Relational Database Decomposition Criteria

Robert Keller
11 February 2004
Type of Decomposition

- The type of decomposition considered here consists of **projecting** relations onto subsets of their attributes.

- It is desired to be able to reconstruct the original relation from the components of the decomposition using the **natural join** operator, described elsewhere.
Projection

To project a relation on a subset of its attributes, simply discard columns that are not in the set, then eliminate duplicates from the result.

If $S$ is the subset, the projection is denoted $\pi_s(R)$. 
Projection Example

\[ \{D,C,Y\}(R) \]

\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\textbf{S} & \textbf{D} & \textbf{C} & \textbf{X} & \textbf{Y} & \textbf{M} & \textbf{I} \\
\textbf{student} & \textbf{dept.} & \textbf{course} & \textbf{section} & \textbf{year} & \textbf{semester} & \textbf{instructor} \\
\hline
John & CS & 60 & 1 & 2003 & fall & Dodds \\
Susan & CS & 60 & 1 & 2003 & fall & Dodds \\
Fred & CS & 60 & 1 & 2004 & spring & Stone \\
John & CS & 60 & 2 & 2004 & spring & Hadas \\
Susan & CS & 70 & 1 & 2004 & spring & O'Neil \\
Susan & Math & 55 & 1 & 2004 & spring & Yong \\
\hline
\end{tabular}

\begin{tabular}{|c|c|c|}
\hline
\textbf{D} & \textbf{C} & \textbf{Y} \\
\textbf{dept.} & \textbf{course} & \textbf{year} \\
\hline
CS & 60 & 2003 \\
CS & 60 & 2003 \\
CS & 60 & 2004 \\
CS & 60 & 2004 \\
CS & 70 & 2004 \\
Math & 55 & 2004 \\
\hline
\end{tabular}
The join of two relations has as its set of attributes the union of the sets of attributes of the original relations.

The set of tuples in the join are all those tuples formed from pairs of tuples, one from each of the original relations, where the values of the attributes common to both tuples are the same and the values of all attributes are the same as in the two tuples.
Join Example

D  C  Y
(CS, 60, 2003)

(John, CS, 60)

S  D  C

(John, CS, 60, 2003)

\{D, C\} are common
Join Example

\[ \text{join}(S, T) \]

\[ \{D, C\} \text{ in common} \]

<table>
<thead>
<tr>
<th>S</th>
<th>D</th>
<th>C</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dept.</td>
<td>course</td>
<td>year</td>
</tr>
<tr>
<td>CS</td>
<td>60</td>
<td>2003</td>
<td></td>
</tr>
<tr>
<td>CS</td>
<td>60</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>CS</td>
<td>70</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>Math</td>
<td>55</td>
<td>2004</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T</th>
<th>S</th>
<th>D</th>
<th>C</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>student</td>
<td>dept.</td>
<td>course</td>
<td></td>
<td></td>
</tr>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
<td></td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>70</td>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>Susan</td>
<td>Math</td>
<td>55</td>
<td>2004</td>
<td></td>
</tr>
</tbody>
</table>
Degenerate Joins

- Normally, those sets have at least one attribute in comment. If not, we have a degenerate join, a **Cartesian product**.
Two Desirable Properties when Decomposing

- Losslessness (what UW calls “recovery”)
- Dependency-preserving
Lossless Decomposition

- This means that when the projections are joined, we get exactly the original relation and no more.

- It is not difficult to see for oneself that we can’t get less.
A Relation that Projects to the Previous Example

<table>
<thead>
<tr>
<th>S student</th>
<th>D dept.</th>
<th>C course</th>
<th>Y year</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>70</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>Math</td>
<td>55</td>
<td>2004</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T student</th>
<th>D dept.</th>
<th>C course</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>70</td>
</tr>
<tr>
<td>Susan</td>
<td>Math</td>
<td>55</td>
</tr>
</tbody>
</table>
The Previous Example: Not Lossless

\[
\text{join}(S, T) = \{D, C\} \text{ in common}
\]

\[
\begin{array}{|c|c|c|}
\hline
\text{dept.} & \text{course} & \text{year} \\
\hline
\text{CS} & 60 & 2003 \\
\text{CS} & 60 & 2004 \\
\text{CS} & 70 & 2004 \\
\text{Math} & 55 & 2004 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
\text{student} & \text{dept.} & \text{course} \\
\hline
\text{John} & \text{CS} & 60 \\
\text{Susan} & \text{CS} & 60 \\
\text{Fred} & \text{CS} & 60 \\
\text{Susan} & \text{CS} & 70 \\
\text{Susan} & \text{Math} & 55 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|c|}
\hline
\text{student} & \text{dept.} & \text{course} & \text{year} \\
\hline
\text{John} & \text{CS} & 60 & 2003 \\
\text{John} & \text{CS} & 60 & 2004 \\
\text{Susan} & \text{CS} & 60 & 2003 \\
\text{Susan} & \text{CS} & 60 & 2004 \\
\text{Fred} & \text{CS} & 60 & 2003 \\
\text{Fred} & \text{CS} & 60 & 2004 \\
\text{Susan} & \text{CS} & 70 & 2004 \\
\text{Susan} & \text{Math} & 55 & 2004 \\
\hline
\end{array}
\]
The Preceding Decomposition is Not Lossless

### Original

<table>
<thead>
<tr>
<th>student</th>
<th>dept.</th>
<th>course</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>70</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>Math</td>
<td>55</td>
<td>2004</td>
</tr>
</tbody>
</table>

### Join

<table>
<thead>
<tr>
<th>student</th>
<th>dept.</th>
<th>course</th>
<th>year</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>John</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
<td>2003</td>
</tr>
<tr>
<td>Fred</td>
<td>CS</td>
<td>60</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>CS</td>
<td>70</td>
<td>2004</td>
</tr>
<tr>
<td>Susan</td>
<td>Math</td>
<td>55</td>
<td>2004</td>
</tr>
</tbody>
</table>
Dependency Preserving

- When a relation is projected, some of its FD’s might not be representable in the projection, because the attributes are no longer present.

- The effective FD’s in the decomposed scheme are those derived from the FD’s in the projections.

- If these are the same as the originals, the decomposition is dependency preserving.
Dependency Preserving Example

- Consider relation ABCD, with FD’s: A->B, B->C, C->D
- Decompose into two relations: ABC and CD.
- ABC supports the FD’s A->B, B->C.
- CD supports the FD C->D.
- All the original dependencies are preserved.
Non-Dependency Preserving Example

- Consider relation ABCD, with FD’s: A->B, B->C, C->D
- Decompose into two relations: ACD and BC.
  - ACD supports the FD C->D and the implied FD A->C.
  - BC supports the FD B->C.
- However, no relation supports A->B, so that dependency is not preserved.
Normal Form Decompositions Contrasted

- 3NF Decomposition:
  - Lossless
  - Dependency preserving

- BCNF Decomposition:
  - Lossless
  - Not necessarily dependency-preserving
  - Component relations are all BCNF, and thus 3NF

- 4NF Decomposition:
  - Lossless
  - Not necessarily dependency-preserving
  - Component relations are all 4NF, and thus BCNF and 3NF

- No decomposition is guaranteed to preserve all MVD’s
Remembering Definitions

- Let $FD(X, A)$ mean $X \rightarrow A$ **non-trivially**!
- Let $SK(X)$ mean $X$ is a SuperKey.
- Let $P(X)$ mean $X$ is prime (in some key)

- **BCNF:** $(X, F) \quad FD(X, A) \quad SK(X)$
- **3NF:** $(X, F) \quad FD(X, A) \quad (SK(X) \quad P(A))$

- **$\beta$ BCNF:** $(X, F) \quad FD(X, A) \quad \beta \quad SK(X)$
- **$\gamma$ 3NF:** $(X, F) \quad FD(X, A) \quad (\gamma \quad SK(X) \quad \gamma \quad P(A))$
3NF Decomposition

- Any set of attributes not involved in any FD becomes a relation by itself.
- Assume a **minimal** set of dependencies (no implied dependencies in the set).
- A relation violates 3NF when there is an FD $X \rightarrow A$ with $X$ not a superkey and $A$ non-prime.
- For each such FD, create a relation consisting of only the attributes in the FD.
- Add a separate relation for the **overall key** if needed, and delete any relation subsumed.
3NF Example

- CTHRSG (course-teacher-hour-room-student-grade) relation
- C -> T, HR -> C, HT -> R, CS -> G, HS -> R
3NF Example

- C -> T
- HR -> C
- HT -> RCS -> G
- HS -> R

- Keys: HS
- Violations: all but HS -> R
- 3NF decomp:
  - HRS, CT,
  - HRC, HRT,
  - CSG

- key is present in HRS
Same Example but BCNF

- C -> T
- HR -> C
- HT -> R
- CS -> G
- HS -> R

**Keys:** HS

**BCNF decomposes:**
- HRS, CT, CHS, CSG

**vs. 3NF decomposes:**
- HRS, CT, HRC, HRT, CSG

(worth checking)
Fifth Normal Form?

- Also known as Projection-Join Normal Form (PJNF).
- 3NF, BCNF, and 4NF decompose **pairwise**.
- Accomodates relations having no pairwise lossless decomposition, but with a 3-way decomposition.

<table>
<thead>
<tr>
<th>S</th>
<th>P</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td>J2</td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td>J1</td>
</tr>
<tr>
<td>S2</td>
<td>P1</td>
<td>J1</td>
</tr>
<tr>
<td>S1</td>
<td>P1</td>
<td>J1</td>
</tr>
</tbody>
</table>
How to Check for Dependency Preservation

- Compute all implied FD’s.
- Determine which FD’s are represented by the projections.
- Compute the FD’s that are implied by the ones represented by the projections.
- See if they are the same as the original implied FD’s.
How to Check Losslessness

Tableau Method

- The collective set of attributes head the columns of a table.
- The relation names head the rows of the table.
- In the $i^{th}$ row corresponding to each relation:
  - Put $a_j$ in the $j^{th}$ column if the $j^{th}$ attribute is in the relation.
  - Put $b_{ij}$ otherwise.
Lossless Check (cont’d)

- Repeatedly consider the FD’s. Whenever two rows agree in their LHS columns, force the RHS columns to agree by changing one to the other.
- Continue until no more changes can be made to the table.
- If any row ends up with all a’s, the decomposition is lossless.
Lossless Check Example

- Five attributes: ABCDE
- Three relations: ABC, AD, BDE
- FD’s: A -> BD, B -> E

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>b14</td>
<td>b15</td>
</tr>
<tr>
<td>AD</td>
<td>a1</td>
<td>b22</td>
<td>b23</td>
<td>a4</td>
<td>b25</td>
</tr>
<tr>
<td>BDE</td>
<td>b21</td>
<td>a2</td>
<td>b33</td>
<td>a4</td>
<td>a5</td>
</tr>
</tbody>
</table>
# Lossless Check Example

**FD’s:** \( A \rightarrow BD, \ B \rightarrow E \)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>b14</td>
<td>b15</td>
</tr>
<tr>
<td>AD</td>
<td>a1</td>
<td>b22</td>
<td>b23</td>
<td>a4</td>
<td>b25</td>
</tr>
<tr>
<td>BDE</td>
<td>b21</td>
<td>a2</td>
<td>b33</td>
<td>a4</td>
<td>a5</td>
</tr>
</tbody>
</table>

Consider FD: \( A \rightarrow BD \)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>a4</td>
<td>b15</td>
</tr>
<tr>
<td>AD</td>
<td>a1</td>
<td>a2</td>
<td>b23</td>
<td>a4</td>
<td>b25</td>
</tr>
<tr>
<td>BDE</td>
<td>b21</td>
<td>a2</td>
<td>b33</td>
<td>a4</td>
<td>a5</td>
</tr>
</tbody>
</table>
Lossless Check Example

FD’s: A -> BD, B -> E

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>a4</td>
<td>b15</td>
</tr>
<tr>
<td>AD</td>
<td>a1</td>
<td>a2</td>
<td>b23</td>
<td>a4</td>
<td>b25</td>
</tr>
<tr>
<td>BDE</td>
<td>b21</td>
<td>a2</td>
<td>b33</td>
<td>a4</td>
<td>a5</td>
</tr>
</tbody>
</table>

consider FD: B -> E

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>a1</td>
<td>a2</td>
<td>a3</td>
<td>a4</td>
<td>a5</td>
</tr>
<tr>
<td>AD</td>
<td>a1</td>
<td>a2</td>
<td>b23</td>
<td>a4</td>
<td>a5</td>
</tr>
<tr>
<td>BDE</td>
<td>b21</td>
<td>a2</td>
<td>b33</td>
<td>a4</td>
<td>a5</td>
</tr>
</tbody>
</table>

lossless