Tutorial 9:
Functional Dependencies
Summary
Outline

• Schema design
  – Introduction
  – Notation
• Functional dependencies
• Armstrong’s axioms
• Minimal cover
• Keys
Outline

• Decomposition
• Preserving information
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
## Schema Design

### Customer:

<table>
<thead>
<tr>
<th>Cust_Id</th>
<th>Faculty</th>
<th>Track</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>CS</td>
<td>Software</td>
</tr>
<tr>
<td>45678</td>
<td>EE</td>
<td>Hardware</td>
</tr>
<tr>
<td>11111</td>
<td>IE</td>
<td>IS</td>
</tr>
<tr>
<td>22222</td>
<td>IE</td>
<td>Accounting</td>
</tr>
</tbody>
</table>

### Ordered:

<table>
<thead>
<tr>
<th>Cust_Id</th>
<th>Book_Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>Database Systems</td>
</tr>
<tr>
<td>45678</td>
<td>Anatomy</td>
</tr>
<tr>
<td>12345</td>
<td>Database And Knowledge</td>
</tr>
<tr>
<td>11111</td>
<td>Anatomy</td>
</tr>
<tr>
<td>22222</td>
<td>Intro. to Economy</td>
</tr>
</tbody>
</table>

### CustOrders:

<table>
<thead>
<tr>
<th>Cust_Id</th>
<th>Faculty</th>
<th>Track</th>
<th>Book_Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td>CS</td>
<td>Software</td>
<td>Database Systems</td>
</tr>
<tr>
<td>45678</td>
<td>EE</td>
<td>Hardware</td>
<td>Anatomy</td>
</tr>
<tr>
<td>12345</td>
<td>CS</td>
<td>Software</td>
<td>Database And Knowledge</td>
</tr>
<tr>
<td>11111</td>
<td>IE</td>
<td>IS</td>
<td>Anatomy</td>
</tr>
<tr>
<td>22222</td>
<td>IE</td>
<td>Accounting</td>
<td>Intro. To Economy</td>
</tr>
</tbody>
</table>

Which design is better?
• Disadvantages of one table:
  – Redundant storage
    • Harder to update
    • Consistency issues
  – How should one represent a customer that has not ordered any book?
Outline

• Schema design
  – Introduction
  – Notation
• Functional dependencies
• Armstrong’s axioms
• Minimal cover
• Keys
Functional Dependencies - definitions

- Let
  \( R = \{A_1, \ldots, A_n\} \) be a relational schema, let
  \( r \) be a relation over \( R \) and let
  \( X, Y \subseteq R \) be sets of attributes

- \( r \) is said to satisfy the functional dependency \( X \rightarrow Y \) if
  - every two tuples that have the same value in \( X \) have the same values in \( Y \)
  - We denote this by \( r \models X \rightarrow Y \)
Functional Dependencies – definitions

• Let
  – R be a relational schema and let
  – r be a relation over R and let
  – F be a set of FD’s over R

• r satisfies F (r \models F) if
  – for every f in F, r satisfies f (r \models f)

• f is entailed from F (F \models f) if
  – for every relation r over R it holds that if r \models F then r \models f
Outline

• Schema design
  – Introduction
  – Notation
• Functional dependencies
• Armstrong’s axioms
• Minimal cover
• Keys
Armstrong’s Axioms

• Three axioms for proving existence of dependencies. Given $X,Y,Z \subseteq R$:
  – Reflexivity: if $X \subseteq Y$ then $Y \rightarrow X$
  – Inclusion: if $X \rightarrow Y$ then $XZ \rightarrow YZ$
  – Transitivity: if $X \rightarrow Y$ and $Y \rightarrow Z$ then $X \rightarrow Z$

• Other modus ponens:
  – Union: if $X \rightarrow Y$ and $X \rightarrow Z$ then $X \rightarrow YZ$
  – Decomposition: if $X \rightarrow Y$ and $Z \subseteq Y$ then $X \rightarrow Z$
  – Semi-transitivity: if $X \rightarrow Y$ and $WY \rightarrow Z$ then $WX \rightarrow Z$
Armstrong’s Axioms - example

• \( F = \{ \text{Cust}_\text{Id} \rightarrow \text{Track}, \ \text{Track} \rightarrow \text{Faculty} \} \)
• We show that
  – \( F \vdash \text{Cust}_\text{Id} \rightarrow \{ \text{Track, Faculty} \} \)

1. \( \text{Track} \rightarrow \text{Faculty} \) \( \in F \)
2. \( \text{Track} \rightarrow \{ \text{Track, Faculty} \} \) \( \text{Inclusion}, 1 \)
3. \( \text{Cust}_\text{Id} \rightarrow \text{Track} \) \( \in F \)
4. \( \text{Cust}_\text{Id} \rightarrow \{ \text{Track, Faculty} \} \) \( 2,3,\text{Trans.} \)
Outline

• Schema design
  – Introduction
  – Notation
• Functional dependencies
• Armstrong’s axioms
• Minimal cover
• Keys
Closure of FD’s

• Let F be a set of FD’s
  – The closure of F (F⁺) is the set
    \[ \{X \rightarrow Y \mid F \models X \rightarrow Y\} \]

• Example: F={A \rightarrow B, B \rightarrow C}, the following FD’s are in F⁺:
  – A \rightarrow C, AB \rightarrow C, AC \rightarrow C, B \rightarrow B, A \rightarrow B, \emptyset \rightarrow \emptyset, C \rightarrow \emptyset
  – Note that F⁺ contains also other FD’s

• Note that the set F⁺ is exponential and therefore we will try to avoid computing it.
Closure of a property

- Let
  - $X$ be a set of properties and let
  - $F$ be a set of FD’s

- The closure of $X$ with respect to $F$ ($X^+_F$) is the set $\{A \mid F \models X \Rightarrow A\}$
  - Note that $A$ is a single attribute
Minimal Cover

• A set of FD’s might have “redundant” information, for instance the sets
  – $F = \{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$
  – $G = \{A \rightarrow B, B \rightarrow C\}$

are equivalent in the sense that $F^+ = G^+$
  – The dependency $A \rightarrow C$ is redundant

• Our goal: A unified form for FD’s
Minimal set of FD’s

- Let F be a set of FD’s, F is minimal if for every FD $X \rightarrow Y \in F$ the following hold:
  - $|Y| = 1$
  - $F^+ \neq (F \setminus \{X \rightarrow Y\})^+$
  - For all $Z \subset X$ the following hold:
    - $F^+ \neq ((F \setminus \{X \rightarrow Y\}) \cup \{Z \rightarrow Y\})^+$
    - i.e., F does not contain an FD $X \rightarrow A$ such that $X$ includes redundant attributes
• Schema design
  – Introduction
  – Notation
• Functional dependencies
• Armstrong’s axioms
• Minimal cover
• Keys
• Let
  – R be a relational schema and let
  – X ⊆ R be a subset of attributes and let
  – F be a set of FD’s

• X is a superkey of R if and only if F ⊨ X → R, or equivalently:
  – X is a superkey of R iff X → R ∈ F^+
  – X is a superkey of R iff X^+_F = R
• Let $R = \{A, B, C, D\}$ and $F = \{A \rightarrow C, B \rightarrow D\}$
  – ABC is a superkey of $R$
    • However, it is not unique
    • And not minimal

• $X$ is a key of $R$ if
  – It is a superkey of $R$
  – There does not exist $Y \subset X$ such that $Y$ is also a superkey
Outline

• Decomposition
• Preserving information
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
A decomposition of R is a set \( \{R_1, \ldots, R_n\} \) such that \( \bigcup_{i=1}^{n} R_i = R \).

Motivation:
- Allows a better modeling of the database

Characteristics of a good modeling:
- Preserves information (necessary)
- Preserves dependencies (not necessary, desired)
Outline

• Decomposition
• **Preserving information**
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
Preserving Information

- $R$ - a relational schema
- $F$ – a set of FD’s
- $P = \{R_1, \ldots, R_n\}$ decomposition

- $P$ preserves information with respect to $F$ if for every relation $r$ over $R$ such that $r \not\models F$ the following holds:

\[ \bowtie_{i=1..n} \pi_{R_i}(r) = r \]
Algorithm for information preserving

- R - a relational schema and P={R₁, ..., Rₙ} a decomposition
- Does P preserves information?

- We show the algorithm’s run on the following example R(A, B, C, D, E, H)
- P={R₁(A, B), R₂(A, C, D), R₃(B, E, H)}
Algorithm for information preserving

• 1st step - Initialization:
  – Create a relation r over R such that:
    • Every schema R_i has its own tuple t_i
    • For every attribute A:
      – If A∈R_i then t[A]=a
      – Else t[A]=a_i
      – Similarly with b for B, c for C, etc.
Algorithm for information preserving

• In our example:

\[
\begin{array}{cccccccc}
\text{t}_1 & A & B & C & D & E & H \\
a & b & c_1 & d_1 & e_1 & h_1 \\
\text{t}_2 & a & b_2 & c & d & e_2 & h_2 \\
a_3 & b & c_3 & d_3 & e & h \\
\end{array}
\]

R(A, B, C, D, E, H)

F = \{A \rightarrow B, C \rightarrow D, B \rightarrow EH\}

\rho = \{R_1(A, B), R_2(A, C, D), R_3(B, E, H)\}
Algorithm for information preserving

• 2nd step - chase
  – While the table changes do:
    • Look for an FD violation and equate the conclusions
    • “Equate” = change every occurrence of one to the other
      – When equating $a_j$ with $a$, change $a_j$ with $a$.

• 3nd step:
  – Return true if and only if there is a row without indexes.
Algorithm for information preserving

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>t₁</strong></td>
<td>a</td>
<td>b</td>
<td>c₁</td>
<td>d₁</td>
<td>e₁</td>
<td>h₁</td>
</tr>
<tr>
<td><strong>t₂</strong></td>
<td>a</td>
<td>b₂</td>
<td>c</td>
<td>d</td>
<td>e₂</td>
<td>h₂</td>
</tr>
<tr>
<td><strong>t₃</strong></td>
<td>a₃</td>
<td>b</td>
<td>c₃</td>
<td>d₃</td>
<td>e</td>
<td>h</td>
</tr>
</tbody>
</table>

\[ F = \{ A \rightarrow B, C \rightarrow D, B \rightarrow EH \} \]

\[ A \rightarrow B \ (t₁,t₂) \]
Algorithm for information preserving

Note that we have a tuple without subscripts and thus the decomposition preserves information.
Outline

• Decomposition
• Preserving information
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
Projection of FD’s

- R - a relational schema
- F – a set of FD’s
- S ⊆ R
- The projection of F on S, $\pi_S F$ is the set:
  
  $\{ X \rightarrow Y \mid X \rightarrow Y \in F^+ \land X \cup Y \subseteq S \}$.  

  – Intuitively, this is the set of FD’s that are relevant to S
Preserving Dependencies

• Intuition:
  – Each FD from R appears (or can be derived from) $R_i$.

• Goal: simple updates
  – If a decomposition does not preserve dependencies then when have to check the other tables.
  – Therefore we can live without it (although it is desired).
Preserving Dependencies - example

- \( R(\text{Phone, AreaCode, City}) \)
- \( F=\{ \text{City} \rightarrow \text{AreaCode, (AreaCode, Phone)} \rightarrow \text{City} \} \)
- \( P=\{ R1(\text{Phone, City}), R2(\text{AreaCode, City}) \} \)

- This decomposition preserves information
- Does it preserve dependencies?
Preserving Dependencies

• A functional dependency \( f \) is preserved if
  – There exists a schema that includes all of the attributes in \( f \)
  – \( f \) can be deduced from other dependencies that are preserved in the decomposition
Outline

• Decomposition
• Preserving information
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
Normal Forms

• Normal Form is a characteristic of relational schema that captures the “quality” of the schema in the following sense:
  – a schema is better if it prevents duplications

• We will discuss the following Normal Forms (NF):
  – BCNF
  – 3NF
BCNF – Boyce-Codd NF

- A schema R with a set F of FD’s is in BCNF if every nontrivial FD implied by F has a superkey on its premise (lhs)
  - That is, every $X \rightarrow Y$ in $F^+$ is such that – $X$ is a superkey; or
  - $Y \subseteq X$
BCNF – example

• $R = \{\text{Id, Name, Address}\}$
• $F = \{\text{Id} \rightarrow \text{Name}, \text{Name} \rightarrow \text{Address}\}$

• $R$ is not BCNF w.r.t. $F$ since
  – The only key is Id
  – Name $\rightarrow$ Address does not satisfy the condition

• For $F' = \{\text{Id} \rightarrow \text{Name}, \text{Id} \rightarrow \text{Address}\}$
  – $R$ is BCNF w.r.t. $F'$
How to check whether a schema is BCNF?

By definition:

• Compute $F^+$

• For every FD $X \rightarrow Y \in F^+$ check whether $X$ is a superkey.

Problem: the size of $F^+$ is exponential in $R$

Theorem: If $R$ is not BCNF with respect to $F$ (there exists an FD $X \rightarrow Y \in F^+$ that violates the conditions), then there exists an FD $Z \rightarrow W \in F$ that violates the conditions.
• Often we need to choose between BCNF and Preserving Dependencies
  – How should we choose?
  • If we have lots of updates of attributes that have duplications in the original database (for instance AreaCode)
    – BCNF prevents duplications
  • If we want to add/update attributes that appear in an FD that is not preserved (for instance Phone)
    – We use 3NF
3NF

• R – a relational schema
• F – a set of FD’s over R

• R is in 3NF if for every FD $X \rightarrow A \in F^+$ such that $A \notin X$
  – X is a superkey of R or
  – A is contained in a key of R
Algorithm for finding 3NF decomposition

- Given a minimal cover of FD’s
  1. If there exists an FD in F that includes all of the attributes in R, return \{R\}.
  2. For every set of FD’s of the form \( X \rightarrow A_1 \), \( X \rightarrow A_2 \),..., \( X \rightarrow A_n \), create a schema \( \{X, A_1,\ldots, A_n\} \).
  3. If none of the schemas contains a superkey of R, add a schema which is a key of R.

- Note that this algorithm finds a decomposition that preserves information and dependencies.
Outline

• Decomposition
• Preserving information
  – Example
  – Algorithm for checking
• Preserving dependencies
  – Projecting FD’s
• Normal Forms
  – BCNF
  – 3NF
• Questions
**Question**

- Given a schema $R(A, B, C, D)$ and FDs:
  
  \[
  \begin{align*}
  AB &\rightarrow CD, \\
  D &\rightarrow B, \\
  CB &\rightarrow A, \\
  BC &\rightarrow D
  \end{align*}
  \]

  Decompose using 3NF algorithm

- Answer: $R(A, B, C, D)$

- Never forget (the 1st step) to check for FD with all attributes.
• Given a schema $R(A, B, C, D, E, H)$ and FDs:
  \{AB \rightarrow CD, D \rightarrow HB, C \rightarrow H, CH \rightarrow AE, BC \rightarrow D \} \}

1. Find 3 keys
2. Find Minimal Cover
3. Decompose using the 3NF Algorithm
• \( R(A, B, C, D, E, H) \)
• \( \{AB \rightarrow CD, D \rightarrow HB, C \rightarrow H, CH \rightarrow AE, BC \rightarrow D \} \)
• Find 3 keys:

\[ K = (A,B,C,D,E,H) \]

Let’s try \( K \setminus H \):

\[ (C \rightarrow H) \]
\[ K = (A,B,C,D,E) \]

\( K \setminus E? \)
\[ (C \rightarrow E) \]
\[ K = (A,B,C,D) \]

\( K \setminus D? \)
\[ (BC \rightarrow D) \]
\[ K = (A,B,C) \]

\( K \setminus C? \)
\[ (AB \rightarrow C) \]
\[ K = (A,B) \]
• \( R(A, B, C, D, E, H) \)
• \( \{ AB \rightarrow CD, D \rightarrow HB, C \rightarrow H, CH \rightarrow AE, BC \rightarrow D \} \)
• Find 3 keys:

We found one! Let’s go back a step and see if it helps:
K = (A,B) —*one step back* → K = (A,B,C)

K\B?

Doesn’t help 😞

K\A

C→A 😊

K = (B,C)
• $R(A, B, C, D, E, H)$
• \{\begin{align*} AB &\rightarrow CD, D \rightarrow HB, C \rightarrow H, CH \rightarrow AE, BC \rightarrow D \end{align*}\}

2. Find Minimal Cover

1\textsuperscript{st} step: Break left side
\begin{align*} \{AB &\rightarrow C, AB \rightarrow D, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \end{align*}

2\textsuperscript{nd} step: Iterate
\begin{align*} \{AB &\rightarrow C, AB \rightarrow D, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \end{align*}
No 😞

\begin{align*} \{AB &\rightarrow C, AB \rightarrow D, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \end{align*}
Yes ($AB \rightarrow C, INCLUSION, AB \rightarrow BC, BC \rightarrow D, TRANSITIVITY, AB \rightarrow D$)

\begin{align*} \{AB &\rightarrow C, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \end{align*}
2. Find Minimal Cover

\{AB \rightarrow C, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \\
No more moves of first type, onward to second:

\{AB \rightarrow C, D \rightarrow H, D \rightarrow B, C \rightarrow H, CH \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \\
Yes (C \rightarrow H, CH \rightarrow A, SEMI - TRANSITIVITY, C \rightarrow A) \\

\{AB \rightarrow C, D \rightarrow H, D \rightarrow B, C \rightarrow H, C \rightarrow A, CH \rightarrow E, BC \rightarrow D \} \\
Yes (In same manner) \\

\{AB \rightarrow C, D \rightarrow H, D \rightarrow B, C \rightarrow H, C \rightarrow A, C \rightarrow E, BC \rightarrow D\}
3. 3NF Decompose

1\textsuperscript{st} step (never forget): check if there is an FD with all attributes

2\textsuperscript{nd} step: decompose

• $R_1 = (A, B, C)$
• $R_2 = (D, H, B)$
• $R_3 = (C, H, A, E)$
• $R_4 = (B, C, D)$

3\textsuperscript{rd} step: check for superkey

$R_1 \, \text{👍}$
A student who didn’t listen to the TA (Ahem ahem) tried to check for 3NF in a schema like this:

Find a (minimal) key $Z$ of $R$
For each nontrivial FD $X \rightarrow Y$ in $F$ {
    If $X$ is not a superkey then {
        For each $A$ in $Y \setminus X$ {
            If $A$ is not in $Z$ then return false
        }
    }
}

Return true
• If his algorithm returned **TRUE**, is the schema in 3NF

• Yes, each FD $X \rightarrow Y$ maintains:
  – Triviality
  – $X$ is a superkey
  – $Y$ is part of key $Z$
• If his algorithm returned **FALSE**, is the schema **necessarily not** in 3NF

• No, it is possible that the right side of the FD is part of a different key
• Schema $R(A, B, C, D, E, H)$
  
  \[ F = \{ E \rightarrow AB, AH \rightarrow B, C \rightarrow D, AC \rightarrow H, D \rightarrow E \} \]

• Prove that the decomposition \{B, E\}, \{C, D, E, H\}, \{A, B, D, H\} does not maintain dependencies

• Dependencies that are maintained:
  - \( E \rightarrow B \)
  - \( AH \rightarrow B \)
  - \( C \rightarrow D \)
  - \( D \rightarrow E \)

• Dependencies that aren’t:
  - \( E \rightarrow A \) (A never appears on right hand side)
  - \( AC \rightarrow H \) (H never appears on right hand side)
• Schema \( R(A, B, C, D, E, H) \)
  \[ F = \{ E \rightarrow AB, AH \rightarrow B, C \rightarrow D, AC \rightarrow H, D \rightarrow E \} \]

• Does the decomposition \{B, E\}, \{C, D, E, H\}, \{A, B, D, H\} preserve information?
• Schema $R(A, B, C, D, E, H)$
  
  $$F = \{ E \rightarrow AB, AH \rightarrow B, C \rightarrow D, AC \rightarrow H, D \rightarrow E \}$$

• How can we make the decomposition
  
  \{B, E\}, \{C, D, E, H\}, \{A, B, D, H\} preserve everything?

• Information – already preserved. No harm if we add more schemas.

• Dependencies – We’re missing \{E \rightarrow A, AC \rightarrow H\}. As such, any number of schemas which have A,C,H together and E,A together will work.
• Schema $R(A_1, ..., A_n)$ $n > 2$

• Show FD such that:
  – $|FD| = 2$
  – $R$ is 3NF
  – $R$ is not BCNF

• $A_1, ..., A_{n-1} \rightarrow A_n$
  $A_n \rightarrow A_1$
• $R(A, B, C)$

• $F = \{ A \to B, B \to C, BC \to A \}$

• Find minimal cover

• $F_{\text{min}} = \{ A \to B, B \to C, B \to A \}$
• \( R(A, B, C) \)
• \( F = \{ A \rightarrow B, B \rightarrow C, BC \rightarrow A \} \)
• \( F_{min} = \{ A \rightarrow B, B \rightarrow C, B \rightarrow A \} \)
• Prove that for every possible \( F_{min} \):
  \( \forall f \in F_{min} \) - \( f \) has one attribute on each side (no \( AB \rightarrow C \))

• Right side – obvious
• Left side – Suppose there is an \( AB \rightarrow C \) which we can’t minimize, then contradict using \( F_{min} \) from previous slide