Introduction to Data Management
CSE 344

Lectures 18: BCNF
Announcements

• Midterm is posted and grades are up
  – Have until 8pm tonight for regrade requests

• WQ6 - 11pm July 31 (Monday)
• HW6 is up due Aug 1 (Tuesday)
  – Database design

• Will drop HW8 - not enough time to get to Spark lecture before it is due.
Review: Relation Decomposition

Break the relation into two:

<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>
</tbody>
</table>

Anomalies have gone:
- No more repeated data
- Easy to move Fred to “Bellevue” (how ?)
- Easy to delete all Joe’s phone numbers (how ?)
Review: Functional Dependencies (FDs)

**Definition**: \( A_1, \ldots, A_m \Rightarrow B_1, \ldots, B_n \) holds in \( R \) if:

\[
\forall t, t' \in R, \quad (t.A_1 = t'.A_1 \quad \ldots \quad t.A_m = t'.A_m \Rightarrow t.B_1 = t'.B_1 \quad \ldots \quad t.B_n = t'.B_n)
\]

<table>
<thead>
<tr>
<th>R</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A_1</td>
<td>...</td>
<td>A_m</td>
<td>B_1</td>
<td>...</td>
<td>B_n</td>
</tr>
<tr>
<td>t</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>t'</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

if \( t, t' \) agree here then \( t, t' \) agree here
Closure Algorithm

\[ X = \{ A_1, \ldots, A_n \} \]

Repeat until \( X \) doesn’t change do:
  
  if \( B_1, \ldots, B_n \rightarrow C \) is a FD and \( B_1, \ldots, B_n \) are all in \( X \)
  then add \( C \) to \( X \).

Example:

1. name \( \rightarrow \) color
2. category \( \rightarrow \) department
3. color, category \( \rightarrow \) price

\[ \{ \text{name, category} \}^+ = \{ \text{name, category, color, department, price} \} \]

Hence: name, category \( \rightarrow \) color, department, price
Keys

• A **superkey** is a set of attributes $A_1, \ldots, A_n$ s.t. for any other attribute $B$ in the same relation, we have $A_1, \ldots, A_n \rightarrow B$

• A **key** is a minimal superkey
  – A superkey and for which no subset is a superkey
Computing (Super)Keys

• For all sets $X$, compute $X^+$

• If $X^+ = \{\text{all attributes}\}$, then $X$ is a superkey

• Try reducing to the minimal $X$’s to get the key
Example

Product(name, price, category, color)

name, category → price
category → color

What is the key?
Example

Product(name, price, category, color)

\[
\begin{align*}
\text{name, category } & \rightarrow \text{ price} \\
\text{category } & \rightarrow \text{ color}
\end{align*}
\]

What is the key?

\[(\text{name, category})^+ = \{ \text{name, category, price, color} \}\]

Hence (name, category) is a key
How To Eliminate Anomalies

Main idea:

- $X \rightarrow A$ is OK if $X$ is a (super)key
- $X \rightarrow A$ is not OK otherwise
  - Need to decompose the table, but how?

Boyce-Codd Normal Form
Boyce-Codd Normal Form

There are no “bad” FDs:

**Definition.** A relation R is in BCNF if:

Whenever $X \rightarrow B$ is a non-trivial dependency, then $X$ is a superkey.

Equivalently:

**Definition.** A relation R is in BCNF if:

$\forall X \text{ in } X \rightarrow B,$

either $X^+ = X$ or $X^+ = [\text{all attributes}]$
BCNF Decomposition Algorithm

Normalize(R)
find X in X→B s.t.: X ≠ X⁺ and X⁺ ≠ [all attributes]
if (not found) then R is in BCNF
let Y = X⁺ - X; Z = [all attributes] - X⁺
decompose R into R1(X Y) and R2(X Z)
Normalize(R1); Normalize(R2);
Want $X$ in $X \rightarrow B$ s.t.: $X = X^+$ or $X^+ = [\text{all attributes}]$

## 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>

$\text{SSN} \rightarrow \text{Name, City}$

The only key is: $\{\text{SSN, PhoneNumber}\}$

Hence $\text{SSN} \rightarrow \text{Name, City}$ is a “bad” dependency

In other words:

$\text{SSN}^+ = \text{SSN, Name, City}$ and is neither $\text{SSN}$ nor $\text{All Attributes}$
Want $X$ in $X \rightarrow B$ s.t.: $X = X^+$ or $X^+ = [\text{all attributes}]$
Find $X$ in $X \rightarrow B$ s.t.: $X \neq X^+$ and $X^+ \neq \{\text{all attributes}\}$

Example BCNF Decomposition

Person(name, SSN, age, hairColor, phoneNumber)

SSN $\rightarrow$ name, age
age $\rightarrow$ hairColor
Find $X$ in $X \rightarrow B$ s.t.: $X \neq X^+$ and $X^+ \neq \text{[all attributes]}$

**Example BCNF Decomposition**

$\text{Person}(\text{name, SSN, age, hairColor, phoneNumber})$

$\text{SSN} \rightarrow \text{name, age}$

$\text{age} \rightarrow \text{hairColor}$

**Iteration 1:**

**Person:** $\text{SSN}^+ = \text{SSN, name, age, hairColor}$

**Decompose into:** $P(\text{SSN, name, age, hairColor})$

$\text{Phone}(\text{SSN, phoneNumber})$
Find $X$ in $X \rightarrow B$ s.t.: $X \neq X^+$ and $X^+ \neq [\text{all attributes}]

**Example BCNF Decomposition**

$\text{Person}(\text{name}, \text{SSN}, \text{age}, \text{hairColor}, \text{phoneNumber})$

$\text{SSN} \rightarrow \text{name, age}$
$\text{age} \rightarrow \text{hairColor}$

Iteration 1: $\text{Person}$: $\text{SSN}^+ = \text{SSN, name, age, hairColor}$
Decompose into: $P(\text{SSN, name, age, hairColor})$
$\text{Phone}(\text{SSN, phoneNumber})$

Iteration 2: $P$: $\text{age}^+ = \text{age, hairColor}$
Decompose: $\text{People}(\text{SSN, name, age})$
$\text{Hair}(\text{age, hairColor})$
$\text{Phone}(\text{SSN, phoneNumber})$

What are the keys?
Find $X$ in $X \rightarrow B$ s.t.: $X \neq X^+$ and $X^+ \neq [\text{all attributes}]$

**Example BCNF Decomposition**

$\text{Person(name, SSN, age, hairColor, phoneNumber)}$

- $\text{SSN} \rightarrow \text{name, age}$
- $\text{age} \rightarrow \text{hairColor}$

**Iteration 1:** $\textbf{Person:}$ $\text{SSN}^+ = \text{SSN, name, age, hairColor}$

Decompose into: $\text{P(SSN, name, age, hairColor)}$

- $\text{Phone(SSID, phoneNumber)}$

**Iteration 2:** $\textbf{P:}$ $\text{age}^+ = \text{age, hairColor}$

Decompose: $\textbf{People(SSID, name, age)}$

- $\text{Hair(age, hairColor)}$
- $\text{Phone(SSID, phoneNumber)}$
Example: BCNF

Recall: find $X$ s.t. $X \neq X^+ \neq \text{[all-attrs]}$
Example: BCNF

R(A,B,C,D)

Recall: find X s.t.
X ≠ X⁺ ≠ [all-attrs]

R(A,B,C,D)
A⁺ = ABC ≠ ABCD
R(A,B,C,D)

Example: BCNF

Recall: find X s.t. X \neq X^+ \neq [all-attrs]

R(A,B,C,D)
A^+ = ABC \neq ABCD

R_1(A,B,C)

R_2(A,D)

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

Example: BCNF

Recall: find $X$ s.t. $X \neq X^+ \neq \text{[all-attrs]}$

$R(A,B,C,D)$

$A^+ = ABC \neq ABCD$

$R_1(A,B,C)$

$B^+ = BC \neq ABC$

$R_2(A,D)$

$A \rightarrow B$

$B \rightarrow C$
Example: BCNF

R(A,B,C,D)

A $\rightarrow$ B
B $\rightarrow$ C

Recall: find X s.t.
X $\neq$ X$^+$ $\neq$ [all-attrs]

R(A,B,C,D)
A$^+$ = ABC $\neq$ ABCD

R$_1$(A,B,C)
B$^+$ = BC $\neq$ ABC

R$_{11}$(B,C)

R$_{12}$(A,B)

R$_2$(A,D)

What are the keys?

What happens if in R we first pick B$^+$? Or AB$^+$?
Decompositions in General

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

\[ S_1(A_1, \ldots, A_n, B_1, \ldots, B_m) \quad S_2(A_1, \ldots, A_n, C_1, \ldots, C_p) \]

\[ S_1 = \text{projection of } R \text{ on } A_1, \ldots, A_n, B_1, \ldots, B_m \]
\[ S_2 = \text{projection of } R \text{ on } A_1, \ldots, A_n, C_1, \ldots, C_p \]

Not all decompositions are good - some are lossy
A Lossless Decomposition

<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>
</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>
A Lossy Decomposition

What is lossy here?

<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>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>

<table>
<thead>
<tr>
<th>Price</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.99</td>
<td>Gadget</td>
</tr>
<tr>
<td>24.99</td>
<td>Camera</td>
</tr>
<tr>
<td>19.99</td>
<td>Camera</td>
</tr>
</tbody>
</table>
Lossy Decomposition

<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>

CSE 344 - Summer 2017
Decomposition in General

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

\[ S_1(A_1, ..., A_n, B_1, ..., B_m) \]

\[ S_2(A_1, ..., A_n, C_1, ..., C_p) \]

Let:

\[ S_1 = \text{projection of } R \text{ on } A_1, ..., A_n, B_1, ..., B_m \]

\[ S_2 = \text{projection of } R \text{ on } A_1, ..., A_n, C_1, ..., C_p \]

The decomposition is called \textit{lossless} if \[ R = S_1 \cup S_2 \]

Fact: If \[ A_1, ..., A_n \rightarrow B_1, ..., B_m \] then the decomposition is lossless

It follows that every BCNF decomposition is lossless
Schema Refinements = Normal Forms

- 1st Normal Form = all tables are flat
- 2nd Normal Form = obsolete
- Boyce Codd Normal Form = no bad FDs
- 3rd and 4th Normal Form = see book
  - BCNF is lossless but can cause loss of ability to check some FDs (see book 3.4.4)
  - 3NF fixes that (is lossless and dependency-preserving), but some tables might not be in BCNF – i.e., they may have redundancy anomalies
Normalization IRL

- It’s good to know about BCNF (and maybe 3NF).
- Should every table always be in BCNF?

R(Item, OrderNo, Customer, Address, Zip Code, State)

What are the FDs:
Normalization IRL

R( Item, OrderNo, Customer, Address, Zip Code, State)

What are the FDs:

- OrderNo -> Customer
- Customer -> Address, Zip Code, State
- Zip Code -> State

What is BCNF:
Normalization IRL

R( Item, OrderNo, Customer, Address, Zip Code, State)

What is BCNF:

R(Item, OrderNo, Customer, Address, Zip Code, State)  
OrderNo⁺ = Customer, Address, Zip, Code, State

R₁(Item, OrderNo)

R₂(OrderNo, Customer, Address, Zip Code, State)

R₃(OrderNo, CustomerId)

R₄(CustomerId, Name, Address, Zip Code, State)

Are We Done?
Normalization IRL

R( Item, OrderNo, Customer, Address, Zip Code, State)

What is BCNF:

$R(\text{Item, OrderNo, Customer, Address, Zip Code, State})$

$\text{OrderNo}^+ = \text{Customer, Address, Zip, Code, State}$

$R_1(\text{Item, OrderNo})$

$R_2(\text{OrderNo, Customer, Address, Zip Code, State})$

$R_3(\text{OrderNo, CustomerId})$

$R_4(\text{CustomerId, Name, Address, Zip Code})$

$R_5(\text{Zip Code, State})$

Is this a good idea?