Functional Dependencies continued

Outline

- Relational Algebra (6.1)
- E/R Model (7.2 - 7.4)
- E/R Diagrams (7.5)
- Reduction to Schema (7.6)
- Relational Database Design (7.7)
- Functional Dependencies (8.1 – 8.4)
- Normalization (8.5 – 8.7)
2. Closure of an attribute set

- Given a set of attributes \( \alpha \) and a set of FDs \( F \), closure of \( \alpha \) under \( F \) is the set of all attributes implied by \( \alpha \).
- In other words, the largest \( \beta \) such that: \( \alpha \rightarrow \beta \)
- Redefining super keys:
  - *The closure of a super key is the entire relation schema*
- Redefining candidate keys:
  - It is a super key
  - No subset of it is a super key

Computing the closure for \( \alpha \)

- Simple algorithm:
  1. Start with \( \beta = \alpha \).
  2. Go over all functional dependencies, \( \delta \rightarrow \gamma \), in \( F^+ \)
  3. If \( \delta \subseteq \beta \), then
     - Add \( \gamma \) to \( \beta \)
  4. Repeat till \( \beta \) stops changing
Example

- $F = \{ A \rightarrow B$
  - $A \rightarrow C$
  - $CG \rightarrow H$
  - $CG \rightarrow I$
  - $B \rightarrow H\}$

- $(AG)^+$?
  - 1. $\beta = AG$
  - 2. $\beta = ABG$ (A $\rightarrow$ B and A $\subseteq$ AG)
  - 3. $\beta = ABCG$ (A $\rightarrow$ C and A $\subseteq$ ABG)
  - 4. $\beta = ABCGH$ (CG $\rightarrow$ H and CG $\subseteq$ ABCG)
  - 5. $\beta = ABCGHI$ (CG $\rightarrow$ I and CG $\subseteq$ ABCGH)
  - done

- Is $(AG)$ a candidate key?
  - 1. It is a super key.
  - 2. $(A^+) = ABCH$, $(G^+) = G$.
  - YES.

Uses of attribute set closures

- Determining superkeys and candidate keys

- Determining if $\alpha \rightarrow \beta$ is a valid FD
  - Does $\alpha^+$ contain $\beta$?

- Can be used to compute $F^+$
3. Extraneous Attributes

Consider $F$, and a functional dependency, $\alpha \rightarrow \beta$.

“Extraneous”: Any attributes in $\alpha$ or $\beta$ that can be safely removed?

Without changing the constraints implied by $F$

- $\sigma$ is extraneous in $\alpha$ if:
  1. $\sigma$ is in $\alpha$, and
     - $F$ logically implies $F'$ (show that $F$ implies $(\alpha - \sigma) \rightarrow \beta$)
     - where $F' = (F - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - \sigma) \rightarrow \beta\}$, or
  2. show $(\alpha - \sigma)^+$ includes $\beta$ under $F$

- $\sigma$ is extraneous in $\beta$ if:
  1. $\sigma$ is in $\beta$, and
     - $F'$ logically implies $F$, and
     - $F' = (F - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - \sigma)\}$
  2. show $\alpha^+$ includes $\sigma$ under $F'$

Example: Given $F = \{A \rightarrow C, AB \rightarrow CD\}$, show $C$ extra in $AB \rightarrow CD$

- $F' = \{A \rightarrow C, AB \rightarrow D\}$

- Using Armstrong’s:
  (show $F' \rightarrow F$)
  - We know:
    - $AB \rightarrow D$ (F')
    - $ABC \rightarrow CD$ (aug)
  - also:
    - $A \rightarrow C$ (F')
    - $AB \rightarrow BC$ (aug w/ B)
    - $AB \rightarrow ABC$ (aug w/ A)
  - then:
    - $AB \rightarrow ABC \rightarrow CD$ (trans)

done.

$\sigma$ is extraneous in $\alpha$ iff:
$F \rightarrow F'$, or
$(\alpha - \sigma)^+$ includes $\beta$ under $F$

$\sigma$ is extraneous in $\beta$ iff:
$F' \rightarrow F$, or
$\alpha^+$ includes $\sigma$ in $F'$
### 3. Extraneous Attributes

- **Example:** Given \( F = \{ A \rightarrow C, AB \rightarrow CD \}, \) *show C extra in AB \( \rightarrow \) CD*
  - \( F' = \{ A \rightarrow C, AB \rightarrow D \} \)
  - Using Armstrong’s:
    - (show \( F' \rightarrow F \))
      - We know:
        - \( AB \rightarrow D \) \( (F') \)
        - \( ABC \rightarrow CD \) \( \text{(aug)} \)
        - also:
          - \( A \rightarrow C \) \( (F') \)
          - \( AB \rightarrow BC \) \( \text{(aug w/ B)} \)
          - \( AB \rightarrow ABC \) \( \text{(aug w/ A)} \)
        - then:
          - \( AB \rightarrow ABC \rightarrow CD \) \( \text{(trans)} \)
      - done.
  - Attribute closures (show \( \alpha + \) includes \( C \) under \( F' \)):
    - \( (AB)^+ = AB \)
    - \( = ABC \quad \text{(A \rightarrow C)} \)
      - done.

---

### 3. Extraneous Attributes

- **Example:** Given \( F = \{ A \rightarrow BE, B \rightarrow C, C \rightarrow D, AC \rightarrow DE \}, \) *remove extraneous attributes*
  - For left side of \( AC \rightarrow DE \)
    - A extraneous?
      - NO: \( C^+ = CD, \) NOT include DE
    - C extraneous?
      - YES: \( A^+ = ABCDE, \) includes DE
  - Now \( F = A \rightarrow BE, B \rightarrow C, C \rightarrow D, A \rightarrow DE \)
  - For right side,
    - B extraneous in \( A \rightarrow BE \)?
      - \( F' = A \rightarrow E, B \rightarrow C, C \rightarrow D, A \rightarrow DE \)
      - NO: \( A' = ADE, \) not include B.
    - E extraneous in \( A \rightarrow BE \)?
      - \( F' = A \rightarrow B, B \rightarrow C, C \rightarrow D, A \rightarrow DE \)
      - YES: \( A^+ = ABCDE, \) includes E.
  - Now \( F = A \rightarrow B, B \rightarrow C, C \rightarrow D, A \rightarrow DE \)
    - D extraneous in right side of \( A \rightarrow DE ? \)
      - \( F' = A \rightarrow B, B \rightarrow C, C \rightarrow D, A \rightarrow E \)
      - YES: \( A^+ = ABCDE, \) so does include D
  - Now \( F = A \rightarrow B, B \rightarrow C, C \rightarrow D, A \rightarrow E \)

\[ \sigma \text{ is extraneous in } \alpha \text{ iff:} \]
\[ F \rightarrow F', \text{ or} \]
\[ (\alpha - \sigma)^* \text{ includes } \beta \text{ under } F \]

\[ \sigma \text{ is extraneous in } \beta \text{ iff:} \]
\[ F' \rightarrow F, \text{ or} \]
\[ \alpha^+ \text{ includes } \sigma \text{ in } F' \]
4. Canonical Cover

- A canonical cover for $F$ is a set of dependencies $F_c$ such that
  - $F$ logically implies all dependencies in $F_c$, and
  - $F_c$ logically implies all dependencies in $F$, and
  - No functional dependency in $F_c$ contains an extraneous attribute, and
  - Each left side of functional dependency in $F_c$ is unique

- In some (vague) sense, it is a minimal version of $F$

- Create as follows:
  - **repeat**
    1. use union rule to merge right sides
    2. eliminate extraneous attributes
  - until $F_c$ does not change

\[ A \rightarrow B, A \rightarrow C, C \rightarrow D, AC \rightarrow BD \]

\[
\text{Cover:}
\]

\[
\begin{align*}
\text{A} & \rightarrow \text{B}, \text{A} \rightarrow \text{C}, \text{C} \rightarrow \text{D}, \text{AC} \rightarrow \text{BD} \\
\text{A} & \rightarrow \text{BC}, \text{C} \rightarrow \text{D}, \text{AC} \rightarrow \text{BD} \quad \text{(union)} \\
\text{A} & \rightarrow \text{BC}, \text{C} \rightarrow \text{D}, \text{A} \rightarrow \text{BD} \\
\text{A} & \rightarrow \text{BCD}, \text{C} \rightarrow \text{D} \quad \text{(union)} \\
\text{A} & \rightarrow \text{B} \text{C} \text{D, C} \rightarrow \text{D} \\
\text{A} & \rightarrow \text{B} \text{C} \text{D, C} \rightarrow \text{D} \quad \text{(union)} \\
\end{align*}
\]

- $\sigma$ is extraneous in $\alpha$ iff:
  - $F \rightarrow F'$, or
  - $(\alpha - \sigma)^* \text{ includes } \beta \text{ under } F$

- $\sigma$ is extraneous in $\beta$ iff:
  - $F' \rightarrow F$, or
  - $\alpha^* \text{ includes } \sigma \text{ in } F'$
Mechanisms and definitions to work with FDs
- Closures, candidate keys, canonical covers etc...
- Armstrong axioms

Decompositions
- Loss-less decompositions, Dependency-preserving decompositions

BCNF
- How to achieve a BCNF schema
- BCNF may not preserve dependencies
- 3NF: Solves the above problem
- BCNF allows for redundancy
- 4NF: Solves the above problem

**Loss-less Decompositions**

Definition: A decomposition of $R$ into $(R_1, R_2)$ is called *lossless* if, for all legal instance of $r(R)$:

$\quad r = \Pi_{R_1} (r) \bowtie \Pi_{R_2} (r)$

or

$\quad \text{(select * from (select R1 from r) natural join (select R2 from r))}$

In other words, projecting on $R_1$ and $R_2$, and joining back, results in the relation you started with

Rule: A decomposition of $R$ into $(R_1, R_2)$ is lossless, iff:

$\quad R_1 \cap R_2 \rightarrow R_1$ or $\quad R_1 \cap R_2 \rightarrow R_2$

in $F^*$.

$(R_1 \cap R_2)$ must be key for $R_1$ or $R_2$
Dependency-preserving Decompositions

- Is it easy to check if dependencies in $F$ hold?
  - Yes if dependencies can be checked in the same table.

- Consider $R = (A, B, C)$, and $F = \{A \rightarrow B, B \rightarrow C\}$

- 1. Decompose into $R_1 = (A, B)$, and $R_2 = (A, C)$
  - Lossless?
    - Yes: $AB \cap AC = A$, which is a key for $R_1$
    - But harder to check for $B \rightarrow C$ as the data is in multiple tables.

- 2. On the other hand, $R_1 = (A, B)$, and $R_2 = (B, C)$,
  - is both lossless and dependency-preserving

---

Definition:

- Consider decomposition of $R$ into $R_1$, ..., $R_n$.
- Let $F_i$ be dependencies using just attributes in $R_i$.

- The decomposition is dependency preserving, if
  $$(F_1 \cup F_2 \cup ... \cup F_n)^+ = F^+$$
Outline

- Mechanisms and definitions to work with FDs
  - Closures, candidate keys, canonical covers etc...
  - Armstrong axioms
- Decompositions
  - Loss-less decompositions, Dependency-preserving decompositions
- BCNF
  - How to achieve a BCNF schema
- BCNF may not preserve dependencies
  - 3NF: Solves the above problem
- BCNF allows for redundancy
  - 4NF: Solves the above problem

Normalization
**BCNF**

- Recall that $R$ is in BCNF if every FD, $\alpha \rightarrow \beta$, is either:
  1. Trivial, or
  2. $\alpha$ is a superkey of $R$
- *No redundancy*

- What if the schema is not in BCNF?
  - Decompose (split) the schema into two pieces.
  - Careful: you want the decomposition to be lossless

---

**Achieving BCNF Schemas**

- For all dependencies $\alpha \rightarrow \beta$ in $F^+$, check if $\alpha$ is a superkey
  - (attribute closure)

- If not, then
  - Choose a dependency in $F^+$ that breaks the BCNF rules, say $\alpha \rightarrow \beta$
  - Create $R_1 = \alpha\beta$
  - Create $R_2 = R - (\beta - \alpha)$.
  - Note that: $R_1 \cap R_2 = \alpha$ and $\alpha \rightarrow \alpha\beta$, so:
    - $\alpha$ is a superkey of $R_1$
    - lossless decomposition

- Repeat for $R_1$, and $R_2$
  - Define $F_i$ to be all dependencies in $F^+$ that contain only attributes in $R_i$

---

*Note:*

$(R - (\beta - \alpha)) = (R - \beta)$

if no extraneous attributions in FDs

*We use $(R - \beta)$ in this course.*
Achiving BCNF Schemas

Example 1

\[ R = (A, B, C) \]
\[ F = \{ A \rightarrow B, B \rightarrow C \} \]
Candidate keys = \{A\}

BCNF? No. B \rightarrow C violates.

\[ B \rightarrow C \]

R1 = (B, C)
F1 = \{ B \rightarrow C \}
Candidate keys = \{B\}
BCNF = true

R2 = (A, B)
F2 = \{ A \rightarrow B \}
Candidate keys = \{A\}
BCNF = true

Dependency preservation ???
Yes

Example 2a

\[ R = (A, B, C, D, E) \]
\[ F = \{ A \rightarrow B, BC \rightarrow D \} \]
Candidate keys = \{ACE\}

BCNF = Violated by \{A \rightarrow B, BC \rightarrow D\}

\[ A \rightarrow B \]

R1 = (A, B)
F1 = \{ A \rightarrow B \}
Candidate keys = \{A\}
BCNF = true

R2 = (A, C, D, E)
F2 = \{ \}
Candidate keys = \{ACDE\}
BCNF = true

Dependency preservation ???
No: lost BC \rightarrow D
\[ R = (A, B, C, D, E) \]
\[ F = \{ A \rightarrow B, BC \rightarrow D \} \]
Candidate keys = \{ACE\}
BCNF = Violated by \{A \rightarrow B, BC \rightarrow D\} etc…

\[ A \rightarrow B \]

\[ R1 = (A, B) \]
\[ F1 = \{ A \rightarrow B \} \]
Candidate keys = \{A\}
BCNF = true

\[ R2 = (A, C, D, E) \]
\[ F2 = \{ \} \]
Candidate keys = \{ACDE\}
BCNF = true

Dependency preservation ???
No: we lost BC \rightarrow D
So this is not a dependency-preserving decomposition.