Database Systems: The Complete Book

Hector Garcia-Molina
Jeffrey D. Ullman
Jennifer Widom

Department of Computer Science
Stanford University

An Alan R. Apt Book

Prentice Hall
Upper Saddle River, New Jersey 07458

About the Authors

JEFFREY D. ULLMAN is the Stanford W. Ascherman Professor of Computer Science at Stanford University. He is the author or co-author of 16 books, including *Elements of ML Programming* (Prentice Hall 1998). His research interests include data mining, information integration, and electronic education. He is a member of the National Academy of Engineering, and recipient of a Guggenheim Fellowship, the Karl V. Karlstrom Outstanding Educator Award, the SIGMOD Contributions Award, and the Knuth Prize.

JENNIFER WIDOM is Associate Professor of Computer Science and Electrical Engineering at Stanford University. Her research interests include query processing on data streams, data caching and replication, semistructured data and XML, and data warehousing. She is a former Guggenheim Fellow and has served on numerous program committees, advisory boards, and editorial boards.

HECTOR GARCIA-MOLINA is the L. Bosack and S. Lerner Professor of Computer Science and Electrical Engineering, and Chair of the Department of Computer Science at Stanford University. His research interests include digital libraries, information integration, and database application on the Internet. He was a recipient of the SIGMOD Innovations Award and is a member of PITAC (President's Information-Technology Advisory Council).

Table of Contents

Chapter 1

The Worlds of Database Systems

Databases today are essential to every business. They are used to maintain internal records, to present data to customers and clients on the World-Wide-Web, and to support many other commercial processes. Databases are likewise found at the core of many scientific investigations. They represent the data gathered by astronomers, by investigators of the human genome, and by biochemists exploring the medicinal properties of proteins, along with many other scientists.

The power of databases comes from a body of knowledge and technology that has developed over several decades and is embodied in specialized software called a *database management system*, or *DBMS*, or more colloquially a "database system." A DBMS is a powerful tool for creating and managing large amounts of data efficiently and allowing it to persist over long periods of time, safely. These systems are among the most complex types of software available. The capabilities that a DBMS provides the user are:

1. *Persistent storage.* Like a file system, a DBMS supports the storage of very large amounts of data that exists independently of any processes that are using the data. However, the DBMS goes far beyond the file system in providing flexibility, such as data structures that support efficient access to very large amounts of data.

2. *Programming interface.* A DBMS allows the user or an application program to access and modify data through a powerful query language. Again, the advantage of a DBMS over a file system is the flexibility to manipulate stored data in much more complex ways than the reading and writing of files.

3. *Transaction management.* A DBMS supports concurrent access to data, i.e., simultaneous access by many distinct processes (called "transac-

tions") at once. To avoid some of the undesirable consequences of simultaneous access, the DBMS supports *isolation*, the appearance that transactions execute one-at-a-time, and *atomicity*, the requirement that transactions execute either completely or not at all. A DBMS also supports *durability*, the ability to recover from failures or errors of many types.

1.1 The Evolution of Database Systems

What is a database? In essence a database is nothing more than a collection of information that exists over a long period of time, often many years. In common parlance, the term *database* refers to a collection of data that is managed by a DBMS. The DBMS is expected to:

1. Allow users to create new databases and specify their *schema* (logical structure of the data), using a specialized language called a *data-definition language.*
2. Give users the ability to *query* the data (a "query" is database lingo for a question about the data) and modify the data, using an appropriate language, often called a *query language* or *data-manipulation language.*
3. Support the storage of very large amounts of data — many gigabytes or more — over a long period of time, keeping it secure from accident or unauthorized use and allowing efficient access to the data for queries and database modifications.
4. Control access to data from many users at once, without allowing the actions of one user to affect other users and without allowing simultaneous accesses to corrupt the data accidentally.

1.1.1 Early Database Management Systems

The first commercial database management systems appeared in the late 1960's. These systems evolved from file systems, which provide some of item (3) above; file systems store data over a long period of time, and they allow the storage of large amounts of data. However, file systems do not generally guarantee that data cannot be lost if it is not backed up, and they don't support efficient access to data items whose location in a particular file is not known.

Further, file systems do not directly support item (2), a query language for the data in files. Their support for (1) — a schema for the data — is limited to the creation of directory structures for files. Finally, file systems do not satisfy (4). When they allow concurrent access to files by several users or processes, a file system generally will not prevent situations such as two users modifying the same file at about the same time, so the changes made by one user fail to appear in the file.

The first important applications of DBMS's were ones where data was composed of many small items, and many queries or modifications were made. Here are some of these applications.

Airline Reservations Systems

In this type of system, the items of data include:

1. Reservations by a single customer on a single flight, including such information as assigned seat or meal preference.
2. Information about flights — the airports they fly from and to, their departure and arrival times, or the aircraft flown, for example.
3. Information about ticket prices, requirements, and availability.

Typical queries ask for flights leaving around a certain time from one given city to another, what seats are available, and at what prices. Typical data modifications include the booking of a flight for a customer, assigning a seat, or indicating a meal preference. Many agents will be accessing parts of the data at any given time. The DBMS must allow such concurrent accesses, prevent problems such as two agents assigning the same seat simultaneously, and protect against loss of records if the system suddenly fails.

Banking Systems

Data items include names and addresses of customers, accounts, loans, and their balances, and the connection between customers and their accounts and loans, e.g., who has signature authority over which accounts. Queries for account balances are common, but far more common are modifications representing a single payment from, or deposit to, an account.

As with the airline reservation system, we expect that many tellers and customers (through ATM machines or the Web) will be querying and modifying the bank's data at once. It is vital that simultaneous accesses to an account not cause the effect of a transaction to be lost. Failures cannot be tolerated. For example, once the money has been ejected from an ATM machine, the bank must record the debit, even if the power immediately fails. On the other hand, it is not permissible for the bank to record the debit and then not deliver the money if the power fails. The proper way to handle this operation is far from obvious and can be regarded as one of the significant achievements in DBMS architecture.

Corporate Records

Many early applications concerned corporate records, such as a record of each sale, information about accounts payable and receivable, or information about employees — their names, addresses, salary, benefit options, tax status, and

S

V

W

X

Y

Z