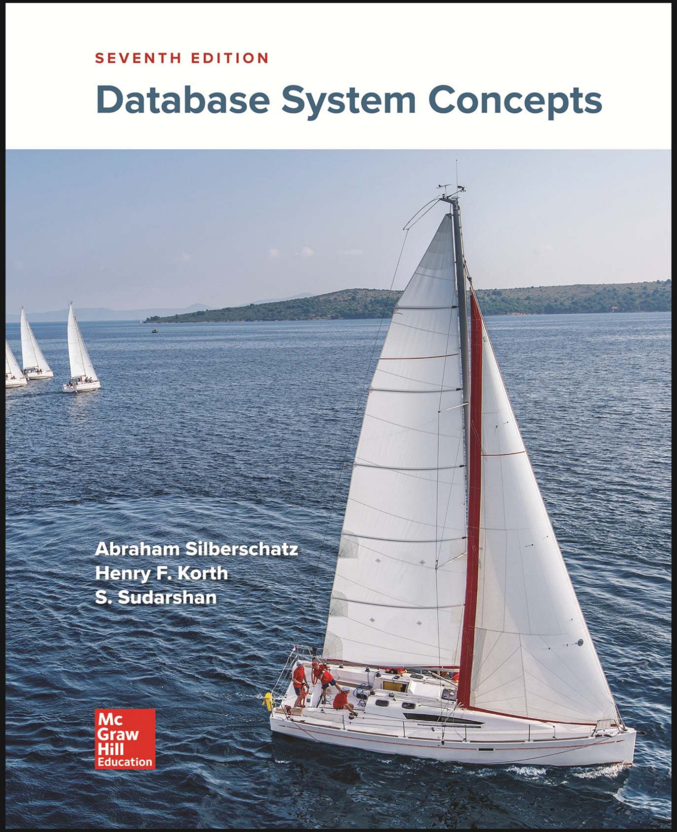
IF2240 – Basis Data

Relational Database Design

KNOWLEDGE & SOFTWARE ENGINEERING



Sumber

- Silberschatz, Korth, Sudarshan: "Database System Concepts", 7th Edition
  - Chapter 7: Relational Database Design

Capaian

- Mahasiswa dapat menghasilkan desain basis data relasional yang baik berdasarkan prinsip-prinsip yang diberikan

Outline

- Features of Good Relational Design
- Functional Dependencies
- Decomposition Using Functional Dependencies
- Normal Forms
- Functional Dependency Theory
- Algorithms for Decomposition using Functional Dependencies
- Database-Design Process

Features of Good Relational Designs

ID	name	salary	dept_name	building	budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Decomposition

ID	name	salary	dept_name	building	budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

ID	name	salary	dept_name
22222	Einstein	95000	Physics
12121	Wu	90000	Finance
32343	El Said	60000	History
45565	Katz	75000	Comp. Sci.
98345	Kim	80000	Elec. Eng.
76766	Crick	72000	Biology
10101	Srinivasan	65000	Comp. Sci.
58583	Califieri	62000	History
83821	Brandt	92000	Comp. Sci.
15151	Mozart	40000	Music
33456	Gold	87000	Physics
76543	Singh	80000	Finance

dept_name	building	budget
Physics	Watson	70000
Finance	Painter	120000
History	Painter	50000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Biology	Watson	90000
Music	Packard	80000

Decomposition

Not all decompositions are good.

employee(ID, name, street, city, salary)

ID	name	street	city	salary
…				
57766	Kim	Main	Perryridge	75000
98776	Kim	North	Hampton	67000
…				

lossy decomposition

employ

ID	name
…	
57766	Kim
98776	Kim
…	

2(name, street, city, salary)

name	street	city	salary
…			
Kim	Main	Perryridge	75000
Kim	North	Hampton	67000
…			

employee1 ⋈ *employee2*

ID	name	street	city	salary
…				
57766	Kim	Main	Perryridge	75000
57766	Kim	North	Hampton	67000
98776	Kim	Main	Perryridge	75000
98776	Kim	Nort	Hampton	67000
…				

Lossless Decomposition

Let R be a relation schema and let R_1 and R_2 form a decomposition of R.

That is $R = R_1 \cup R_2$

A **lossless decomposition** if

There is no loss of information by replacing R with the two relation schemas $R_1 \cup R_2$

Formally,

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

And, conversely a decomposition is lossy if

$$r \subset \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r)$$

Lossless Decomposition

Let R be a relation schema and let R_1 and R_2 form a decomposition of R.

That is R = $R_1 \cup R_2$

A lossless decomposition if

There is no loss of information by replacing R with the two relation schemas $R_1 \cup R_2$

Formally,

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

And, conversely a decomposition is lossy if

$$r \subset \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r)$$

$R(A,B,C)$

A	B	C
α	1	A
β	2	B

$R1(A,B) = \Pi_{A,B}(r)$

A	B
α	1
β	2

$R2(B,C) = \Pi_{B,C}(r)$

B	C
1	A
2	B

$R1 \bowtie R2$

A	B	C
α	1	A
β	2	B

Normalization Theory

Decide whether a particular relation R is in "good" form.

In the case that a relation R is not in "good" form, decompose it into set of relations $\{R_1, R_2, \ldots, R_n\}$ such that

- Each relation is in good form
- The decomposition is a lossless decomposition

Our theory is based on:

- Functional dependencies
- Multivalued dependencies

Functional Dependencies

There are usually a variety of constraints (rules) on the data in the real world.

For example, some of the constraints that are expected to hold in a university database are:

Students and instructors are uniquely identified by their ID.	Each student and instructor has only one name.	Each instructor and student is (primarily) associated with only one department.	Each department has only one value for its budget, and only one associated building.

Functional Dependencies

- An instance of a relation that satisfies all such real-world constraints is called a **legal instance** of the relation;
- A legal instance of a database is one where all the relation instances are legal instances
- Constraints on the set of legal relations.
- Require that the value for a certain set of attributes determines uniquely the value for another set of attributes.
- A functional dependency is a generalization of the notion of a *key*.

Functional Dependencies Definition

Let R be a relation schema

$\alpha \subseteq R$ *and* $\beta \subseteq R$

The **functional dependency** $\alpha \rightarrow \beta$

holds on R if and only if for any legal relations $r(\mathrm{R})$, whenever any two tuples t_1 and t_2 of r agree on the attributes α, they also agree on the attributes β.

That is,

$t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$

Functional Dependencies Definition

Let R be a relation schema

$\alpha \subseteq R$ *and* $\beta \subseteq R$

The **functional dependency** $\alpha \rightarrow \beta$

holds on R if and only if for any legal relations r(R), whenever any two tuples t_1 and t_2 of r agree on the attributes α, they also agree on the attributes β.

That is,

$t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$

Consider $R(A,B)$ with the following instance

A	B
1	4
1	5
3	7

$B \rightarrow A$ ✅

$A \rightarrow B$ ❌

Functional Dependencies Example

Each student and instructor has only one name.	SID → SNAME IID → INAME
Each instructor and student is (primarily) associated with only one department.	SID → DNAME IID → DNAME
Each department has only one value for its budget, and only one associated building.	DNAME → BUILDING BUDGET

Closure of a Set of Functional Dependencies

Given a set F of functional dependencies, there are certain other functional dependencies that are logically implied by F.

The set of **all** functional dependencies logically implied by F is the **closure** of F.

We denote the *closure* of F by F^+.

NIM → PRODI

PRODI → FAKULTAS

NIM → FAKULTAS

Latihan

Tuliskan himpunan *functional dependencies* F yang mungkin terdefinisi bagi relasi ini.

Cabang	Petugas	Akun	Nasabah
Dago	Andra	12345671	Thomas
Sukajadi	Endang	65412342	Indra
Dago	Andra	12356563	Diah
Dago	Andra	12387654	Thomas
Sukajadi	Nisrin	65490905	Diah
Sukajadi	Nisrin	65412346	Wildan
Buahbatu	Indah	34567897	Thomas

Solusi

KNOWLEDGE & SOFTWARE ENGINEERING

Keys and Functional Dependencies

K is a superkey for relation schema R if and only if

- $K \rightarrow R$

K is a candidate key for R if and only if

- $K \rightarrow R$, and
- for no $\alpha \subset K$, $\alpha \rightarrow R$

Keys and Functional Dependencies

K is a superkey for relation schema R if and only if

- $K \rightarrow R$

K is a candidate key for R if and only if

- $K \rightarrow R$, and
- for no $\alpha \subset K$, $\alpha \rightarrow R$

STUDENTS(ID, NAME, DOB, MAJOR)

ID NAME → ID NAME DOB MAJOR
∴ {ID,NAME} is a superkey of STUDENTS

ID → ID NAME DOB MAJOR
∴ {ID} is a superkey and also a candidate key of STUDENTS

Keys and Functional Dependencies

K is a superkey for relation schema R if and only if

- $K \rightarrow R$

K is a candidate key for R if and only if

- $K \rightarrow R$, and
- for no $\alpha \subset K$, $\alpha \rightarrow R$

Functional dependencies allow us to express constraints that cannot be expressed using superkeys

Consider the schema:

in_dep (*ID*, *name*, *salary*, *dept_name*, *building*, *budget*).

- $dept\_name \rightarrow building$ ✅
- $ID \rightarrow building$ ✅
- $dept\_name \rightarrow salary$ ❌

Use of Functional Dependencies

To test relations to see if they are legal under a given set of functional dependencies.

↓

If a relation *r* is legal under a set *F* of functional dependencies, we say that *r* **satisfies** *F*.

To specify constraints on the set of legal relations

↓

We say that *F* **holds on** *R* if all legal relations on *R* satisfy the set of functional dependencies *F*.

Note: A specific instance of a relation schema may satisfy a functional dependency even if the functional dependency does not hold on all legal instances.

- For example, a specific instance of *instructor* may, by chance, satisfy $name \rightarrow ID$.

Trivial Functional Dependencies

- *A* functional dependency is **trivial** if it is satisfied by all instances of a relation
- Example:

  ID, name → *ID*   *name* → *name*

- In general, $\alpha \rightarrow \beta$ is trivial if $\beta \subseteq \alpha$