The Normalization Process

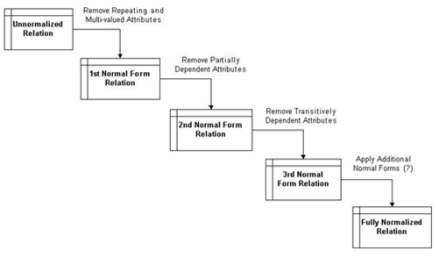
Chapter 6

Normalization

- ***Normalization***: A technique for producing a set of relations with desirable properties, given the data requirements of an enterprise
  - A bottom-up approach to database design
  - First developed by E.F. Codd around 1972, with 3 normal forms
  - Additional normal forms subsequently developed by Boyce-Codd and Fagin
  - Primarily intended for relations (tables) that are used by business transactions (order processing, payroll, etc.).
    - However, some of the principles apply to databases for other uses.

The Normalization Process

F06-01: The Normalization Process



Advantages of Normalization

- Greater overall database organization
- Reduced redundant data
- Better data integrity
- More flexible design
- Easier security management
- Removal of data anomalies

Data Anomalies

- ***Insertion Anomaly***: Insert operation blocked by an artificial dependency (e.g. cannot insert a new project without an employee to assign to it)
- ***Deletion Anomaly***: Delete operation destroys unintended information (e.g. delete of last employee on project destroys project information)

Data Anomalies

- ***Modification (Update) Anomaly***: Changing a single data value requires changes to multiple tuples (e.g. changing a project due date requires a change to the record of every person assigned to the project)
- A primary purpose of normalization is to remove these anomalies

Unnormalized View Anomalies

Employee ID: 172 **Employee Name**: Werdna J. Leppo **Mgr ID**: 090 **Mgr Name**: Lisa Fong **Week Begin**: 12/13/2015 **Week End**: 12/19/2015

Project ID	Project Name	Task ID	Task Name	Sun	Mon	Tue	Wed	Thu	Fri	Sat
127A	Order Entry Rewrite	T0001	Data Requirements		2	1				
127A	Order Entry Rewrite	T0021	Conceptual Model		1	4	1			
187B	Marketing Portal	T0100	Dashboard Layout		3	1	4	1		
187B	Marketing Portal	T0125	Star Schema Model				1	5		
187B	Marketing Portal	T0225	ETL Specification						6	

- Insert Anomalies:
  - A new project cannot be added unless there is an employee on the project with at least one task assigned to them.
  - A new task cannot be added unless there is an employee on the project to which the task can be assigned.

Unnormalized View Anomalies

Employee ID: 172 **Employee Name**: Werdna J. Leppo **Mgr ID**: 090 **Mgr Name**: Lisa Fong **Week Begin**: 12/13/2015 **Week End**: 12/19/2015

Project ID	Project Name	Task ID	Task Name	Sun	Mon	Tue	Wed	Thu	Fri	Sat
127A	Order Entry Rewrite	T0001	Data Requirements		2	1				
127A	Order Entry Rewrite	T0021	Conceptual Model		1	4	1			
187B	Marketing Portal	T0100	Dashboard Layout		3	1	4	1		
187B	Marketing Portal	T0125	Star Schema Model				1	5		
187B	Marketing Portal	T0225	ETL Specification						6	

- Delete Anomalies:
  - If we delete the timecard for the last employee working on a project, we lose any record of the project.
  - If we delete the timecard for the last employee on a project task, we lost any record of that task for that project.

Unnormalized View Anomalies

Employee ID: 172 **Employee Name**: Werdna J. Leppo **Mgr ID**: 090 **Mgr Name**: Lisa Fong **Week Begin**: 12/13/2015 **Week End**: 12/19/2015

Project ID	Project Name	Task ID	Task Name	Sun	Mon	Tue	Wed	Thu	Fri	Sat
127A	Order Entry Rewrite	T0001	Data Requirements		2	1				
127A	Order Entry Rewrite	T0021	Conceptual Model		1	4	1			
187B	Marketing Portal	T0100	Dashboard Layout		3	1	4	1		
187B	Marketing Portal	T0125	Star Schema Model				1	5		
187B	Marketing Portal	T0225	ETL Specification						6	

- Modification Anomalies:
  - If we change an employee name, we may have to update multiple timecards (depending on when the change is to be effective).
  - If we change a manager's name, we will have to apply to at least every current timecard for every employee managed by the manager.
  - If we change a project name, we must update every row that references that project.
  - If we change a task name, we must update every row that references that project task.

Disadvantages of Normalization

- Potentially slower performance
- More joins required (a bit more difficult to program)
  - Some view "more tables" as "more complicated", but logically, normalized data is simpler

Functional Dependency

- ***Functional Dependency***: "B" is functionally dependent on "A" if for every value of "A", there is exactly one value of "B"
  - Mathematically, we say that "A" ***determines*** "B"
  - Physically, we might say that "A" is a unique identifier for "B"

Functional Dependency

- ***Full Functional Dependency***: "B" is functionally dependent on "A" but not on any subset of "A"
- ***Transitive Dependency***: If "B" and "C" are both dependent on "A", but "C" is also dependent on "B", we say that "C" is ***transitively dependent*** on "B"

Normalization Process

- Formal technique for analyzing relations based on their primary key and functional dependencies
- Often executed as a number of steps, each step corresponding to a specific normal form

Normalization: Choosing a Unique Identifier

- Normalization requires that we choose a *unique identifier* for each relation
- *Unique Identifier:* a collection of one or more attributes that uniquely identifies each occurrence (each *tuple*) of a relation
  - *Natural identifiers* have real-world meaning
  - *Surrogate (artificial) identifiers* are meaningless replacements for real-world identifers

Criteria for Choosing a Unique Identifier

- If there is only one candidate, choose it
- Choose the candidate least like to have its value changed
- Choose the simplest candidate
- Choose the shortest candidate
- Invent a unique identifer

First Normal Form (1NF)

- ***Unnormalized Relation***: a relation that contains repeating groups (or that has not been tested for 1NF)
- ***First Normal Form***: A relation where the intersection of each row and column contains only one value (i.e. a relation with no repeating groups of attributes and no multi-valued attributes)

Transformation to 1NF

- Assign a unique identifier to each relation
- Move repeating group to a new relation, copying the original unique identifier and adding to it so that it is unique.
- For a multi-valued attribute (a repeating group of only one attribute), the attribute itself may be added to the original primary key to achieve uniqueness.

Example: Unnormalized Relation

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, (product number, description, quantity, unit price, extended amount), total order amount

Example: 1NF

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, total order amount

INVOICE NUMBER, PRODUCT NUMBER, description, quantity, unit price, extended amount

Second Normal Form (2NF)

- ***Second Normal Form***: A relation in 1NF where every non-key attribute is fully functionally dependent on the entire key
  - A 1NF relation with a single attribute primary key is automatically in 2NF

Example: 2NF

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, total order amount

INVOICE NUMBER, PRODUCT NUMBER, quantity, sale price, extended amount

PRODUCT NUMBER, description, unit price

Third Normal Form (3NF)

- ***Third Normal Form***: a relation that is in 1NF and 2NF and in which no non-primary-key attributes are transitively dependent
  - Calculated attributes are transitively dependent since they are determined by other attributes

Transformation to 3NF

- Calculated attributes may simply be removed (document the algorithm or formula)
- Non-calculated attributes that are transitively dependent are placed in a relation (new or existing) where the primary key is their principal determinant

Example: 3NF

INVOICE NUMBER, customer number, terms, ship via, order date
INVOICE NUMBER, PRODUCT NUMBER, quantity, sale price
PRODUCT NUMBER, description, unit price
CUSTOMER NUMBER, name, address, city, state, zip, telephone

Reasonableness: The Zip Code Dilemma

- A 5-digit zip code **DOES NOT** uniquely determine a city, and may not always determine a state
  - Piedmont, CA shares 94620 with Oakland, CA
  - In the past, zip codes have crossed state lines
- A 9-digit zip code does seem to determine a single city and state.
  - However, the USPS does not guarantee alignment with political boundaries

The Zip Code Dilemma (Example)

- So, do we always move City and State to a Zip Code table? The reasonableness check is in the anomalies:
  - Insertion Anomaly: Do we care about a new city or state if we have no addresses in it?
  - Deletion Anomaly: Do we care about losing a city or state name when the last address we have for it is deleted?
  - Update Anomaly: How often does a City or State change its name?

Normalization Summary

In a Third Normal Form relation,

Every non-key attribute depends on the key, the whole key and nothing but the key,

So help me Codd

Boyce-Codd Normal Form

- A stronger version of third normal form
- Two Requirements:
  - Relation must be in third normal form
  - No determinants exist that are not either the primary key or a candidate key for the relation (i.e. a non-key attribute may not uniquely identify [determine] any other attribute)

Boyce-Codd NF Example

Table with Customer ID, Product Line and Support Specialist must be split because:

- Customer ID plus Product Line forms a unique identifier, and relation then passes third NF
- Support Specialists are restricted to certain Product Lines

Fourth Normal Form

- Combination of several multi-valued attributes in the same relation creates an implied dependency
- If removal of multi-valued dependencies in solving for first NF is done carefully, this problem never arises

Fourth Normal Form Example

Employee ID, Office Skill, Language Skill

- Combination of either Employee ID and Office Skill or Employee ID and Language Skill is unique
- Either unique identifier choice implies a relationship that does not exist (office skills do not determine language skills, nor vice versa)

Fifth Normal Form

- Deals with concept of *join dependency* that requires knowledge of relational calculus to understand
- Some authors have confused fourth and fifth normal forms
- No clear evidence that join dependencies have practical value in business applications

Domain-Key Normal Form

- Introduced by R. Fagin in a 1981 research paper
- Step-by-step procedure or rules were never published
- Designers have no indication of when DKNF has been achieved
- Not in widespread use

Denormalization

- While frowned upon in modern systems, techniques include:
  - Recombining relations that were split to satisfy normalization rules
  - Storing redundant data in tables
  - Storing summarized data in tables

Beyond Third Normal Form

- We will ignore Fourth, Fifth and Boyce-Codd Normal Form because:
  - The anomalies they solve are extremely rare in business data
  - The concepts apply better to an advanced course rather than an introductory one

Class Exercise

Class Exercise Instructions

1. Normalize each of the user views into 3rd Normal Form relations. NOTE: OK to add attributes to make keys simpler.
2. List attributes for each normalized relation with primary keys in all capital letters.
3. Draw a preliminary ERD that shows all the normalized relations.

Class Exercise - User Views

Student Report:

ID	Name	Mailing Address		Home Phone
4567	Helen Wheels	127 Essex Drive	Hayward CA 94545	510-599-2859
4973	Barry Bookworm	P.O. B ox 45	Oakland CA 94601	510-486-9403
6758	Carla Coed	South Hall #23	Berkeley CA 94623	510-976-DORM

Class Exercise - User Views

Instructor Report:

ID	Name	Home Address	Home Phone	Office Phone	Courses
756	Werdna Leppo	12 Main St. Alameda CA 94501	510-976-CLAS	x-7463	X408 X422
795	Cora Coder	32767 Binary Way Abend CA 21304	510-101-1010	x-5382	X301 X302
801	Tillie Talker	123 Forms Rd. Paperwork CA 95684	510-888-BLAB	x-3547	X100

Class Exercise - User Views

Section Report:

Year: 1994 Semester: Spr Building: Evans Room: 70 Day(s): Tu Time(s): 7-10

Instructor: 756, Werdna Leppo Course: X408 Credits: 4

Student ID	Student Name	Grade
4567	Helen Wheels	A
5768	Carla Cond	B+

Year: 1994 Semester: Spr Building: SFO Room: 70 Day(s): We Time(s): 7-10

Instructor: 756, Werdna Leppo Course: X408 Credits: 4

Student ID	Student Name	Grade
4973	Barry Bookworm	B+

Year: 1994 Semester: Spr Building: Evans Room: 70 Day(s): M, Fr Time(s): 7-9

Instructor: 801, Tillie Talker Course: X100 Credits: 4

Student ID	Student Name	Grade

Class Exercise - Solution

COURSE:

COURSE NUMBER, course title, course description, course number credits

INSTRUCTOR:

INSTRUCTOR ID, instructor name, instructor home address, instructor home phone, instructor office phone

COURSE-SECTION:

EDP NUMBER, section year, section semester, course number, section building, section room, section meeting day, section meeting time, instructor id

Class Exercise - Solution

STUDENT:

STUDENT ID, student name, student address, student phone

STUDENT-SECTION:

STUDENT ID, EDP NUMBER, grade

COURSE - PREREQUISITE COURSE:

COURSE NUMBER, PREREQUISITE COURSE NUMBER

COURSE - INSTRUCTOR QUALIFIED:

INSTRUCTOR ID, COURSE NUMBER