2

Introduction to Relational Databases

Large corporate computing shops have been using complex and expensive database products for years. These full-featured, heavily optimized software systems are the only way for a big organization to manage its volumes of corporate information.

Home computer users haven't traditionally needed database products at all. They house their data—addresses, to-do lists, etc.—on their systems in small files or in specialized off-the-shelf spreadsheet and phonebook applications.

A new category of computer users who fall between these two extremes has come into play. These persons maintain moderate-sized information sets required for small organizations, such as new businesses or nonprofit organizations. Alternatively, such users may be a geographically isolated part of a larger company. Or they be individuals interested in maintaining complex personal data, such as a list of songs from favorite bands that can be served up on a personal web page. If you are the kind of person who wants a database, who is willing to do some work to set one up, but who does not want to spend six figures on a product and a fleet of programmers to maintain it, this book is for you.

This book introduces you to the world of small-scale database development through one popular database product: MySQL. We start by introducing you to relational databases and MySQL. We then proceed to show you how to get up and running with MySQL and how to administer it. The rest of the book covers the use of MySQL to design, build and support the type of applications important to users like you.

What Is a Database?

A *database* is, simply put, a collection of data. An example of a non-electronic database is the public library. The library stores books, periodicals, and other documents. When you need to locate some data at the library, you search through the card catalog or the periodicals index, or maybe you even ask the librarian. Another similar example is the unsorted

pile of papers you might find on your desk. When you need to find something, you rifle through the stack until you find the scrap of paper you are looking for. This database works (or maybe it doesn't) because the size of the database is incredibly small. A stack of papers certainly would not work with a larger set of data, such as the collections in the library. In the library, without the card catalog, periodicals index, and librarian, the library would still be a database; it would just be an unusable database. A database therefore generally requires some sort of organization to be of value. Your pile of papers would be much more reliable if you had some sort of filing system (then maybe you would not have lost that phone number!). So, restating our definition, we will define a database as an *organized* collection of data.

The library and the stack of papers have many similarities. They are both databases of documents. It makes no sense, however, to combine them because your papers are only interesting to you and the library contains documents of general interest. Both databases have a specific purpose and they are organized according to that purpose. We will therefore amend our definition a bit further: a database is a collection of data that is *organized* and stored according to some *purpose*.

Traditional paper-based databases have many disadvantages. They require a tremendous amount of physical space. Libraries occupy entire buildings and searching a library is relatively slow. Anyone who has spent time in a library knows that it can consume a non-trivial amount of time to find the information you seek. Libraries are also tedious to maintain and an inordinate amount of time is spent keeping the catalogs and shelves consistent. Electronic storage of a database helps to address these issues.

MySQL is not a database, per se. It is computer software that enables a user to create, maintain, and manage electronic databases. This category of software is known as a Database Management System (DBMS). A DBMS acts as a broker between the physical database and the users of that database.

When you first began managing electronic information, you almost certainly used a flat file such as a spreadsheet file. The file system is the electronic version of the pile of papers on your desk. You likely came to the conclusion that this sort of ad hoc electronic database didn't meet your needs any more. A DBMS is the logical next step for your database needs, and MySQL is the first stepping stone into the world of relational database management systems.

What Is a Relational Database?

According to our definition, a database is an organized collection of data. A relational database organizes data into tables and represents relationships between those tables. These relationships allow you to combine data from multiple tables to provide different "views" of the data.It is probably easier to illustrate the concepts of tables and relationships than try to explain them. Table 1-1 is an example of a table that might appear in a book database.

Table 1-2: A Table of Books (continued)

ISBN	Title	Author
0-446-67424-9	L.A. Confidential	James Ellroy
0-201-54239-X	An Introduction to Database Systems	C.J. Date
0-87685-086-7	Post Office	Charles Bukowski
0-941423-38-7	The Man with the Golden Arm	Nelson Algren

Table 1-3 and Table 1-5 demonstrate two tables that might appear in an NBA database.

Table 1-4: A Table of NBA Teams (continued)

Team #	Name	Coach
1	Sacramento Kings	Rick Adelman
2	Minnesota Timberwolves	Flip Saunders
3	L.A. Lakers	Phil Jackson
4	Portland Trailblazers	Mike Dunleavy

Table 1-6: A Table of NBA Players (continued)

Name	Position	Team #
Vlade Divac	Center	1
Kevin Garnett	Forward	2
Kobe Bryant	Guard	3
Rasheed Wallace	Forward	4
Damon Stoudamire	Guard	4
Shaquille O'Neal	Center	3

We'll get into the specifics about tables later on, but you should note a few things about these examples. Each table has a name, several columns, and rows containing data for each of the columns. A relational database represents all of your data in tables just like this and provides you with retrieval operations that generate new tables from existing ones. As a result, the user sees the entire database in the form of tables.

Also note that the "Team #" column appears in both tables. It encodes a relationship between a player and a team. By linking the "Team #" columns you can determine that Vlade Divac plays for the Sacramento Kings. You could also figure out all the players on the Portland Trailblazers. This linking of tables is called a "relational join", or "join" for

short.A DBMS for a relational system is often called a Relational Database Management System (RDBMS). MySQL is an example of an RDBMS.

Where does SQL fit into all of this? We need some way to interact with the database. We need to define tables and retrieve, add, update, or delete data. SQL (Structured Query Language) is a computer language used to express database operations for data organized in a relational form.

> SQL is commonly pronounced "Sequel". MySQL is pronounced "MySequel".

SQL is the industry standard language that most database programmers speak, and it is used by most RDBMS packages. As the name indicates, MySQL is a SQL database engine. However, it only supports a subset of the current SQL standard, SQL2. We will discuss exactly how MySQL support for SQL differs from the standard in later chapters.

Applications and Databases

According to our definition, a database is an organized collection of data that serves some purpose. Simply having a DBMS is not sufficient to give your database purpose. How you use your data defines its purpose. Imagine a library where nobody ever reads the books. There would not be much point in storing and organizing all those books if they're never used. Now, imagine a library where you could not change or add to the collection. The utility of the library as a database would decrease over time since obsolete books could never be replaced and new books could never be added. In short, a library exists so that people may read the books and find the information they seek.

Databases exist so that people can interact with them. In the case of electronic databases, the interaction occurs not directly with the database, but instead indirectly through software applications. Before the emergence of the World Wide Web, databases typically were used by large corporations to support various business functions: accounting and financials, shipping and inventory control, manufacturing planning, human resources, and so on. The web and more complex home computing tasks have helped move the need for database applications outside the realm of the large corporation.

Databases and the Web

The area in which databases have experienced the most explosive growth—an area where MySQL excels—is in web application development. As the demand for more complex and robust web applications grows, so does the need for databases. A database backend can support many critical functions on the web. Virtually any web content can be driven by a database.

Consider the example of a catalog retailer who wants to publish on the web and accept orders online. If the contents of the catalog are entered directly into one or more HTML files, someone has to hand edit the files each time a new item is added to the catalog or a price is changed. If the catalog information is instead stored in a relational database, it becomes possible to publish real-time catalog updates simply by changing the product or price data in the database. It also becomes possible to integrate the online catalog with

existing electronic order-processing systems. Using a database to drive such a web site thus has obvious advantages for both the retailer and the customer.

Here's how a web page typically interacts with a database. The database is on your web server or another machine that your server can talk to (a good DBMS makes this kind of distributed responsibility easy). You put a form on one web page that the user fills in with a query or data to submit. When the user submits the form to your server, the server runs a program that you've written to extract the data from the form. These programs are most often written as CGI scripts and Java servlets, but they can also be implemented by embedding programming commands right inside the HTML page. We will look at all of these methods in this book.

Now your program knows what the user is asking for or wishes to add to the database. The program issues an SQL query or update, and the database magically takes care of the rest. Any results obtained from the database can be formatted by your program into a new HTML page to send back to the user.

Summary

MySQL is an SQL-based Relational Database Management System (RDMS). Relational databases are organized into tables and relationships between the tables. These tables are further broken up into columns and rows. Each row in a table represents one record of data. Each column represents one "piece" of data for each record. Some columns are used to represent a relationship between two tables, thus storing that relationship a another "piece" of data. SQL stands for **S**tructured **Q**uery **L**anguage. This is a standard language that is used to insert, retrieve, update and delete data from your relational database. It is also used to define the structure of your database tables.

That's all there is to it. The basic concepts behind relational databases are quite simple. Therein lies the power: these simple building blocks – tables, columns, rows, relationships, and SQL – can be combined to create powerful applications.

