Database Systems for Management Third edition

James F. Courtney

David B. Paradice

Kristen L. Brewer

Julia C. Graham

For any questions about this text, please email: drexel@uga.edu

The Global Text Project is funded by the Jacobs Foundation, Zurich, Switzerland

This book is licensed under a Creative Commons Attribution 3.0 License

DATABASE SYSTEMS FOR MANAGEMENT, THIRD EDITION
James F. Courtney, David B. Paradice
Kristen L. Brewer, and Julia C. Graham

PART ONE

Database Fundamentals

CHAPTER CONTENTS

CHAPTER 1

Introduction to Database Systems

Organizations capture and process data for a variety of reasons. Accounting data are used to measure the financial health and vitality of the enterprise: It is useful to internal corporate management in keeping track of how the firm is performing; it is useful to people outside the firm, such as investors interested in the firm's stock and bankers to whom the firm has applied for a loan. Records on a host of other organizational entities, including employees, inventories, customers, and suppliers, are required as well.

One of the main uses of data relates to making decisions. A manager must decide, for example, on the price of the firm's product or on how many units of the product to produce. An investor must decide whether to invest in a firm; a banker must decide whether to make a loan. The Federal Reserve Board must decide whether to increase or decrease interest rates, or hold them stable. In all of these cases, it is desirable to have reliable information on which to base the decision. Information, and the data on which it is based, are valuable organizational assets that warrant careful management.

DATA AS AN ORGANIZATIONAL ASSET

Major functional areas in organizations have evolved from recognition that the function being managed is of vital importance. Financial and accounting departments, for example, manage predominantly monetary assets of an organization. Few would argue that these assets are unimportant to the organization. Marketing departments handle the advertising for new and existing products, carry out research studies, and distribute products. These activities are obviously important. Production, personnel, and other departments also specialize in efficient and effective management of the assets in these areas. Similarly, organizational data and information must be regarded as an asset of vital importance to the organization. A few examples should make clear how important data and information can be.

Consider a company that produces a newsmagazine with national distribution. It can collect data about its subscribers that can be used in many ways to gain an advantage in the marketplace; it can, for example, use data about personal interests to help select the stories to be printed or to entice organizations to purchase advertising space.

Or imagine an organization that, thanks to a secret formula, manufactures the most popular product in a highly competitive market. In fact, the company maintains the results of hundreds of consumer tests of variations of the formula in a database. These can be analyzed whenever consumer preferences appear to be changing, giving the company the appearance of an uncanny ability to anticipate consumer preferences. Were this data to fall into the hands of competitors, all advantage in the marketplace would be lost.

Or consider such routine information as salary or production schedule data. Salary data in the hands of a competing organization could result in the loss of top employees through offers of higher compensation. Production data could allow a competitor to anticipate an organization's strategy in the near future. Each of these actions could have devastating results on an organization.

These examples are intended to emphasize the point that data are indeed a valuable asset to an organization. You can doubtlessly think of other examples. As with any valuable asset, the organization should have persons and procedures to effectively manage the asset. The management of organizational data is the main topic of concern in this book.

MANAGING ORGANIZATIONAL DATA

Organizations of all sizes now use computers to perform the data processing functions required to provide information for management decision-making. Experience with business applications of data processing has shown that the data itself is a valuable organizational resource that must be carefully managed. The task of managing organizational data and data processing functions is often called **data management** or **information management.**

The data resources of an organization are usually stored in **databases,** which are highly integrated sets of files shared by users throughout the organization. Because it is impractical to store all the data for even relatively small organizations in one database, organizations design **database systems,** which are integrated collections of databases along with users who share the databases, personnel who design and manage databases, techniques for designing and managing databases, and computer systems to support them. These are rather simple definitions of databases and database systems, but they will suffice for now. These definitions are expanded throughout the book.

This book provides an introduction to management information systems with an emphasis on special technologies and techniques for designing and managing organizational database systems. After studying the material in this book, you should have mastered at least the basic principles underlying information systems, the technology used to manage databases and the techniques used to design and implement effective database systems.

When reading this chapter, think about what is meant by data as an organizational resource and about how managers use data. Keep in mind the idea that **the main goal of database systems is to provide managers with information so that they can make effective decisions about how to run the organization.** The basic theme of this book is that reliable information and accurate data are required for effective managerial decisions. The object of the book is to demonstrate how to design and manage effective database systems.

The users of database systems are the customers of the information systems department. If these customers do not use the product (the database system), then the system has not been designed effectively. To emphasize the viewpoint of database system users, each chapter includes a section entitled "The User's View" in which we discuss the concerns of users as related to that chapter's topics.

DATA VERSUS INFORMATION

Information and data are not the same thing. Information is sometimes defined as processed data, but simply because data has been processed does not mean that it is useful to an organization. The average price of corn in Iowa may be meaningful to thousands of farmers but useless to an automobile manufacturer. For the purposes of this book, **data** is defined simply as numbers, words, names, and other symbols that can be stored in a computer system. **Information** is simply useful data. If a single data value (such as the price of a competitor's product) is useful, then it is information. Several data values may be manipulated or processed to create information (such as in computing the average price of several competing products). If the resulting data value is not useful, then it is not information, no matter how much it has been processed.

ELEMENTS OF DATABASE SYSTEMS

The principal elements in a database environment are people, computer hardware, computer software, databases, and techniques for planning, designing, implementing, and managing databases. Each of these elements is examined in more detail in the next chapter. A brief overview is given here.

People

The people involved with database systems can be divided into two groups: those who use information provided by the system and those who design, develop, and manage the system itself. If a database system is to be of value to the organization, it is imperative that it be designed to support the needs of information users. This point is emphasized repeatedly throughout the book; it cannot be overemphasized. The design of user-oriented database systems is considered in detail in the chapters on system design, implementation, and administration.

Those people responsible for the design, development, and administration of database systems are often referred to as **database administrators.** The position of database administrator is highly challenging and should be a high-level managerial position. Database administrators must possess well-developed skills in both technical and managerial aspects of information management, because they are the primary liaisons between the community of users and the systems development staff.

The personnel responsible for developing database systems are information analysts or database analysts. **Information analysts** work closely with users of information to carefully define information requirements and to structure these requirements into a logical form. **Database analysts** use database technology to design systems satisfying those requirements. This book is aimed primarily at those who seek to develop successful database systems and who may ultimately become database administrators, but it is of interest to users of database systems as well. Thus the book is concerned with both technical and managerial aspects of information management. Database administration is discussed in detail in Chapters 10 and 11.

Computer Hardware and Software

Computer hardware and software are two different elements of a database environment, but they are discussed together in this chapter. Hardware and software form the technological foundation for database systems. This book contains a limited discussion of the hardware for database management; hardware is an extensive and complex topic requiring special treatment far beyond the scope of the book.

Software for information management can be separated into two groups: applications software and database management systems. **Database management systems (DBMS)** are commercial software packages designed to provide the basis for building database systems. Many such packages are on the market today, and thousands of organizations use them. These packages are part of the technology of information management and usually depend heavily on the operating system for many of their basic functions.

Organizations acquire database management systems and develop or acquire applications software to satisfy their particular database processing requirements. You are undoubtedly familiar with the way ordinary programming languages such as Visual Basic (VB) describe data files. As is illustrated shortly, this approach may not provide ready access to information. The database approach is intended to overcome these problems.

Databases

As shown in Figures 1-1 and 1-2, database management systems are designed to allow various programs to share the data they have in common. In Figure 1-1, Joe has his files and his data, and Sue has her files and her data. Joe uses the manufacturing budget in his program, but Sue does not. Sue uses salary data, but Joe does not. In fact, Joe is not even allowed to see or access the salary data; it is privileged information. Both programmers use product prices.

In Figure 1-2 the programmers have merged their data into an integrated database ("our" data). The integrated database is defined in the **schema,** which describes all the data items and relationships in the entire database. Note, however, that each programmer still has some data (salary and manufacturing) that is inaccessible to the other, even though it is stored in the integrated database.

Each programmer's individual view of the database is defined in a **subschema.** The use of a database management system to develop an integrated database is discussed later in this chapter. For now, note that a database is an integrated collection of shared files. The database management system acts as a sophisticated buffer between applications programs and the data those programs need. This approach separates data from applications programs and makes data management (the treatment of data as an important organizational resource) possible.

Database Planning and Design Techniques

Since database systems involve people from all parts of the organization with a variety of information needs, the development and operation of database systems must be very carefully planned and managed. The databases themselves must be carefully designed to provide efficient access to information required by the various users. The first step in converting to a database approach should be the development of (1) a master plan that specifies in general terms the various applications and databases to be included in the overall system and (2) a schedule for detailed design and implementation of these applications and databases.

Detailed database design consists of three major phases: information requirements analysis, logical design, and physical design. **Information requirements analysis** is work done with users to define information needs. **Logical design** is the development of schema and subschema definitions. **Physical design** establishes exactly how the data will be organized and stored on the storage devices. Database planning and design are previewed in the next chapter and examined extensively in other chapters of the text.

DATABASE SYSTEMS AND OTHER ORGANIZATIONAL INFORMATION SYSTEMS

Databases and database management systems provide the infrastructure on which other organizational information systems are built. Organizational information systems include transaction processing systems (TPS), management information systems (MIS), and decision support systems (DSS). Relative to the database system, these are applications programs that derive data from the database via the DBMS.

The relationships among these various systems are illustrated in Figure 1-3. The entire set of organizational information systems is shown as a pyramid with the DBMS and databases below the base. **Transaction processing systems** form the bottom of the pyramid and are the lifeblood systems of the organization because they support the processing of "production" data--for example, for inventory maintenance, accounts receivable and payable, and other accounting functions. These transactions provide most of the internal data used as the basis for generating information for managerial decision-making. Transactions themselves are usually well defined; forms displayed on terminal screens can be used to gather and display data. These systems are used primarily by clerical personnel and first-line managers.

Management information systems are in the middle of the pyramid and are used primarily by middle management to control the organization. These systems derive much of their information by summarizing and abstracting data from transaction processing systems. They tend to be report oriented; standard reports are produced periodically (weekly, monthly, or annually) for use by middle managers to support tasks such as budget decisions and personnel assignments.

Decision support systems are designed to provide information for managerial decision making in cases where the decision is not clear-cut (**ill-structured** or **semi-structured** in DSS jargon). These problems tend to occur at the apex of the organizational pyramid. Decision support systems often use mathematical and statistical techniques to manipulate and analyze data. It is difficult to anticipate information needs in a DSS environment, so these systems must be flexible and adaptable.

Database management technology has been used as the basis for some decision support systems. These are referred to as data-oriented decision support systems. Many of the examples in this book are data-oriented decision support systems.

Since decision support systems have become quite popular and often rely on data extracted from database systems, it is important to understand their relationship to database management systems. Decision support systems are discussed in Chapter 13.

EFFECTIVE SYSTEMS AND EFFICIENT SYSTEMS

It is useful to distinguish between effective database systems and efficient database systems. **Effective** systems provide correct, current information that is relevant to the decision at hand. Ahituv and Neumann (1986) refer to effectiveness as "doing the right thing." To provide effective database systems, analysts must work closely with managers and other users to carefully specify information requirements. The system cannot be effective unless information needs are accurately determined and the database system designed to serve those needs.

Efficient systems, on the other hand, perform a task in a cost-effective manner. A database system must provide the required information at a reasonable cost. Ahituv and Neumann call this "doing the thing right."

Of the two, the concern for effectiveness predominates. If a database system does not provide correct, current, relevant information, it does not matter how efficiently the data are processed--it is still useless. Conversely, if the system is effective, it may contain certain inefficiencies but still be highly useful.

Effective database systems are expensive to develop, but an organization cannot afford to go bankrupt in developing effective systems. The value of the information provided must exceed the cost of providing that information.

To illustrate some of the concepts already introduced in this chapter and to explain the need for effective data management, two case examples are presented next. The first is that of the Conventional National Bank, which uses the customary file approach to data management. The second contains a portfolio management example that illustrates the database approach.

Case Example: Data Management at the Conventional National Bank

The Conventional National Bank (CNB) acquired a computer in the mid-1960s to automate check processing. The bank soon realized that the computer was useful for many other purposes, such as maintaining information on mortgage loans, auto loans, savings accounts, and portfolios of common stocks, bonds, and other securities. During the late 1960s and early 1970s, the bank converted these applications from manual to computer-based systems. Most of the software to support these applications was written in COBOL. By the mid-1970s, thousands of COBOL programs were in use. Some of these programs were enormous--several thousand lines long. Together, all the programs comprised well over a million lines of COBOL code.

The bank is organized on the basis of the functions it performs: there are departments for checking accounts, savings accounts, auto loans, mortgage loans, trusts, and so on. To reduce the need for interdepartmental communication, separate data files are maintained for each department. Because the bank has several thousand customers, and most customers have several accounts, these files now occupy several billion characters of disk storage.

Tia Fuentes has been a regular customer of CNB for 15 years. Tia began her association with the bank when she graduated from college and moved to Conventional City to begin a position as manager trainee for Silver Flatware Products. Over the years, Tia has accumulated some savings, which she keeps in a passbook savings account with the bank. Two years ago she purchased a new car with a loan she got at CNB. Three months ago, Tia was promoted to assistant manager, and shortly thereafter she moved into a new home. Tia now resides comfortably in her new estate near the flatware plant.

After an important staff meeting one day, Tia returned to her office to find a telephone message from Gilda Kerr, an auto loan officer at CNB. Tia returned the call only to discover that Gilda was calling to ask why Tia had not made car payments for the past three months. After all, CNB had sent the usual computer-prepared reminder each month with a preprinted return envelope for her convenience.

Tia was stunned! In the excitement of her promotion and move, she had completely forgotten about the car payment. She had called the bank to give them her new address, and she had received her monthly bank statements and notices on the mortgage loan payments. She had not received any notices on the car payments, and they had completely slipped her mind. Gilda knew Tia to be honest and reliable, so she said she would look into the matter.

The manager of Information Services (IS), Henry Lew, showed Gilda some diagrams (Figure 1-4) and explained that all the departmental files had originally been set up independently and that Information Services got updates for departmental files from the departments themselves. IS was not at fault because they did not get a notice to change Tia's address from the auto loan department.

Gilda asked why one update form did not suffice, since the system could use the data from that form to change all occurrences of the address. Henry explained that no one in IS had any way of knowing if a person's name and address were in a file. A program could be written to search all the files, but it would be extremely inefficient because so many files would have to be examined.

Gilda wondered aloud why CNB could not just have one file of names and addresses that was shared by all the departments in the bank. Henry replied that they could, but then all the existing programs using names and addresses would have to be changed, and IS did not really even know all the programs that used that data. Just tracking down those programs and the programmers who wrote them (if they were still at CNB) would be a big job.

In desperation, Gilda asked if IS could not at least find each name occurring in more than one file and then compare the addresses to see if they were consistent in all files. Henry said that IS would be glad to do that, but that it would take a special program. IS was already overloaded with a waiting list of two years' worth of programming. With luck, however, they might be able to develop the program in 18 months.

Gilda was flabbergasted! How could it take 18 to 24 months to produce something as simple as a list of people whose names and addresses appeared in more than one place in the computer system? She left the data processing department more discouraged than ever about computers. Why should the bank waste time putting data into the computer system, when it took forever to get it out in a usable form, and even then was wrong half the time?

The intent of the preceding scenario is to illustrate some of the problems that arose from the conventional file approach to data management. Special programs have to be written for simple tasks. It may be difficult to extract data in the system if it resides in separate files. Users (and data processing personnel) find these problems exasperating and inefficient. The next scenario shows some of the advantages of the database approach.

Case Example: Byer Kaufman's Portfolio Management System

Byer Kaufman was known as the child prodigy of his high school class. While still in high school he studied the stock market and developed a highly successful investment strategy that he implemented on his personal computer. Using his strategy, Byer had become a teenage millionaire.

Instead of going to college, getting married, or joining the army as most of his friends had, Byer started his own investment counseling firm and was doing extremely well. He had converted his investment software to a larger computer and now maintained data on thousands of securities in his database. In addition, he had added a simple query language that allowed his clients to access the database and his software directly so that they could select their own investments and manage their own portfolios if they chose to do so. His clients could even copy ("download") portions of his database to their own personal computer (PC) and manipulate the data there with a PC version of his query language. Many of his clients, however, were busy with other tasks--they preferred to have Byer manage their portfolios for them.

Byer is presently demonstrating his system to a potential new client, Joan Yasuda. Joan is interested in searching Byer's database for some underpriced stocks: "What would I do to find a stock in the airline industry whose price is less than five times its earnings?"

"That's a good question," Byer replied, "One that requires what database people call an 'ad hoc' query. It's ad hoc because the database designer may not have anticipated that precisely that question would be posed to the system. Nevertheless, the system can still answer it. The name of the stock is

FIRM_NAME in the system, and it's stored in a table named STOCK_DATA_TABLE along with current price, earnings, and dividends and the firm's industry [see Figure 1-5a]. We can compute price-earnings ratios just by dividing price by earnings, so you would type in this:"

```
SELECT FIRM_NAME
FROM STOCK_DATA_TABLE
WHERE INDUSTRY = 'AIRLINE'
AND (PRICE / EARNINGS) < 5.00
```

Almost immediately the system displayed a list of firms meeting these criteria.

"Next," said Byer, "I'll show you how I manage portfolios for my clients who don't have time to do all the analysis and so forth themselves."

"That's a good idea," said Joan.

"Okay," said Byer, "My system works like this. First, I have an enormous database consisting of 20 years of data from financial statements and stock market data on about 5000 companies. This database is about 2.5 billion characters in size, and it is updated every evening. I have several mathematical models based on sophisticated statistical techniques to analyze market trends and the behavior of individual securities to pick 200 or so that are good investments at any one time. I also have software based on management science models that I use to select and revise portfolios for my clients from the 200 good securities. I developed these models myself and the software is legally protected as a trade secret, so no one else can provide you with my approach.

"Data for individual portfolios is maintained in separate files, so I can manage them separately. These portfolios contain only the name of the securities you hold and the number of shares of each [see Figure 1-5b]. However, they are all integrated into a common database with the stock data, so I can extract prices, earnings, and dividends from the STOCK-DATA-TABLE and send you monthly reports on the value of your account; that is, I know the firm name from your portfolio file and can relate that to the stock data file to retrieve current prices to compute the current value of your portfolio. The database system makes it easy to do that either on-line or via a program.

"At the end of the year, I send you an annual report to help your accountant prepare your income tax return. This diagram [see Figure 1-6] shows how all the pieces fit together. Let me show you how the portfolio selection system works."

Byer hit the key labeled "MAIN SCREEN," and the main menu appeared on the screen (Figure 1-7). He selected the option to initiate a new portfolio by entering "2," because he wanted the system to continue displaying menus so that Joan could see the options available at that point. He could have entered the word "initiate" and the system would have entered Command Mode, which is faster for experienced users because it does not display options but expects the user to type in commands without special prompts.

Next the prompt in Figure 1-8 appeared. Byer had developed these menus and prompts to assist in gathering the data for new clients. Byer filled in the blanks as shown in Figure 1-9. Since Joan had plenty of income from other sources, he put in growth as the primary objective; meaning Joan wants to maximize the increase in the money invested ($1,000,000) rather than trying to get immediate cash. Also, because Joan has plenty of money, she can accept high risk in the hopes of getting a high return. The system responded,

OK. I'M GOING TO SELECT A HIGH-RISK, GROWTH PORTFOLIO FOR YOUR $ 1.0 MILLION DOLLARS. THAT IS WHAT YOU WANT ME TO DO, ISN'T IT?

Byer typed in "yes," and the system displayed:

PLEASE WAIT WHILE I SELECT A GOOD PORTFOLIO FOR YOU. IT WILL TAKE A FEW SECONDS TO DO ALL THE SEARCHING AND CALCULATIONS.

After a brief pause, the following message appeared on the screen, along with a list of securities and the amount to be invested in each:

I HAVE A PORTFOLIO FOR YOU. I EXPECT IT TO INCREASE YOUR $ 1.0 MILLION TO $ 1.23 MILLION IN SIX MONTHS. BUT, THIS IS RISKY. YOU COULD LOSE YOUR MONEY. THE PORTFOLIO CONSISTS OF THE FOLLOWING INVESTMENTS:

"This looks great!" said Joan. "Sign me up, and put a million in that portfolio."
"Fine," Byer replied. "I'll draw up a contract."

OBJECTIVES OF DATABASE SYSTEMS

The preceding scenarios may be used to illustrate some problems that occur in traditional file-oriented systems, as well as the objectives of database systems. These objectives are derived from the overriding goal of managing data as an important organizational resource. For each objective, there is a corresponding problem arising in a file-oriented approach. Database system objectives and problems in file-oriented systems are discussed together. Database objectives include providing flexible access to information, maintaining data integrity, protecting the data from destruction and unauthorized use, providing data shareability and relatability, reducing data redundancy (duplicated data), making data independent of applications programs, standardizing data item definitions, and increasing the productivity of information systems personnel. Each of these objectives is discussed fully in the following sections.

Access Flexibility

Access flexibility allows for easy retrieval of selected items in a database and presentation of that data in a variety of formats. This flexibility is one of the most important reasons for having database management systems in the first place--to have information readily available for managerial purposes. The part of the system that provides for communication with the person using the system is often called the **user interface.**

In older, conventional programming environments (such as that of the Conventional National Bank, where COBOL was in use), the only user interface provided is for programmers. In such environments, special programs must be written to perform even simple tasks such as listing those people whose names and addresses appear in two or more files. Most database packages have a special **query language** designed to perform such tasks with a few simple commands. Query languages are much easier to learn and use than programming languages, and they are suitable for use by many managers and other end-users who have neither the time nor the inclination to learn to program.

Byer's portfolio management system is an example of a system that provides flexible access. The retrieval command given previously is an example of a query in the **Structured Query Language (SQL),** the standard query language for relational database systems.

The portfolio management system also uses menus and fill-in-the-blank operations to guide unfamiliar users through the system. Experienced users can bypass these by entering commands directly. Also, an on-line "help" facility is available to give more complete explanations of system usage.

Although query languages are easy to learn and use, they lack the power and versatility of programming languages. To make up for this deficiency, most database management systems also provide a special **data manipulation language** consisting of commands that may be embedded in applications programs to add, retrieve, or change data values. Some packages allow SQL statements to be used in programs; others use the data manipulation language designed by various subcommittees of the Conference on Data and Systems Languages (CODASYL), which is discussed in Appendix C. Still other packages have their own proprietary language.

Most database management systems have a language that is intermediate in power between query languages and data manipulation languages. Such languages are called **report generators,** because they are designed to create printed reports. Report generators have special commands for creating headings, titles, rows, columns, sums, and other elements frequently found in reports. The programming of reports is often greatly simplified by using such facilities.

Many database packages also have a **screen generator** to help analysts write routines to create forms on a terminal screen. Users fill in these forms to retrieve, modify, or add data to the database. These facilities simplify the process of capturing data and making it readily accessible to users.

Finally, database management systems usually work closely with the host operating system to provide rapid and flexible access to the database. Some packages have their own access routines, but most rely on the routines of the host operating system.

Data Integrity

Data integrity ensures that data values are correct, consistent, and current. This is a critical aspect of information management. The problem that Tia had in getting notices of payments due is one example of a lack of data integrity. Even more critical to a bank is the need to maintain accurate account balances. How long would a bank survive if it credited deposits to the wrong account or reported incorrect account balances?

Managers insist on accurate data for decision-making, and often they will not use information from a computer system that has produced incorrect data. Such misgivings may be well founded. Suppose you were a portfolio manager using Byer's system and it indicated that the price of a stock was going down, and you sold it--when it was actually still going up. In the future, you would probably be suspicious of the data in that system, if you even used it at all.

One of the best ways to assure data integrity is to make sure that the data values are correctly entered in the first place. This may be done through a variety of methods, such as carefully setting up manual procedures for originally capturing the data, or programming the system to check the reasonableness of data values when they are entered into the machine. For example, you may know that hourly wages in a payroll system never exceed $30.00 per hour. The data input program should check to see that hourly wages entered never exceed this value.

Data Security

Data security is needed to protect data from unauthorized access and from accidental or intentional damage or destruction. It might be nice (but also illegal) to secretly change one's bank balance or one's grades. Clearly, Byer would not want a person using his portfolio management system to be able to access the portfolios of other clients. Security is another critical aspect of database systems; responsible managers are justifiably reluctant to put sensitive data into computer systems unless every reasonable precaution is made to protect it from unauthorized access. Security will be discussed in greater detail later in Chapter 11.

Another need for security is related to inadvertent destruction of the database by natural disasters such as fires, floods, tornadoes, hurricanes, and volcanoes. Yes, even volcanoes. The fine dust from the Mount St. Helen's eruption in Washington played havoc with many computer systems in that area. To provide security from such events, it is customary to keep a log of changes made to the database so that they may be reapplied to "back up" copies of the database that are made occasionally and stored in a safe place. **Recovery** is the process of using logs and backup copies to recreate a damaged database. For critical databases, such as payroll files, it is desirable to store the backup in a different building, perhaps in specially designed rooms.

Other security facilities include passwords for individual users that allow different types of access to the database (for example, read only or read and write), and passwords for databases, data records, and even individual data items. These measures and others as well are discussed in Chapter 10.

Data Independence and the ANSI/SPARC Model (Are we leaving this in?)

Data independence has two dimensions: logical and physical. To explain the difference between logical and physical data independence, it is convenient to use the American National Standards Institute/Standards Planning and Requirements Committee (ANSI/SPARC) three-level architecture of database systems. As illustrated in Figure 1-10, the uppermost level in the architecture is the **external level.** The external level is closest to the users of the system and refers to the way users view the data. This may be in terms of printed reports, forms, or other documents that contain information useful to people using the system. Different users may have different, but overlapping, data needs and may view the data differently, as did Joe and Sue in Figure 1-2. The data as thought of by an individual user or group of users with similar needs is called a **user view.**

The middle level in the architecture is the **conceptual level.** The conceptual level represents the union of all the user views at the external level. Thus the conceptual representation is an integrated view or schema of the entire database. At this level, the representation of data still reflects human views of data. The database at this level is described in a **logical schema.**

The third level, the **internal level,** corresponds to the actual representation of data within the computer system and the methods used to access the data. This may also be referred to as the **physical level,** for it specifies how the data are represented physically in the storage devices of the machine.

In a system with **physical data independence,** the conceptual level schema is independent of changes in the internal or physical level; that is, the way the data are actually stored or accessed in the system can be changed without requiring a change in the logical schema. The database management system is responsible for insulating the conceptual level from changes in the physical level by providing means of translating physical representations into conceptual representations.

In a system with **logical data independence,** the external level is independent of changes to the conceptual level. One user may wish to change or add a user view, which changes the conceptual level, but this should have no impact on other users and their views.

One important result of data independence is that the close link between data files and the programs using those files is weakened. In conventional situations where ordinary programming languages are used, programs and the files they access are very closely associated. When the U.S. Postal Service converted from five- to nine-digit ZIP codes, for example, a tremendous number of files in the computer systems of organizations throughout the country had to be restructured. Moreover, all of the programs accessing those files had to be modified and recompiled.

Database management systems seek to circumvent problems such as these by making the data independent of applications programs that use the data. This is done by putting the data under the control of the database management software. Applications programs access the data, not directly, but rather via the data manipulation language of the database management system. The system is responsible for satisfying the requests for data retrievals, changes, and additions.

Data independence is a major objective of database management systems. Separating data files from programs is essential if data are to be managed as an independent resource. Logical data independence is necessary for database systems to provide adaptability to changing user requirements. In an environment where files are shared, it is important to ensure that changes made in response to one user have no major implications for others. Even though it is virtually impossible to achieve absolute logical and physical independence, current database systems provide much more data independence than file-oriented systems, thus enabling better management of organizational data resources.

Reduced Data Redundancy

Data redundancy is storage of the same piece of data in more than one place in the computer system. Another objective of database management is to reduce data redundancy. The master files in the Conventional National Bank scenario (see Figure 1-4) can be used to illustrate some problems that arise when redundancy is allowed. Note that each of the files contains fields for last name, first name, middle name, and social security number. If the same person has several accounts with the bank, then his or her name and social security number are duplicated in several different files. This is clearly a waste of storage space.

When files are updated, data redundancy often leads to problems of data integrity. Because the same data are duplicated in several places, it must be changed in each place it occurs; otherwise inconsistent data values will arise. In the CNB case, the error of sending Tia's payment notice to the old address would have been avoided if her address had been correctly stored in one location and shared with all necessary departments.

Data redundancy leads to processing inefficiencies as well. Updating the same data in different files requires extra processing time. Because each department generally has its own update programs, several programs contain procedures for updating the same fields and must be run to update files. If the redundant data are stored in only one place, both storage space and processing time for updates can be saved.

Figure 1-11a, referred to as a **data structure diagram** or **Bachman diagram,** shows how CNB's files might be reorganized to reduce data redundancy. Name and social security number are stored only once, in the master record. The master record contains account numbers indicating the type of accounts owned by each person. Data items such as the account numbers in the master file are called **pointers** because they "point to" associated records in other files. Another representation of CNB's database is shown in Figure 1-11b. This is a **relational model** of CNB's data; data elements are shown in tables called <u>relations.</u> Relationships between tables are represented by common data items in different tables: in Figure 1-11b, customer number (Customer#) is shown in each table. Data for any customer can be located by matching on common customer numbers in different tables.

In both the data structure diagram and the relational model (two approaches that are examined in detail in this book), some data elements have been reproduced in the reorganized database. (This may or may not result in <u>actual</u> data redundancy; the customer number may be stored at one location with pointers in other files to that location.) In designing databases, we seek to control data redundancy, since in most cases it is not possible, or even desirable, to totally eliminate it. It may be desirable to allow redundancy, for example, to increase access flexibility and response time.

Data Shareability

Data shareability allows different users or user groups to use the same (non-redundant) data. If data redundancy is to be reduced, then it is necessary to share data among various organizational departments. Tia's address is one example. Byer's security database, which is accessed by several clients, is another.

Users are not always enthusiastic about sharing data. Reluctance to share data often arises from fear of unauthorized use of private data or from resistance to making performance data (such as grades) available to others. One who used Byer's portfolio management system would appreciate shared access to public information about securities but would probably not want other users to have access to his or her private portfolios and investment holdings. Likewise, different groups in educational institutions--the registrar's office, advisors, or financial aid officers--need access to student grades, and so it is reasonable that they share grade information. But few others have such needs, and access to such information must be tightly controlled.

Potential providers of data may have legitimate reasons for resisting the placement of data into a shared system. Designers of database systems must be aware of these reasons and be sure to take them into account during system development.

Data Relatability

Data relatability is the ability to establish logical relationships between different types of records, usually in different files. The pointers in Figure 1-11, for example, show the relationships between the master files and account files in CNB's database. Because customers have checking accounts, and checking accounts have balances, a relationship exists between the files containing these two data items.

In the case of the CNB, it was not possible to perform the simple task of relating names and addresses in various files without writing a special program to do so. In cases where ordinary programming languages alone are used to manage data, such problems arise frequently. If a database system is to achieve the goal of providing flexible access to information, it must also provide the ability to relate data that resides in different parts of the system.

In many instances, some of the most important information in the database must be derived from relationships between database items. Joan's query into Byer's security database, for example, required information on the relationship between price and earnings for each security. Price and earnings data might be kept in different records in the system because price data must be updated more frequently than earnings and updating could be more efficient if the two were stored separately. But if a query involves a relationship such as price divided by earnings, both records must be retrieved to create the desired information.

As another example, Byer would need information such as current prices, dividends, and earnings to create monthly and annual reports for client portfolios. If these data values were stored every time a security was included in a portfolio, it would result in a great deal of data redundancy because the same security might be held in many portfolios. This data redundancy could be reduced by storing the security data once and providing the ability to relate the information on the relationship between portfolios and securities via the database management system.

Standardization

Standardization refers to the need for common definitions of data items, in terms of both the precise definition of a data item name and its storage format in the database. Most database management systems provide a **data dictionary** facility to define data item names and to state the internal storage format of the data items in the database. They may, for example, permit the specification of **synonyms** or **aliases** when more than one name is used to refer to the same data item. This strategy is useful in supporting shareability when different user groups refer to the same data with different names. Thus "marks" might be an alias for the more common term "grades."

Data dictionaries may provide means for controlling access to data as well. Some users may be permitted to access certain data items while others are denied access to those items or are not even informed that the items exist in the database. In some cases it may be desirable to allow selected users to retrieve, but not to change, data in the system; such an approach can prevent inadvertent destruction of data by inexperienced users. Other facilities of data dictionaries are discussed in the chapters on database administration.

A related problem in the development of database systems is the tendency of users to have slightly different definitions (or imprecise definitions) of potentially shareable data. In the development of a database system to help manage research proposals submitted by a research and development organization, for example, it was discovered that different departments had different ideas of what constituted a proposal. Most departments thought of a proposal as a formal document spelling out the details of a research project for a potential funding agency. Other departments, however, operated on a much less formal basis, with only a letter or even a telephone call constituting the proposal in some cases. A standardized definition of "proposal" had to be developed as part of the system design effort.

Personnel Productivity

Any organization must deal with constantly changing information needs. Information needs may change as a result of new governmental reporting requirements, tax laws, accounting procedures, or economic conditions. The modification of existing programs to satisfy changing information requirements is referred to as **program maintenance.** Large organizations have discovered that the lack of data independence in file-oriented environments severely hampers the ability to respond to changing information requirements and results in large maintenance expenditures. Martin (1983, p.48) claims that as much as 80 percent of some corporations' programming budget goes into maintenance, while only 20 percent goes to the development of new systems. One can assume this percentage is roughly still accurate, especially considering the high costs associated with the development and administration of new systems. The bulk of this maintenance expense is accounted for by personnel time. Another major objective of database systems is to reduce the amount of personnel time it takes to respond to changing information needs.

Database management systems can lead to increased productivity in several ways. Simple requests for data can be handled via the query language, obviating the need to write a program. In many cases, users themselves can use the query facility, circumventing the need for systems personnel entirely.

Even if a more sophisticated report is required, it may be possible to "program" it using a report generator. This may take far less time than writing an equivalent report in a conventional language. Or, again, users may be able to write their own reports.

As new applications are added to the system, the data those applications require may already be in the database. Thus the time and expense associated with designing file formats and collecting data are bypassed.

DISADVANTAGES OF THE DATABASE APPROACH

Just like any other technology, database systems are not without their costs and disadvantages. An organization may find that a database approach is expensive at first. Database management software is expensive. Some mainframe packages with minimal features may be purchased for $20,000 to $30,000. Full feature packages, however, may run as much as $300,000. A package with typical capabilities costs around $100,000. Generally, the number of users determines the overall cost. Personal computer packages are available for a few hundred dollars or so, but these do not support data sharing and many of the other objectives described in this chapter. These systems are not really comparable to multi-user, mainframe packages.

Multi-user packages are complex and often have extensive memory requirements. As new applications are converted to the database system, storage requirements may increase. Thus new hardware may be required to accommodate the increased memory and storage needs.

Additional costs arise because database programming may be more complex than file-oriented programming in some circumstances, and programmers must be trained to use the package. Also, analysts must be trained in database design techniques or experienced analysts must be hired. Careful plans for development of the database system must be developed and the databases and applications programs must be acquired, designed, or converted.

There are, of course, many benefits that offset these costs. As more applications are converted to the database system, costs begin to drop because most or all of the data required is already in the database. Simple applications may be handled with query languages or report generators, resulting in much less programming time. Furthermore, maintenance expenditures are reduced because the independence of data and programs allows programs to be changed without concern for file structures and the effect on other users or programs. It may be possible to cut maintenance expenditures in half, or even more. Management must realize, however, that such results are obtained, not overnight, but only after a period of two years or so. Thus conversion to a database environment requires a long-term commitment on the part of the organization.

THE USER'S VIEW

Database systems exist to support the needs of information users. The needs of users are thus paramount in the development of an effective system. Different users of a database system have different needs and different perspectives on the system and the information and data it provides. Different people may refer to the same entity with different terms or, perhaps worse, may use the same term to refer to a different entity. The terms "customer," "patron," and "client," for example, may all refer to the same entity--people who buy the organization's product or service.

In an integrated database environment, it is unusual for any one user to need access to every item in the database. Thus the perspective of users is often limited. The primary concern of each user is to get the information he or she needs. Information analysts are concerned with ensuring that the information requirements of users are satisfied. Thus the objective of information analysts is to provide an effective system—one that does the right thing—by supplying the information required to make decisions.

Database analysts are concerned with designing databases that provide the required information efficiently. Effective, efficient organizational information is best provided through integrated, shared database systems. In database management, the different perspectives of users are referred to as user views, hence the name of this section. The User's View in each chapter is designed to help you keep in mind why database systems are developed in the first place and to reinforce the main goal as we examine the trees of the database forest.

Metropolitan National Bank

The Metropolitan National Bank scenario is a continuing case study you can work throughout the text. It provides a general idea of the organizational environment and the type of activities prevalent in the information management field. Each chapter contains a specific project that provides an opportunity to apply the concepts illustrated in that chapter.

The Metropolitan National Bank, or MetNat as it is called for short, is a large financial institution in the town of Lincoln, Texas. Lincoln is a moderate-size town of approximately 500,000 people at the base of the Texas panhandle. The rural areas around Lincoln are predominantly ranching and farming areas. MetNat provides checking, savings, and several types of money market accounts to its customers. It also provides a wide range of loan services, from large business loans to small personal loans. MetNat has safe deposit boxes and can access a statewide automated teller network.

MetNat currently services approximately 70,000 checking accounts, 20,000 saving accounts, and 25,000 money market accounts. MetNat has approximately 5,000 commercial loans and 3,000 personal loans outstanding. MetNat also issues a bank credit card. Because the credit card has a low annual fee ($50) and an extremely attractive annual percentage rate (8. 00%), the bank has almost 50,000 cardholders.

At one time, MetNat performed all of its accounting functions manually. This system was replaced in the early 1960s by the first of many computer-based systems. The most recent system began operation in the early 1980s, and has been updated several times over the years with changing technology. MetNat's information processing orientation has always has always emphasized bank administration efficiency. MetNat's information processing management has never been particularly concerned with using information processing for any type of competitive advantage. However, the competitive nature of the banking industry has caused the Board of Directors to make some changes in the information processing function. A new position has been created, a Chief Information Officer, and you have been hired to fill the position.

In each of the chapters that follow you have an opportunity to apply the concepts of the chapter to the scenario at MetNat. Along the way you will be building, piece by piece, a comprehensive analysis and design of the MetNat information processing system. You should always review prior MetNat assignments prior to beginning a new one, since the new ones frequently build upon the old ones. By the end of the book, you should have a considerably well-developed project that can demonstrate your information systems and design capabilities. Good luck!

CHAPTER SUMMARY

Data consists of numbers, letters, words, names, and other symbols that may be stored in a computer system. Information is useful data. Organizations now realize that data are a valuable resource that deserves careful management. Information for managing the organization can be derived from data, which resides in integrated collections of shared files called databases.

Database management systems are software packages designed to support information management. The major components of database management software, summarized in Figure 1-12, include a query language for simple, ad hoc access to the database, a report generator to create printed reports, a data manipulation language for accessing the database via programming languages, facilities for providing security, integrity, backup, and recovery, and a data dictionary facility to catalog all the data items under control of the database system. Each of these major components is examined in detail in the remainder of this book.

Many packages include several other features as well, including screen generators for designing terminal screens, intelligent systems for facilitating application development, and support for databases distributed over several computer systems. These features are described later in the book.

Database administrators are responsible for managing database systems. Database analysts and information analysts are staff members who use database technology to develop database systems.

It is critical that database systems be effective in the sense that they must provide correct, current, and appropriate information for managerial decision making. A database system should also provide this information efficiently.

The conventional approach to managing data with a close link between applications programs and files often leads to problems of redundant data, processing duplication, lack of data consistency and integrity, and the inability to relate data items stored in different files. Database systems seek to overcome these problems by separating data from applications programs, providing flexible access to the data, protecting the data from unauthorized access and malicious or unintentional damage, controlling data redundancy, and improving data integrity, relatability, shareability, and standardization and increasing programmer productivity.

QUESTIONS AND EXERCISES

1. Define the following terms:
   - a. data
   - b. information
   - c. information management
   - d. database
   - e. database system
   - f. database management system
   - g. database administrator
   - h. database analyst
   - i. decision support system
   - j. system effectiveness

k. system efficiency
l. user view
m. logical schema
n. physical schema

2. Explain why data are an important organizational resource.

3. What is the main goal of a database system?

4. If, as a manager in an organization, you needed information to perform your job, would you want your data processing department to emphasize effectiveness or efficiency in its database systems?

5. As a database administrator, would you emphasize effectiveness or efficiency in the development of database systems?

6. List and briefly discuss the objectives of database systems.

7. Describe the three levels of the ANSI/SPARC database architecture.

8. Why might people be reluctant to share data in an organization? How might you convince them to put data into a shared database?

9. Name and briefly describe three applications that are particularly well-suited to the database approach. What are some advantages and disadvantages of the database approach for these applications?

10. Why is planning more important in a database environment than in a conventional file-oriented environment?

11. Determine whether each of the following relates to logical independence or physical independence:

a. The ability to convert to new storage devices without changing applications programs.
b. The ability to provide one user access to additional data items already in the database, without affecting other users.
c. One user wants a data item displayed with no decimal places; a second user wants the same data item displayed with two decimal places.
d. The ability to change the access method (see the Appendix) used to retrieve a data record without changing applications programs.

12. Model a short schema and subschema for two users in the following situation: Bart needs access to customer name, address, and account balance; Lee needs access to customer name, address, and credit rating.

13. Elizabeth Barella and Andrew Wu both work for the Best Desserts Shop. Elizabeth calls clients of the shop "patrons"; Andrew calls them "customers." Assume that you are developing a database system for Best Desserts. How might you handle the problem of data definition?

14. Name two ways in which use of database systems can lead to increased personnel productivity.

15. Database management systems are expensive. Why do you think so many organizations use them in spite of the expense?

16. Discuss disadvantages of the database approach. What might be done to offset these disadvantages?

17. Indicate whether each of the following examples is a transaction processing system, a management information system, or a decision support system.

a. A system for entering, maintaining, and distributing student grades at a university.
b. A system for reporting differences in budgeted costs and actual costs associated with projects at a software consulting firm.
c. A system to help decide whether a potential new product should be manufactured and sold.
d. A system to record data from sales transactions.
e. A system that takes data from sales transactions and plugs it into a sales forecasting model to produce information used in deciding on raw material orders.

18. Give an example of a transaction processing system, a management information system, and a decision support system.

19. What are some unique security problems that might arise in a university system that uses one computer system to support its administrative systems (faculty and staff, student grades, budgets, and so on) and academic systems (teaching programming, software packages, and so forth)? What might be done to control security problems?

20. Indicate whether normal or tight security measures would be justified for each of the following types of data and briefly explain your answer.

a. Student names and addresses at a university.
b. Student grades at a university.
c. Names and addresses of a firm's employees.
d. Salary and wage data for a firm's employees.
e. Travel itinerary of a U.S. senator traveling in the United States.
f. Travel itinerary of a U.S. senator traveling in countries with known terrorist activities.

FOR FURTHER READING

Ahituv, Niv, and Neumann, Seev. <u>Principles of Information Systems for Management</u> (2nd ed.). William C. Brown, Publishers, Dubuque, Iowa, 1986. A good explanation of the difference between effective and efficient systems in Chapter 4.

Cardenas, Alfonso F. <u>Data Base Management Systems</u> (2nd ed.). Allyn and Bacon, Boston, 1985. A thorough treatment of database system objectives is found in Chapters 1 and 3.

Kanter, Jerome. <u>Management Information Systems</u> (3rd ed.). Prentice-Hall, Englewood Cliffs, New Jersey, 1984. The relationship of database systems and management information systems is discussed thoroughly in Chapter 4. Information resource management is treated in Chapter 11.

Martin, James. <u>Managing the Data-Base Environment.</u> Prentice-Hall, Englewood Cliffs, New Jersey, 1983. One of the first and still one of the best books on the managerial aspects of database systems. The first six chapters provide a very readable view of managerial issues in a database environment.

FIGURE 1-1

Conventional Programming Environment with Program-file Dependence. Both Joe and Sue have access to product prices. Only Joe has access to the manufacturing budget and only Sue has access to salary data.

FIGURE 1-2
An Integrated Database with Both Shared and Private Data. Both Joe and Sue have access to product prices via their respective subschemas, but only Joe can access the manufacturing budget and only Sue can access salary data.

FIGURE 1-3
The Relationship of Database Systems to Transaction Processing Systems, Management Information Systems, and Decision Support Systems

FIGURE 1-4
Conventional National Bank Departmental Master File Formats

FIGURE 1-5
(a) Data Elements in Byer's STOCK-DATA-TABLE; (b) Data Elements in Individual Portfolios. By matching firm name in the portfolio to firm name in the STOCK-DATA-TABLE, the current price for each stock in a portfolio can be found and the value of the portfolio can be computed by multiplying shares held by the current price and summing over all securities.

FIGURE 1-6
Structure of Byer's Portfolio Management System

FIGURE 1-7
Main Menu for Byer's Portfolio Management System

FIGURE 1-8
Prompts for Data to Initiate a New Portfolio

FIGURE 1-9
Byer's Responses to the Prompts of Figure 1-8

FIGURE 1-10
ANSI/SPARC Three-level Database Architecture. The external level consists of data as the users view it. The conceptual level is a logical composite of the entire database and is described in the logical schema. The stored data are described in the physical schema at the internal level.

FIGURE 1-11
(a) Data Structure for CNB Database (the lines indicate pointers showing relationships between record types in the reorganized database); (b) Relational Representation of the CNB Database (relationships between files are given by including customer number as a data item in each table).

FIGURE 1-12
Components of a Database Management System

FIGURE 1-11 (continued)

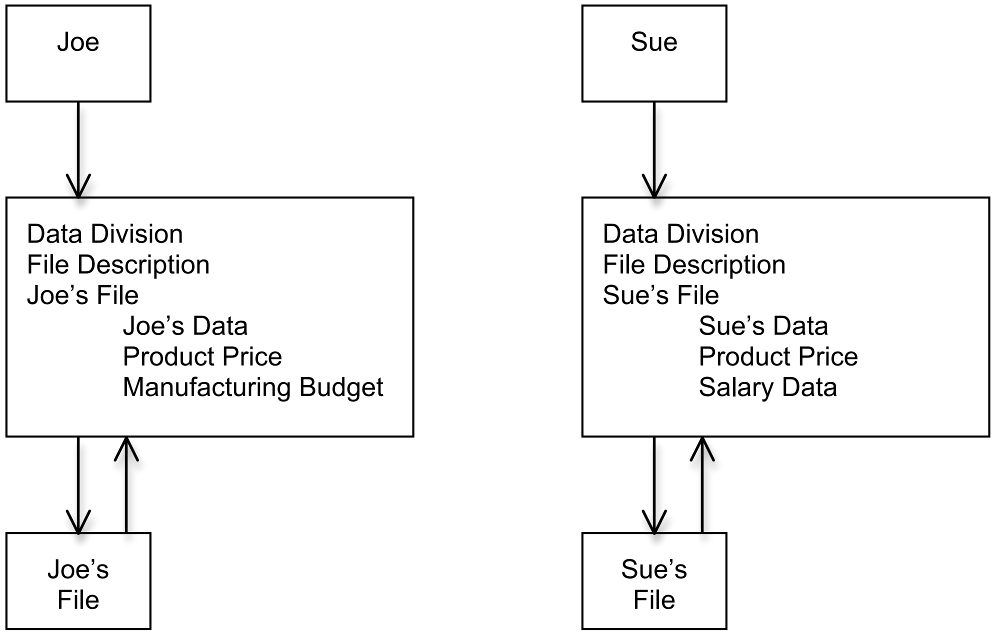
Joe
Data Division
File Description
Joe’s File
Joe’s Data
Product Price
Manufacturing Budget
Joe’s
File
Sue
Data Division
File Description
Sue’s File
Sue’s Data
Product Price
Salary Data
Sue’s
File

Figure 1-1

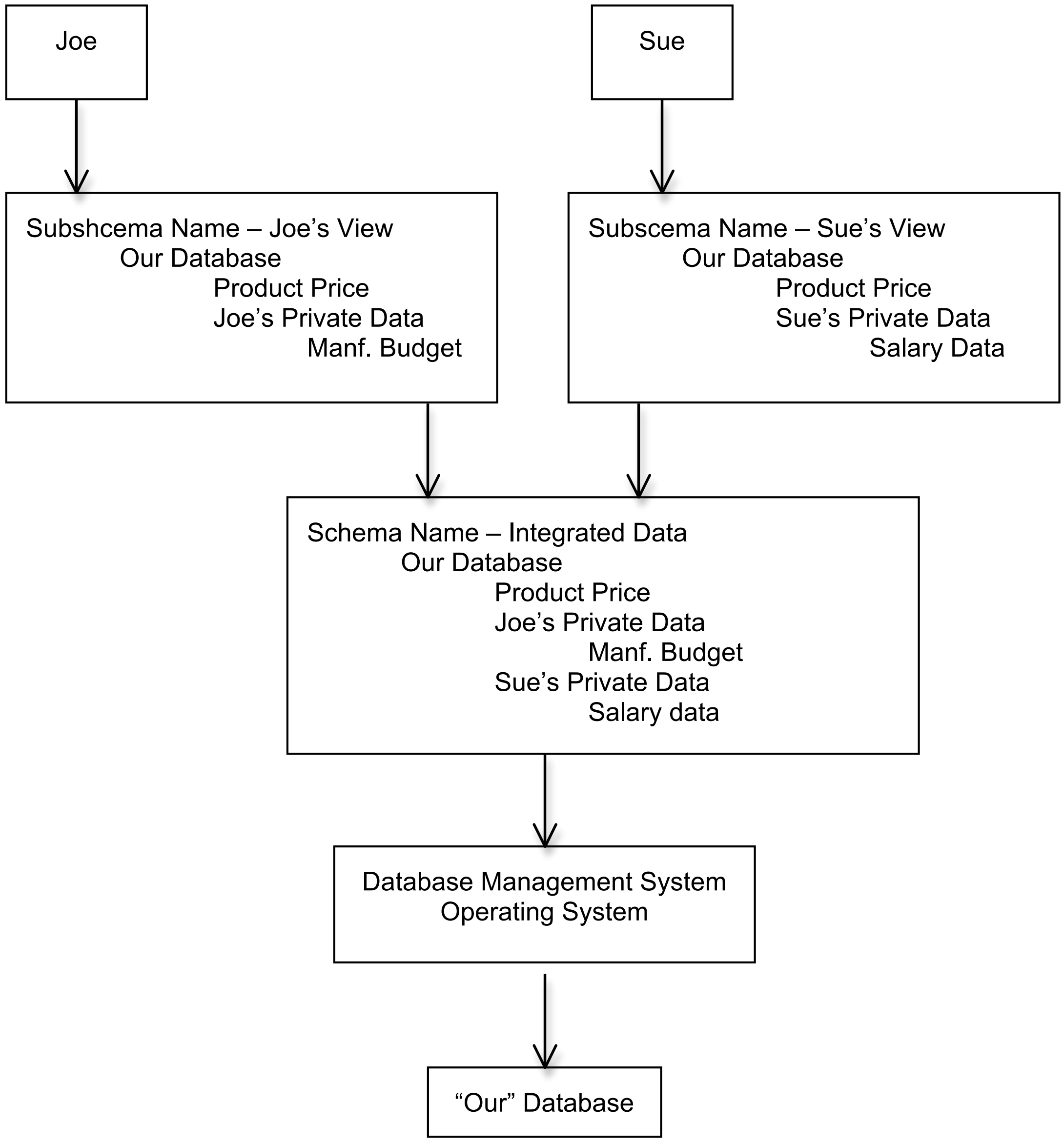
Joe
Sue
Subshcema Name – Joe’s View
Our Database
Product Price
Joe’s Private Data
Manf. Budget
Subscema Name – Sue’s View
Our Database
Product Price
Sue’s Private Data
Salary Data
Schema Name – Integrated Data
Our Database
Product Price
Joe’s Private Data
Manf. Budget
Sue’s Private Data
Salary data
Database Management System
Operating System
“Our” Database

Figure 1-2

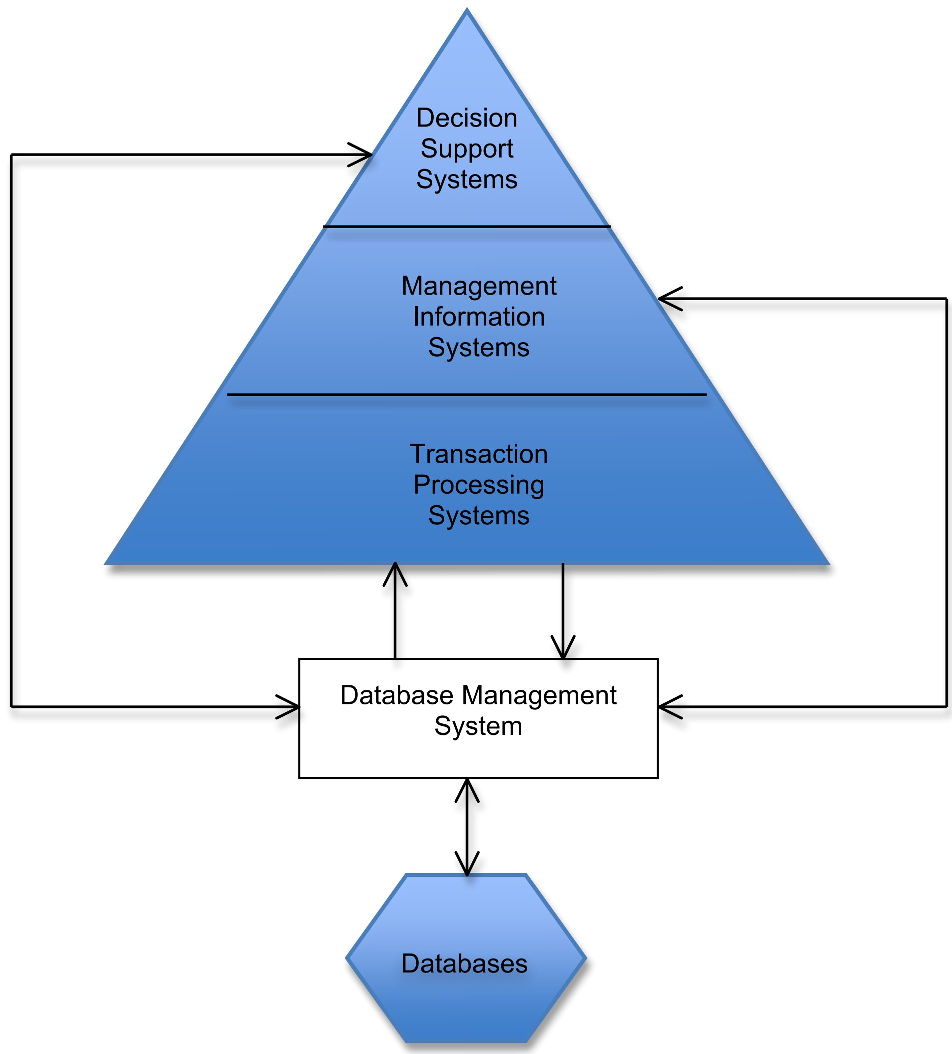
Decision
Support
Systems
Management
Information
Systems
Transaction
Processing
Systems
Database Management
System
Databases

Figure 1-3

Checking Account Master File Format

Acct#	Last Name	First Name	Middle Name	SS#	Checking Acct data

Savings Account Master File Format

Acct#	Last Name	First Name	Middle Name	SS#	Savings Acct data

Auto Loan Master File Format

Acct#	Last Name	First Name	Middle Name	SS#	Savings Acct data

Personal Loan Master File Format

Acct#	Last Name	First Name	Middle Name	SS#	Savings Acct data

Figure 1-4

STOCK-DATA-TABLE (a)

Firm-Name	Industry	Price	Earnings	Dividends

YASUDA-PORTFOLIO (b)

Firm-Name	Shares-Held

Figure 1-5a
Figure 1-5b

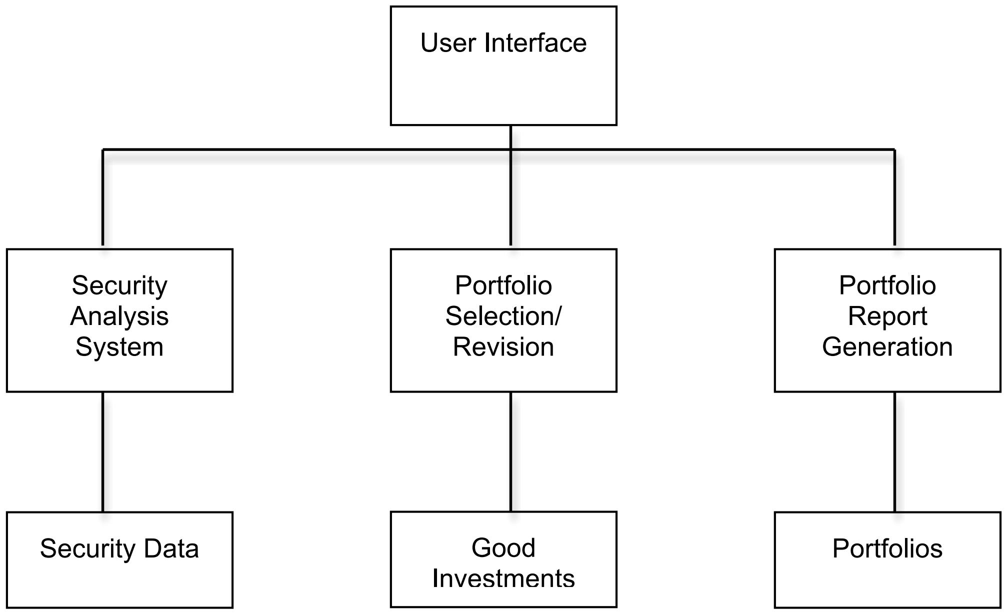
User Interface
Security Analysis System
Portfolio Selection/ Revision
Portfolio Report Generation
Security Data
Good Investments
Portfolios

Figure 1-6

Main Menu

Your Choices Are:

1. Analyze Securities to Find Good Investments
2. Initiate a New Portfolio
3. Revise and Existing Portfolio
4. Examine existing portfolios
5. Run Monthly or Annual Reports for Portfolios
6. Help on How to Use the System
7. Exit From the System

Enter a number from 1 to 7 corresponding to your choice, or enter the first word of the line to enter command mode.

What is your choice? 2

Figure 1-7

Enter the Following:

Clients Name:

Portfolio Name:

Risk Acceptability (Low, Normal, or High):

Primary Objective (Growth, Income, or Principal Protection):

Initial Investment:

Figure 1-8

Enter the Following:

Clients Name: Joan Yasuda

Portfolio Name: Gigabuck

Risk Acceptability (Low, Normal, or High): High

Primary Objective (Growth, Income, or Principal Protection): Growth

Initial Investment: 1,000,000

Figure 1-9

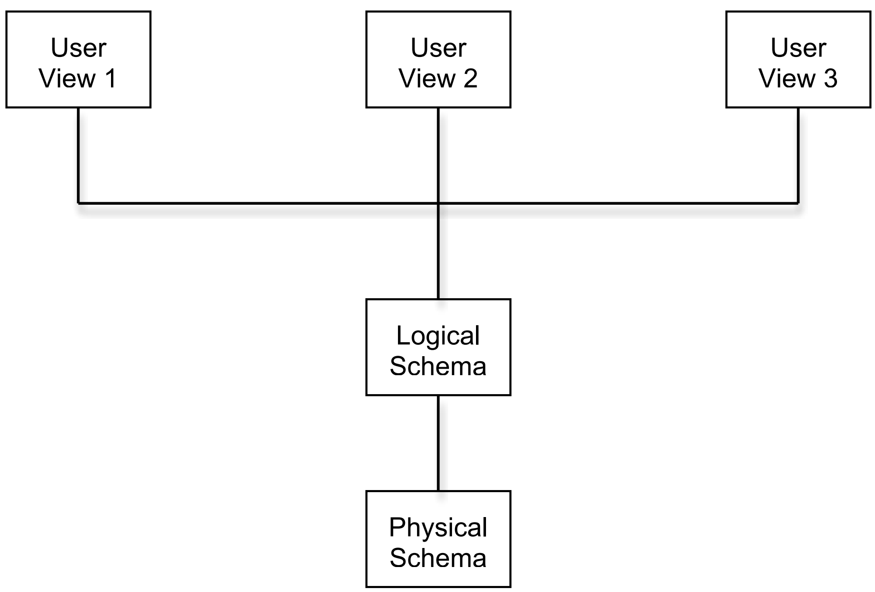

Figure 1-10

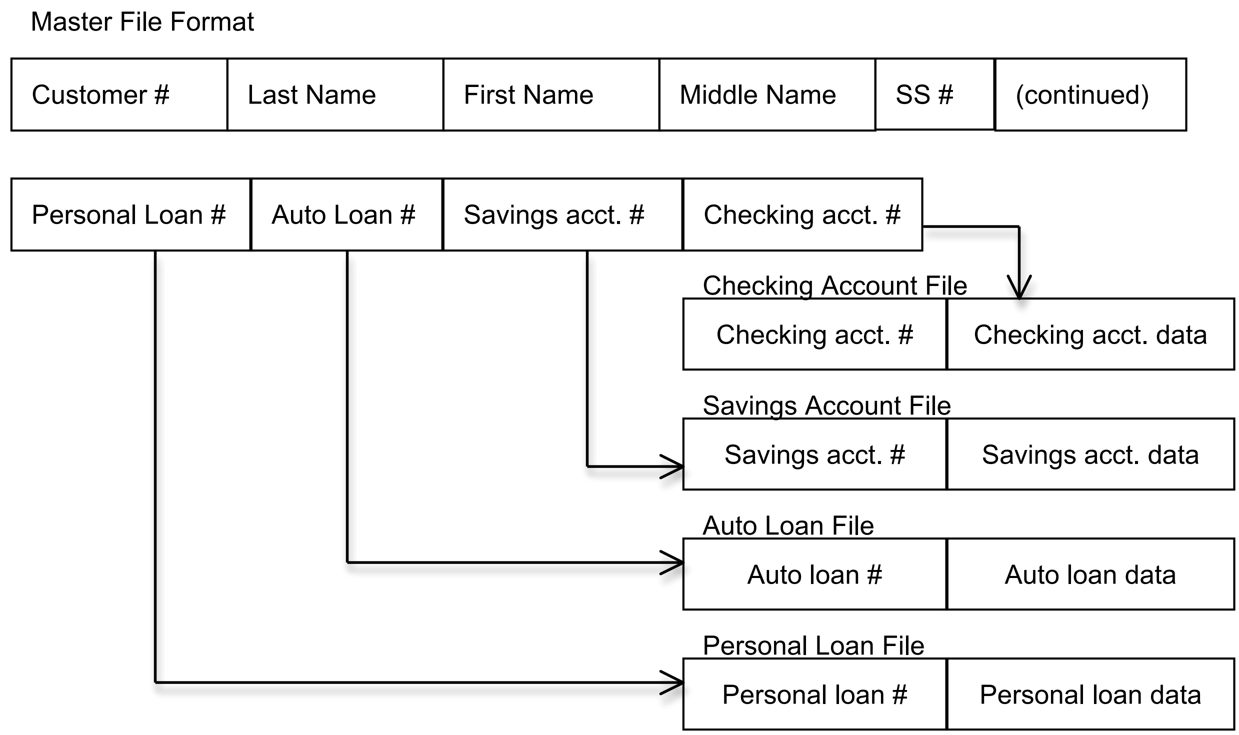
Master File Format
Customer #
Last Name
First Name
Middle Name
SS #
(continued)
Personal Loan #
Auto Loan #
Savings acct. #
Checking acct. #
Checking Account File
Checking acct. #
Checking acct. data
Savings Account File
Savings acct. #
Savings acct. data
Auto Loan File
Auto loan #
Auto loan data
Personal Loan File
Personal loan #
Personal loan data

Figure 1-11a

CUSTOMER_DATA

Customer#	Last_Name	First_Name	Middle_Name	SS#

CHECKING_ACCOUNT_DATA

Customer#	Checking_Account#	Checking acct. data

SAVINGS_ACCOUNT_DATA

Customer#	Savings_Account#	Savings acct. data

AUTO_LOAN DATA

Customer#	Auto_Loan#	Auto loan data

PERSONAL_LOAN_DATA

Customer#	Personal_Loan#	Personal loan data

Figure 1-11b

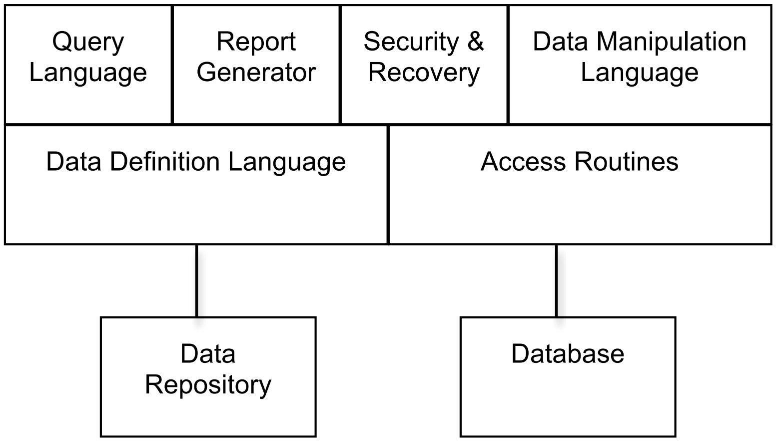
Query Language
Report Generator
Security & Recovery
Data Manipulation Language
Data Definition Language
Access Routines
Data Repository
Database

Figure 1-12