Lecture 15
Properties of Decompositions

Decomposition of a Relation Schema

- A *decomposition* of R consists of replacing R by two or more relations such that:
  - Each new relation schema contains a subset of the attributes of R, and
  - Every attribute of R appears as an attribute of one or more new relations
- Store instances of the relation schemas produced by the decomposition, instead of instances of R
- Decompositions should be used only when needed

Decomposition Example

- Consider a relation obtained from Hourly_Emps:
  - Hourly_Emps (*ssn*, *name, lot, rating, hrly_wages*, *hrs_worked*)
  - Denote this relation schema by listing the attributes: SNLRWH
  - SNLRWH has FDs: S $\rightarrow$ SNLRWH and R $\rightarrow$ W
  - R $\rightarrow$ W causes violation of 3NF
  - Create a relation RW to store the associations and remove W from the main schema (decompose SNLRWH into SNLRH and RW)
- If we just store the projections of SNLRWH tuples onto SNLRH and RW, are there any potential problems that we should be aware of?

Problems with Decompositions

- Three potential problems:
  1. Some queries become more expensive
     - e.g., How much did sailor Joe earn? (salary = W*H)
  2. Given instances of the decomposed relations, we may not be able to reconstruct the corresponding instance of the original relation!
  3. Checking some dependencies may require joining the instances of the decomposed relations
- Tradeoff: must consider these issues vs. redundancy

Problem 2: Example

A	B	C
1	2	3
4	5	6
7	2	8

A	B
1	2
4	5
7	2

B	C
2	3
5	6
2	8

A	B	C
1	2	3
4	5	6
7	2	8
1	2	8
7	2	3

Lossless Join Decomposition

- Decomposition of R into X and Y has the lossless-join property w.r.t. a set of FDs F if for every instance *r* of R that satisfies F:
  $\pi_X(r) \bowtie \pi_Y(r) = r$
- It is always true that $r \subseteq \pi_X(r) \bowtie \pi_Y(r)$. If the other direction holds, the decomposition is lossless-join
- Definition extends to decomposition into 3 or more relations

Lossless Join Decomposition (Cont.)

- Repeated decompositions: R is decomposed into R_1 and R_2 through a lossless join decomposition, and R_1 is decomposed into R_{11} and R_{12} through another lossless join decomposition. Then the decomposition of R into R_{11}, R_{12}, and R_2 is lossless join
- All decompositions used to deal with redundancy should be lossless! (Avoids problem 2)

Testing Binary Decomposition for Lossless Join

- R_1, R_2 is a lossless join decomposition of R with respect to F iff at least one of the following dependencies is in F^+(the closure of F)
  - $(R_1 \cap R_2) \rightarrow R_1$
  - $(R_1 \cap R_2) \rightarrow R_2$
- For example, the decomposition of R into UV and R - V is lossless if $U \rightarrow V$ holds over R

Example

Id#	Name	Address	C#	Description	Grade
124	Jones	Phila	Phil7	Plato	A
789	Brown	Boston	Math8	Topology	C

- Given the FD set:

  Id# $\rightarrow$ Name, Address
  C# $\rightarrow$ Description
  Id#, C# $\rightarrow$ Grade

- Is (Id#, Name, Address) and (Id#, C#, Description, Grade) a lossless decomposition?
- What happens if we decompose on (Id#, Name, Address) and (C#, Description, Grade)?

Dependency Preserving Decomposition

- Consider a relation: Contracts (contractid, supplierid, projectid, deptid, partid, qty, value)
  - CSJDPQV, C is the key, JP $\rightarrow$ C and SD $\rightarrow$ P
  - BCNF decomposition: CSJDQV and SDP
  - Problem: Checking JP $\rightarrow$ C requires a join!
- If R is decomposed into X, Y and Z through a dependency preserving decomposition, and if we enforce the FDs that hold on X, on Y and on Z, then all FDs that were given to hold on R must also hold. (Avoids problem 3)

Dependency Preserving Decomposition (Cont.)

- Suppose we need to update a relation in a database. Can we easily check whether an FD $X \rightarrow Y$ is violated?
  - We can if $X \cup Y$ is contained within the set of attributes
- The projection of an FD set F onto a set of attributes Z

  $$F_Z = \{X \rightarrow Y \mid X \rightarrow Y \in F^+ \text{ and } X \cup Y \in Z\}$$

- A decomposition $R_1, \ldots, R_k$ is dependency preserving if $F^+ = (F_{R_1} \cup \ldots \cup F_{R_k})^+$
- Dependency preserving decomposition hasn't "lost" any essential FDs

Dependency Preserving Decompositions (Cont.)

- Decomposition of R into X and Y is *dependency preserving* if $(F_X \cup F_Y)^+ = F^+$
  - i.e., if we consider only dependencies in the closure F^+ that can be checked in X without considering Y, and in Y without considering X, these imply all dependencies in F^+
- Important to consider F^+, not F
  - ABC, $A \rightarrow B$, $B \rightarrow C$, $C \rightarrow A$, decomposed into AB and BC
  - Is this dependency preserving? Is $C \rightarrow A$ preserved?
- Dependency preserving does not imply lossless join and vice versa!
  - ABC, $A \rightarrow B$, decomposed into AB and BC

Example 1

- Given a relation schema: {Sname, Sadd, City, Zip, Item, Price} and an FD set:
  - FD1: Sname $\rightarrow$ Sadd, City
  - FD2: Sadd, City $\rightarrow$ Zip
  - FD3: Sname, Item $\rightarrow$ Price
- Consider the decomposition: {Sname, Sadd, City, Zip} and {Sname, Item, Price}
  - Is it lossless?
  - Is it dependency preserving?
  - What if we replaced FD1 by Sname, Sadd $\rightarrow$ City ?

Example 2

- Given a relation schema {Student, Teacher, Subject} and an FD set
  - FD1: Teacher $\rightarrow$ Subject
  - FD2: Student, Subject $\rightarrow$ Teacher
- Consider the decomposition: {Student, Teacher} and {Teacher, Subject}
  - Is it lossless?
  - Is it dependency preserving?

Minimum Sets of Functional Dependencies

Equivalence of FD Sets

- Two sets of FDs, F and G, are equivalent if $F^+ = G^+$
- Example: $\{AB \rightarrow C, A \rightarrow B\}$ and $\{A \rightarrow C, A \rightarrow B\}$ are equivalent
- F^+ contains a huge number of FDs (exponential in the size of the schema). One naturally looks for small equivalent
- A set of FDs F covers another set of FDs E if every FD in E is also in F^+

Is E covered by F when every dependency in E can be inferred from F?

Yes

No

Is E covered by F when every dependency in E can be inferred from F?

Yes

0%

No

0%

Is E covered by F when every dependency in E can be inferred from F?

Yes

0%

No

0%

Minimal Cover

- Minimal cover G for a set of FDs F:
  - $F^+ = G^+$
  - RHS 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
- Every FD in minimal cover 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 a minimal cover: {$A \rightarrow B$, $ACD \rightarrow E$, $EF \rightarrow G$, $EF \rightarrow H$)

Minimal Cover (Cont.)

- Formal Definition: A FD set F is minimal if
  1. Every FD in F is of the form $X \rightarrow A$, where A is a single attribute
  2. No redundant attributes from LHS of FDs
  3. No redundant FDs
- Every dependency is required and is as small as possible
  - Each attribute on the left side is necessary
  - Right side is a single attribute
- Example:

  $\{A \rightarrow C, A \rightarrow B\}$ is a minimal cover for $\{AB \rightarrow C, A \rightarrow B\}$

Finding a Minimal Cover

1. Replace each functional dependency $X \rightarrow \{A_1, A_2, \ldots, A_n\}$ in F by the *n* functional dependencies $X \rightarrow A_1$, $X \rightarrow A_2, \ldots, X \rightarrow A_n$
2. Remove all redundant attributes from LHS of FDs
3. Remove all redundant FDs

Example

Find a minimal cover for {$ABH \rightarrow C$, $A \rightarrow D$, $C \rightarrow E$, $BGH \rightarrow F$, $F \rightarrow AD$, $E \rightarrow F$, $BH \rightarrow E$}

Step1: RHS are single attribute. Applying decomposition rule:
$ABH \rightarrow C$, $A \rightarrow D$, $C \rightarrow E$,$BGH \rightarrow F$, $F \rightarrow A$, $F \rightarrow D$, $E \rightarrow F$, $BH \rightarrow E$

Step2: Remove redundant attributes from LHS:

$BH \rightarrow C$ $\quad (BH)^+ \Rightarrow (BEH) \Rightarrow (BEFH) \Rightarrow (ABEFH) \Rightarrow (ABCEFH)$

$BH \rightarrow E \quad E \rightarrow F \quad F \rightarrow A \quad ABH \rightarrow C$

$A \rightarrow D$

$C \rightarrow E$

$BH \rightarrow F$

$F \rightarrow A$

$F \rightarrow D$

$E \rightarrow F$

$BH \rightarrow E$

Example (Cont.)

Step 3: Remove redundant FDs: $BH \rightarrow F$, $F \rightarrow D$, and $BH \rightarrow E$ since:

$BH \rightarrow F$ can be derived from $BH \rightarrow C$, $C \rightarrow E$, $E \rightarrow F$

$F \rightarrow D$ can be derived from $F \rightarrow A$, $A \rightarrow D$

$BH \rightarrow E$ can be derived from $BH \rightarrow C$, $C \rightarrow E$

A minimal cover is

$\{BH \rightarrow C, A \rightarrow D, C \rightarrow E, F \rightarrow A, E \rightarrow F\}$

Is it possible that a set of FDs can have several minimal covers?

Yes

No

Is it possible that a set of FDs can have several minimal covers?

Yes	0%
No	0%

Is it possible that a set of FDs can have several minimal covers?

Yes

0%

No

0%

Exercise

- Find a minimal cover for $\{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$
- Find a minimal cover for $\{A \rightarrow B, ABCD \rightarrow E, EF \rightarrow G, EF \rightarrow H, ACDF \rightarrow EG\}$