

Lecture 16

Decompositions

Lossless Join Decomposition into BCNF

- Whether a relation schema Q is in BCNF or not?
 - For each $X \rightarrow Y$ in Q , whether X^+ fails to include all the attributes in Q
- Consider a relation R with FDs F . If $X \rightarrow Y$ violates BCNF, decompose R into $R - Y$ and XY . Repeat this for all other FDs that violate BCNF will give us:
 - A collection of relations that are in BCNF
 - Lossless join decomposition
- In general, several dependencies may cause violation of BCNF. The order in which we “deal with” them could lead to very different sets of relations!

Example 1

$R = (ABCDEFGH)$, $F = \{ABH \rightarrow C, A \rightarrow DE, BGH \rightarrow F, F \rightarrow ADH, BH \rightarrow GE\}$

Step 1: Find a FD that violates BCNF

Not $ABH \rightarrow C$ since $(ABH)^+$ includes all attributes

$A \rightarrow DE$ violates BCNF since A is not a superkey ($A^+ = ADE$)

Step 2: Split R into:

$R_1 = (ADE)$

$R_2 = (ABCFGH)$

- R_1 is in BCNF
- Decomposition is lossless since A is a key of R_1
- Is this lossless join decomposition dependency preserving?

Example 1 (Cont.)

$R_2 = (ABCFGH)$, $F_2 = \{ABH \rightarrow C, BGH \rightarrow F, F \rightarrow AH, BH \rightarrow G\}$

Step 1: Find a FD that violates BCNF

Not $ABH \rightarrow C$ or $BGH \rightarrow F$ since BH is a key of R_2

$F \rightarrow AH$ violates BCNF since F is not a superkey ($F^+ = AFH$)

Step 2: Split R_2 into:

$R_{21} = (FAH)$

$R_{22} = (BCFG)$

- Both R_{21} and R_{22} are in BCNF
- The decomposition is lossless since F is a key of R_{21}
- Is this lossless join decomposition dependency preserving?

Example 2

- CSJDPQV, key C, JP \rightarrow C, SD \rightarrow P, J \rightarrow S
- To deal with SD \rightarrow P, decompose into SDP, CSJDQV
- To deal with J \rightarrow S, decompose CSJDQV into JS and CJDQV

BCNF and Dependency Preserving

- A BCNF decomposition is *not necessarily* dependency-preserving
- In example 2, adding JPC to the collection of relations gives a dependency preserving decomposition
 - JPC tuples stored only for checking FD
(*Redundancy!*)

Dependency-Preserving Decomposition into 3NF

- Compromise: Not all redundancy removed, but dependency-preserving decompositions are always possible
- 3NF dependency-preserving decomposition is based on a *minimal cover*

3NF Synthesis Algorithm

- Given a schema $R = (ABCDEFGH)$ with functional dependencies $F = \{ABH \rightarrow C, A \rightarrow D, C \rightarrow E, BGH \rightarrow F, F \rightarrow AD, E \rightarrow F, BH \rightarrow E\}$
 1. Compute a minimal cover G for F
 - $G = \{BH \rightarrow C, A \rightarrow D, C \rightarrow E, F \rightarrow A, E \rightarrow F\}$
 2. For each LHS X of a functional dependency in G , create a relation schema in D with attributes $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$, where $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$ are the only dependencies in G with X as LHS (X is the key of this relation)
 - $R_1 = (BHC), R_2 = (AD), R_3 = (CE), R_4 = (FA), R_5 = (EF)$

3NF Synthesis Algorithm (Cont.)

3. Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property
- Every relation schema created by this algorithm is in 3NF

Synthesis

- The 3NF synthesis algorithm does preserve the original dependencies, but it makes no guarantee of preserving all of the information (may not be lossless join)
- It is called a relational **synthesis** algorithm, because each relation schema R_i in the decomposition is synthesized (constructed) from the set of functional dependencies in minimum cover G with the same LHS X

Dependency-Preserving and Lossless Join Decomposition into 3NF

- Keep step 1 and 2 from the algorithm of dependency-preserving decomposition into 3NF
- 3. If none of the relation schemas in D contains a key of R , then create one more relation schema in D that contains attributes that form a key of R
- 4. Eliminate redundant relations from the resulting set of relations
 - A relation R is redundant if R is a projection of another relation S in the schema (R is subsumed by S)



Is there always a lossless join and dependency preserving decomposition into 3NF?

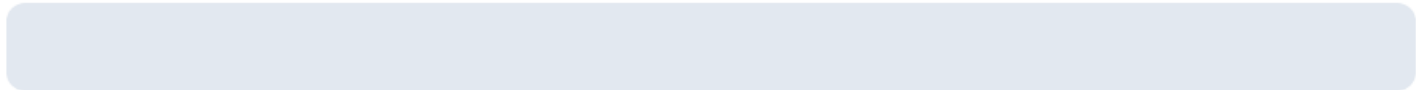
Yes

No



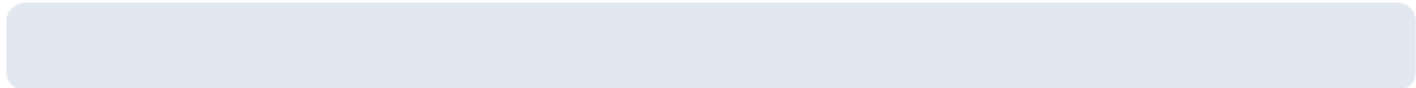
Is there always a lossless join and dependency preserving decomposition into 3NF?

Yes



0%

No

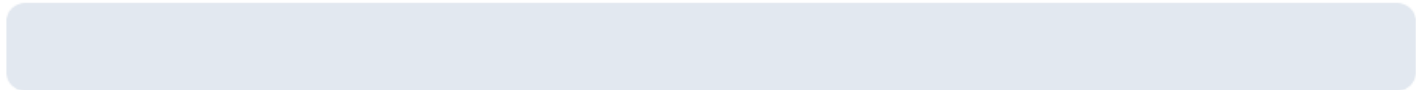


0%



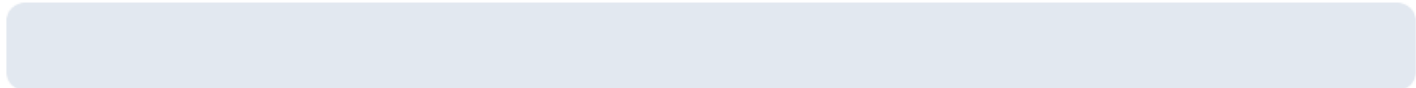
Is there always a lossless join and dependency preserving decomposition into 3NF?

Yes



0%

No



0%

Example

- Consider the Contracts relation with attributes CSJDPQV and the following FDs
 - $C \rightarrow CSJDPQV$, $JP \rightarrow C$, $SD \rightarrow P$, and $J \rightarrow S$
- Find a minimal cover
 - First replace $C \rightarrow CSJDPQV$ with the FDs:
 - $C \rightarrow S$, $C \rightarrow J$, $C \rightarrow D$, $C \rightarrow P$, $C \rightarrow Q$, and $C \rightarrow V$
 - No redundant attribute on the LHS of each FDs
 - $C \rightarrow P$ is deleted since it is implied by $C \rightarrow S$, $C \rightarrow D$, and $SD \rightarrow P$
 - $C \rightarrow S$ is deleted since it is implied by $C \rightarrow J$ and $J \rightarrow S$
 - The minimal cover is:
 - $C \rightarrow J$, $C \rightarrow D$, $C \rightarrow Q$, $C \rightarrow V$, $JP \rightarrow C$, $SD \rightarrow P$, and $J \rightarrow S$
- Apply 3NF synthesis algorithm, we have the relational schema:
 - $R_1=(CDJQV)$, $R_2=(JPC)$, $R_3=(SDP)$, $R_4=(JS)$
 - Since $(CDJQV)$ is a superkey, we are done

Exercise

- Find a dependency-preserving and lossless join decomposition into 3NF
 1. R1 (Emp_ssn, Pno, Esal, Ephone, Dno, Pname, Plocation)
FDs that hold on R1:
 $\text{Emp_ssn} \rightarrow \text{Esal}, \text{Ephone}, \text{Dno},$
 $\text{Pno} \rightarrow \text{Pname}, \text{Plocation},$
 $\text{Emp_ssn}, \text{Pno} \rightarrow \text{Esal}, \text{Ephone}, \text{Dno}, \text{Pname}, \text{Plocation}$
 2. R2 (Property_id, Lot#, County, Area) represented by PLCA
FDs that hold on R2: $\{P \rightarrow LCA, LC \rightarrow AP, A \rightarrow C\}$

Finding a Key

- Given: a relation R and a set of functional dependencies F on the attributes of R
 1. Set $K=R$
 2. For each attribute A in K
 - {Compute $(K-A)^+$ with respect to F ;
 - if $(K-A)^+$ contains all the attributes in R ,
 - then set $K=K-A$ }

Summary of Decompositions

- If a relation is in BCNF, it is free of redundancies that can be detected using FDs
 - Trying to ensure that all relations are in BCNF is a good heuristic
- Must consider whether all information and FDs are preserved
 - If a lossless-join, dependency preserving decomposition into BCNF is not possible (or unsuitable, given typical queries), should consider decomposition into 3NF
- Decompositions should be carried out and/or re-examined while keeping *performance requirements* in mind