Chapter 8

Designing Your Web Database

Advantages of Databases

- Faster
- Easily queried
- Concurrency
- Random access
  - As opposed to sequential access
- Built-in privilege system
- RDBMS: Relational Database Management System
  - There are also database *engines*, which may be separate from the DBMS

Basic DB Concepts

- *Tables*
  - Also called *relations*
- Tables are divided into *columns*
  - Also called *fields* or *attributes*
  - Each column has a data type
- Tables are also divided into *rows*
  - Also called *records* or *tuples*
  - Each row in a table has the same attributes
- *Schema*
  - All the tables for the database
  - Like a blueprint
  - Can be expressed in text or in entity relationship diagrams

Keys

- Each row needs to have a unique identifier (primary key)
  - Called a *key*
  - Avoids duplication, confusion
  - Any column or combination of columns can be the key
- Appropriate choice of keys is vital
  - Must be both unique and defined for all records in the table
    - Names
    - Social security numbers
  - Poor choice of key numbers can bite you years later
    - Duplicate SSNs!
  - Best practice is to assign an arbitrary ID number to every record
    - Many RDBMSs support this automatically
    - L numbers
    - ISBNs
- Can have compound keys, where a combination of columns is the key
  - Sometimes useful, but less common than single-column keys
- Foreign key

  key in one field in a table that is a primary key in another table

Designing Schemas

- Generally, each real-world object gets its own table
  - Like a class in OOP
- Make sure all the data needed to answer interesting questions is stored

Designing Schemas (cont'd)

- Avoid null values
  - Wastes space
  - Difficult to query
- Avoid redundancy!
  - Wastes space
  - Makes consistency difficult to maintain
    - What if the same data is stored in two places that contradict each other? BAD!
  - Avoid redundancy!
- Reference tables in other tables to minimize redundancy and null values

Relationships

- Relationships can exist between two tables
  - Redundancy or null values probably mean you need two separate tables
- Rows in one table can reference rows in another table
  - UserID can be the primary key column in the Users table, and also a column in the Orders table
    - Ties the user to the order!
  - Another reason having arbitrary ID numbers is helpful
- Three kinds of relationships
  - One-to-one
  - One-to-many
  - Many-to-many

One-to-One Relationships

- Each row in table A is tied to exactly one row in table B, and vice versa
  - Each table has a column containing the other table's primary key
- Example: each customer has one address, each address has one customer
  - Any one-to-one relationship could be combined into a single table
    - May not want to for performance reasons

One-to-Many Relationships

- Each row in table A may be referenced by many rows in table B
  - No reference in table A to table B
- Each row in table B only references one row in table A
  - Table B has a column containing the primary key of table A
- Example:
  - Each class has one room
  - Each room has many classes

Many-to-Many Relationships

- Each row in table A can be tied to many rows in table B, and vice versa
- Need a third table, called a *bridge* table
  - Each row contains one key from table A and one key from table B
  - Bridge table's key is the combination of table A and table B's keys
- Example:
  - Each book can have many authors
  - Each author can write many books
  - Bridge table stores each combination

Designing your Web Database

- Think about the Real-World Objects You are modeling
  - Each class of real-world objects needs its own table (table stores objects with the same look)
  - See figure 8.3 p. 212
- Avoid Storing Redundant Data
  - Wastes space
  - Can lead to update anomalies (situations in which you change the database and end up with inconsistent data)

3 Types of Anomalies to Avoid

- Modification
  - Occur when you try to modify the database like forget to change each time when it is in more than one place
- Insertion
  - Occurs when data is being inserted and problem if it exists somewhere else and it not get changed
- Deletion
  - Occurs when you remove a row from the database – might delete an address and not have for the next time you need it

More Rules on Designing a Web Database

- Use Atomic Column Values
  - You can only store one thing in each attribute in each row
  - Use multiple tables rather than a table within a table
- Choose Sensible Keys
  - Must be unique
- Think about what you want to ask the database
  - Be sure the database contains all the required data to answer these questions.

More Rules on Designing a Web Database

- Avoid Designs with Many Empty Attributes (null value)
  - Wastes storage
  - Causes problems when working out totals and other functions on numerical column
  - Can be avoided by using an alternate design
- Summary of Table Types
  - Two types of tables
    - Simple tables that describe a real-world object
    - Linking tables that describe a many-to-many relationship between two real objects

Normalization

- Complex formal topic
  - 1NF, 2NF, 3NF, 4NF, 5NF, 6NF, BCNF, DKNF
  - Database admins need to understand these
    - I'm not teaching you to be DBAs
- Basic idea is to reduce redundancy
  - Improve storage efficiency
- Sometimes you want to denormalize, increase redundancy!
  - Usually for applications where speed is vital, mostly read, not often written
  - Reduces number of lookups necessary to complete query

Web Database Architecture

- P. 217 figure 8.8
- P. 217 figure 8.9 and steps below

Stages in Web Database Transaction

- User's web browser issues an HTTP request for a given web page
- Web server receives the request, retrieves the file, and passes it to the PHP engine for processing
- PHP engine begins parsing the script. Inside the script is a command to connect to the database and execute a query. PHP opens a connection to the MySQL server and sends the query
- MySQL server receives the database query, processes it, and sends the results back to the PHP engine
- PHP engine finishes running the script (normally formats the results). Returns resulting HTML to the web server
- Web server passes the HTML back to the browser

Schema Example

- Rules
  - Underline the primary key
  - Italicize the foreign key
- Look ahead at chapter 9
  - P. 219
  - 5 tables