $F'^+ \equiv F^+$, then the decomposition is dependency preserving.

**BCNF and Lossless Decomposition of $R$ Algorithm:**

1. Set $D = \{R\}$

2. while $\exists$ schema $Q \in D$ that is not BCNF
   
   
   \{ 
   
   (a) choose $Q \in D$ that is not BCNF
   
   (b) find FD $X \rightarrow Y$ that violates BCNF
   
   (c) replace $Q$ by the two schemas $(Q - Y)$ and $(X \cup Y)$
   
   \}

Note, this decomposition is NOT guaranteed to be dependency preserving.

**3NF, Lossless, and Dependency Preserving Decomposition of $R$ Algorithm:**

1. Find a minimal cover $F_c$ for $F$

2. For each left hand side $X \in F_c$, create a relation scheme $(X \cup A_1 \cup A_2 \cup \ldots A_N)$ where $X \rightarrow A_1, X \rightarrow A_2, \ldots X \rightarrow A_N$ are all the dependencies in $F_c$ with $X$ in the left hand side.