Anomalies

Hourly_emps( ssn, name, lot, rating, hourly_wages, hours_worked)

Suppose hourly wages is determined by rating:

\[
\text{rating} \rightarrow \text{hourly}_wages
\]

<table>
<thead>
<tr>
<th>ssn</th>
<th>name</th>
<th>lot</th>
<th>rating</th>
<th>hourly_wages</th>
<th>hours_worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>brutus</td>
<td>48</td>
<td>8</td>
<td>10</td>
<td>40</td>
</tr>
<tr>
<td>85</td>
<td>art</td>
<td>22</td>
<td>8</td>
<td>10</td>
<td>30</td>
</tr>
<tr>
<td>95</td>
<td>bob</td>
<td>35</td>
<td>5</td>
<td>7</td>
<td>30</td>
</tr>
<tr>
<td>96</td>
<td>frodo</td>
<td>35</td>
<td>5</td>
<td>7</td>
<td>32</td>
</tr>
<tr>
<td>22</td>
<td>dustin</td>
<td>35</td>
<td>8</td>
<td>10</td>
<td>40</td>
</tr>
</tbody>
</table>

- Redundant storage: association between rating 8 and hourly wages 10 repeated 3 times.
- Update anomalies: hourly_wages updated in first tuple but not second
- Insertion anomalies: must know hourly_wage for rating value
- Deletion anomalies: delete all tuples with certain rating value, lost assoc.
Relational decomposition

rating $\rightarrow$ hourly_wages

<table>
<thead>
<tr>
<th>ssn</th>
<th>name</th>
<th>lot</th>
<th>rating</th>
<th>hourly_wages</th>
<th>hours_worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>brutus</td>
<td>48</td>
<td>8</td>
<td>10</td>
<td>40</td>
</tr>
<tr>
<td>85</td>
<td>art</td>
<td>22</td>
<td>8</td>
<td>10</td>
<td>30</td>
</tr>
<tr>
<td>95</td>
<td>bob</td>
<td>35</td>
<td>5</td>
<td>7</td>
<td>30</td>
</tr>
<tr>
<td>96</td>
<td>frodo</td>
<td>35</td>
<td>5</td>
<td>7</td>
<td>32</td>
</tr>
<tr>
<td>22</td>
<td>dustin</td>
<td>35</td>
<td>8</td>
<td>10</td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>rating</th>
<th>hourly_wages</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>
Decompositions in General

\[ R(A_1, \ldots, A_n, B_1, \ldots, B_m, C_1, \ldots, C_p) \]

\[ R_1(A_1, \ldots, A_n, B_1, \ldots, B_m) \]

\[ R_2(A_1, \ldots, A_n, C_1, \ldots, C_p) \]

\[ R_1 = \text{projection of } R \text{ on } A_1, \ldots, A_n, B_1, \ldots, B_m \]

\[ R_2 = \text{projection of } R \text{ on } A_1, \ldots, A_n, C_1, \ldots, C_p \]
Problems With Decomposition

• Can we get the data back correctly?
  – Lossless decomposition
  – Discuss next

• Can we recover the FD’s on the ‘big’ table from the FD’s on the small tables?
  – Dependency-preserving decomposition
  – Please read 19.5.2
Decompositions in General

\[ R(A_1, \ldots, A_n, B_1, \ldots, B_m, C_1, \ldots, C_p) \]

\[ R_1(A_1, \ldots, A_n, B_1, \ldots, B_m) \]

\[ R_2(A_1, \ldots, A_n, C_1, \ldots, C_p) \]

\[ R_1 = \text{projection of } R \text{ on } A_1, \ldots, A_n, B_1, \ldots, B_m \]

\[ R_2 = \text{projection of } R \text{ on } A_1, \ldots, A_n, C_1, \ldots, C_p \]

Decomposition is **lossless** if: \[ R_1 \text{ join } R_2 = R \]
Lossless Decomposition

• Sometimes it is correct:

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>19.99</td>
<td>Gadget</td>
</tr>
<tr>
<td>OneClick</td>
<td>24.99</td>
<td>Camera</td>
</tr>
<tr>
<td>Gizmo</td>
<td>19.99</td>
<td>Camera</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>19.99</td>
</tr>
<tr>
<td>OneClick</td>
<td>24.99</td>
</tr>
<tr>
<td>Gizmo</td>
<td>19.99</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Gadget</td>
</tr>
<tr>
<td>OneClick</td>
<td>Camera</td>
</tr>
<tr>
<td>Gizmo</td>
<td>Camera</td>
</tr>
</tbody>
</table>
Lossy Decomposition

• Sometimes it is not:

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>19.99</td>
<td>Gadget</td>
</tr>
<tr>
<td>OneClick</td>
<td>24.99</td>
<td>Camera</td>
</tr>
<tr>
<td>Gizmo</td>
<td>19.99</td>
<td>Camera</td>
</tr>
</tbody>
</table>

What’s wrong ??
Decompositions in General

R(A_1, ..., A_n, B_1, ..., B_m, C_1, ..., C_p)

R_1(A_1, ..., A_n, B_1, ..., B_m)
R_2(A_1, ..., A_n, C_1, ..., C_p)

Theorem The decomposition is lossless IFF
A_1, ..., A_n \rightarrow B_1, ..., B_m \text{ or } A_1, ..., A_n \rightarrow C_1, ..., C_p \text{ holds.}

Example: name \rightarrow price, hence the first decomposition is lossless.
What happens to FDs after decomposition?

- We must “project” F onto each relation resulting from the decomposition.
- Suppose F is a set of FDs over attributes in U.
- If Z is a subset of U, the projection of F on Z is:
  - all FDs $X \rightarrow Y$ in $F^+$ such that $XY \subseteq Z$
  - $F=\{A \rightarrow B, B \rightarrow C\}$ projected onto AC?
Normal Forms

• Decomposition into Boyce Codd Normal Form (BCNF)
  – Lossless

• Decomposition into 3rd Normal Form
  – Lossless
  – Dependency preserving
Boyce-Codd Normal Form

Given relation R, and set of FD’s F on R.

R is in BCNF if:

For each FD $X \rightarrow A$, one of following is true:

- A is in X (i.e. it is a trivial FD), or
- X is a superkey for R

Intuitively: the only non-trivial dependencies are those in which a key determines some attributes
BCNF Decomposition Algorithm

Repeat
   choose $A_1, \ldots, A_m \rightarrow B_1, \ldots, B_n$ that violates the BNCF condition
   split $R$ into $R_1(A_1, \ldots, A_m, B_1, \ldots, B_n)$ and $R_2(A_1, \ldots, A_m, [\text{rest}])$
   continue with both $R_1$ and $R_2$
Until no more violations

Heuristics:
choose $B_1, \ldots, B_n$
“as large as possible”
## BCNF Example

<table>
<thead>
<tr>
<th>Name</th>
<th>SSN</th>
<th>PhoneNumber</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred</td>
<td>123-45-6789</td>
<td>206-555-1234</td>
<td>Seattle</td>
</tr>
<tr>
<td>Fred</td>
<td>123-45-6789</td>
<td>206-555-6543</td>
<td>Seattle</td>
</tr>
<tr>
<td>Joe</td>
<td>987-65-4321</td>
<td>908-555-2121</td>
<td>Westfield</td>
</tr>
<tr>
<td>Joe</td>
<td>987-65-4321</td>
<td>908-555-1234</td>
<td>Westfield</td>
</tr>
</tbody>
</table>

FD: \( \text{SSN} \rightarrow \text{Name, City} \)

Key: \{\text{SSN, PhoneNumber}\}

Is it in BCNF?
BCNF Example

<table>
<thead>
<tr>
<th>Name</th>
<th>SSN</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred</td>
<td>123-45-6789</td>
<td>Seattle</td>
</tr>
<tr>
<td>Joe</td>
<td>987-65-4321</td>
<td>Westfield</td>
</tr>
</tbody>
</table>

SSN → Name, City

Let’s check anomalies:
• Redundancy ?
• Update ?
• Delete ?
Example

- $R(A,B,C,D)$  $A \rightarrow B$,  $B \rightarrow C$

- Key:  $AD$
- Violations of BCNF:
  $A \rightarrow B$,  $A \rightarrow C$,  $A \rightarrow BC$,  $B \rightarrow C$

- Pick $A \rightarrow BC$ first: split into $R_1(A,B,C)$  $R_2(A,D)$
- In $R_1$:  $B \rightarrow C$; split into $R_{11}(A,B)$, $R_{12}(B,C)$

- Final answer: $R_{11}(A,B)$, $R_{12}(B,C)$, $R_2(A,D)$
BCNF and Dependencies

FD’s:  Unit $\rightarrow$ Company;  Company, Product $\rightarrow$ Unit
So, there is a BCNF violation, and we decompose.

In BCNF we lose the FD:  Company, Product $\rightarrow$ Unit
Solution: 3rd Normal Form (3NF)

A simple condition for removing anomalies from relations:

R is in 3NF if:

For each FD \( X \rightarrow A \), one of following is true:

- A is in X (i.e. it is a trivial FD), or
- X is a superkey for R
- A is part of some key for R

Please read in the book!
3NF Discussion

• 3NF decomposition v.s. BCNF decomposition:
  – Use same decomposition steps, for a while
  – 3NF may stop decomposing, while BCNF continues

• Tradeoffs
  – BCNF = no anomalies, but may lose some FDs
  – 3NF = keeps all FDs, but may have some anomalies
Summary of Schema Refinement

- If a relation is in BCNF, it is free of redundancies that can be detected using FDs. Thus, trying to ensure that all relations are in BCNF is a good heuristic.

- If a relation is not in BCNF, we can try to decompose it into a collection of BCNF relations.
  - Must consider whether all FDs are preserved. If a lossless, 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.