Database Management Systems INFO 210

Conceptual Modeling Lecture 12

Franz Wotawa
TU Graz, Institut for Software Technologie
Inffeldgasse 16b/2
wotawa@ist.tugraz.at

What we already discussed

- Functional dependencies
- Normal forms:
  - 1NF
  - BCNF
- Decomposition to assure fulfillment of BCNF
- What we will see today is that BCNF is sometimes too strong!
  - Will use 3NF instead!

Example of data inconsistencies

- Change of the address of a lecture hall!

Lect_num	Type	Name	Hours	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	Franz Wotawa	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	Franz Wotawa	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....	.....	....	....	

Change 16 to 17!

Lect_num	Type	Name	Hours	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	Franz Wotawa	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	Franz Wotawa	LH i11	Infeldgasse 17
117.111	UE	Compiler Construction	1.0	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....	.....	....	....	

$\neq$

1. Normal form (1NF)

- *"A relation is in first normal form if and only if the **domain** of each attribute **contains only atomic (indivisible) values**, and the value of each attribute contains only a single value from that domain."*
- The 1NF enables queries and sorting!
- There is one extension: *"... and each relation must have a primary key."*

Example 1NF

Lect_num	Type	Name	Hours	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	Franz Wotawa, Birgit Hofer	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	Franz Wotawa	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....	.....	....	....	

The relation above is not in 1NF!!!

Lect_num	Type	Name	Hours	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	Franz Wotawa	LH i7	Inffeldgasse 25
117.101	VO	Software Maintenance	3.0	Birgit Hofer	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	Franz Wotawa	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....	.....	....	....	

2. Normal form (2NF)

- *"A relation in 1NF is in 2NF if and only if no **non-prime attribute** is **dependent** on any **proper subset** of any (candidate) key of the relation.*

  ***A non-prime attribute** of a relation is an attribute that is **not part of any** (candidate) **key** of the relation."*

Every non-prime attribute has to be dependent on the key only

- A database that is not in 2NF comprises redundancies!

Example 2NF

- Let us consider the following table with primary key {Lect_num,SSN}:

Lect_num	Type	Name	Hours	SSN	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	1000	Franz Wotawa	LH i7	Inffeldgasse 25
117.101	VO	Software Maintenance	3.0	1002	Birgit Hofer	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	1000	Franz Wotawa	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	1002	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....	.....		....	....	

- Dependencies:
  - {Lect_num} → *{Type, Name, Hours, Room}*
  - {SSN} → *{Teacher}*
  - *{Room}* → *{Address}*

Example 2NF

Redundancy

This table is not in 2NF!!!
- Type, name, and hours of a lecture do only depend on the lecture number (Lect_num) and not on the teacher!

Lect_num	Type	Name	Hours	SSN	Teacher	Room	Address
117.101	VO	Software Maintenance	3.0	1000	Franz Wotawa	LH i7	Inffeldgasse 25
117.101	VO	Software Maintenance	3.0	1002	Birgit Hofer	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	1000	Franz Wotawa	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	1002	Birgit Hofer	LH i11	Inffeldgasse 16
....	......	....			....	....	

Make 3 tables!!!

Lect_num	Type	Name	Hours	Room	Address
117.101	VO	Software Maintenance	3.0	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	LH i11	Inffeldgasse 16
....	......	....	.....	....	

Lect_num	SSN
117.101	1000
117.101	1002
117.102	1000
117.111	1002
....	....

SSN	Teacher
1000	Franz Wotawa
1002	Birgit Hofer
....	....

3. Normal form (3NF)

- *"A relation in 2NF is in 3NF if and only if all the attributes in a table are determined only by the candidate keys of that relation and not by any non-prime attributes.*

  No non-prime attributes are allowed to be transitive dependent on a prime attribute*!"*

- Eliminates problems occurring when changing information!

Example 3NF

{Lect_num} →*{Type, Name, Hours, Room}*
{Room} →*{Address}*

- **This table is not in 3NF!!!**
  - The room of the lecture depends functionally on the lecture number (Lect_num). The address of the room depends functionally on the room itself. Hence, we have a transitive dependency between a key and an attribute that is not in the key!

Lect_num	Type	Name	Hours	Room	Address
117.101	VO	Software Maintenance	3.0	LH i7	Inffeldgasse 25
117.101	VO	Software Maintenance	3.0	LH i7	Inffeldgasse 25
117.102	VO	Compiler Construction	2.0	LH i11	Infeldgasse 16
117.111	UE	Compiler Construction	1.0	LH i11	Inffeldgasse 16
....	......	....	.....	....	

Make 2 tables!!!

Lect_num	Type	Name	Hours	Room
117.101	VO	Software Maintenance	3.0	LH i7
117.101	VO	Software Maintenance	3.0	LH i7
117.102	VO	Compiler Construction	2.0	LH i11
117.111	UE	Compiler Construction	1.0	LH i11
....	......	....	.....	....

Room	Address
LH i7	Inffeldgasse 25
LH i7	Inffeldgasse 25
LH i11	Infeldgasse 16
LH i11	Inffeldgasse 16
....	

Outline: Normalization

- Anomalies ✓
- Boyce-Codd Normal Form
- 3rd Normal Form

Anomalies

The goal of relational schema design is to avoid anomalies and redundancy:

- *Update anomaly* : one occurrence of a fact is changed, but not all occurrences
- *Deletion anomaly* : a valid fact is lost when a tuple is deleted

Examples of Bad Design

Drinkers(name, addr, beersLiked, manf, favBeer)

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	???	WickedAle	Pete's	???
Spock	Enterprise	Bud	???	Bud

Data is redundant, because
each of the ???s can be figured out by using the FDs
- name → addr favBeer
- beersLiked → manf

Such Bad Design Also Exhibits Anomalies

name	addr	beersLiked	manf	favBeer
Janeway	Voyager	Bud	A.B.	WickedAle
Janeway	Voyager	WickedAle	Pete's	WickedAle
Spock	Enterprise	Bud	A.B.	Bud

- Update anomaly: if Janeway is transferred to *Intrepid*, will we remember to change each of her tuples?
- Deletion anomaly: If nobody likes Bud, we lose track of the fact that Anheuser-Busch manufactures Bud.

Outline

- Anomalies
- Boyce-Codd Normal Form ✓
- 3rd Normal Form

Boyce-Codd Normal Form

A relation *R* is in Boyce-Codd Normal Form (*BCNF*) if
whenever $X \rightarrow A$ is a *nontrivial* FD that holds in *R*,
then *X* is a *superkey*

Remember:

- *nontrivial* means $A \notin X$
- a *superkey* is any superset of a key
  (not necessarily a strict superset)

"Each attribute must describe the key, the whole key, and nothing but the key"

Example

Drinkers(name, addr, beersLiked, manf, favBeer)

FDs: name → addr favBeer, beersLiked → manf

- The only key is {name, beersLiked}
- In each FD above, the left side is *not* a superkey

⇒ Any one of these FDs shows Drinkers is not in BCNF

Each of the above FDs is a partial dependency,
i.e., the right side depends only on a *part of the key*

Yet Another Example

Beers(name, manf, manfAddr)

FD's: name → manf, manf → manfAddr

- The only key is {name}
- name → manf does not violate BCNF,
- … but manf → manfAddr does

The second FD is a transitive dependency, because
manfAddr depend on manf, which is not part of any key

Decomposition into BCNF

Given: relation R with FDs F

Goal: decompose R into relations $R_1,\ldots,R_m$ such that

- each R_i is a *projection* of R
- each R_i is in *BCNF*
  (wrt the projection of F)
- R is the *natural join* of $R_1,\ldots,R_m$

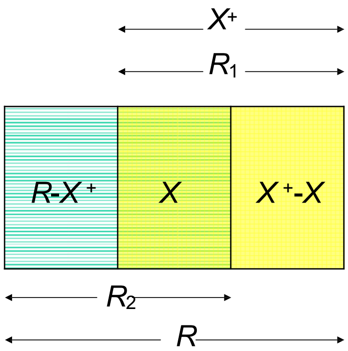
Intuition: R is broken into pieces

- that contain the same information as R,
- but are free of redundancy

The Algorithm: Divide and Conquer

- Look in F for an FD $X \rightarrow B$ that violates BCNF
  (If any FD following from F violates BCNF, then there is surely an FD in F itself that violates BCNF)
- Compute X^+
  (X^+ does not contain all attributes of R, otherwise X would be superkey)
- Decompose R using $X \rightarrow B$, i.e., replace R by relations with schemas
  $R_1 = X^+$
  $R_2 = (R - X^+) \cup X$
- Compute the projections F_1, F_2 of F on R_1, R_2
- Continue with R_1, F_1 and R_2, F_2

Decomposition Picture



Example

Drinkers(name, addr, beersLiked, manf, favBeer)

F = {name → addr, name → favBeer, beersLiked → manf}

- Pick the BCNF violation name → addr
- Close the left side: $\{\text{name}\}^+$ = {name, addr, favBeer}
- Decomposed relations:
  Drinkers1(name, addr, favBeer)
  Drinkers2(name, beersLiked, manf)

Example (cntd)

Projecting FDs:

- For Drinkers1(name, addr, favBeer), the only relevant FDs are name → addr and name → favBeer:
  ⇒ the only key is {name}
  ⇒ Drinkers1 is in BCNF

- For Drinkers2(name, beersLiked, manf), the only relevant FD is beersLiked → manf:
  ⇒ the only key is {name, beersLiked}
  ⇒ violation of BCNF (because there is partial dependency)

- Continue with Drinkers2

Example (cntd)

Drinkers2(name, beersLiked, manf)

F_2 = {beersLiked → manf}

- Pick the BCNF violation beersLiked → manf
- Close the left side: $\{\text{beersLiked}\}^+$ = {beersLiked, manf}
- Decomposed relations:
  Drinkers3(beersLiked, manf)
  Drinkers4(name, beersLiked)

Example (cntd)

Projecting FDs:

- For Drinkers3(beersLiked, manf), the only relevant FD is beersLiked $\rightarrow$ manf:
  $\Rightarrow$ the only key is {beersLiked}
  $\Rightarrow$ Drinkers3 is in BCNF

- For Drinkers4(name, beersLiked), there is no relevant FD:
  $\Rightarrow$ the only key is {name, beersLiked}
  $\Rightarrow$ Drinkers4 is in BCNF

Example (concluded)

We have decomposed

Drinkers(name, addr, beersLiked, manf, favBeer)

into

- Drinkers1(<u>name</u>, addr, favBeer)
- Drinkers3(<u>beersLiked</u>, manf)
- Drinkers4(<u>name</u>, <u>beersLiked</u>)

Notice:

- Drinkers1 is about drinkers
- Drinkers3 is about beers
- Drinkers4 is about the relationship between drinkers and the beers they like

Outline

- Anomalies
- Boyce-Codd Normal Form
- 3rd Normal Form ✓

Third Normal Form — Motivation

There is one structure of FDs that causes trouble when we decompose:

- $AB \rightarrow C$ and $C \rightarrow B$

  Examples:
  - A = street address, B = city, C = zip code
  - A = lecturer, B = hour, C = course

- There are two keys, $\{A, B\}$ and $\{A, C\}$
- $C \rightarrow B$ is a BCNF violation, so we must decompose into
  - AC
  - BC

We Cannot Enforce FDs

If we decompose *ABC* into *AC* and *BC*,
then we cannot enforce $AB \to C$
by checking FDs in the decomposed relations

Example with *A* = street,
B = city
C = zip

on the next slide

29

An Unenforceable FD

street	zip
545 Tech Sq.	02138
545 Tech Sq.	02139

city	zip
Cambridge	02138
Cambridge	02139

Join tuples with equal zip codes.

street	city	zip
545 Tech Sq.	Cambridge	02138
545 Tech Sq.	Cambridge	02139

Although no FDs were violated in the decomposed relations,
FD street city $\to$ zip is violated by the database as a whole

30

3NF Lets Us Avoid This Problem

3rd Normal Form (3NF) modifies the BCNF condition so we do not have to decompose in this problem situation

- An attribute is *prime* if it is a member of *any* key
- $X \rightarrow A$ violates 3NF if and only if
  - X is *not a superkey*
  - and also A is *not prime*

Back to the *ABC* Example

On ABC, we have FDs $AB \rightarrow C$ and $C \rightarrow B$

$\Rightarrow$ There are two keys: AB and AC

$\Rightarrow$ A, B, and C are each prime

$\Rightarrow$ Although $C \rightarrow B$ violates BCNF,
it does not violate 3NF

What 3NF and BCNF Give You

There are two important properties of a decomposition:

◆ *Losslessness*: It should be possible
  - ◆to project the original relation onto the decomposed schema
  - ◆and then reconstruct the original by a natural join

◆ *Dependency Preservation*: It should be possible to check in the projected relations whether all the given FDs are satisfied

3NF and BCNF Continued

- We can get (1) with a BCNF decomposition
- We can get both (1) and (2) with a 3NF decomposition
- But we can't always get (1) and (2) with a BCNF decomposition

  "street-city-zip" is an example

Exercise

Consider the relation PI (= passengerInfo) with the attributes

PI(FlightNo, Date, DepartureTime, SeatNo, TicketNo, Name, Address, LuggageId, Weight)

and the FDs

- FlightNo, Date → DepartureTime
- FlightNo, Date, TicketNo → SeatNo
- TicketNo → Name Address
- LuggageId → Weight Date TicketNo

- Is the relation in Boyce-Codd normal form?
- If not, decompose into relation that are in BCNF. Is the resulting decomposition dependency preserving?

26

Exercise

Let R be a relation with attributes *ABCD*. Consider the following combinations of FDs on R:

- $AB \to C$, $C \to D$, $D \to A$
- $B \to C$, $B \to D$
- $AB \to C$, $BC \to D$, $CD \to A$, $AD \to B$
- $A \to B$, $B \to C$, $C \to D$, $D \to A$

For each collection of FDs do the following:

1. Indicate all the BCNF violations
2. Decompose the relations into collections of relations that are in BCNF
3. Are the decompositions dependency preserving?

36