CSC 261/461 – Database Systems
Lecture 11

Fall 2017
• Read the textbook!
  – Chapter 8:
    • Will cover later; But self-study the chapter
    • Everything except Section 8.4
  – Chapter 14:
    • Section 14.1 – 14.5
  – Chapter 15:
    • Section 15.1 – 15.4
Superkeys and Keys
A **superkey** is a set of attributes $A_1, ..., A_n$ s.t. for any other attribute $B$ in $R$, we have $\{A_1, ..., A_n\} \rightarrow B$

A **key** is a *minimal* superkey

I.e. all attributes are *functionally determined* by a superkey

Meaning that no subset of a key is also a superkey
Finding Keys and Superkeys

• For each set of attributes $X$

  1. Compute $X^+$

  2. If $X^+ = \text{set of all attributes}$ then $X$ is a superkey

  3. If $X$ is minimal, then it is a key

Do we need to check all sets of attributes? Which sets?
Example of Finding Keys

Product(name, price, category, color)

{name, category} \rightarrow price
{category} \rightarrow color

What is a key?
Example of Keys

Product(name, price, category, color)

{name, category} → price
{category} → color

{name, category}⁺ = {name, price, category, color}
= the set of all attributes
⇒ this is a superkey
⇒ this is a key, since neither name nor category alone is a superkey
Today’s Lecture

1. 2NF, 3NF and Boyce-Codd Normal Form

2. Decompositions
Prime and Non-prime attributes

- A **Prime attribute** must be a member of *some* candidate key.
- A **Nonprime attribute** is not a prime attribute—that is, it is not a member of any candidate key.
Now that we know how to find FDs, it’s a straight-forward process:

1. Search for “bad” FDs

2. If there are any, then keep decomposing the table into sub-tables until no more bad FDs

3. When done, the database schema is normalized
Main idea is that we define “good” and “bad” FDs as follows:

- $X \rightarrow A$ is a “good FD” if $X$ is a (super)key
  - In other words, if $A$ is the set of all attributes

- $X \rightarrow A$ is a “bad FD” otherwise

We will try to eliminate the “bad” FDs!
  - Via normalization
Second Normal Form

• Uses the concepts of FDs, primary key
• Definitions
  – **Full functional dependency:**
    • a FD $Y \rightarrow Z$ where removal of any attribute from $Y$ means the FD does not hold any more
Second Normal Form (cont.)

- Examples:
  - \( \{\text{Ssn, Pnumber} \} \rightarrow \text{Hours} \) is a full FD since neither
    - \( \text{Ssn} \rightarrow \text{Hours} \) nor \( \text{Pnumber} \rightarrow \text{Hours} \) hold
  - \( \{\text{Ssn, Pnumber} \} \rightarrow \text{Ename} \) is not a full FD (it is called a partial dependency) since \( \text{Ssn} \rightarrow \text{Ename} \) also holds
Second Normal Form (2)

• A relation schema R is in **second normal form (2NF)** if every non-prime attribute A in R is fully functionally dependent on the primary key

• R can be decomposed into 2NF relations via the process of 2NF normalization or “second normalization”
Third Normal Form (1)

• Definition:
  – **Transitive functional dependency**:  
    • a FD $X \rightarrow Z$ that can be derived from two FDs $X \rightarrow Y$ and $Y \rightarrow Z$

• Examples:
  – $Ssn \rightarrow Dmgr\_ssn$ is a transitive FD  
    • Since $Ssn \rightarrow Dnumber$ and $Dnumber \rightarrow Dmgr\_ssn$ hold
  – $Ssn \rightarrow Ename$ is non-transitive  
    • Since there is no set of attributes $X$ where $Ssn \rightarrow X$ and $X \rightarrow Ename$
Third Normal Form (2)

- A relation schema R is in **third normal form (3NF)** if it is in 2NF and no non-prime attribute A in R is transitive dependent on the primary key.
- R can be decomposed into 3NF relations via the process of 3NF normalization.
Normalizing into 2NF and 3NF

(a) EMP_PROJ

2NF Normalization

(b) EMP_DEPT

3NF Normalization
Figure 14.12 Normalization into 2NF and 3NF. (a) The LOTS relation with its functional dependencies FD1 through FD4. (b) Decomposing into the 2NF relations LOTS1 and LOTS2. (c) Decomposing LOTS1 into the 3NF relations LOTS1A and LOTS1B. (d) Progressive normalization of LOTS into a 3NF design.
Normal Forms Defined Informally

• **1st normal form**
  – All attributes depend on **the key**

• **2nd normal form**
  – All attributes depend on **the whole key**

• **3rd normal form**
  – All attributes depend on **nothing but the key**
• A relation schema \( R \) is in **second normal form (2NF)** if every non-prime attribute \( A \) in \( R \) is fully functionally dependent on *every* key of \( R \).

• A relation schema \( R \) is in **third normal form (3NF)** if it is in 2NF *and* no non-prime attribute \( A \) in \( R \) is transitively dependent on *any* key of \( R \).
1. BOYCE-CODD NORMAL FORM
What you will learn about in this section

1. Conceptual Design

2. Boyce-Codd Normal Form

3. The BCNF Decomposition Algorithm
5. BCNF (Boyce-Codd Normal Form)

• A relation schema R is in Boyce-Codd Normal Form (BCNF) if whenever an FD $X \rightarrow A$ holds in R, then $X$ is a superkey of R.

• Each normal form is strictly stronger than the previous one
  – Every 2NF relation is in 1NF
  – Every 3NF relation is in 2NF
  – Every BCNF relation is in 3NF
Figure 14.13 Boyce-Codd normal form

(a) LOTS1A

<table>
<thead>
<tr>
<th>Property_id#</th>
<th>County_name</th>
<th>Lot#</th>
<th>Area</th>
</tr>
</thead>
</table>

FD1

FD2

FD5

BCNF Normalization

LOTS1AX

<table>
<thead>
<tr>
<th>Property_id#</th>
<th>Area</th>
<th>Lot#</th>
</tr>
</thead>
</table>

LOTS1AY

<table>
<thead>
<tr>
<th>Area</th>
<th>County_name</th>
</tr>
</thead>
</table>

(b) \( R \)

\[
\begin{array}{ccc}
A & B & C \\
\hline
FD1 & & \\
FD2 & & \\
\end{array}
\]

Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. \( C \to B \).
• A relation schema $R$ is in **second normal form (2NF)** if every non-prime attribute $A$ in $R$ is fully functionally dependent on *every* key of $R$

• A relation schema $R$ is in **third normal form (3NF)** if it is in 2NF *and* no non-prime attribute $A$ in $R$ is transitively dependent on *any* key of $R$
4.3 Interpreting the General Definition of Third Normal Form (2)

- ALTERNATIVE DEFINITION of 3NF: We can restate the definition as:

  A relation schema $R$ is in third normal form (3NF) if, whenever a nontrivial FD $X \rightarrow A$ holds in $R$, either
  a) $X$ is a superkey of $R$ or
  b) $A$ is a prime attribute of $R$

  The condition (b) takes care of the dependencies that “slip through” (are allowable to) 3NF but are “caught by” BCNF which we discuss next.
5. BCNF (Boyce-Codd Normal Form)

- Definition of 3NF:
  - A relation schema R is in 3NF if, whenever a nontrivial FD \( X \rightarrow A \) holds in R, either
    - a) \( X \) is a superkey of R or
    - b) \( A \) is a prime attribute of R

- A relation schema R is in Boyce-Codd Normal Form (BCNF) if whenever an FD \( X \rightarrow A \) holds in R, then
  - a) \( X \) is a superkey of R
  - b) There is no

- Each normal form is strictly stronger than the previous one
  - Every 2NF relation is in 1NF
  - Every 3NF relation is in 2NF
  - Every BCNF relation is in 3NF
Boyce-Codd normal form

(a) LOTS1A

<table>
<thead>
<tr>
<th>Property_id#</th>
<th>County_name</th>
<th>Lot#</th>
<th>Area</th>
</tr>
</thead>
</table>

FD1

FD2

FD5

BCNF Normalization

LOTS1AX

<table>
<thead>
<tr>
<th>Property_id#</th>
<th>Area</th>
<th>Lot#</th>
</tr>
</thead>
</table>

LOTS1AY

<table>
<thead>
<tr>
<th>Area</th>
<th>County_name</th>
</tr>
</thead>
</table>

(b) $R$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
</table>

FD1

FD2

Figure 14.13 Boyce-Codd normal form. (a) BCNF normalization of LOTS1A with the functional dependency FD2 being lost in the decomposition. (b) A schematic relation with FDs; it is in 3NF, but not in BCNF due to the f.d. $C \rightarrow B$. 
A relation TEACH that is in 3NF but not in BCNF

- Two FDs exist in the relation TEACH:
  - \{\text{student, course}\} \rightarrow \text{instructor}
  - \text{instructor} \rightarrow \text{course}

- \{\text{student, course}\} is a candidate key for this relation
- So this relation is in 3NF but not in BCNF
- A relation NOT in BCNF should be decomposed
  - while possibly forgoing the preservation of all functional dependencies in the decomposed relations.
Achieving the BCNF by Decomposition

- Three possible decompositions for relation TEACH
  - D1: \{student, instructor\} and \{student, course\}
  - D2: \{course, instructor\} and \{course, student\}
  - D3: \{instructor, course\} and \{instructor, student\}
Boyce-Codd Normal Form

BCNF is a simple condition for removing anomalies from relations:

A relation $R$ is in BCNF if:

if $\{X_1, \ldots, X_n\} \rightarrow A$ is a non-trivial FD in $R$
then $\{X_1, \ldots, X_n\}$ is a superkey for $R$

In other words: there are no “bad” FDs
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>

{SSN} → {Name, City}

This FD is \textit{bad} because it is \textit{not} a superkey

⇒ \textbf{Not} in BCNF

\textit{What is the key?} {SSN, PhoneNumber}
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>Madison</td>
</tr>
</tbody>
</table>

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

This FD is now good because it is the key.

Let’s check anomalies:
- Redundancy?
- Update?
- Delete?

Now in BCNF!
BCNF Decomposition Algorithm

BCNFDcomp(R):

Find \( X \) s.t. \( X + \neq X \) and \( X + \neq \{\text{all attributes}\} \)

if (not found) then Return \( R \)

let \( Y = X + - X, Z = (X +) C \)

decompose \( R \) into \( R_1(X + Y) \) and \( R_2(X + Z) \)

Return BCNFDcomp(\( R_1 \)), BCNFDcomp(\( R_2 \))
BCNF Decomposition Algorithm

BCNFDecomp(R):

Find a set of attributes X s.t.: X⁺ ≠ X and X⁺ ≠ [all attributes]

Find a set of attributes X which has non-trivial “bad” FDs, i.e. is not a superkey, using closures
BCNF Decomposition Algorithm

BCNFDcomp(R):
   Find a set of attributes X s.t.: X⁺ ≠ X and X⁺ ≠ [all attributes]

   if (not found) then Return R

   let Y = X⁺ - X, Z = (X⁺)C

   decompose R into R₁(X È Y) and R₂(X È Z)

   Return BCNFDcomp(R₁), BCNFDcomp(R₂)

If no “bad” FDs found, in BCNF!
BCNF Decomposition Algorithm

BCNFD decomp(R):

Find a set of attributes X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

if (not found) then Return R

let $Y = X^+ - X$, $Z = (X^+)^C$

Let Y be the attributes that $X$ functionally determines (+ that are not in X)
And let Z be the other attributes that it doesn’t
BCNFDecomp(R):
Find a set of attributes $X$ s.t.: $X^+ \neq X$ and $X^+ \neq [\text{all attributes}]$

**if** (not found) **then** Return $R$

**let** $Y = X^+ - X$, $Z = (X^+)^C$

**decompose** $R$ into $R_1(X \cup Y)$ and $R_2(X \cup Z)$

Split into one relation (table) with $X$ plus the attributes that $X$ determines ($Y$)…
BCNF Decomposition Algorithm

**BCNFDecomp**(R):

Find a set of attributes $X$ s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

**if** (not found) **then** Return R

**let** $Y = X^+ - X$, $Z = (X^+)^C$

**decompose** R into $R_1(X \cup Y)$ and $R_2(X \cup Z)$

And one relation with X plus the attributes it *does not* determine (Z)
BCNF Decomposition Algorithm

BCNFDecomp(R):
  Find a set of attributes X s.t.: $X^+ \neq X$ and $X^+ \neq$ [all attributes]

  if (not found) then Return R

  let $Y = X^+ - X$, $Z = (X^+)^C$
  decompose R into $R_1(X \cup Y)$ and $R_2(X \cup Z)$

  Return BCNFDecomp($R_1$), BCNFDecomp($R_2$)

Proceed recursively until no more “bad” FDs!
BCNFDecomp(R):
If $X \rightarrow A$ causes BCNF violation:

Decompose $R$ into

$R1 =XA$
$R2 = R -A$

(Note: $X$ is present in both $R1$ and $R2$)
BCNFDecomp(R):
Find a set of attributes $X$ s.t.: $X^+ \neq X$ and $X^+ \neq [all attributes]$

if (not found) then Return $R$

let $Y = X^+ - X$, $Z = (X^+)^C$
decompose $R$ into $R_1(X \cup Y)$ and $R_2(X \cup Z)$

Return BCNFDecomp($R_1$), BCNFDecomp($R_2$)

BCNFDecomp(R):
If $X \rightarrow A$ causes BCNF violation:

Decompose $R$ into

$R_1 = XA$
$R_2 = R - A$

(Note: $X$ is present in both $R_1$ and $R_2$)
Example

\[ R(A,B,C,D,E) \]
\[
\{A\}^+ = \{A,B,C,D\} \neq \{A,B,C,D,E\}
\]

\[ R_1(A,B,C,D) \]
\[
\{C\}^+ = \{C,D\} \neq \{A,B,C,D\}
\]

\[ R_{11}(C,D) \]
\[ R_{12}(A,B,C) \]

\[ R_2(A,E) \]

\[ \{A\} \rightarrow \{B,C\} \]
\[ \{C\} \rightarrow \{D\} \]
Acknowledgement

• Some of the slides in this presentation are taken from the slides provided by the authors.
• Many of these slides are taken from cs145 course offered by Stanford University.
• Thanks to YouTube, especially to Dr. Daniel Soper for his useful videos.