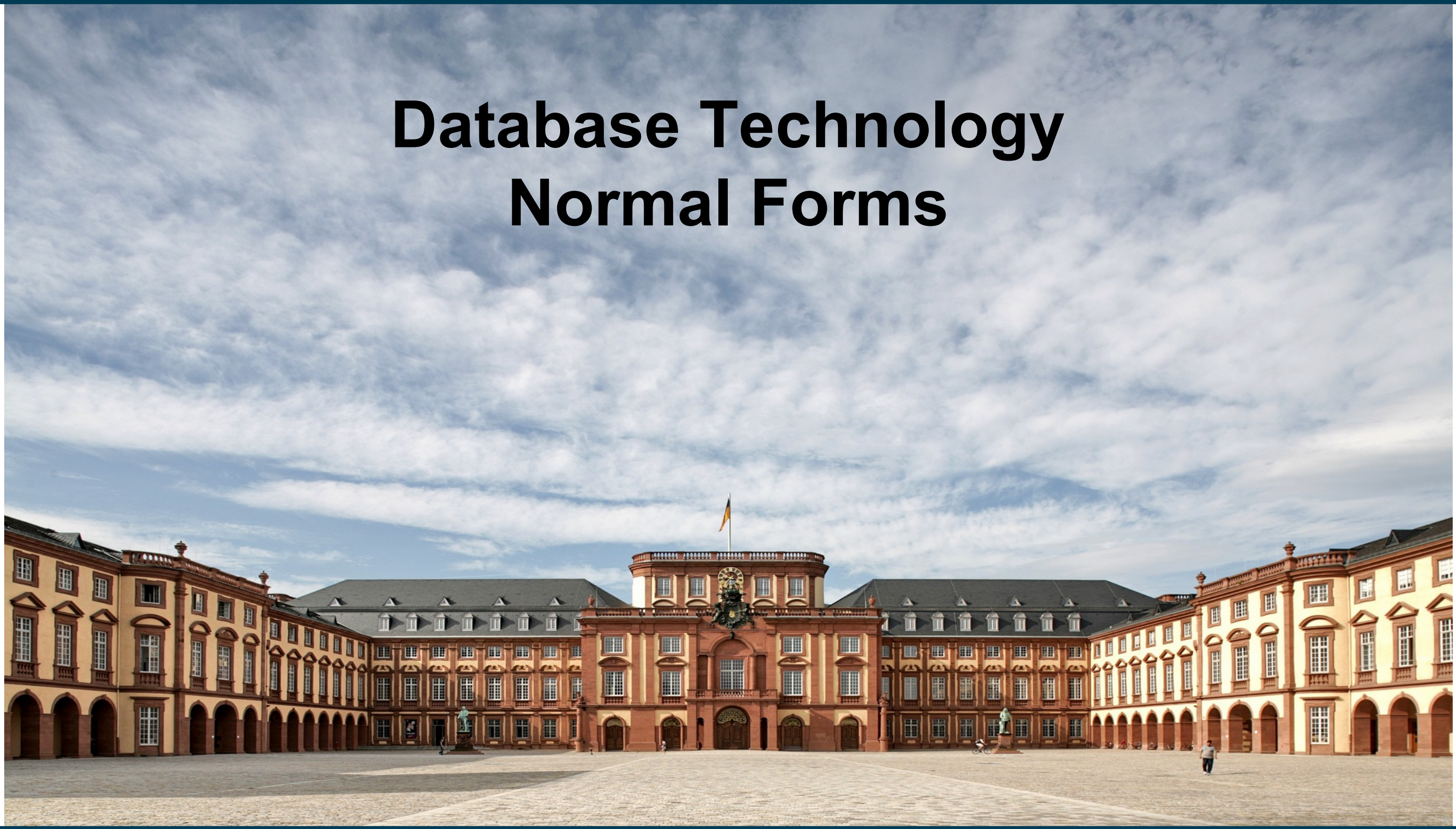
UNIVERSITÄT MANNHEIM



Heiko Paulheim

Previously on Database Technology

- Designing databases with ER diagrams
  - Modeling a domain as a collection of *entities* and *relationships*
  - Entity: a “thing” or “object” in the domain
    - Described by a set of *attributes*
  - Relationship: an association among several entities
  - Represented diagrammatically by an *entity-relationship diagram*

Today

- More about database design
  - Features of *good* Relational design
  - Atomic domains and first normal form
  - Decomposition using functional dependencies
    - 2nd, 3rd normal form, and Boyce Codd normal form
  - Decomposition using multi-valued dependencies
    - 4th normal form
  - Even more normal forms

The Normalization Process

To the foregoing I should perhaps add the following. As far as I know, Codd himself never mentioned, in his early writings on the subject, his reasons for introducing the terminology of normal forms or normalization. But many years afterward, he did go on record with his own explanation:[33]

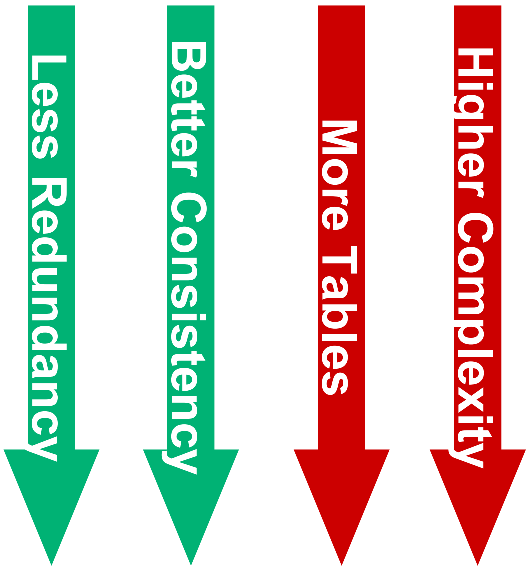
> ***Interviewer:*** *Where did "normalization" come from?*
>
> **Codd:** It seemed to me essential that some discipline be introduced into database design. I called it normalization because then President Nixon was talking a lot about normalizing relations with China. I figured that if he could normalize relations, so could I.

The Normalization Process

- Iteratively improve the database design
  - Rule out non-atomic values
  - Eliminate redundancies
- Iterations
  - Move database design from one normal form to the next
- In each iteration
  - The design is changed (usually: smaller, but more relations)
  - Some typical problems are eliminated

The Normalization Process

- Levels of normalization based on the amount of redundancy in the database
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - **Third Normal Form (3NF)**
  - **Boyce-Codd Normal Form (BCNF)**
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain Key Normal Form (DKNF)



Atomicity

- Consider the following relation

ID	Name	Instructor	Programs
CS-101	Introduction to Computer Science	Melanie Smith	CS, DS, Math, CE
CS-205	Introduction to Databases	Mark Johnsson	DS, Soc
CS-374	Data Mining	Mark Johnsson	CS, DS, Soc
MA-403	Linear Algebra	Mary Williams	Math, CS
...	...	...	...

- Task:
  - Find all courses in the DS program

Atomicity

- Find all courses in the DS program
  - Requires processing all the strings of the *Programs* attribute
  - String processing is expensive

- Notion of *atomicity*:
  - Attribute is *atomic* (also: *scalar*) if its values are considered to be indivisible units
  - Examples of non-atomic attributes
    - Set-valued attributes (like *Programs*)
    - Composite attributes (like *Instructor*)
    - Identification numbers like CS-101 that can be broken up into (meaningful) parts

Atomicity

- In the database, all values, also strings, are considered indivisible
- SQL queries will only return strings
  - e.g., “CS-101” or “CS, DS, Math, CE”

- If we further analyze them
  - Extract department “CS” from “CS-101”
  - Find program “DS” in “CS, DS, Math, CE”
- ...we move the semantics from the database to the application logic!

First Normal Form

- Definition: A relational schema R is in **first normal form** if the domains of all attributes of R are atomic
- Rationale: Non-atomic values
  - complicate storage and encourage redundant storage of data
  - complicate processing of complex attributes
  - move data semantics to processing code
- From here on, we assume all relations are in first normal form

First Normal Form – Decomposition

- Replace composite attributes by single attributes
- Replace multi-valued attributes by a new relati
  - before: R= (<u>ID</u>, …, mva)
  - after: R = (<u>ID</u>, …), R_{mva} = (<u>ID</u>,mva)

ID_dept	ID_no	Name	
CS	101	Introduction to Computer Science	Mela
CS	205	Introduction to Databases	Mark
CS	374	Data Mining	Mark
MA	403	Linear Algebra	Mary
...		...	...

multi-attribute key

ID_dept	ID_no	Program
CS	101	CS
CS	101	DS
	101	Math
	101	CE
CS	205	DS
CS	205	Soc
CS	374	CS
CS	374	DS
CS	374	Soc
MA	403	Math
MA	403	CS
...	...	...

Functional Dependencies

- Functional dependencies
  - are a means to identify potential redundancies
  - also: a means to identify primary keys
- Definition:
  - If one set of attributes A determines another set of attributes B
  - Then B is functionally dependent on A
- Less formal:
  - If we know A, we also know B

Functional Dependencies

- Consider the example below
- The course ID (two parts) determine
  - The course name
  - The instructor
- Functional dependency:
  - {ID_dept, ID_no} $\rightarrow$ {Name, Instructor_first, Instructor_last}

ID_dept	ID_no	Name	Instructor_first	Instructor_last
CS	101	Introduction to Computer Science	Melanie	Smith
CS	205	Introduction to Databases	Mark	Johnsson
CS	374	Data Mining	Mark	Johnsson
MA	403	Linear Algebra	Mary	Williams
...		...	...	...

Functional Dependencies

- Note:
  - Functional dependencies are not only determined from the data
  - But the domain knowledge

- Example: by chance, each instructor teaches only one course
  - {Instructor_first, Instructor_last} → {ID_dept, ID_no, Name}

ID_dept	ID_no	Name	Instructor_first	Instructor_last
CS	101	Introduction to Computer Science	Melanie	Smith
CS	205	Introduction to Databases	Mark	Johnsson
CS	374	Data Mining	John	Stevens
MA	403	Linear Algebra	Mary	Williams
...		...	...	...

Towards the Second Normal Form

- Assume we have the following relation
  - The department is represented by *ID_dept* as part of the course ID
  - ...and in fully written form in the *Department* attribute
- Suppose we insert the tuple

  (CS, 102, Mathematics, Programming)

  → We can create an inconsistency!

ID_dept	ID_no	Department	Name
CS	101	Computer Science	Introduction to Computer Science
CS	205	Computer Science	Introduction to Databases
CS	374	Computer Science	Data Mining
MA	403	Mathematics	Linear Algebra
...			...

Second Normal Form

- A relation is in second normal form if
  - It is in first normal form, and
  - All *nonkey* attributes are functionally dependent on the entire primary key
- Relation: (ID_dept, ID_no, Departmen
- Functional dependencies:
  - {ID_dept} → {Department}, {ID_dept, ID_no} → {Name}

Violation: Department only depends on part of the primary key!

ID_dept	ID_no	Department	Name
CS	101	Computer Science	Introduction to Computer Science
CS	205	Computer Science	Introduction to Databases
CS	374	Computer Science	Data Mining
MA	403	Mathematics	Linear Algebra
...			...

Decomposition to Second Normal Form

- Determine the primary key PK of a relation R
- Write down all functional dependencies for the relation
- For each FD $A \rightarrow B$, where $A \subset PK$
  - Create a new Relation RB (A,B)
  - Remove B from R

Each department name now only appears once

ID_dept	ID_no	Name
CS	101	Introduction to Computer Science
CS	205	Introduction to Databases
CS	374	Data Mining
MA	403	Linear Algebra
...		...

ID_dept	Department
CS	Computer Science
MA	Mathematics
...	

Decomposition to Second Normal Form

- Original problem: we could create an inconsistency by inserting

  (CS, 102, Mathematics, Programming)

- This is no longer possible: we cannot insert

  (CS, Mathematics)

  into

  (ID_dept, Department)

ID_dept	ID_no	Name
CS	101	Introduction to Computer Science
CS	205	Introduction to Databases
CS	374	Data Mining
MA	403	Linear Algebra
...		...

ID_dept	Department
CS	Computer Science
MA	Mathematics
...	

Lossless Join Decomposition

- Each decomposition should be lossless join
  - i.e., a natural join should reconstruct the original table
  - Consider the (wrong) example below
    - ID_dept is omitted from the first relation
  - Natural join creates cross product of both relations!

ID_no	Name
101	Introduction to Computer Science
205	Introduction to Databases
374	Data Mining
403	Linear Algebra
	...

ID_dept	Department
CS	Computer Science
MA	Mathematics
...	

Determining Keys w/ 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
  - there is no $K' \subset K$ with $K' \rightarrow R$

- There may be more than one candidate key
  - Example: student_assistant(matriculation_no, social_sec_no, name, ...)
  - Each one may be equally well picked as a primary key

Further Dependencies in Relations

- Assume we had another attribute in our relation
  - i.e., the personnel ID of the instructor
- Suppose we inserted
  - (CS, 379, Web Mining, 143970, Steven, Smith)
- Is that consistent?

ID_dept	ID_no	Name	Instr_ID	Instr_first	Instr_last
CS	101	Introduction to Computer Science	143273	Melanie	Smith
CS	205	Introduction to Databases	143970	Mark	Johnsson
CS	374	Data Mining	143970	Mark	Johnsson
MA	403	Linear Algebra	141784	Mary	Williams
...		...		...	...

Further Dependencies in Relations

- Observation 1: this relation is in 2NF
  - It is is 1NF
  - {ID_dept, ID_no} → {Name, Instr_Id, Instr_first, Instr_last} holds
  - There is no non-key attribute a for which {ID_dept} → {a} or {ID_no} → {a} holds
- Still, inconsistent inserts are possible

ID_dept	ID_no	Name	Instr_ID	Instr_first	Instr_last
CS	101	Introduction to Computer Science	143273	Melanie	Smith
CS	205	Introduction to Databases	143970	Mark	Johnsson
CS	374	Data Mining	143970	Mark	Johnsson
MA	403	Linear Algebra	141784	Mary	Williams
...		...		...	...

Third Normal Form

- Observation 2: there is a second functional dependency
  - {Instr_ID} → {Instr_first, Instr_last}
- Definition: Third Normal Form
  - Relation is in Second Normal Form, and
  - No attribute is *transitively dependent* on the primary key

ID_dept	ID_no	Name	Instr_ID	Instr_first	Instr_last
CS	101	Introduction to Computer Science	143273	Melanie	Smith
CS	205	Introduction to Databases	143970	Mark	Johnsson
CS	374	Data Mining	143970	Mark	Johnsson
MA	403	Linear Algebra	141784	Mary	Williams
...		...		...	...

Transitivity of Functional Dependencies

- We observe that
  - {ID_dept, ID_no} → {Instr_ID}
  - {Instr_ID} → {Instr_first, Instr_last}
- Hence, there is a *transitive* dependency of {Instr_first, Instr_last} on the primary key
- Functional dependencies are transitive by nature

ID_dept	ID_no	Name	Instr_ID	Instr_first	Instr_last
CS	101	Introduction to Computer Science	143273	Melanie	Smith
CS	205	Introduction to Databases	143970	Mark	Johnsson
CS	374	Data Mining	143970	Mark	Johnsson
MA	403	Linear Algebra	141784	Mary	Williams
...		...		...	...

Third Normal Form: Decomposition

- Identify transitive dependency in R
  - PK → A and A → B
- Create new relation $R_A(\underline{A}, B)$
- Remove from B from R

Instr_ID	Instr_first	Instr_last
143273	Melanie	Smith
143970	Mark	Johnsson
141784	Mary	Williams
	...	...

ID_dept	ID_no	Name	I
CS	101	Introduction to Computer Science	1
CS	205	Introduction to Databases	1
CS	374	Data Mining	1
MA	403	Linear Algebra	141784
...		...	

Third Normal Form: Decomposition

- Result: the new relations are now in 3NF
- There is no transitive dependency in R
  - {Name} → {Instr_ID} and {Instr_ID} → {Name} do not hold
- There is no transitive dependency in the new relation
  - {Instr_first} → {Instr_last} and {Instr_last} → {Instr_first} do not hold

Instr_ID	Instr_first	Instr_last
143273	Melanie	Smith
143970	Mark	Johnsson
141784	Mary	Williams
	...	...

ID_dept	ID_no	Name	I[illegible]
CS	101	Introduction to Computer Science	1[illegible]
CS	205	Introduction to Databases	1[illegible]
CS	374	Data Mining	1[illegible]
MA	403	Linear Algebra	141784
...		...	

Third Normal Form: Decomposition

- Initial problem: inconsistent insert

  (CS, 379, Web Mining, 143970, Steven, Smith)

- Now, inserting

  (143970, Steven, Smith)

  into the new relation is no longer possible!

Instr_ID	Instr_first	Instr_last
143273	Melanie	Smith
143970	Mark	Johnsson
141784	Mary	Williams
	...	...

ID_dept	ID_no	Name	I[illegible]
CS	101	Introduction to Computer Science	1[illegible]
CS	205	Introduction to Databases	1[illegible]
CS	374	Data Mining	1[illegible]
MA	403	Linear Algebra	141784
...		...	

Dependencies between Candidate Keys

- Recap:
  - 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
    - there is no $K' \subset K$ with $K' \rightarrow R$
- There may be more than one candidate key
  - Example: student_assistant(matriculation_no, social_sec_no, name, ...)
  - Each one may be equally well picked as a primary key

Multiple Candidate Keys in 3NF

- Given this example

  student_assistant(matriculation_no, social_sec_no, name, …)

- For 3NF, we would decompose this relation
  - If we pick matriculation_no as primary key, then

    {matriculation_no} → {social_sec_no} → {name}

  - If we pick social_sec_no as primary key, then

    {social_sec_no} → {matriculation_no} → name

  - i.e., in any case, there is a transitive functional dependency!

Dependencies between Candidate Keys

- Assume the following scenario
  - Departments have (one or more) secretaries
  - Secretaries work for (one or more) departments
  - Each secretary may have one phone number per department s/he works for, or just one phone number for all
  - Secretaries' basic data (name etc.) have already been decomposed

Institute	Secr_ID	Phone
CS	0001	5073
CS	0002	5074
Soc	0001	6010
Soc	0003	6011
Eng	0003	6011
...	...	...

Dependencies between Candidate Keys

- Neither Institute, Secr_ID, nor Phone are a superkey
- This relation has two candidate keys
  - {Institute, Secr_ID}
  - {Institute, Phone}

Institute	Secr_ID	Phone
CS	0001	5073
CS	0002	5074
Soc	0001	6010
Soc	0003	6011
Eng	0003	6011
...	...	...

Dependencies between Candidate Keys

- With each candidate key as a primary key, the relation is in 3NF
- {Institute, Secr_ID}:
  - {Institute, Secr_ID} - > {Phone}
  - Neither {Institute} → {Phone} nor {Secr_ID} → {Phone} holds
  - No transitive dependency

same for {Institute, Phone}

Institute	Secr_ID	Phone
CS	0001	5073
CS	0002	5074
Soc	0001	6010
Soc	0003	6011
Eng	0003	6011
...	...	...

Dependencies between Candidate Keys

- However, inconsistent inserts are still possible:

  (CS, 0001, 5075)

  if {Institute, Phone} is chosen as the primary key, or

  (Soc, 0002, 5073)

  if {Institute, Secr_ID} is chosen as the primary key

Two phone numbers for the same secretary and department

Same phone number for two secretaries in different departments

Institute	Secr_ID	Phone
CS	0001	5073
CS	0002	5074
Soc	0001	6010
Soc	0003	6011
Eng	0003	6011
...	...	...

Dependencies between Candidate Keys

- Observation: we have different functional dependencies here
  - {Institute, Secr_ID} → {Phone}
  - {Institute, Phone} → {Secr_ID}
- None of them violate the 3NF
- These are dependencies between different candidate keys

Institute	Secr_ID	Phone
CS	0001	5073
CS	0002	5074
Soc	0001	6010
Soc	0003	6011
Eng	0003	6011
...	...	...

Boyce-Codd Normal Form

- Definition: A relation is in Boyce-Codd Normal Form if
  - It is in 3NF
  - There is no functional dependency between attributes that belong to different candidate keys
  - i.e., if CK $\rightarrow$ {a}, then a must not be part of a candidate key
- BCNF is equivalent to 3NF unless
  - A relation has two or more candidate keys
  - At least two of the candidate keys are composed of more than one attribute
  - The candidate keys are not disjoint, i.e., the composite candidate keys share some attributes

Decomposition to BCNF

- Create a single relation for each composite candidate key
- Place the remaining attributes (if any) in one of the relations
  - Based on their dependency

Institute	Secr_ID
CS	0001
CS	0002
Soc	0001
Soc	0003
Eng	0003
...	...

Institute	Phone
CS	5073
CS	5074
Soc	6010
Soc	6011
Eng	6011
...	...

Decomposition to BCNF

- Problem with inconsistent inserts:

  (CS, 0001, 5075)
  (Soc, 0002, 5073)

- This is no longer possible
  - First violates primary key in R1
  - Second violates primary key in R2

Institute	Secr_ID
CS	0001
CS	0002
Soc	0001
Soc	0003
Eng	0003
...	...

Institute	Phone
CS	5073
CS	5074
Soc	6010
Soc	6011
Eng	6011
...	...

BCNF and Dependency Preservation

- We have lost some dependency during this decomposition
- We could try to reintroduce it
  - e.g., add a third relation (Secr_ID, Phone)
  - but this would lead to new sources of inconsistencies

Institute	Secr_ID
CS	0001
CS	0002
Soc	0001
Soc	0003
Eng	0003
...	...

Institute	Phone
CS	5073
CS	5074
Soc	6010
Soc	6011
Eng	6011
...	...

Which phone no. belongs to which secretary?

Boyce-Codd NF vs. 3NF

- If decomposition does not cause any loss of information it is called a *lossless decomposition*
- If a decomposition does not cause any dependencies to be lost it is called a *dependency-preserving decomposition*
- Any relation can be decomposed in a lossless way into a collection of smaller relations that are in BCNF form
  - However, *dependency preservation* is not guaranteed
- Any table can be decomposed in a lossless way into 3NF that also preserves the dependencies
  - 3NF may be better than BCNF in some cases

How Good is BCNF/3NF?

- There are database schemas in BCNF that do not seem to be sufficiently normalized
- Consider a relation

  employee (ID, cost_center, phone)
- where an employee may have more than one phone and can have multiple cost centers

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

How Good is BCNF/3NF?

- Consider a relation

  employee (ID, cost_center, phone)

- Functional dependency

  {ID} → {cost_center, phone}

- Superkey: {ID, cost_center, phone}
  - this would also be the primary key of the table

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

How Good is BCNF/3NF?

- There are database schemas in BCNF that do not seem to be sufficiently normalized
- Insertion anomaly:
  - Suppose that an employee gets a new phone number
  - That would require one insert per cost center
  - Performing only some of those inserts would cause an anomaly

This looks odd

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

How Good is BCNF/3NF?

- There are database schemas in BCNF that do not seem to be sufficiently normalized
- It looks like decomposition into (ID, Cost Center) and (ID, Phone) would be a good idea
  - But BCNF does not suggest this
  - There is only one candidate key, i.e., {ID, Cost Center, Phone}

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

Fourth Normal Form

- Here, both Cost Center and Phone are multi-valued attributes
  - i.e., for each combination of ID and Cost Center, there are multiple values for Phone
  - same for combinations of ID and Phone
- Note: cost center and phone are independent from each other
  - i.e., neither {Cost Center} → {Phone} nor {Phone} → {Cost Center} holds

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

Fourth Normal Form

- Definition: A relation is in Fourth Normal Form if
  - it is in Boyce-Codd Normal Form
  - it does not contain more than one multi-valued attribute
    - in the sense that the multiple values are independent

ID	Cost Center	Phone
1000	10020	512-555-1234
1000	10030	512-555-1234
1000	10020	512-555-4321
1000	10030	512-555-4321

Fourth Normal Form: Decomposition

- Decomposition algorithm:
  - Create a separate relation for each multi-valued attribute
  - Identify a suitable primary key
- Note: now we can safely insert a new phone number for an employee
  - Requires exactly one insert operation (as expected)
  - Does not lead to inconsistencies!

ID	Cost Center
1000	10020
1000	10030

ID	Phone
1000	512-555-1234
1000	512-555-4321

Further Decompositions

- Consider the relation below
  - Each course is offered in different years
    - But there are additional constraints, e.g., Data Mining is only offered in the second semester
  - There might be different offerings in the same year by different lecturers
  - Although all of them are multi-valued, they are *not* independent
- Primary key: {Course, Inst_ID, Semester}

Course	Inst_ID	Semester
Introduction to Computer Science	13001	1st
Introduction to Computer Science	13001	2nd
Data Mining	15743	2nd
Data Mining	14233	2nd
Linear Algebra	14233	1st
...		

Further Decompositions

- Suppose we want to insert a third offering for data mining for the 2nd semester
  - But we do not know the instructor yet
- We cannot simply insert (Data Mining, null, 2nd)
  - Why?

Course	Inst_ID	Semester
Introduction to Computer Science	13001	1st
Introduction to Computer Science	13001	2nd
Data Mining	15743	2nd
Data Mining	14233	2nd
Linear Algebra	14233	1st
...		

Further Decompositions

- However, we could easily insert (Data Mining, 12874, 1st)
  - Although data mining is only offered for the 2nd semester
  - That knowledge is not explicit in the schema

Course	Inst_ID	Semester
Introduction to Computer Science	13001	1st
Introduction to Computer Science	13001	2nd
Data Mining	15743	2nd
Data Mining	14233	2nd
Linear Algebra	14233	1st
...		

Fifth Normal Form

- Also known as *project-join normal form (PJ/NF)*
- A relation is 5NF if
  - It is in 4NF, and
  - If it cannot be decomposed and re-joined based on the keys, without removing or adding information

Course	Inst_ID	Semester
Introduction to Computer Science	13001	1st
Introduction to Computer Science	13001	2nd
Data Mining	15743	2nd
Data Mining	14233	2nd
Linear Algebra	14233	1st
...		

Fifth Normal Form

- Decomposition into Fifth Normal Form
- For each PK with three values (A,B,C)
  - Try to decompose three relations (A,B), (B,C), (A,C)
  - Analyze whether their natural join is equivalent to (A,B,C)

Course	Semester
Introduction to Computer Science	1st
…on to Computer S…	[illegible]
…ng	
…ng	
…gebra	

Course	Inst_ID
Introduction to Computer Science	13001
Introduction to Computer Science	13001
Data Mining	15743
Data Mining	14233
Linear Algebra	14233
...	

Inst_ID	Semester
13001	1st
13001	2nd
15743	2nd
14233	2nd
14233	1st

Fifth Normal Form

- Suppose we want to insert a third offering for data mining for the 2nd semester
  - We can do this now by inserting into the (course, semester) relation

Course	Semester
Introduction to Computer Science	1st
...on to Computer S...	
...ng	
...ng	
...gebra	

Course	Inst_ID
Introduction to Computer Science	13001
Introduction to Computer Science	13001
Data Mining	15743
Data Mining	14233
Linear Algebra	14233
...	

Inst_ID	Semester
13001	1st
13001	2nd
15743	2nd
14233	2nd
14233	1st

Fifth Normal Form

- Inconsistent insert (because data mining is only offered in the 2nd semester):

  (Data Mining, 12874, 1st)

- Requires three inserts
  - We may restrict the access to (course, semester)!

Course	Semester
Introduction to Computer Science	1st
…on to Computer S…	
…ng	
…ng	
…gebra	

Course	Inst_ID
Introduction to Computer Science	13001
Introduction to Computer Science	13001
Data Mining	15743
Data Mining	14233
Linear Algebra	14233
...	

Inst_ID	Semester
13001	1st
13001	2nd
15743	2nd
14233	2nd
14233	1st

6th Normal Form (or Domain Key Normal Form)

- Generally:
  - A relation is in DKNF when there can be no insertion or deletion anomalies in the database
  - i.e., all constraints must be encoded in the database
- Consider the employee relation below
  - Additional constraint: students may not work more than 80h

ID	Type	Hours
10032	Lecturer	80
10432	Student	40
10483	Secretary	160
...	...	...

6th Normal Form (or Domain Key Normal Form)

- Recap: domain of an attribute
  - can be used to define valid ranges
- Solution:
  - Decompose into individual relations by employee type
  - Impose domain constraint on attribute *S_Hours*

Lecturer_ID	Hours
10032	80
...	...

Secretary_ID	Hours
10483	160
...	...

Student_ID	S_Hours
10432	40
...	...

Incidental Denormalization

- Consider the relation below (courses and prerequisites)
  - Not normalized
  - Violates 2NF
    - Primary Key: {ID, Prereq_ID}
    - {ID} → {Name}, {Prereq_ID} → {Prereq_Name}

ID	Name	Prereq_ID	Prereq_Name
110	Data Mining	100	Databases
110	Data Mining	101	Programming
...	...	...	...

Incidental Denormalization

- Normalizing to 2NF breaks this into three tables
  - Now, displaying a list of courses with prerequisites requires two joins
  - Costly in terms of performance
- Not normalizing may be better in terms of performance
  - Alternative: materialized view

The same as (ID, Name), but normalization does not tell us

ID	Name
110	Data Mining
...	...

ID	Prereq_ID
110	100
110	101
...	...

Prereq_ID	Prereq_Name
100	Databases
101	Programming
...	...

Extreme Example

- 1NF: find non-atomic attributes

ISBN	Author	Title	Publisher	Year
978-0-857-52009-8	Stephen Baxter, Terry Pratchett	The Long Earth	Doubleday	2012
978-0-06-206777-7	Stephen Baxter, Terry Pratchett	The Long War	Harper	2013
978-0-575-07434-7	Alastair Reynolds	Absolution Gap	Gollancz	2003
...	...	...	...	...

Extreme Example

- 1NF: find non-atomic attributes
  - Break *ISBN* in parts (<u>ISBN1</u>, <u>ISBN2</u>, <u>ISBN3</u>, <u>ISBN4</u>, ISBN5)
  - Move *Author* to own relation
  - Break author's *Name* in parts

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Author_First	Author_Last
978	0	857	52009	8	Stephen	Baxter
978	0	857	52009	8	Terry	Pratchett
978	0	06	206777	7	Stephen	Baxter
						ett
						ds

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Title	Publisher	Year
978	0	857	52009	8	The Long Earth	Doubleday	2012
978	0	06	206777	7	The Long War	Harper	2013
978	0	575	07434	7	Absolution Gap	Gollancz	2003
...	...	...	...	...	...	...	...

Extreme Example

- Functional dependencies on partial key
  - Relation book: publisher only depends on ISBN1, ISBN2, ISBN3
  - Violation of 2NF

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Author_First	Author_Last
978	0	857	52009	8	Stephen	Baxter
978	0	857	52009	8	Terry	Pratchett
978	0	06	206777	7	Stephen	Baxter
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	ett
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	ds

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Title	Publisher	Year
978	0	857	52009	8	The Long Earth	Doubleday	2012
978	0	06	206777	7	The Long War	Harper	2013
978	0	575	07434	7	Absolution Gap	Gollancz	2003
...	...	...	...	...	...	...	...

Extreme Example

- Functional dependencies on partial key
  - Relation book: publisher only depends on ISBN1, ISBN2, ISBN3
  - Violation of 2NF
  - Resolution: move publisher to own relation

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Author_First	Author_Last
978	0	857	52009	8	Stephen	Baxter
978	0	857	520			
978	0	06	206			

ISBN1	ISBN2	ISBN3	Publisher
978	0	857	Doubleday
			Harper
			Gollancz
			...

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Title	Year
978	0	857	52009	8	The Long Earth	2012
978	0	06	206777	7	The Long War	2013
978	0	575	07434	7	Absolution Gap	2003
...	...	...	...	...	...	...

Extreme Example

- Observation: we still store the authors' names multiple times
  - DKNF would create author as a single table
  - Note: we need an artificial key

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Author_ID
978	0	857	52009	8	1
978	0	857	52009	8	2
978	0	06			

ISBN1	ISBN2	ISBN3	Publisher
			Doubleday

ISBN1	ISBN2	ISBN3	ISBN4	ISBN5	Title	Year
978	0	857	52009	8	Th	
978	0	06	206777	7	Th	
978	0	575	07434	7	Ab	
...	...	...	...	...	...	

Author_ID	Author_First	Author_Last
1	Stephen	Baxter
2	Terry	Pratchett
3	Alastair	Reynolds
...	...	

Extreme Example

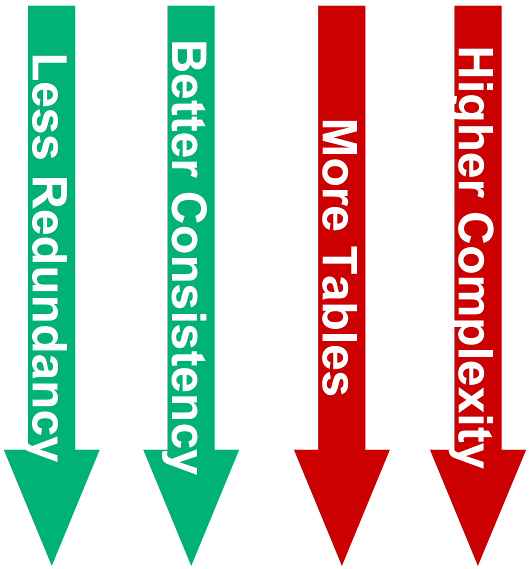
- Normalizing broke a relatively small table into four
- Discussion
  - Is it useful to break the ISBN?
  - Which of the three additional tables do we actually need?
  - Notion of atomicity/scalarity can be very subtle

ER Models vs. Normal Forms

- Note: the relation between authors and books is an n:m relation
- ER models
  - n:m relations are represented by their own table in the database
- Normalization
  - ultimately creates a table for the n:m relation, too

The Normalization Process

- Levels of normalization based on the amount of redundancy in the database
- Various levels of normalization are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - **Third Normal Form (3NF)**
  - **Boyce-Codd Normal Form (BCNF)**
  - Fourth Normal Form (4NF)
  - Fifth Normal Form (5NF)
  - Domain Key Normal Form (DKNF)



Normal Forms in a Nutshell

- Notions:
  - 1NF: based on atomic/scalar values
  - 2NF, 3NF, BCNF: based on keys and functional dependencies
  - 4NF: based on keys and multi-valued dependencies
  - 5NF: based on join dependencies
  - DKNF: based on domain definitions

- In practice
  - 3NF/BCNF is most used
  - The other NFs are rather of academic interest
    - e.g., 3NF relations that are not 4NF are rather rare

Trade-Offs

- Normalization is a trade-off
- Pro:
  - Avoid inconsistencies
  - Reduce storage
- Con:
  - Increase complexity
  - Decrease performance
- 3NF vs. BCNF
  - Pro: more inconsistencies avoided
  - Con: some dependencies lost

Summary

- How to obtain a *good* database design
  - Avoiding redundancy
  - Avoiding inconsistency
- Normalization
  - Step-by-step modification of your database design
  - Successively refines the design
- Caveat
  - Normalization until the bitter end also has shortcomings…
  - Never lose the use cases out of sight

Questions?