The Relational Model

A Formal View on the RM, Basics of Functional Dependency Theory

Olaf Hartig

David R. Cheriton School of Computer Science
University of Waterloo

CS 640
Principles of Database Management and Use
Winter 2013

Some of these slides are based on a slide set provided by M. T. Öszu.

Notes

Outline

Notes

Problems due to Badly Designed Schemas

ProfLectures

ProfID	Name	Rank	Room	LecID	Title	Hours
2125	Sokrates	C4	226	4052	Ethics	2
2132	Popper	C3	52	5041	Logics	4
2132	Popper	C3	52	5259	Databases	4
2238	Platon	C4	221	?	?	?

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 Platon without a lecture? (Notice, SQL `NULL` is unsuitable: Is it unknown whether Platon has a lecture or unknown what the lecture is?)

Notes

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).

Notes

Basic Definitions

Universe: $\mathcal{DOM}$ 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 $\text{dom}(A) \subseteq \mathcal{DOM}$.

Tuple: A *tuple* on a set of attributes $R = \{A_1, \ldots, A_k\}$ is a mapping

$$u : R \to \left(\text{dom}(A_1) \cup \ldots \cup \text{dom}(A_k)\right)$$

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.

Notes

Basic Definitions (Example)

Publication

PubID	Title
3	Mathematical Logic
153	Query Languages
1	Database Systems

Example Schema: Publication $= \{\text{PubID}, \text{Title}\}$ with

- $\text{dom}(\text{PubID}) = \mathit{Int}$
- $\text{dom}(\text{Title}) = \mathit{Str}$

  (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 $u(\text{Title}) =$ "Mathematical Logic"
- $v(\text{PubID}) = 153$ and $v(\text{Title}) =$ "Query Languages"
- $w(\text{PubID}) = 1$ and $w(\text{Title}) =$ "Database Systems"

Notes

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:

u:

PubID	Title
3	Mathematical Logic

$\Longrightarrow$ $u[\{\text{PubID}\}]$:

PubID
3

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

Notes

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:

If $u \neq v$, 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

Notes

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:

ProfID	Name	Rank	Room	LecID	Title	Hours
2125	Sokrates	C4	226	4052	Ethics	2
2132	Popper	C3	52	5041	Logics	4
2132	Popper	C3	52	5259	Databases	4

$$\{\text{ProfID}\} \rightarrow \{\text{Name}, \text{Rank}, \text{Room}\}$$

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 $Y \subseteq X$.

Notes

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:

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

Notes

Functional Dependencies (Example)

ProfLectures

ProfID	Name	Rank	Room	LecID	Title	Hours
2125	Sokrates	C4	226	4052	Ethics	2
2132	Popper	C3	52	5041	Logics	4
2132	Popper	C3	52	5259	Databases	4

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

Notes

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** Σ, if I satisfies all $\sigma \in \Sigma$.

Definition

Let Σ be a set of FDs over attribute set R, and let σ be an FD over R.

Σ **implies** σ, denoted by

$$\Sigma \models \sigma,$$

if any relational instance I on R that satisfies Σ, also satisfies σ.

Example: Let $\Sigma = \{\{\mathsf{ProfID}\} \rightarrow \{\mathsf{Name, Room}\}, \{\mathsf{Room}\} \rightarrow \{\mathsf{Building}\}\}$.

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

How do we know what are all the additional FDs that are implied?

Notes

Closure of FD Sets

Definition

Let Σ be a set of FDs over attribute set R.

The **closure** of Σ, denoted by Σ^+, is the set of all FDs that are satisfied by every relational instance on R that satisfies Σ.

$$\Sigma^+ := \{\sigma \mid \Sigma \models \sigma\}$$

Properties:

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

Relationship to keys:

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

Notes

Reasoning About FDs

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

Reflexivity: $Y \subseteq X \implies X \to Y$

Augmentation*: $X \to Y \implies XZ \to YZ$

Transitivity: $X \to Y,\ Y \to Z \implies X \to Z$

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

These rules are:

- sound (anything derived from Σ is in Σ^+) and
- complete (anything in Σ^+ can be derived from Σ).

Additional rules can be derived:

Union: $X \to Y,\ X \to Z \implies X \to YZ$

Decomposition: $X \to YZ \implies X \to Y$

Notes

Reasoning About FDs (Example)

$$\text{Let } \Sigma = \{\ \begin{array}{l} \{\text{SIN, PNum}\} \to \{\text{Hours}\}, \\ \{\text{PNum}\} \to \{\text{PName, Loc}\}, \\ \{\text{Loc, Hours}\} \to \{\text{Allowance}\} \quad \}. \end{array} \quad \begin{array}{r} 1 \\ 2 \\ 3 \end{array}$$

A derivation of $\{\text{SIN, PNum}\} \to \{\text{Allowance}\}$:

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

Reflexivity:	$Y \subseteq X \implies X \to Y$
Augmentation:	$X \to Y \implies XZ \to YZ$
Transitivity:	$X \to Y,\ Y \to Z \implies X \to Z$
Union:	$X \to Y,\ X \to Z \implies X \to YZ$
Decomposition:	$X \to YZ \implies X \to Y$

Notes

Using the Closure of FD Sets?

Now we know how to compute Σ^+.

Hence, we could use a set of FDs to compute a key.

(Recall:

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

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

Hold on, not all is lost...

Notes

Attribute Closure

Definition

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

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

$$cl_\Sigma(X) := \left\{ A \,\middle|\, \Sigma \models X \to \{A\} \right\}$$

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

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

Notes

Computing Attribute Closures

function $ComputeAttrClosure(X, \Sigma)$
begin
  $X^+ := X$;
  while there exists an FD $(Y \to Z) \in \Sigma$ such that
    (i) $Y \subseteq X^+$, and (ii) $Z \not\subseteq X^+$ **do**
   $X^+ := X^+ \cup Z$;
  end while;
  return X^+;
end

Notes

Computing Attribute Closures (Example)

Let $R = \{$SIN, PNum, EName, PName, Loc, Allowance$\}$

and $\Sigma = \{$ $\{$SIN$\} \to \{$EName$\}$, 1
$\{$PNum$\} \to \{$PName, Loc$\}$, 2
$\{$Loc, Hours$\} \to \{$Allowance$\}$ $\}$. 3

Compute $cl_\Sigma(\{$PNum, Hours$\})$:

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

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

Notes

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 *ComputeAttrClosure*) we can
  - efficiently test implication (i.e. given a set Σ of FDs and an FD σ, does $\Sigma \models \sigma$ hold?)
  - and therefore we can efficiently compute all candidate keys.

Notes

Outlook

Recall:

Goals

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

1. Normal forms
2. Decomposition

- Moshe Y. Vardi: **Fundamentals of Dependency Theory**. In *Trends in Theoretical Computer Science*. ed. E. Borger, Computer Science Press (1987).

Notes