Normalization of Database Tables
ISM 602
Dr. Hamid Nemati

Objectives
- The idea of Dependencies of Attributes
- Normalization and Database Design
- Understand concepts of normalization (Higher-Level Normal Forms)
- Learn how to normalize tables
- Understand normalization and database design issues
- Denomalization

Functional Dependency
- A Functional Dependency: Is A Relationship Between Or Among Attributes Such That The Values Of One Attribute Depend On, Or Are Determined By, The Values Of The Other Attribute(s).
- Partial Dependency: Is A Relationship Between Attributes Such That The Values Of One Attribute Is Dependent On, Or Determined By, The Values Of Another Attribute Which Is Part Of The Composite Key.
- Partial Dependencies Are Not Good Due To duplication Of Data And Update Anomalies;

Examples of Functional Dependencies:
- If we know an ISBN, then we know the Book Title and the author(s)
- ISBN → Book Title
- ISBN → Author(s)
- If we know the VIN, then we know who is the Auto owner
- VIN → Auto_Owner
- If we know Student-ID (SID), then we can uniquely determine his/her Name
- SID → S_Name

Transitive Dependencies
- Is A Relationship Between Attributes Such That The Values Of One Attribute Is Dependent On, Or Determined By, The Values Of Another Attribute Which Is Not A Part Of The Key.
- Exist when a nonkey attribute value is functionally dependent upon another nonkey value in the record. For example:
  - EMPLOYEE_ID → JOB_CATEGORY
  - JOB_CATEGORY → HOURLY_RATE
- An employee data table that includes the “hourly pay rate” would require searching every employee record to properly update an hourly rate for a particular job category.

So Now what is Normalization?
- GOLDEN RULE OF NORMALIZATION: Enter The Minimum Data Necessary, Avoiding Duplicate Entry Of Information, With Minimum Risks To Data Integrity.
- Goals Of Normalization:
  - Eliminate Redundancies Caused By:
    - Fields Repeated Within A File
    - Fields Not Directly Describing The Key Entity
    - Fields Derived From Other Fields
  - Avoid Anomalies In Updating (Adding, Editing, Deleting)
  - Represent Accurately The Items Being Modeled
  - Simplify Maintenance And Retrieval Of Info
Normalization is a process for assigning attributes to entities. It reduces data redundancies and helps eliminate the data anomalies. Normalization works through a series of stages called normal forms:
- First normal form (1NF)
- Second normal form (2NF)
- Third normal form (3NF)
- The highest level of normalization is not always desirable.

The attribute values in a relational table should be functionally dependent (FD) on the primary key value. A relationship is functionally dependent when one attribute value implies or determines the attribute value for the other attribute.

EM_SS_NUM → EM_NAME

Corollaries
- Corollary 1: No repeating groups allowed in relational tables.
- Corollary 2: A relational table should not have attributes involved in a transitive dependency relationship with the primary key.

The attribute values in a relational table should be functionally dependent (FD) on the primary key value.

- A relationship is functionally dependent when one attribute value implies or determines the attribute value for the other attribute.
- EM_SS_NUM → EM_NAME

Corollaries
- Corollary 1: No repeating groups allowed in relational tables.
- Corollary 2: A relational table should not have attributes involved in a transitive dependency relationship with the primary key.

The attribute values in a relational table should be functionally dependent (FD) on the primary key value.

- A relationship is functionally dependent when one attribute value implies or determines the attribute value for the other attribute.
- EM_SS_NUM → EM_NAME

Corollaries
- Corollary 1: No repeating groups allowed in relational tables.
- Corollary 2: A relational table should not have attributes involved in a transitive dependency relationship with the primary key.

Facilitates data integration.
- Reduces data redundancy.
- Provides a robust architecture for retrieving and maintaining data.
- Compliments data modeling.
- Reduces the chances of data anomalies occurring.

The Need for Normalization
- Case of a Construction Company
  - Building project -- Project number, Name, Employees assigned to the project.
  - Employee -- Employee number, Name, Job classification
  - The company charges its clients by billing the hours spent on each project. The hourly billing rate is dependent on the employee’s position.

Problems with the Table 5.1
- The project number is intended to be a primary key, but it contains nulls.
- The table displays data redundancies.
- The table entries invite data inconsistencies.
- The data redundancies yield the following anomalies:
  - Update anomalies.
  - Addition anomalies.
  - Deletion anomalies.

Deletion Anomaly
- Occurs when the removal of a record results in a loss of important information about an entity.
- Example:
  - All the information about a customer is contained in an order file. If the order is canceled, all the customer information could be lost when the order record is deleted
- Solution:
  - Create two tables—one table contains order information and the other table contains customer information.
Update Anomaly
- Occurs when a change of a single attribute in one record requires changes in multiple records
- Example:
  - A staff person changes their telephone number and every potential customer that person ever worked with has to have the corrected number inserted.
- Solution:
  - Put the employees telephone number in one location--as an attribute in the employee table.

Insertion Anomaly
- Occurs when there does not appear to be any reasonable place to assign attribute values to records in the database. Probably have overlooked a critical entity.
- Example:
  - Adding new attributes or entire records when they are not needed. Where do you place information on new Evaluator’s? Do you create a dummy Lead.
- Solution:
  - Create a new table with a primary key that contains the relevant or functional dependent attributes.

Database Tables and Normalization
- Conversion to First Normal Form
  - A relational table must not contain repeating groups.
  - Repeating groups can be eliminated by adding the appropriate entry in at least the primary key column(s). (See Database Table 5.3)

DB Table 5.2 The Evergreen Data

- 1NF Definition
  - The term first normal form (1NF) describes the tabular format in which:
    - All the key attributes are defined.
    - There are no repeating groups in the table.
    - All attributes are dependent on the primary key.

Database Tables and Normalization
- Conversion to Second Normal Form
  - Starting with the 1NF format, the database can be converted into the 2NF format by:
    - Writing each key component on a separate line, and then writing the original key on the last line and writing the dependent attributes after each new key.
  - PROJECT (PROJ_NUM, PROJ_NAME)
  - EMPLOYEE (EMP_NUM, EMP_NAME, JOB_CLASS, CHG_HOUR)
  - ASSIGN (PROJ_NUM, EMP_NUM, HOURS)

Dependency Diagram
- The arrows above the entity indicate that the entity’s attributes are dependent on the combination of PROJ_NUM and EMP_NUM.
- The arrows below the dependency diagram indicate less desirable dependencies based on only a part of the primary key -- partial dependencies.
Introduction to Normalization of Database Tables

Database Tables and Normalization

- **2NF Definition**
  - A table is in 2NF if:
    - It is in 1NF and
    - It includes no partial dependencies; that is, no attribute is dependent on only a portion of the primary key.
  - Note:
    - It is still possible for a table in 2NF to exhibit transitive dependency; that is, one or more attributes may be functionally dependent on nonkey attributes.
  - See figure 5.2 page 290.

Conversion to Third Normal Form

- Create a separate table with attributes in a transitive functional dependence relationship.

<table>
<thead>
<tr>
<th>PROJECT (PROJ_NUM, PROJ_NAME)</th>
<th>ASSIGN (PROJ_NUM, EMP_NUM, HOURS)</th>
<th>EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL, JOB_CODE)</th>
<th>JOB (JOB_CODE, JOB_DESCRIPTION, JOB_CHG_HOUR)</th>
</tr>
</thead>
</table>

Normalization and Database Design

- **3NF Definition**
  - A table is in 3NF if:
    - It is in 2NF and
    - It contains no transitive dependencies.

Normalization and Database Design

- **Database Design and Normalization Example:** (Construction Company)
  - Summary of Operations:
    - The company manages many projects.
    - Each project requires the services of many employees.
    - An employee may be assigned to several different projects.
    - Some employees are not assigned to a project and perform duties not specifically related to a project. Some employees are part of a labor pool, to be shared by all project teams.
    - Each employee has a (single) primary job classification. This job classification determines the hourly billing rate.
    - Many employees can have the same job classification.

Normalization and Database Design

- **Two Initial Entities:**
  - PROJECT (PROJ_NUM, PROJ_NAME)
  - EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL, JOB_DESCRIPTION, JOB_CHG_HOUR)

- **Three Entities After Transitive Dependency Removed**
  - PROJECT (PROJ_NUM, PROJ_NAME)
  - EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL, JOB_CODE)
  - JOB (JOB_CODE, JOB_DESCRIPTION, JOB_CHG_HOUR)
Normalization and Database Design

- Creation of the Composite Entity ASSIGN

Figure 5.8 The Modified E-R Diagram for a Contacting Company

Figure 5.9 The Final (implementable) E-R Diagram for the Contracting Company

- Attribute ASSIGN_HOUR is assigned to the composite entity ASSIGN.
- "Manages" relationship is created between EMPLOYEE and PROJECT.

EMPLOYEE (EMP_NUM, EMP_LNAME, EMP_FNAME, EMP_INITIAL, EMP_HIREDATE, JOB_CODE)
JOB (JOB_CODE, JOB_DESCRIPTION, JOB_CHG_HOUR)
ASSIGN (ASSIGN_NUM, ASSIGN_DATE, PROJ_NUM, EMP_NUM, ASSIGN_HOURS)

Figure 5.10 The Relational Schema for the Contracting Company

Summary
A Journey of Normalization

- First Normal Form (1NF) - Remove "Repeating Groups"
- Second Normal Form (2NF) - Remove "Partial Functional Dependency"
- Third Normal Form (3NF) - Remove "Transitive Functional Dependency"
- Higher order Normal Forms - Remove "All Remaining Functional Dependency"

Denormalization

- Denormalization is only one of many database design goals.
- Normalized (decomposed) tables require additional processing, reducing system speed.
- Normalization purity is often difficult to sustain in the modern database environment. The conflict between design efficiency, information requirements, and processing speed are often resolved through compromises that include denormalization.