Normality Algorithms

Preliminaries

Theorem. If R is a relation schema with only two attributes, AB, then R is in BCNF
pf: There are only four possibilities for F...∅, \{A→B\}, \{B→A\} or \{ A→B, B→A\}. R is
BCNF in all of these.

Lemma:
If R is a relation schema then R is not in BCNF if there are attributes A and B s.t. R-
AB→A
Pf: Let X-A → A be a nontrivial FD where X is not a superkey (Otherwise R would be
in BCNF). There must exist some attribute B in R s.t. B is not in X ∪ A (if not, X would
be a superkey).

Therefore X-AB→A and consequently R-AB→A.

Procedure Decompose(R, F, A, B,*R1,*R2)
input: relation schemata R,R 1,R 2;
set of FD's F
attributes A, B // (R-AB) → A
BEGIN
Y = R-B
WHILE there are A' and B' s.t. Y-A'B'→ A',
Y = Y-B'
END_WHILE
R 1 = R-A;
R 2 = Y
F 1 = subset of F that still applies to R 1;
F 2= subset of F that still applies to R 2;
END
// Y is in BCNF by theorem.

BCNFs from R and F

Algorithm: Lossless join decomposition into BCNF (R, F)

Input: relation schema R
set of FD's F

Output: the decomposition ρ.

BEGIN
Z = R;
ρ = ∅;
IF there are no attributes A,B s.t. Z-AB→A, 
THEN ρ = {Z} 
ELSE 
    WHILE there exist A,B s.t. Z-AB→A 
        Decompose(Z,F,A,B,R₁,R₂); 
        Z = R₁; 
        ρ = ρ ⊃ {R₂}; 
    END_WHILE; 
    ρ = ρ ⊃ {Z}. 
END

Extended example: R₀(CTHRSG)

\[F = \{C \rightarrow T, \text{HR} \rightarrow C, \text{HT} \rightarrow R, \text{CS} \rightarrow G, \text{HS} \rightarrow R, \text{HR} \rightarrow T\}\]

A candidate key is HS, since

\[C^\perp = CT; \quad \text{HR}^\perp = HRCT; \quad \text{HT}^\perp = HTCR; \quad \text{CS}^\perp = CSGT; \quad \text{HS}^\perp = HSRCTG;\]

Let A be played by R, and B be played by G. Since HT→R, then

CTHS → R, ie R₀-GR → R, so 
Decompose (R₀, F, R, G, R₁, R₂)

After first application of the while loop, R₁=R₀-R and Y=CTHRS

Now HT→R so attributes C and S can be discarded in successive applications of the loop, and the final result of decompose is

\[R₁ = R₀-R = CTHSG\]
\[Y = \text{HT}R, \text{so...}\]
\[\rho = \{\text{HT}R\}\]
\[Z = R₁;\]

repeat the algorithm A being played by G and B by H.

/* CS→G means that R₁-HG→G */

Decompose (R₁, F₁, G, Y, R₁₁, R₁₂)

R₁₁ = R₁ - H and Y winds up being CSG
\[\rho = \rho \cup \{\text{CSG}\} = \{\text{HT}R, \text{CSG}\}\]
\[Z = R₁₁(CTHS)\]

repeat algorithm with A being played by T and B by H

R₁₁₁ is CHS and Y is CT
\[\rho = \{\text{HT}R, \text{CSG}, \text{CT}\} \text{ and}\]
\[Z = \text{CHS which has no further qualifying pairs so the final}\]
\[\rho \text{ is } \{\text{HT}R, \text{CSG}, \text{CT}, \text{CHS}\}.\]
NOTA BENE: The algorithm does not guarantee that the decomposition will preserve all the dependencies...in fact in our example..

\[ F_1 = \{ HT \rightarrow R \}, F_2 = \{ CS \rightarrow G \}, F_3 = \{ C \rightarrow T \}, \text{ and } F_4 = \emptyset \ldots \]

try to deduce the FD HR \( \rightarrow \) C from those three...

### 3NFs from R and F

**Algorithm:** Dependency preserving decomposition into 3NFs.

```
Input: relational schema R,  
      Set of FD's F  
Output: the decomposition.
```

**Method:**

a) If there are any attributes of R not involved in any dependency of F, either on the left or right, then any such attribute can, in principle, form a relation scheme by itself, and we shall eliminate it from R.

b) If one of the dependencies in F involves all the attributes of R, then the output is R itself.

c) Otherwise, find a minimal cover of F, \( F_{\text{min}} \), and the decomposition \( \rho \) to be output consists of scheme XA for each dependency \( X \rightarrow A \) in \( F_{\text{min}} \).

In our previous example, following the algorithm we obtain 5 tables \{CT, HRC, HTR, CSG, HSR\}.

In fact we have for \( R_0(CTHRSG) \):

\[ F = \{ C \rightarrow T, HR \rightarrow C, HT \rightarrow R, CS \rightarrow G, HS \rightarrow R, HR \rightarrow T \} \]

a) No elimination of attributes
b) No FD involves all attributes of \( R_0 \)
c) Find \( F_{\text{min}} \):

   **Step 1:** All FDs have rhs with only one attribute
   **Step 2:** Could we remove any attribute in the lhs of any FD?

   \[ F = \{ C \rightarrow T, HR \rightarrow C, HT \rightarrow R, CS \rightarrow G, HS \rightarrow R, HR \rightarrow T \} \]

   Consider HR \( \rightarrow \) C: Drop H? \( R^+ = R \) in \( F \), but \( R^+ = R \), in \( J_1 \)
   Drop R? \( H^+ = H \) in \( F \), but \( H^+ = HC \), in \( J_2 \)

   Consider HT \( \rightarrow \) R: Drop H? \( T^+ = T \) in \( F \), but \( T^+ = TR \), in \( J_3 \)
   Drop T? \( H^+ = H \) in \( F \), but \( H^+ = HR \), in \( J_4 \)

   Consider CS \( \rightarrow \) G: Drop C? \( S^+ = S \) in \( F \), but \( S^+ = SG \), in \( J_5 \)
   Drop S? \( C^+ = CT \) in \( F \), but \( C^+ = CGT \), in \( J_6 \)

   Consider HS \( \rightarrow \) R: Drop H? \( S^+ = S \) in \( F \), but \( S^+ = SR \), in \( J_7 \)
   Drop S? \( H^+ = H \) in \( F \), but \( H^+ = HR \), in \( J_8 \)

   Consider HR \( \rightarrow \) T: Drop H? \( R^+ = R \) in \( F \), but \( R^+ = RT \), in \( J_9 \)
   Drop R? \( H^+ = H \) in \( F \), but \( H^+ = HT \), in \( J_{10} \)

So no changes
Step 3: Find inessential FDs:
- Remove C → T; C⁺ = C, so this one is essential
- Remove HR → C; HR⁺ = HRT, so this one is essential
- Remove HT → R; HT⁺ = HT, so this one is essential
- Remove CS → G; CS⁺ = CST, so this one is essential
- Remove HS → R; HS⁺ = HS, so this one is essential
- Remove HR → T; HR⁺ = HRCT, so this one is inessential

Step 4: Therefore the decomposition is
\[ R = \{CT\} \cup \{HRC\} \cup \{HTR\} \cup \{CSG\} \cup \{HSR\} \]

Decomposition into 3NF with a Lossless Join and Preservation of Dependencies.

**Theorem:** If \( \sigma \) is the 3NF decomposition of \( R \) given by the previous algorithm, and \( X \) is a key for \( R \). Then \( \mu = \sigma \cup \{X\} \) is a decomposition of \( R \) with all relation schema in 3NF; the decomposition preserves dependencies and has the lossless join property.

**Proof:** Another time.

**Remark:** In many cases, the decomposition \( \mu \) is not the smallest one having the two properties. We can remove relation schemas in \( \mu \), one at a time, as long as both properties are preserved. In fact, if any key of \( R \) is already in one of the relations, we don’t need to add it to the set \( \sigma \), to obtain both properties.

**Example:** We study the relation: \( R \) (ABCDEFG), with the set of FDs

\[ F = \{AB \rightarrow C, D \rightarrow EG, ABC \rightarrow A, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CG \rightarrow BD, ACD \rightarrow B, CE \rightarrow AG\} \]

We obtain the decomposition

\[ R = \{ABC\} \cup \{DEG\} \cup \{CA\} \cup \{BEC\} \cup \{BCD\} \cup \{CEG\} \cup \{ACDB\} \]

Because

\[ F_{\text{min}} = \{AB \rightarrow C, D \rightarrow EG, C \rightarrow A, BE \rightarrow C, BC \rightarrow D, CE \rightarrow G, ACD \rightarrow B\} \]

This decomposition is not lossless join decomposition. We need to find a key for \( R \)

\[ A^+ = A; AB^+ = ABCDEG; ABF^+ = ABCDEFG \]
Therefore our decomposition of \( R \) wit a lossless join and preserving dependencies is:

\[
R = \{ABC\} \cup \{DEG\} \cup \{CA\} \cup \{BEC\} \cup \{BCD\} \cup \{CEG\} \cup \{ACDB\} \cup \{ABF\}
\]

Major exercise: use the algorithms to find 3NF and BCNF decompositions of the relation

Lots (Id#, County, Lot#, Area, Price, Taxrate)

\( F = \{I \rightarrow CLAPT, CL \rightarrow IAPT, C \rightarrow T, A \rightarrow P, A \rightarrow C\} \)