Normalization

Jing Hua Ye
jinghua.ye@cit.ie

November 20, 2016

Well-Structured Relations

- Contains minimal redundancy
- Allows users to insert, modify, and delete the rows in a table without errors or inconsistencies
- The employee1 table is a well-structured relation:

emp_id	name	dept_name	salary
100	Margaret Simpson	Marketing	48000
140	Allen Beeton	Accounting	52000
110	Chris Lucero	Info Systems	43000
190	Lorenzo Davis	Finance	55000
150	Susan Martin	Marketing	42000

  - Each row contains data describing one employee
  - Any modification to an employee's data is confined to one row of the table

Well-Structured Relations

- The employee2 table is not a well-structured relation:

emp_id	name	dept_name	salary	course_title	date_completed
100	Margaret Simpson	Marketing	48000	SPSS	2005-06-19
100	Margaret Simpson	Marketing	48000	Surveys	2006-10-07
140	Allen Beeton	Accounting	52000	Tax Acc	2004-12-08
110	Chris Lucero	Info Systems	43000	Visual Basic	2001-01-12
110	Chris Lucero	Info Systems	43000	C++	2003-04-22
190	Lorenzo Davis	Finance	55000		
150	Susan Martin	Marketing	42000	SPSS	2005-06-19
150	Susan Martin	Marketing	42000	Java	2009-08-12

  - Values for emp_id, name, dept_name, and salary appear in two separate rows for employees 100, 110 and 150
  - If the salary for employee 100 changes, we must record this fact in two rows

Anomaly

- Relational databases rely on the existence of a certain amount of data redundancy.
- The data redundancy is in the form of copies of primary keys acting as foreign keys in related relations to enable the modeling of relationship between data.
- An error or inconsistency that may result when an user attempts to update a table that contains redundant data
- three types of anomalies:
  - Insertion Anomalies
  - Deletion Anomalies
  - Modification/Update Anomalies

Properties for Decomposition

- Two properties associated with decomposition of a larger relation into small relations:

  Lossless-join property: any instance of the original relation can be identified from corresponding instances in the smaller relation

  Dependency preservation property: a constraint on the original relation can be maintained by enforcing some constraint on each of the smaller relations. In other words, we do not need to perform joins on the smaller relations to check whether a constraint on the original relation is violated.

Functional Dependency

- A functional dependency is a constraint between two **attributes** or two sets of attributes
- Attribute B is functionally dependent on attribute A if the value of A **uniquely determines** the value of B
- A functional dependency is **not** a mathematical dependency: B cannot computed from A
- The functional dependency of B on A is represented as $A \rightarrow B$
- **If you know the value of A there can be only one value for B** In other words, if we know the value of A and we examine the relation that holds this dependency, we find only one value of B in all tuples that have a given value of A
- An attribute may be functionally dependent on a combination of two (or more) attributes
- When two tuples have the same value of A, they also have the same value of B. **For a given value of B there may be several different values of A**

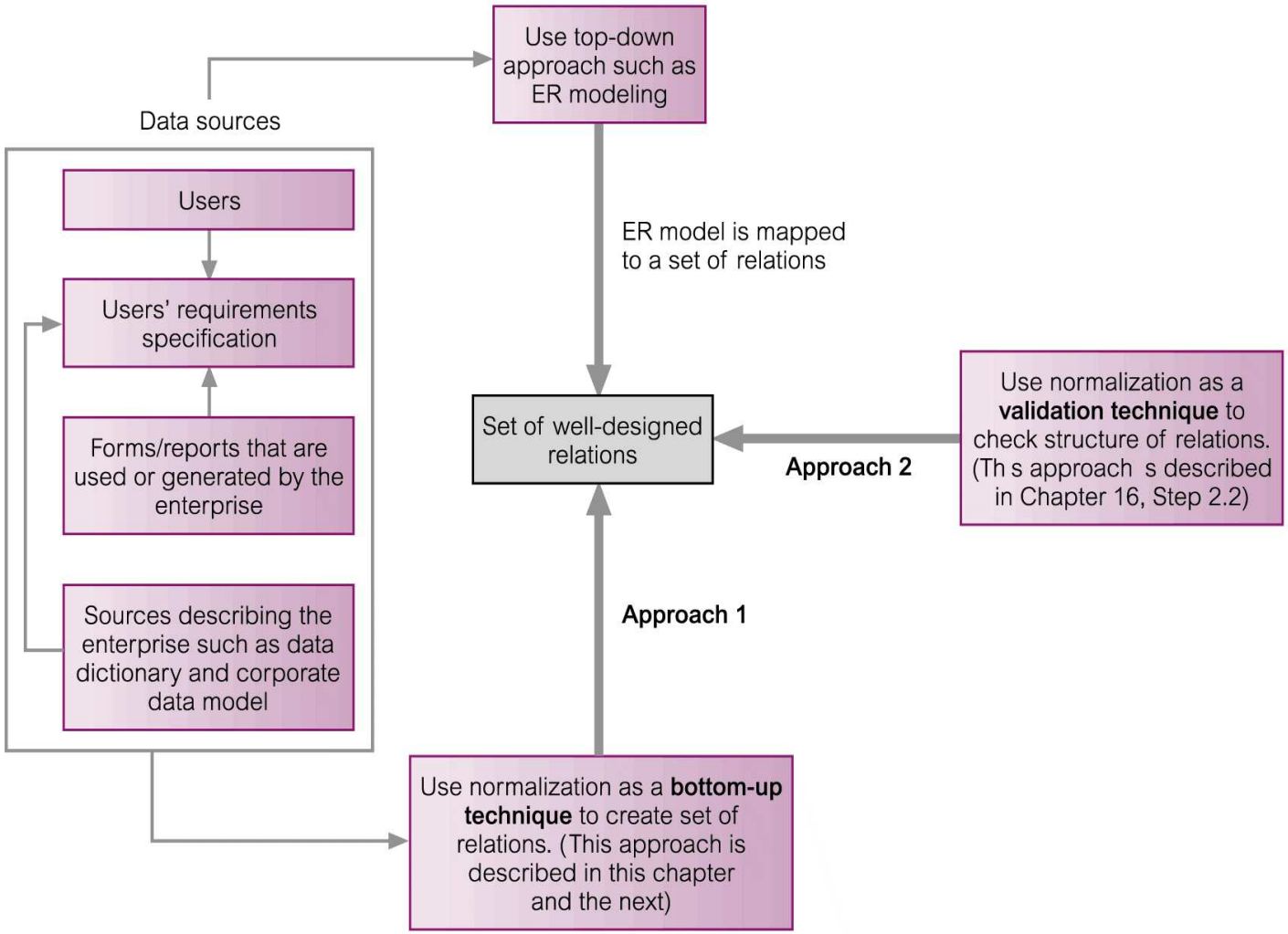
Normalization

- There is no guarantee that all anomalies are removed
- Normalization is a formal process for deciding which attributes should be grouped together in a relation so that all anomalies are removed.
- Normalization produces smaller and well-structured relations
- The goal of normalization:
  - Minimize data redundancy
  - Simplify the enforcement of referential integrity constrants
  - Make it easier to maintain data (insert, update and delete)
  - Provide a better design that is an improved representation of the real world and a stronger basis for future growth
- Normalization makes no assumptions about how data will be used in displays, queries, or reports.
- Normalization defines rules of business not data usage.
- Normalization places no constraints on how data can or should be physically stored.

How Normalization Supports Database Design

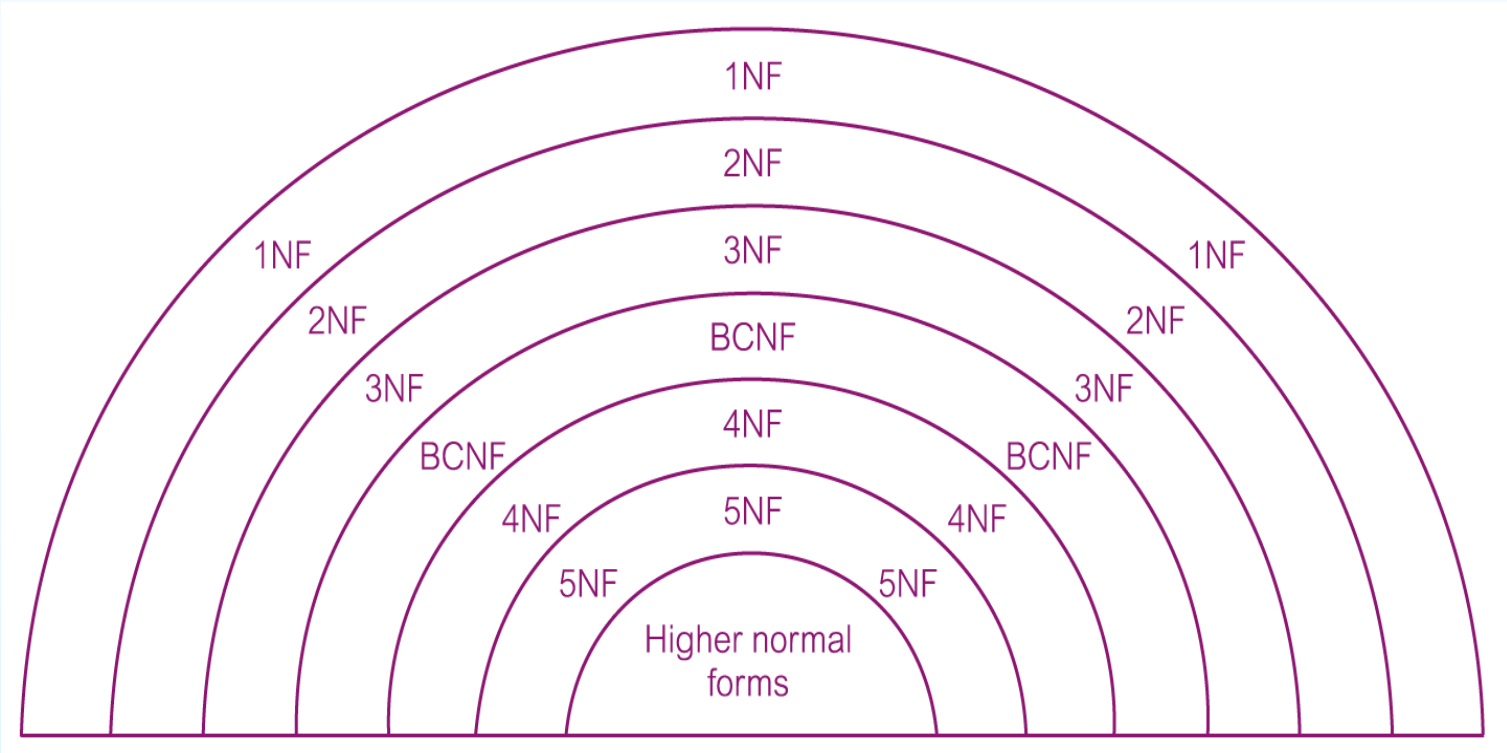
- Two main approaches for using normalization:
  - Normalization can be used as a bottom-up standalone database design technique
  - Normalization can be used as a validation technique to check the structure of relations, which may have been created using a top-down approach.
- It's possible to design a database based on the information taken directly from other data sources such as forms and reports.
- Same data source can be used for both approaches.
- Using normalization as a bottom-up technique is often limited by the level of detail that the designer is reasonably expected to manage

How Normalization Supports Database Design



The Process of Normalization

As normalization proceeds, the relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies.



The Process of Normalization