### Outline

- **Motivation**
- **Basics**
- **Defining Functional Dependencies**
- **Reasoning about Functional Dependencies**
- **Summary and Outlook**

### Problems due to Badly Designed Schemas

<table>
<thead>
<tr>
<th>ProfID</th>
<th>Name</th>
<th>Rank</th>
<th>Room</th>
<th>LecID</th>
<th>Title</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2125</td>
<td>Sokrates</td>
<td>C4</td>
<td>226</td>
<td>4052</td>
<td>Ethics</td>
<td>2</td>
</tr>
<tr>
<td>2132</td>
<td>Popper</td>
<td>C3</td>
<td>52</td>
<td>5041</td>
<td>Logics</td>
<td>4</td>
</tr>
<tr>
<td>2132</td>
<td>Popper</td>
<td>C3</td>
<td>52</td>
<td>5259</td>
<td>Databases</td>
<td>4</td>
</tr>
<tr>
<td>2238</td>
<td>Plato</td>
<td>C4</td>
<td>221</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

- **Redundancies:** Information about Popper appears multiple times  
  (and, thus, wastes storage space and may cause anomalies)
- **Update Anomalies:** Raising Popper's rank requires multiple changes
- **Delete Anomalies:** Deleting the Ethics course deletes information about Sokrates
- **Insert Anomalies:** Inserting Plato without a lecture?  
  (Notice, SQL NULL is unsuitable: Is it unknown whether Plato has a lecture or unknown what the lecture is?)
Designing Good Databases

- Relations should have semantic unity
- Information repetition and change anomalies should be avoided
- Avoid NULL as much as possible
  - Certainly avoid excessive NULLs
- Avoid unnecessary joins

Can we approach this problem more systematically?

Goals
- A methodology for evaluating schemas (detecting anomalies).
- A methodology for transforming bad schemas into good schemas (repairing anomalies).

Basic Definitions

Universe: $\text{DOM}$ denotes the set of all possible values.
Attributes: $\mathcal{U}$ denotes the set of all possible attributes.
  Each attribute $A \in \mathcal{U}$ has a domain $\text{dom}(A) \subseteq \text{DOM}$.

Tuple: A tuple on a set of attributes $R = \{A_1, \ldots, A_k\}$ is a mapping
  \[ u : R \to (\text{dom}(A_1) \cup \ldots \cup \text{dom}(A_k)) \]
  such that $u(A) \in \text{dom}(A)$ for all $A \in R$.

Relation: A relation instance on a set of attributes $R = \{A_1, \ldots, A_k\}$ is a set of tuples on $R$.

Basic Definitions (Example)

<table>
<thead>
<tr>
<th>Publication</th>
<th>PubID</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>Mathematical Logic</td>
</tr>
<tr>
<td></td>
<td>153</td>
<td>Query Languages</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Database Systems</td>
</tr>
</tbody>
</table>

Example Schema: Publication = (PubID, Title) with
  - $\text{dom}(\text{PubID}) = \text{Int}$
  - $\text{dom}(\text{Title}) = \text{Str}$
  (where Int and Str denote the sets of all integers and of all strings, respectively)

Example Instance: $I = \{u, v, w\}$ with
  - $u(\text{PubID}) = 3$ and $u(\text{Title}) = "\text{Mathematical Logic}"$
  - $v(\text{PubID}) = 153$ and $v(\text{Title}) = "\text{Query Languages}"$
  - $w(\text{PubID}) = 1$ and $w(\text{Title}) = "\text{Database Systems}"$
Some Further Notation

Let \( u \) be a tuple on a set of attributes \( R \) and let \( X \subseteq R \). Then \( u[|X|] \) denotes the restriction of \( u \) to \( X \). Hence, \( u[X] \) is a tuple on \( X \).

Example:

\[
\begin{align*}
\text{u:} & \quad \text{PubID} & \quad \text{Title} \\
& \quad 3 & \quad \text{Mathematical Logic}
\end{align*}
\]

\( \Rightarrow \)

\[
\begin{align*}
\text{u[|\{\text{PubID}\}|]:} & \quad \text{PubID} \\
& \quad 3
\end{align*}
\]

- Suppose \( u \) is a tuple on \( \text{Publication} = \{\text{PubID}, \text{Title}\} \) with \( u(\text{PubID}) = 3 \) and \( u(\text{Title}) = "\text{Mathematical Logic}". 
- Let \( u' = u[\{|\text{PubID}\}|]. \)
- Then, still \( u'(\text{PubID}) = 3 \) but \( u'(\text{Title}) \) is undefined.

Keys Revisited

**Superkey**: a set of attributes for which no pair of distinct tuples in the relation will ever agree on the corresponding values

**Definition**

Let \( R \) be a set of attributes and let \( X \subseteq R \). \( X \) is a superkey of \( R \), if for any pair of tuples \( u, v \) on \( R \) it holds:

\[
\text{If } u \neq v, \text{ then } u[|X|] \neq v[|X|].
\]

**Candidate Key**: a minimal superkey

**Definition**

Let \( R \) be a set of attributes and let \( X \subseteq R \). \( X \) is a key of \( R \), if:

1. \( X \) is a superkey of \( R \), and
2. For all \( Y \subset X \): \( Y \) is not a superkey of \( R \).

**Primary Key**: a designated candidate key

Functional Dependencies Revisited

**Functional Dependency** (informally): \( X \rightarrow Y \) requires that if two tuples agree on the values for attributes in \( X \), they must also agree on the values for attributes in \( Y \).

Example:

\[
\begin{align*}
\text{ProfID} & \quad \text{Name} & \quad \text{Rank} & \quad \text{Room} & \quad \text{LecID} & \quad \text{Title} & \quad \text{Hours} \\
2125 & \quad \text{Sokrates} & \quad \text{C4} & \quad 226 & \quad 4052 & \quad \text{Ethics} & \quad 2 \\
2132 & \quad \text{Popper} & \quad \text{C3} & \quad 52 & \quad 5041 & \quad \text{Logics} & \quad 4 \\
2132 & \quad \text{Popper} & \quad \text{C3} & \quad 52 & \quad 5259 & \quad \text{Databases} & \quad 4
\end{align*}
\]

\( \{\text{ProfID}\} \rightarrow \{\text{Name}, \text{Rank}, \text{Room}\} \)

**Some Terminology**

- \( X \) functionally determines \( Y \) (or, simply \( X \) determines \( Y \)),
- \( Y \) functionally depends on \( X \) (or, simply \( Y \) depends on \( X \)),
- The functional dependency is trivial if \( Y \subseteq X \).
Functional Dependencies Revisited (cont’d)

Functional Dependency (informally): \( X \rightarrow Y \) requires that if two tuples agree on the values for attributes in \( X \), they must also agree on the values for attributes in \( Y \).

Definition

We call \( X \rightarrow Y \) a functional dependency over a set of attributes \( R \), if \( X, Y \subseteq R \).

A relational instance \( I \) on \( R \) satisfies this functional dependency if for any pair of tuples \( u \in I \) and \( v \in I \) it holds:

\[
\text{If } u[X] = v[X], \text{ then } u[Y] = v[Y].
\]

Functional Dependencies (Example)

ProfLectures

<table>
<thead>
<tr>
<th>ProfID</th>
<th>Name</th>
<th>Rank</th>
<th>Room</th>
<th>LecID</th>
<th>Title</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>2125</td>
<td>Sokrates</td>
<td>C4</td>
<td>226</td>
<td>4052</td>
<td>Ethics</td>
<td>2</td>
</tr>
<tr>
<td>2132</td>
<td>Popper</td>
<td>C3</td>
<td>52</td>
<td>5041</td>
<td>Logics</td>
<td>4</td>
</tr>
<tr>
<td>2132</td>
<td>Popper</td>
<td>C3</td>
<td>52</td>
<td>5259</td>
<td>Databases</td>
<td>4</td>
</tr>
</tbody>
</table>

\{ ProfID \} \rightarrow \{ Name, Rank \}
\{ ProfID \} \rightarrow \{ Room \}
\{ LecID \} \rightarrow \{ Title, Hours \}
\{ LecID \} \rightarrow \{ Title \}

Sets of Functional Dependencies

Definition

Let \( \Sigma = \{ \sigma_1, \ldots, \sigma_n \} \) be a set of FDs over attribute set \( R \), and let \( I \) be a relational instance on \( R \). \( I \) satisfies \( \Sigma \), if \( I \) satisfies all \( \sigma \in \Sigma \).

Definition

Let \( \Sigma \) be a set of FDs over attribute set \( R \), and let \( \sigma \) be an FD over \( R \). \( \Sigma \) implies \( \sigma \), denoted by

\[ \Sigma \models \sigma, \]

if any relational instance \( I \) on \( R \) that satisfies \( \Sigma \), also satisfies \( \sigma \).

Example: Let \( \Sigma = \{ \{ \text{ProfID} \} \rightarrow \{ \text{Name, Room} \}, \{ \text{Room} \} \rightarrow \{ \text{Building} \} \}. \)

- Then, it is trivial to see: \( \Sigma \models \{ \text{ProfID} \} \rightarrow \{ \text{Room} \}. \)
- But it also holds that \( \Sigma \models \{ \text{ProfID} \} \rightarrow \{ \text{Building} \}. \)

How do we know what are all the additional FDs that are implied?
Closure of FD Sets

**Definition**
Let \( \Sigma \) be a set of FDs over attribute set \( R \).
The closure of \( \Sigma \), denoted by \( \Sigma^+ \), is the set of all FDs that are satisfied by every relational instance on \( R \) that satisfies \( \Sigma \).

\[
\Sigma^+ := \{ \sigma | \Sigma \vdash \sigma \}
\]

Properties:
- \( \Sigma \subseteq \Sigma^+ \)
- \( \Sigma^+ \) includes all those FDs over \( R \) that are trivial.
- \((\Sigma^+)^+ = \Sigma^+\)

Relationship to keys:
- Suppose \((R, \Sigma)\) is a relational schema (i.e. \( \Sigma \) are FDs over \( R \)).
- \( X \subseteq R \) is a superkey of this schema if and only if \( X \rightarrow R \subseteq \Sigma^+ \).

Reasoning About FDs

Logical implications can be derived by using inference rules called Armstrong's rules:

- **Reflexivity:** \( Y \subseteq X \implies X \rightarrow Y \)
- **Augmentation:** \( X \rightarrow Y \implies XZ \rightarrow YZ \)
- **Transitivity:** \( X \rightarrow Y, Y \rightarrow Z \implies X \rightarrow Z \)

*We use \( XY \) as a short form for \( X \cup Y \).*

These rules are:
- sound (anything derived from \( \Sigma \) is in \( \Sigma^+ \)) and
- complete (anything in \( \Sigma^+ \) can be derived from \( \Sigma \)).

Additional rules can be derived:
- **Union:** \( X \rightarrow Y, X \rightarrow Z \implies X \rightarrow YZ \)
- **Decomposition:** \( X \rightarrow YZ \implies X \rightarrow Y \)

Reasoning About FDs (Example)

Let \( \Sigma = \{ \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{Hours} \}, \ 1 \)
\{ \text{PNum} \} \rightarrow \{ \text{PName}, \text{Loc} \}, \ 2 \)
\{ \text{Loc}, \text{Hours} \} \rightarrow \{ \text{Allowance} \} \} \). \ 3

A derivation of \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{Allowance} \} \):

- using reflexivity: \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{PNum} \} \) \ 4
- using transitivity of 4 and 2: \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{PName}, \text{Loc} \} \) \ 5
- using decomposition of 5: \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{Loc} \} \) \ 6
- using union of 1 and 6: \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{Hours}, \text{Loc} \} \) \ 7
- using transitivity of 7 and 3: \( \{ \text{SIN}, \text{PNum} \} \rightarrow \{ \text{Allowance} \} \) \ 8

**Reflexivity:** \( Y \subseteq X \implies X \rightarrow Y \)

**Augmentation:** \( X \rightarrow Y \implies XZ \rightarrow YZ \)

**Transitivity:** \( X \rightarrow Y, Y \rightarrow Z \implies X \rightarrow Z \)

**Union:** \( X \rightarrow Y, X \rightarrow Z \implies X \rightarrow YZ \)

**Decomposition:** \( X \rightarrow YZ \implies X \rightarrow Y \)
Using the Closure of FD Sets?

Now we know how to compute $\Sigma^+$. Hence, we could use a set of FDs to compute a key.

(Recall: Suppose $(R, \Sigma)$ is a relational schema (i.e. $\Sigma$ are FDs over $R$). $X \subseteq R$ is a superkey of this schema if and only if $X \rightarrow R \in \Sigma^+$.)

Unfortunately, computing $\Sigma^+$ is intractable (the size of $\Sigma^+$ is exponential in the number of attributes).

Hold on, not all is lost...

Attribute Closure

**Definition**
Let $\Sigma$ be a set of FDs over attribute set $R$, and let $X \subseteq R$.

The attribute closure of $X$ w.r.t. $\Sigma$, denoted by $cl_\Sigma(X)$, is the maximum set of attributes functionally determined by $X$.

$$cl_\Sigma(X) := \{ A \mid \Sigma \vdash X \rightarrow \{ A \} \}$$

**Theorem:** $X \rightarrow Y \in \Sigma^+$ if and only if $Y \subseteq cl_\Sigma(X)$.

$cl_\Sigma(X)$ can be computed in polynomial time...

Computing Attribute Closures

```plaintext
function ComputeAttrClosure(X, \Sigma) begin
    X^+ := X;
    while there exists an FD $(Y \rightarrow Z) \in \Sigma$ such that
    (i) $Y \subseteq X^+$, and (ii) $Z \not\subseteq X^+$ do
        X^+ := X^+ \cup Z;
    end while;
    return X^+;
end
```
Computing Attribute Closures (Example)

Let \( R = \{ \text{SIN}, \text{PNum}, \text{EName}, \text{PName}, \text{Loc}, \text{Allowance} \} \)
and \( \Sigma = \{ \{ \text{SIN} \} \rightarrow \{ \text{EName} \}, 1 \}
\{ \text{PNum} \} \rightarrow \{ \text{PName}, \text{Loc} \}, 2 \}
\{ \text{Loc}, \text{Hours} \} \rightarrow \{ \text{Allowance} \} \}. 3 \)

Compute \( c_k(\{ \text{PNum}, \text{Hours} \}): \)

initially: \( X^+ = \{ \text{PNum}, \text{Hours} \} \)
using 2: \( X^+ = \{ \text{PNum}, \text{Hours}, \text{PName}, \text{Loc} \} \)
using 3: \( X^+ = \{ \text{PNum}, \text{Hours}, \text{PName}, \text{Loc}, \text{Allowance} \} \)

... while there exists an FD \( (Y \rightarrow Z) \in \Sigma \) such that
(i) \( Y \subseteq X^+ \), and (ii) \( Z \not\subseteq X^+ \) do
\( X^+ := X^+ \cup Z \);
end while; ...

Summary

- Basic structural elements:
  - relation scheme, attributes, attribute domains
  - relation instance, tuples, attribute values
- Primary key constraints (superkey, candidate key, primary key)
- Functional dependencies
- Using the attribute closure (and algorithm \textit{ComputeAttrClosure})
  we can
  - efficiently test implication (i.e. given a set \( \Sigma \) of FDs and an FD \( \sigma \),
    does \( \Sigma \models \sigma \) hold?)
  - and therefore we can efficiently compute all candidate keys.

Outlook

Recall:

\textbf{Goals}

1. A methodology for evaluating schemas (detecting anomalies).
2. A methodology for transforming bad schemas into good schemas (repairing anomalies).

- Normal forms
- Decomposition