NORMAL FORMS
Normalization

Decomposition techniques for ensuring:
Lossless joins
Dependency preservation
Redundancy avoidance

We will look at some normal forms:
Boyce-Codd Normal Form (BCNF)
3rd Normal Form (3NF)
What is a normal form?

Characterization of schema decomposition in terms of properties it satisfies

BCNF: guarantees no redundancy

Defined:
relation schema \( R \), with FD set, \( F \) is in BCNF if:

For all nontrivial \( X \rightarrow Y \) in \( F^+ \):
\( X \rightarrow R \) (i.e. \( X \) a superkey)
Example: \( R=(A, B, C) \)

\( F = (A \rightarrow B, B \rightarrow C) \)

Is \( R \) in BCNF?

Ans: Consider the non-trivial dependencies in \( F^+ \):

- \( A \rightarrow B \), \( A \rightarrow R \) (A a key)
- \( A \rightarrow C \), -//-
- \( B \rightarrow C \), \( B \rightarrow R \) (B not a superkey)

Therefore not in BCNF
Example:

\[ R = R_1 \cup R_2 \]

\[ R_1 = (A, B), \quad R_2 = (B, C) \]

\[ F = (A \rightarrow B, B \rightarrow C) \]

Are \( R_1, R_2 \) in BCNF?

Ans: Yes, both non-trivial FDs define a key in \( R_1, R_2 \)

Is the decomposition lossless? DP?

Ans: Lossless: Yes. DP: Yes.
Decomposition Algorithm

Algorithm BCNF(R: relation, F: FD set)

Begin
1. Compute F+
2. Result \rightarrow \{R\}
3. While some \( R_i \) in Result not in BCNF Do
   a. Chose \((X \rightarrow Y)\) in F+ s.t.
      \((X \rightarrow Y)\) covered by \( R_i \)
      \( X \nrightarrow R_i \) (\( X \) not a superkey for \( R_i \))
   b. Decompose \( R_i \) on \((X \rightarrow Y)\)
      \( R_{i1} \leftarrow X \cup Y \)
      \( R_{i2} \leftarrow R_i - Y \)
   c. Result \leftarrow Result - \{R_i\} \cup \{R_{i1}, R_{i2}\}
4. return Result
End
BCNF Decomposition

Example:

R = (A, B, C, D)
F = (A \rightarrow B, AB \rightarrow D, B \rightarrow C)

Decompose R into BCNF

Ans: $F_c = \{A \rightarrow BD, B \rightarrow C\}$

R = (A, B, C, D)
B \rightarrow C is covered by R and B not a superkey

R1 = (B, C)
In BCNF: B \rightarrow C and B key

R2 = (A, B, D)
In BCNF: A \rightarrow B, A \rightarrow D, A \rightarrow BD
and A is a key
BCNF Decomposition

Example:

\[ R = (\text{bname, bcity, assets, cname, lno, amt}) \]
\[ F = \{ \text{bname } \rightarrow \text{bcity assets,} \]
\[ \text{lno } \rightarrow \text{amt bname} \} \]

Decompose \( R \) into BCNF

\[ \text{Ans: } F_c = F \]

\[ R, \text{ and } \text{bname } \rightarrow \text{bcity} \text{ covered by } R, \text{ bname not a key} \]

\[ R_1 = (\text{bname, bcity}) \]
\[ \text{In BCNF} \]

\[ R_2 = (\text{bname, assets, cname, lno, amt}) \]
\[ \text{lno } \rightarrow \text{amt bname} \text{ covered by } R_2 \text{ and lno not a key} \]

\[ R_3 = (\text{lno, amt, bname}) \]
\[ \text{In BCNF} \]

\[ R_4 = (\text{assets, cname, lno}) \]
\[ \text{lno } \rightarrow \text{assets} \ldots \]

\[ R_5 = (\text{lno, assets}) \]

\[ R_6 = (\text{lno, cname}) \]

Not DP! \text{bname } \rightarrow \text{assets} \text{ is not covered by any relation AND cannot be implied by the covered FDs.} \]

\[ \text{Covered FDs: } G = \{ \text{bname } \rightarrow \text{bcity, lno } \rightarrow \text{amt bname,} \]
\[ \text{lno } \rightarrow \text{assets} \} \]
Can there be > 1 BCNF decompositions?

Ans: Yes, last example was not DP. But...

Given $F_c = \{ \text{bname} \rightarrow \text{bcity assets}, \quad \text{lno} \rightarrow \text{amt bname} \}$

$R=(\text{bname, bcity, assets, cname, lno, amt})$

$bname \rightarrow \text{bcity assets} \quad \text{and} \quad bname \rightarrow \text{R}$

$R_1 = (\text{bname, bcity, assets})$

BCNF: $bname \rightarrow R_1$

$R_2 = (\text{bname, cname, lno, amt})$

$lno \rightarrow \text{amt bname}, \quad \text{lno} \rightarrow \text{R}_2$

$R_3 = (\text{lno, amt, bname})$

BCNF: $lno \rightarrow R_3$

$R_4 = (\text{lno, cname})$

Is $R = R_1 \cup R_3 \cup R_4$ DP?

Yes!!
Can we decompose on FD’s in Fc to get a DP, BCNF decomposition?

Usually, yes, but ...

Consider: \( R = (J, K, L) \)
\( F = (JK \rightarrow L, \ L \rightarrow K) \) \( (Fc = F) \)

We can apply decomposition either using: \( JK \rightarrow L \), \( L \rightarrow K \) or the opposite

Dec. #1
Using \( L \rightarrow K \)
\( R1 = (L, K) \)
\( R2 = (J, L) \) \( \) Not DP.

Dec. #2
Using \( JK \rightarrow L \)
\( R1 = (J, K, L) \) \( \) not BCNF
\( R2 = (J, K) \)

So, BCNF and DP decomposition may not be possible.
Aside

Is the example realistic?

Consider: BankerName $\rightarrow$ BranchName

BranchName CustomerName $\rightarrow$ BankerName
3NF: An alternative to BCNF

Motivation:
sometimes, BCNF is not what you want

E.g.: street city → zip and zip → city
BCNF: R1 = { zip, city} R2 = { zip, street}

No redundancy, but to preserve 1st FD requires assertion with join

Alternative: 3rd Normal Form
Designed to say that decomposition can stop at {street, city, zip}
3NF: An alternative to BCNF

BCNF test: Given R with FD set, F: For any non-trivial FD, 
\[ X \rightarrow Y \] in F+ and covered by R, then \[ X \rightarrow R \]

3NF test: Given R with FD set, F:

For any non-trivial FD,

\[ X \rightarrow Y \] in F+ and covered by R, then

\[ X \rightarrow R \] or

\[ Y \] is a subset of some candidate key of R

Thus, 3NF a weaker normal form than BCNF:

i.e. \[ R \in BCNF \Rightarrow R \in 3NF \]

but \[ R \in 3NF \nRightarrow R \in BCNF \] (not sure than R is in BCNF)
3NF: An alternative to BCNF

Example:

\[ R = (J, K, L) \quad F = \{JK \rightarrow L, L \rightarrow K\} \]

then \( R \) is 3NF!

Key for \( R \): \( JK \)

- \( JK \rightarrow L \) covered by \( R \), \( JK \rightarrow R \)
- \( L \rightarrow K \), \( K \) is a part of a candidate key
Example:

\[ R = (\text{bname}, \text{cname}, \text{lno}, \text{amt}) \]
\[ F = F_c = \{ \text{Ino} \rightarrow \text{amt bname}, \]
\[ \text{cname bname} \rightarrow \text{lno} \} \]

Q: is \( R \) in BCNF, 3NF or neither?

Ans:

\( R \) not in BCNF: \( \text{Ino} \rightarrow \text{amt} \), covered by \( R \) and \( \text{Ino} -/-> R \)

\( R \) not in 3NF: candidate keys of \( R \): \( \text{Ino} \text{ cname} \)
\[ \text{or} \]
\[ \text{cname bname} \]

\( \text{Ino} \rightarrow \text{amt bname} \) covered by \( R \)
\[ \{\text{amt bname}\} \) not a subset of a candidate key
Example: \[ R = R1 \cup R2 \]
\[ R1 = (lno, amt, bname) \]
\[ R2 = (lno, cname, bname), \quad F=Fc = \{ lno \rightarrow \text{amt bname,} \\
\quad \text{cname bname} \rightarrow lno \} \]

Q: Are \( R1, R2 \) in BCNF, 3NF or neither?

Ans: \( R1 \) in BCNF: \( lno \rightarrow \text{amt bname} \) covered by \( R1 \) and \( lno \rightarrow R1 \)

\( R2 \) not in BCNF: \( lno \rightarrow bname \) and \( lno \rightarrow R2 \)

\( R1 \) in 3NF (since it is in BCNF)

\( R2 \) in 3NF: \( R2 \)'s candidate keys: \( \text{cname bname} \) and \( \text{Ino cname} \)

\( \text{Ino} \rightarrow \text{bname} , \quad \text{bname} \subset \text{c.key} \)
\( \text{cname bname} \rightarrow \text{Ino} \), \( \text{Ino} \subset \text{c. key} \)
Algorithm 3NF ( R: relation, F: FD set)

1. Compute Fc
2. i ← 0
3. For each X→Y in Fc do
   if no Rj (1 <= j <= i) contains X,Y
      i ← i+1
      Ri ← X U Y
4. If no Rj (1<= j <= i) contains a candidate key for R
   i ← i+1
   Ri ← any candidate key for R
5. return (R1, R2, ..., Ri)
Example:
\[ R = (\text{bname, cname, banker, office}) \]
\[ F_c = \{ \text{banker} \rightarrow \text{bname, office}, \]
\[ \text{cname} \rightarrow \text{bname, banker} \} \]

Q1: candidate keys of \( R \): \( \text{cname bname or cnam banker} \)

Q2: decompose \( R \) into 3NF.

Ans: \( R \) is not in 3NF: \( \text{banker} \rightarrow \text{bname, office} \)
\( \{\text{bname, office}\} \) not a subset of a c. key

3NF:
\[ R_1 = (\text{banker, bname, office}) \]
\[ R_2 = (\text{cname, bname, banker}) \]
\[ R_3 = \text{Empty} \text{ (done)} \]
Theory and practice

Performance tuning:

Redundancy not the sole guide to decomposition

Workload matters too!!
  • nature of queries run
  • mix of updates, queries
  •.....

Workload can influence:
  BCNF vs 3NF
  may further decompose a BCNF into (4NF)
  may denormalize (i.e., undo a decomposition or add new columns)