Data Wrangling and Data Analysis

Functional Dependencies

Hakim Qahtan

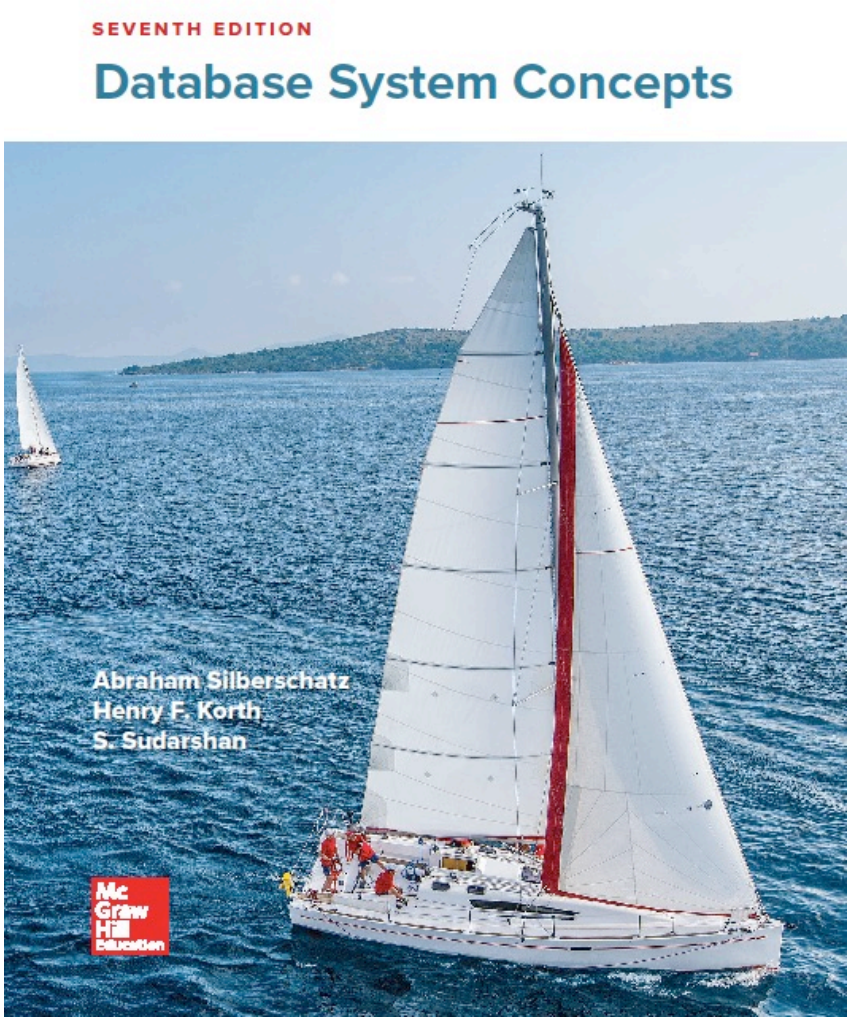
Department of Information and Computing Sciences
Utrecht University

Utrecht University

Reading Material for Today

Database System Concepts (7th Edition)

CH 7.1 - 7.4.1



Designing Good Databases

The Database Design Problem

- How do we know when a design is good?
- A DB schema seems to be good if it helps us to avoid redundancy and inconsistency, but are there more quality issues?
- This question can be answered using the normalization theory
- Basic rules for good DB design
  - 1 table for each entity
  - 1 table for each relationship
  - Each cell contains a single value
  - If you have BIG relations, decompose them
    - This could result in a serious problem

Decomposing Big Relations

- Definition:

  Let R be a relation schema (with constraints).

  A decomposition of R is a set of relation schemas $R_1, R_2, \ldots, R_n$ such that

  i. each R_i consists of attributes in R and

  ii. each attribute of R occurs in at least one R_j

Example

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Consider the two representations of the data

Option 1:

Person

id	name	carId
A	John	2
B	Nick	3
C	Mary	null
A	John	3

Car

carId	brand	color
2	VW	black
3	VW	Red

Option 2:

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Which one do you like?

Person

id	name	carId
A	John	2
B	Nick	3
C	Mary	null
A	John	3

Car

carId	brand	color
2	VW	black
3	VW	Red

PERSON JOIN CAR

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Join followed by decompose

Projection on id, name, carId

Person

id	name	carId
A	John	2
B	Nick	3
C	Mary	null
A	John	3

Projection on carId, brand, color

Car

carId	brand	color
2	VW	black
3	VW	Red

Decompose followed by join

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Projection on id, name, carId

Person

id	name	carId
A	John	2
B	Nick	3
C	Mary	null
A	John	3

Projection on carId, brand, color

Car

carId	brand	color
2	VW	black
3	VW	Red

PERSON JOIN CAR

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Improper decompose followed by join

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	Red
C	Mary	null	null	null
A	John	3	VW	Red

Projection on id, name, carId, brand

Person

id	name	carId	brand
A	John	2	VW
B	Nick	3	VW
C	Mary	null	null
A	John	3	VW

Projection on brand, color

Car

brand	color
VW	black
VW	Red

PERSON JOIN CAR

Sales

id	name	carId	brand	color
A	John	2	VW	black
B	Nick	3	VW	black
C	Mary	null	null	null
A	John	3	VW	black
A	John	2	VW	Red
B	Nick	3	VW	Red
A	John	3	VW	Red

There is clearly a problem here !!

Records in red are called spurious tuples

Dependencies

- The solution for lossy decomposition is to decompose big tables to Normal Forms (lossless decomposition)
- ***Definition***:

  Let R a relation schema (with constraints).

  A decomposition $R_1, R_2, \ldots, R_n$ of R is called lossless *iff.* for each valid relation (instance) $\mathrm{r}(R)$:

$$r = \pi_{R_1(r)} \bowtie \pi_{R_2(r)} \bowtie \ldots \bowtie \pi_{R_n(r)}$$

Functional Dependence (FD)

- Functional dependence (FD): the values of a set of attributes X determine the values of another set of attributes Y
  - Denoted by $X \longmapsto Y$ (X determines Y)
  - If two records has the same set of values for the attributes in X the they should have the same set of values for the attributes in Y
  - In the instructor relation, dept_name is functionally dependent on name ($name \longmapsto dept\_name$)
  - Given the instructor name, I can find *one and only one value* of dept_name
- Constraints on the set of legal relation instances
- Require that the value for a certain set of attributes determines uniquely the value for another set of attributes

Functional Dependence

- Let R be a relation with attributes (A,B, C, D, E)

$$X \subseteq R,\ Y \subseteq R$$

- The functional dependency

$$X \longmapsto Y$$

holds on R if and only if whenever two tuples t_1, t_2 of R agree on the attributes of X, they also agree on the attributes of Y. That is

$$t_1[X] = t_2[X] \implies t_1[Y] = t_2[Y]$$

- Examples:
  - The capital determines the country
  - The country determines the Internet domain

instructor table

ID	name	dept_name	salary
22322	Einstein	Physics	95000
33452	Gold	Physics	87000
21212	Wu	Finance	90000
10101	Brandt	Comp. Sci.	82000
43521	Katz	Comp. Sci.	75000
98531	Kim	Biology	78000
58763	Crick	Elec. Eng.	80000
52187	Mozart	History	65000
32343	El Said	History	86000

Does $name \longmapsto dept\_name$ hold?
Does $name \longmapsto salary$ hold?
Does dept_$name \longmapsto salary$ hold?
Does dept_$name \longmapsto name$ hold?

Note that: $ID \longmapsto A \quad \forall A \in instructor$
In this example: $name \longmapsto A \quad \forall A \in instructor$

Alternative Definition of the Keys

- K is a superkey for relation R if and only if $\mathrm{K} \longmapsto R$
  - This is the *uniqueness* property of "key"
- K is a candidate key for R if and only if
  - $\mathrm{K} \longmapsto R$, and
  - For any $X \subset K, X \nrightarrow R$
    - makes sure key has minimum set of attributes (*minimality*)

Functional Dependencies

- Functional dependencies allow us to express constraints that cannot be expressed using superkeys.
- Example: Consider the instructor relation:

We expect the following set of functional dependencies to hold:

id $\longmapsto$ name
id $\longmapsto$ dept_name
name, dept_name $\longmapsto$ salary

but would not expect the following to hold:
salary $\rightarrow$ name

Closure of a Set of Functional Dependencies

- Given a set of functional dependencies $\mathcal{F}$, there are certain other functional dependencies that are logically implied by $\mathcal{F}$.
- The set of all functional dependencies *logically implied* by $\mathcal{F}$ is the closure of $\mathcal{F}$.
- We denote the closure of $\mathcal{F}$ by $\mathcal{F}^+$.
- We can find all of $\mathcal{F}^+$ by applying Armstrong's Axioms:
  - if $X \subseteq Y$, then $Y \longmapsto X$ (*reflexivity*)
  - if $X \longmapsto Y$, then $AX \longmapsto AY$ (*augmentation*)
  - if $X \longmapsto Y$ and $Y \longmapsto W$, then $X \longmapsto W$ (*transitivity*)

  these rules are sound and complete. A is a set of attributes (could be single attribute)

Functional Dependencies

- FDs can be derived from existing dependencies using Armstrong's Axioms
- Examples:
  - If Y is a subset of X, then $\mathrm{X} \longmapsto Y$ (reflexivity)
  - If X is a key candidate, then $\mathrm{X} \longmapsto Y, \ \forall Y$
  - $\mathrm{X} \longmapsto Y \implies X \longmapsto B \qquad \forall B \in Y$
    - We can restrict the RHS to have only a single attribute

Examples of Armstrong's Axioms

- if $X \subseteq Y$, then $Y \longmapsto X$ (*reflexivity*)

  name $\longmapsto$ name
  name, dept_name $\longmapsto$ name
  name, dept_name $\longmapsto$ dept_name

- if $X \longmapsto Y$, then $AX \longmapsto AY$ (*augmentation*)

  name $\longmapsto$ dept_name
  name, salary $\longmapsto$ dept_name, salary

- if $X \longmapsto Y$ and $Y \longmapsto W$, then $X \longmapsto W$ (*transitivity*)

  id $\longmapsto$ name and

  name $\longmapsto$ dept_name implies id $\longmapsto$ dept_name

More Derived FDs

$X \longmapsto Y$ and $X \longmapsto W$ then $X \longmapsto YW$

$X \longmapsto YW$ then $X \longmapsto Y$ and $X \longmapsto W$ we saw this earlier

$X \longmapsto Y$ and $WY \longmapsto Z$ then $XW \longmapsto Z$

- Can we prove the correctness of $X \longmapsto Y$ and $WY \longmapsto Z$ then $XW \longmapsto Z$?

  $X \longmapsto Y$ and $WY \longmapsto Z$ (given)

  $X \longmapsto Y$ then $XW \longmapsto YW$

  $XW \longmapsto YW$ and $YW \longmapsto Z$ then $XW \longmapsto Z$

- Exercise: can you prove if $X \longmapsto Y$ then $XW \longmapsto Y$?

Boyce-Codd Normal Form

- Trivial FDs
  - An FD $X \longmapsto Y$ is said to be trivial if $Y \subseteq X$
  - It is called trivial because it is satisfied by all relations
- Boyce-Codd Normal Form (BCNF):
  - Definition: A relation R is said to be in BCNF if for all FDs $X \longmapsto Y$, where $X \subseteq R$ and $Y \subseteq R$ then:
    - Either $X \longmapsto Y$ is a trivial FD OR
    - X is a superkey for R

Other Normal Forms

- Third Normal Form (3NF)
  - A relaxation of BCNF
  - A relation R is said to be in 3NF *iff.* for all FDs $X \longmapsto Y$, where $X \subseteq R$ and $Y \subseteq R$ then:
    - Either $X \longmapsto Y$ is a trivial FD, OR
    - X is a superkey for R, OR
    - For any attribute $A \in Y \wedge A \notin X \Rightarrow A \in \tilde{X}$, where $\tilde{X}$ is a candidate key
  - R is in BCNF implies that R in 3NF

Other Normal Forms

- Second Normal Form (2NF)
  - A relation R is said to be in 2NF *iff.* for all FDs $X \longmapsto Y$, where $X \subseteq R$ and $Y \subseteq R$, and X is a candidate key, then:
    - For any attribute $A \in Y$ and any set of attributes $B \subset X$ (proper subset of X), $B \nrightarrow A$
    - (i.e.) no partial dependency
- First Normal Form (1NF)
  - Every cell in the table contains an atomic value (single value)

Use of FDs

- We use functional dependencies to:
  - Test relations to see if they are legal under a given set of functional dependencies.
    - 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
      $name \longmapsto id$
  - Check if a relation decomposition is lossless or not
    - ***Theorem***: For relational schema $\mathrm{R}(XYZ)$, the following holds:
      If $X \longmapsto Y$ then the decomposition $R_1(XY), R_2(XZ)$ is lossless
    - In other words, a decomposition of a relation R into R_1 and R_2 is said to be lossless if at least one of the following holds:
      - $\mathrm{R}_1 \cap R_2 \longmapsto R_1$ OR
      - $\mathrm{R}_1 \cap R_2 \longmapsto R_2$

Use of FDs (Cont.)

- We use functional dependencies to:
  - Detect inconsistencies in the data
  - For example, if we are given that each instructors can work for only one department and:

    $$name \longmapsto dept\_name$$

    Then the highlighted records violate this FD

    When discovering an FD violation, each value can be considered as the source of violation
- Exercise: If you know that in a given relation T

  $$att\_a \longmapsto att\_b$$

  Write python script to check for violations

id	name	dept_name	salary
22322	Einstein	Physics	95000
33452	Gold	Physics	87000
21212	Wu	Finance	90000
10101	Einstein	Comp. Sci.	82000
43521	Katz	Comp. Sci.	75000
98531	Kim	Biology	78000
58763	Crick	Elec. Eng.	80000

The highlighted cells reflect violations of the FD *name* → *dept_name*.

Conditional Functional Dependencies (CFDs)

- In the UK, zip code uniquely determines the street
- The constraint may not hold for other countries
- This constraints can be expressed as follows

  ([country = 44, zip] → street)
- It expresses a fundamental part of the semantics of the data
- It can NOT be expressed as an FD
  - It does not hold on the entire relation; instead, it holds on tuples representing UK customers only

country	area-code	phone	street	city	zip
44	131	1234567	Mayfield	Liverpool	EH4 8LE
44	131	3456789	Crichton	Manchester	EH4 8LE
01	908	3456789	Mountain Ave	NYC	07974

The highlighted cells reflect violations of the CFD *([country = 44, zip] → street)*.

- Summarize what you learned today in 2-minutes

The information in this presentation has been compiled with the utmost care,
but no rights can be derived from its contents.

© Utrecht University