

# Chapter 1

## Database Systems

After completing this chapter, you will be able to:

- Define the difference between data and information
- Describe what a database is, the various types of databases, and why they are valuable assets for decision making
- Explain the importance of database design
- See how modern databases evolved from file systems
- Understand flaws in file system data management
- Outline the main components of the database system
- Describe the main functions of a database management system (DBMS)

## Preview

Organizations use data to keep track of their day-to-day operations. Such data is used to generate information, which in turn is the basis for good decisions. Data is likely to be managed most efficiently when it is stored in a database. Databases are involved in almost all facets and activities of our daily lives: from school to work, medical care, government, nonprofit organizations, and houses of worship. In this chapter, you will learn what a database is, what it does, and why it yields better results than other data management methods. You will also learn about various types of databases and why database design is so important.

Databases evolved from the need to manage large amounts of data in an organized and efficient manner. In the early days, computer file systems were used to organize such data. Although file system data management is now largely outmoded, understanding the characteristics of file systems is important because file systems are the source of serious data management limitations. In this chapter, you will also learn how the database system approach helps eliminate most of the shortcomings of file system data management.

### Data Files and Available Formats

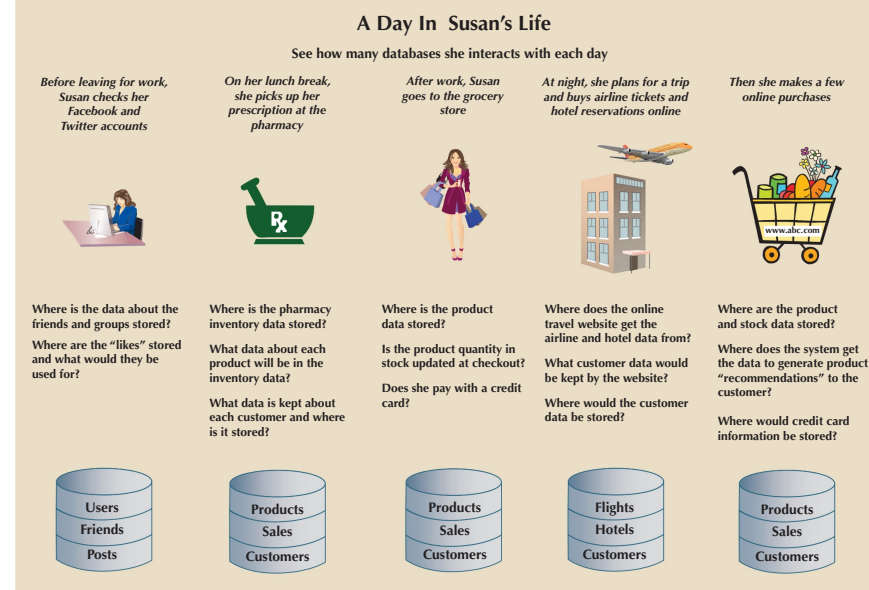
|           | MS Access | Oracle | MS SQL | My SQL |               | MS Access | Oracle | MS SQL | My SQL |
|-----------|-----------|--------|--------|--------|---------------|-----------|--------|--------|--------|
| CH01_Text | ✓         | ✓      | ✓      | ✓      | CH01_Problems | ✓         | ✓      | ✓      | ✓      |

Data Files Available on [cengagebrain.com](http://cengagebrain.com)

## 1-1 Why Databases?

So, why do we need databases? In today's world, data is ubiquitous (abundant, global, everywhere) and pervasive (unescapable, prevalent, persistent). From birth to death, we generate and consume data. The trail of data starts with the birth certificate and continues all the way to a death certificate (and beyond!). In between, each individual produces and consumes enormous amounts of data. As you will see in this book, databases are the best way to store and manage data. Databases make data persistent and shareable in a secure way. As you look at Figure 1.1, can you identify some of the data generated by your own daily activities?

FIGURE 1.1 THE PERVASIVE NATURE OF DATABASES



Data is not only ubiquitous and pervasive; it is also essential for organizations to survive and prosper. Imagine trying to operate a business without knowing who your customers are, what products you are selling, who is working for you, who owes you money, and to whom you owe money. All businesses have to keep this type of data and much more. Just as important, they must have that data available to decision makers when necessary. It can be argued that the ultimate purpose of all business information systems is to help businesses use information as an organizational resource. At the heart of all of these systems are the collection, storage, aggregation, manipulation, dissemination, and management of data.

Depending on the type of information system and the characteristics of the business, this data could vary from a few megabytes on just one or two topics to terabytes covering hundreds of topics within the business's internal and external environment.

Telecommunications companies, such as Sprint and AT&T, are known to have systems that keep data on trillions of phone calls, with new data being added to the system at speeds up to 70,000 calls per second! Not only do these companies have to store and manage immense collections of data but also they have to be able to find any given fact in that data quickly. Consider the case of Internet search staple Google. While Google is reluctant to disclose many details about its data storage specifications, it is estimated that the company responds to over 91 million searches per day across a collection of data that is several terabytes in size. Impressively, the results of these searches are available almost instantly.

How can these businesses process this much data? How can they store it all, and then quickly retrieve just the facts that decision makers want to know, just when they want to know it? The answer is that they use databases. Databases, as explained in detail throughout this book, are specialized structures that allow computer-based systems to store, manage, and retrieve data very quickly. Virtually all modern business systems rely on databases. Therefore, a good understanding of how these structures are created and their proper use is vital for any information systems professional. Even if your career does not take you down the amazing path of database design and development, databases will be a key component of the systems that you use. In any case, you will probably make decisions in your career based on information generated from data. Thus, it is important that you know the difference between data and information.

## 1-2 Data versus Information

To understand what drives database design, you must understand the difference between data and information. **Data** consists of raw facts. The word *raw* indicates that the facts have not yet been processed to reveal their meaning. For example, suppose that a university tracks data on faculty members for reporting to accrediting bodies. To get the data for each faculty member into the database, you would provide a screen to allow for convenient data entry, complete with drop-down lists, combo boxes, option buttons, and other data-entry validation controls. Figure 1.2(a) shows a simple data-entry form from a software package named Sedona. When the data is entered into the form and saved, it is placed in the underlying database as raw data, as shown in Figure 1.2(b). Although you now have the facts in hand, they are not particularly useful in this format. Reading through hundreds of rows of data for faculty members does not provide much insight into the overall makeup of the faculty. Therefore, you transform the raw data into a data summary like the one shown in Figure 1.2(c). Now you can get quick answers to questions such as “What percentage of the faculty in the Information Systems (INFS) department are adjuncts?” In this case, you can quickly determine that 20 percent of the INFS faculty members are adjunct faculty. Because graphics can enhance your ability to quickly extract meaning from data, you show the data summary pie chart in Figure 1.2(d).

**Information** is the result of processing raw data to reveal its meaning. Data processing can be as simple as organizing data to reveal patterns or as complex as making forecasts or drawing inferences using statistical modeling. To reveal meaning, information requires *context*. For example, an average temperature reading of 105 degrees does not mean much unless you also know its context: Is this reading in degrees Fahrenheit or Celsius? Is this a machine temperature, a body temperature, or an outside air temperature? Information can be used as the foundation for decision making. For example, the data summary for the faculty can provide accrediting bodies with insights that are useful in determining whether to renew accreditation for the university.

Keep in mind that raw data must be properly *formatted* for storage, processing, and presentation. For example, dates might be stored in Julian calendar formats within the database, but displayed in a variety of formats, such as day-month-year or month/day/year, for different purposes. Respondents' yes/no responses might need to be converted

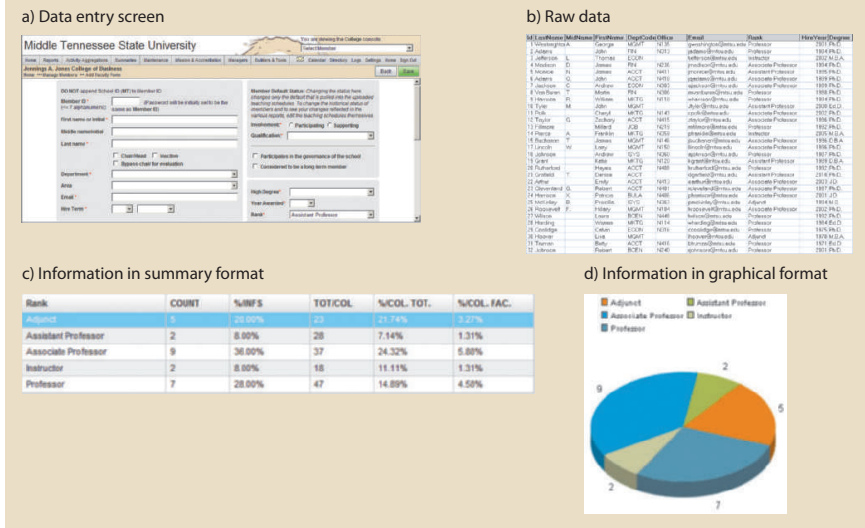
data

Raw facts, or facts that have not yet been processed to reveal their meaning to the end user.

information

The result of processing raw data to reveal its meaning. Information consists of transformed data and facilitates decision making.

FIGURE 1.2 TRANSFORMING RAW DATA INTO INFORMATION



to a Y/N or 0/1 format for data storage. More complex formatting is required when working with complex data types, such as sounds, videos, or images.

In this "information age," production of accurate, relevant, and timely information is the key to good decision making. In turn, good decision making is the key to business survival in a global market. We are now said to be entering the "knowledge age."<sup>1</sup>

Data is the foundation of information, which is the bedrock of **knowledge**—that is, the body of information and facts about a specific subject. Knowledge implies familiarity, awareness, and understanding of information as it applies to an environment. A key characteristic of knowledge is that “new” knowledge can be derived from “old” knowledge.

Let's summarize some key points:

- Data constitutes the building blocks of information.
- Information is produced by processing data.
- Information is used to reveal the meaning of data.
- Accurate, relevant, and timely information is the key to good decision making.
- Good decision making is the key to organizational survival in a global environment.

The previous paragraphs have explained the importance of data, and how the processing of data is used to reveal information that in turn generates “actionable” knowledge. Let’s explore a simple example of how this works in the real world.

In today's information-centric society, we use smartphones on a daily basis. These devices have advanced GPS functionality that constantly tracks your whereabouts. This data is stored and shared with various applications. When you get a new smartphone,

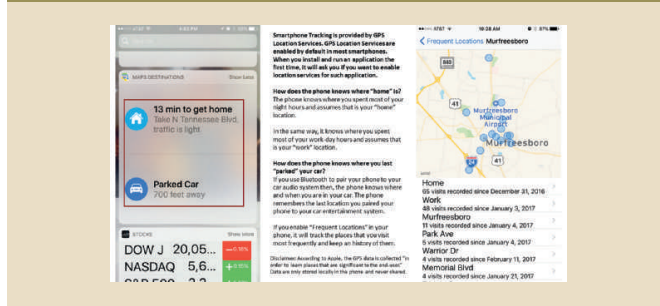
<sup>1</sup> Peter Drucker coined the phrase “knowledge worker” in 1959 in his book *Landmarks of Tomorrow*. In 1994, Esther Dyson, George Keyworth, and Dr. Alvin Toffler introduced the concept of the “knowledge age.”

knowledge

The body of information and facts about a specific subject. Knowledge implies familiarity, awareness, and understanding of information as it applies to an environment. A key characteristic is that new knowledge can be derived from old knowledge.

you can use the map application to go places and to set up your home address (now the phone knows where you live!). The GPS feature in your phone tracks your daily locations. In some cases, the information generated is very helpful: it can help you navigate to various locations and even to find where you parked your car. Figure 1.3 shows screenshots from one of the authors' smartphone. The phone "knows" that this is about the time he goes home and tells him how long it is going to take to get there. It also tells him where he parked his car; if he clicks the Parked Car icon, it will open a map so he can locate the car.

FIGURE 1.3 SMARTPHONE TRACKING



Furthermore, and maybe even scarier in terms of privacy issues, your smartphone may know more about your activities than you imagine. For example, suppose that every Wednesday night you go to the gym and play indoor soccer with your friends. Next Wednesday night, 20 minutes before you leave home, your phone pops up a message saying "19 minutes to [gym address]. Traffic is light." The phone has been storing GPS data on your movements to develop patterns based on days, times, and locations to generate this knowledge. It can then associate such knowledge as your daily activities provide more data points. Imagine that on Wednesday when you go to the Magic Box gym to play soccer, when you arrive you use Facebook on your phone to check in to the gym. Now, your phone also knows the name of the place where you go every Wednesday night.

As you can see from this example, knowledge and information require timely and accurate data. Such data must be properly generated and stored in a format that is easy to access and process. In addition, like any basic resource, the data environment must be managed carefully. **Data management** is a discipline that focuses on the proper generation, storage, and retrieval of data. Given the crucial role that data plays, it should not surprise you that data management is a core activity for any business, government agency, service organization, or charity.

## 1-3 Introducing the Database

Efficient data management typically requires the use of a computer database. A **database** is a shared, integrated computer structure that stores a collection of the following:

- End-user data—that is, raw facts of interest to the end user
- **Metadata**, or data about data, through which the end-user data is integrated and managed

The metadata describes the data characteristics and the set of relationships that links the data found within the database. For example, the metadata component stores information such as the name of each data element, the type of values (numeric, dates, or text) stored on each data element, and whether the data element can be left empty. The metadata provides information that complements and expands the value and use of the data. In short, metadata presents a more complete picture of the data in the database. Given the characteristics of metadata, you might hear a database described as a "collection of *self-describing* data."

A **database management system (DBMS)** is a collection of programs that manages the database structure and controls access to the data stored in the database. In a sense, a database resembles a very well-organized electronic filing cabinet in which powerful software (the DBMS) helps manage the cabinet's contents.

### 1-3a Role and Advantages of the DBMS

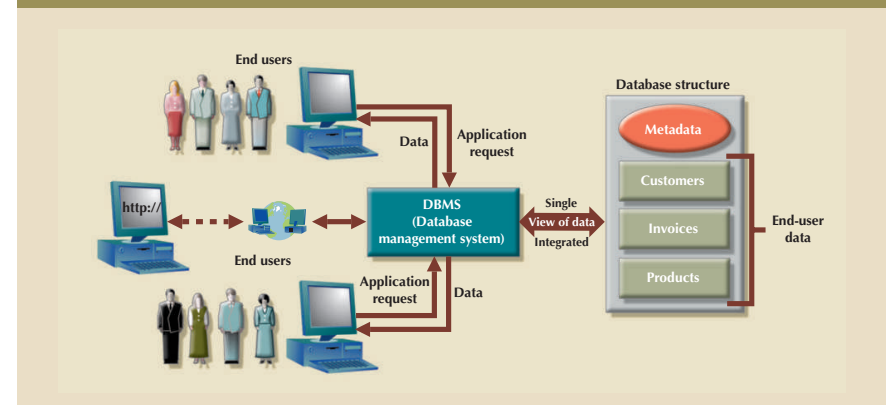
The DBMS serves as the intermediary between the user and the database. The database structure itself is stored as a collection of files, and the only way to access the data in those files is through the DBMS. Figure 1.4 emphasizes the point that the DBMS presents the end user (or application program) with a single, integrated view of the data in the database. The DBMS receives all application requests and translates them into the complex operations required to fulfill those requests. The DBMS hides much of the database's internal complexity from the application programs and users. The application program might be written by a programmer using a programming language, such as Visual Basic.NET, Java, or C#, or it might be created through a DBMS utility program.

Having a DBMS between the end user's applications and the database offers some important advantages. First, the DBMS enables the data in the database *to be shared* among multiple applications or users. Second, the DBMS *integrates* the many different users' views of the data into a single all-encompassing data repository.

#### database management system (DBMS)

The collection of programs that manages the database structure and controls access to the data stored in the database.

FIGURE 1.4 THE DBMS MANAGES THE INTERACTION BETWEEN THE END USER AND THE DATABASE



#### data management

A process that focuses on data collection, storage, and retrieval. Common data management functions include addition, deletion, modification, and listing.

#### database

A shared, integrated computer structure that houses a collection of related data. A database contains two types of data: end-user data (raw facts) and metadata.

#### metadata

Data about data; that is, data about data characteristics and relationships. See also *data dictionary*.

Because data is the crucial raw material from which information is derived, you must have a good method to manage such data. As you will discover in this book, the DBMS helps make data management more efficient and effective. In particular, a DBMS provides these advantages:

- **Improved data sharing.** The DBMS helps create an environment in which end users have better access to more and better-managed data. Such access makes it possible for end users to respond quickly to changes in their environment.
- **Improved data security.** The more users access the data, the greater the risks of data security breaches. Corporations invest considerable amounts of time, effort, and money to ensure that corporate data is used properly. A DBMS provides a framework for better enforcement of data privacy and security policies.
- **Better data integration.** Wider access to well-managed data promotes an integrated view of the organization's operations and a clearer view of the big picture. It becomes much easier to see how actions in one segment of the company affect other segments.
- **Minimized data inconsistency.** **Data inconsistency** exists when different versions of the same data appear in different places. For example, data inconsistency exists when a company's sales department stores a sales representative's name as Bill Brown and the company's personnel department stores that same person's name as William G. Brown, or when the company's regional sales office shows the price of a product as \$45.95 and its national sales office shows the same product's price as \$43.95. The probability of data inconsistency is greatly reduced in a properly designed database.
- **Improved data access.** The DBMS makes it possible to produce quick answers to ad hoc queries. From a database perspective, a **query** is a specific request issued to the DBMS for data manipulation—for example, to read or update the data. Simply put, a query is a question, and an **ad hoc query** is a spur-of-the-moment question. The DBMS sends back an answer (called the **query result set**) to the application. For example, when dealing with large amounts of sales data, end users might want quick answers to questions (ad hoc queries). Some examples are the following:
  - What was the dollar volume of sales by product during the past six months?
  - What is the sales bonus figure for each of our salespeople during the past three months?
  - How many of our customers have credit balances of \$3,000 or more?
- **Improved decision making.** Better-managed data and improved data access make it possible to generate better-quality information, on which better decisions are based. The quality of the information generated depends on the quality of the underlying data. **Data quality** is a comprehensive approach to promoting the accuracy, validity, and timeliness of the data. While the DBMS does not guarantee data quality, it provides a framework to facilitate data quality initiatives. Data quality concepts will be covered in more detail in Chapter 16, Database Administration and Security.
- **Increased end-user productivity.** The availability of data, combined with the tools that transform data into usable information, empowers end users to make quick, informed decisions that can make the difference between success and failure in the global economy.

The advantages of using a DBMS are not limited to the few just listed. In fact, you will discover many more advantages as you learn more about the technical details of databases and their proper design.

## 1-3b Types of Databases

A DBMS can be used to build many different types of databases. Each database stores a particular collection of data and is used for a specific purpose. Over the years, as technology and innovative uses of databases have evolved, different methods have been used to classify databases. For example, databases can be classified by the number of users supported, where the data is located, the type of data stored, the intended data usage, and the degree to which the data is structured.

The number of users determines whether the database is classified as single user or multiuser. A **single-user database** supports only one user at a time. In other words, if user A is using the database, users B and C must wait until user A is done. A single-user database that runs on a personal computer is called a **desktop database**. In contrast, a **multiuser database** supports multiple users at the same time. When the multiuser database supports a relatively small number of users (usually fewer than 50) or a specific department within an organization, it is called a **workgroup database**. When the database is used by the entire organization and supports many users (more than 50, usually hundreds) across many departments, the database is known as an **enterprise database**.

Location might also be used to classify the database. For example, a database that supports data located at a single site is called a **centralized database**. A database that supports data distributed across several different sites is called a **distributed database**. The extent to which a database can be distributed and the way in which such distribution is managed are addressed in detail in Chapter 12, Distributed Database Management Systems.

Both centralized and decentralized (distributed) databases require a well-defined infrastructure (hardware, operating systems, network technologies, etc.) to implement and operate the database. Typically, the infrastructure is owned and maintained by the organization that creates and operates the database. But in recent years, the use of cloud databases has been growing in popularity. A **cloud database** is a database that is created and maintained using cloud data services, such as Microsoft Azure or Amazon AWS. These services, provided by third-party vendors, provide defined performance measures (data storage capacity, required throughput, and availability) for the database, but do not necessarily specify the underlying infrastructure to implement it. The data owners do not have to know, or be concerned about, what hardware and software are being used to support their databases. The performance capabilities can be renegotiated with the cloud provider as the business demands on the database change. For example, 3M Health Information Systems, the world's largest provider of health care analytics software in hospitals, used Amazon's AWS cloud database services to consolidate its multiple IT centers. 3M did not have to buy, install, configure, or maintain any hardware, operating systems, or network devices. It simply purchased storage and processing capacity for its data and applications. As the demands on the databases increased, additional processing and storage capabilities could be purchased as needed. As a result, server provisioning processes that previously took 10 weeks to complete could be done in mere minutes. This allows the company to be more responsive to the needs of customers and innovate faster.

In some contexts, such as research environments, a popular way of classifying databases is according to the type of data stored in them. Using this criterion, databases are grouped into two categories: general-purpose and discipline-specific databases. **General-purpose databases** contain a wide variety of data used in multiple disciplines—for example, a census database that contains general demographic data and the LexisNexis and ProQuest databases that contain newspaper, magazine, and journal articles for a variety of topics. **Discipline-specific databases** contain data focused on specific subject areas. The data in this type of database is used mainly for academic or research purposes

### single-user database

A database that supports only one user at a time.

### desktop database

A single-user database that runs on a personal computer.

### multiuser database

A database that supports multiple concurrent users.

### workgroup database

A multiuser database that usually supports fewer than 50 users or is used for a specific department in an organization.

### enterprise database

The overall company data representation, which provides support for present and expected future needs.

### centralized database

A database located at a single site.

### distributed database

A logically related database that is stored in two or more physically independent sites.

### cloud database

A database that is created and maintained using cloud services, such as Microsoft Azure or Amazon AWS.

### general-purpose database

A database that contains a wide variety of data used in multiple disciplines.

### discipline-specific database

A database that contains data focused on specific subject areas.

### data inconsistency

A condition in which different versions of the same data yield different (inconsistent) results.

### query

A question or task asked by an end user of a database in the form of SQL code. A specific request for data manipulation issued by the end user or the application to the DBMS.

### ad hoc query

A "spur-of-the-moment" question.

### query result set

The collection of data rows returned by a query.

### data quality

A comprehensive approach to ensuring the accuracy, validity, and timeliness of data.



**operational database**

A database designed primarily to support a company's day-to-day operations. Also known as a *transactional database*, *OLTP database*, or *production database*.

**online transaction processing (OLTP) database**

See *operational database*.

**transactional database**

See *operational database*.

**production database**

See *operational database*.

**analytical database**

A database focused primarily on storing historical data and business metrics used for tactical or strategic decision making.

**data warehouse**

A specialized database that stores historical and aggregated data in a format optimized for decision support.

**online analytical processing (OLAP)**

A set of tools that provide advanced data analysis for retrieving, processing, and modeling data from the data warehouse.

**business intelligence**

A set of tools and processes used to capture, collect, integrate, store, and analyze data to support business decision making.

**unstructured data**

Data that exists in its original, raw state; that is, in the format in which it was collected.

**structured data**

Data that has been formatted to facilitate storage, use, and information generation.

**semistructured data**

Data that has already been processed to some extent.

within a small set of disciplines. Examples of discipline-specific databases are financial data stored in databases such as CompuStat or CRSP (Center for Research in Security Prices), geographic information system (GIS) databases that store geospatial and other related data, and medical databases that store confidential medical history data.

The most popular way of classifying databases today, however, is based on how they will be used and on the time sensitivity of the information gathered from them. For example, transactions such as product or service sales, payments, and supply purchases reflect critical day-to-day operations. Such transactions must be recorded accurately and immediately. A database that is designed primarily to support a company's day-to-day operations is classified as an **operational database**, also known as an **online transaction processing (OLTP) database**, **transactional database**, or **production database**. In contrast, an **analytical database** focuses primarily on storing historical data and business metrics used exclusively for tactical or strategic decision making. Such analysis typically requires extensive "data massaging" (data manipulation) to produce information on which to base pricing decisions, sales forecasts, market strategies, and so on. Analytical databases allow the end user to perform advanced analysis of business data using sophisticated tools.

Typically, analytical databases comprise two main components: a data warehouse and an online analytical processing front end. The **data warehouse** is a specialized database that stores data in a format optimized for decision support. The data warehouse contains historical data obtained from the operational databases as well as data from other external sources. **Online analytical processing (OLAP)** is a set of tools that work together to provide an advanced data analysis environment for retrieving, processing, and modeling data from the data warehouse. In recent times, this area of database application has grown in importance and usage, to the point that it has evolved into its own discipline: business intelligence. The term **business intelligence** describes a comprehensive approach to capture and process business data with the purpose of generating information to support business decision making. Chapter 13, Business Intelligence and Data Warehouses, covers this topic in detail.

Databases can also be classified to reflect the degree to which the data is structured. **Unstructured data** is data that exists in its original (raw) state—that is, in the format in which it was collected. Therefore, unstructured data exists in a format that does not lend itself to the processing that yields information. **Structured data** is the result of formatting unstructured data to facilitate storage, use, and generation of information. You apply structure (format) based on the type of processing that you intend to perform on the data. Some data might not be ready (unstructured) for some types of processing, but they might be ready (structured) for other types of processing. For example, the data value 37890 might refer to a zip code, a sales value, or a product code. If this value represents a zip code or a product code and is stored as text, you cannot perform mathematical computations with it. On the other hand, if this value represents a sales transaction, it must be formatted as numeric.

To further illustrate the concept of structure, imagine a stack of printed paper invoices. If you want to merely store these invoices as images for future retrieval and display, you can scan them and save them in a graphic format. On the other hand, if you want to derive information such as monthly totals and average sales, such graphic storage would not be useful. Instead, you could store the invoice data in a (structured) spreadsheet format so that you can perform the requisite computations. Actually, most data you encounter is best classified as semistructured. **Semistructured data** has already been processed to some extent. For example, if you look at a typical webpage, the data is presented in a prearranged format to convey some information. The database types mentioned thus far focus on the storage and management of highly structured data. However, corporations are not limited to the use of structured data.

They also use semistructured and unstructured data. Just think of the valuable information that can be found on company emails, memos, and documents such as procedures, rules, and webpages. Unstructured and semistructured data storage and management needs are being addressed through a new generation of databases known as XML databases. **Extensible Markup Language (XML)** is a special language used to represent and manipulate data elements in a textual format. An **XML database** supports the storage and management of semistructured XML data.

Table 1.1 compares the features of several well-known database management systems.

TABLE 1.1

**TYPES OF DATABASES**

| PRODUCT       | NUMBER OF USERS |           |                      | DATA LOCATION |             | DATA USAGE  |            | XML |
|---------------|-----------------|-----------|----------------------|---------------|-------------|-------------|------------|-----|
|               | SINGLE USER     | WORKGROUP | MULTIUSER ENTERPRISE | CENTRALIZED   | DISTRIBUTED | OPERATIONAL | ANALYTICAL |     |
| MS Access     | X               | X         |                      | X             |             | X           |            |     |
| MS SQL Server | X*              | X         | X                    | X             | X           | X           | X          | X   |
| IBM DB2       | X*              | X         | X                    | X             | X           | X           | X          | X   |
| MySQL         | X               | X         | X                    | X             | X           | X           | X          | X   |
| Oracle RDBMS  | X*              | X         | X                    | X             | X           | X           | X          | X   |

\* Vendor offers single-user/personal or Express DBMS versions

With the emergence of the web and Internet-based technologies as the basis for the new "social media" generation, great amounts of data are being stored and analyzed. **Social media** refers to web and mobile technologies that enable "anywhere, anytime, always on" human interactions. Websites such as Google, Facebook, Twitter, and LinkedIn capture vast amounts of data about end users and consumers. This data grows exponentially and requires the use of specialized database systems. For example, as of 2017, over 648 million tweets were posted every day on Twitter, and that number continues to grow. As a result, the MySQL database Twitter was using to store user content was frequently overloaded by demand.<sup>2</sup> Facebook faces similar challenges. With over 500 terabytes of data coming in each day, it stores over 100 petabytes of data in a single data storage file system. From this data, its database scans over 200 terabytes of data each hour to process user actions, including status updates, picture requests, and billions of "Like" actions.<sup>3</sup> Over the past few years, this new breed of specialized database has grown in sophistication and widespread usage. Currently, this new type of database is known as a NoSQL database. The term **NoSQL** (Not only SQL) is generally used to describe a new generation of DBMS that is not based on the traditional relational database model. NoSQL databases are designed to handle the unprecedented volume of data, variety of data types and structures, and velocity of data operations that are characteristic of these new business requirements. You will learn more about this type of system in Chapter 2, Data Models.

This section briefly mentioned the many different types of databases. As you learned earlier, a database is a computer structure that houses and manages end-user data. One of the first tasks of a database professional is to ensure that end-user data is properly structured to derive valid and timely information. For this, good database design is essential.

<sup>2</sup> [www.internetlivestats.com/twitter-statistics/](http://www.internetlivestats.com/twitter-statistics/)

<sup>3</sup> Josh Constine, "How big is Facebook's data? 2.5 billion pieces of content and 500+ terabytes of data ingested every day," *Tech Crunch*, August 22, 2012, <http://techcrunch.com/2012/08/22/how-big-is-facebooks-data-2-5-billion-pieces-of-content-and-500-terabytes-ingested-every-day/>

**Extensible Markup Language (XML)**

A metalanguage used to represent and manipulate data elements. Unlike other markup languages, XML permits the manipulation of a document's data elements.

**XML database**

A database system that stores and manages semistructured XML data.

**social media**

Web and mobile technologies that enable "anywhere, anytime, always on" human interactions.

**NoSQL**

A new generation of DBMS that is not based on the traditional relational database model.

## 1-4 Why Database Design Is Important

A problem that has evolved with the use of personal productivity tools such as spreadsheets and desktop database programs is that users typically lack proper data-modeling and database design skills. People naturally have a “narrow” view of the data in their environment. For example, consider a student’s class schedule. The schedule probably contains the student’s identification number and name, class code, class description, class credit hours, class instructor name, class meeting days and times, and class room number. In the mind of the student, these various data items compose a single unit. If a student organization wanted to keep a record of the schedules of its members, an end user might make a spreadsheet to store the schedule information. Even if the student makes a foray into the realm of desktop databases, he or she is likely to create a structure composed of a single table that mimics his or her view of the schedule data. As you will learn in the coming chapters, translating this type of narrow view of data into a single two-dimensional table structure is a poor database design choice.

**Database design** refers to the activities that focus on the design of the database structure that will be used to store and manage end-user data. A database that meets all user requirements does not just happen; its structure must be designed carefully. In fact, database design is such a crucial aspect of working with databases that most of this book is dedicated to the development of good database design techniques. Even a good DBMS will perform poorly with a badly designed database.

Data is one of an organization’s most valuable assets. Data on customers, employees, orders, and receipts is all vital to the existence of a company. Tracking key growth and performance indicators are also vital to strategic and tactical plans to ensure future success; therefore, an organization’s data must not be handled lightly or carelessly. Thorough planning to ensure that data is properly used and leveraged to give the company the most benefit is just as important as proper financial planning to ensure that the company gets the best use from its financial resources.

Because current-generation DBMSs are easy to use, an unfortunate side effect is that many computer-savvy business users gain a false sense of confidence in their ability to build a functional database. These users can effectively navigate the creation of database objects, but without the proper understanding of database design, they tend to produce flawed, overly simplified structures that prevent the system from correctly storing data that corresponds to business realities, which produces incomplete or erroneous results when the data is retrieved. Consider the data shown in Figure 1.5, which illustrates the efforts of an organization to keep records about its employees and their skills. Some employees have not passed a certification test in any skill, while others have been certified in several skills. Some certified skills are shared by several employees, while other skills have no employees that hold those certifications.

Based on this storage of the data, notice the following problems:

- It would be difficult, if not impossible, to produce an alphabetical listing of employees based on their last names.
- To determine how many employees are certified in Basic Database Manipulation, you would need a program that counts the number of those certifications recorded in Skill1 and places it in a variable. Then the count of those certifications in Skill2 could be calculated and added to the variable. Finally, the count of those certifications in Skill3 could be calculated and added to the variable to produce the total.
- If you redundantly store the name of a skill with each employee who is certified in that skill, you run the risk of spelling the name differently for different employees. For example, the skill *Basic Database Manipulation* is also entered as *Basic DB Manipulation* for at least one employee in Figure 1.5, which makes it difficult to get an accurate count of employees who have the certification.

www.EBooksWorld.ir

### database design

The process that yields the description of the database structure and determines the database components. The second phase of the database life cycle.

FIGURE 1.5 EMPLOYEE SKILLS CERTIFICATION IN A POOR DESIGN

| ID | EName | Name                  | Title               | HireDate  | Skill                        | SkillDate | Skill2                         | Skill3    | SkillDate                |
|----|-------|-----------------------|---------------------|-----------|------------------------------|-----------|--------------------------------|-----------|--------------------------|
| 1  | 02345 | Brian Coles           | DBA                 | 2/14/1997 | Basic Database Management    | 2/14/2004 | Advanced Database Management   | 2/14/2007 | Basic Web Design         |
| 2  | 06273 | Marco Bianz           | Analyst             | 7/08/2008 | Basic Web Design             | 3/08/2011 | Advanced Process Modeling      | 8/19/2014 |                          |
| 3  | 06234 | Jasmine Patel         | Programmer          | 8/10/2007 | Basic Web Design             | 8/10/2009 | Advanced C# programming        | 8/10/2009 | Basic DB manipulation    |
| 4  | 03373 | Franklin Johnson, Jr. | Purchasing Agent    | 3/15/2004 | Advanced Spreadsheets        | 6/20/2013 |                                |           | 1/29/2014                |
| 5  | 13667 | Almond, Robert        | Analyst             | 9/05/2014 | Basic Process Modeling       | 9/05/2016 | Basic Database Design          | 5/23/2017 |                          |
| 6  | 10282 | Richardson, Amanda    | Clerk               | 4/11/2013 |                              |           |                                |           |                          |
| 7  | 09382 | Susan Mathis          | Database Programmer | 8/2/2012  | Basic DB Design              | 8/2/2014  | Basic Database Manipulation    | 8/2/2014  | Advanced DB Manipulation |
| 8  | 14311 | Duong, Lee            | Programmer          | 9/1/2018  | Basic Web Design             | 9/1/2018  |                                |           | 5/1/2016                 |
| 9  |       |                       |                     |           | Master Database Programming  |           |                                |           |                          |
| 10 |       |                       |                     |           | Basic Spreadsheets           |           |                                |           |                          |
| 11 | 08002 | Wade Gauthier         | Clerk               | 5/20/2012 | Advanced Spreadsheets        | 5/16/2015 | Basic Web Design               | 5/16/2015 |                          |
| 12 | 13383 | Raymond F. Matthews   | Programmer          | 3/12/2014 | Basic C# Programming         | 3/12/2016 |                                |           |                          |
| 13 | 06283 | Chavez, Juan          | Clerk               | 7/4/2012  |                              |           |                                |           |                          |
| 14 | 04893 | Patricia Richards     | DBA                 | 6/11/2005 | Advanced Database Management | 6/11/2008 | Advanced Database Manipulation | 9/20/2014 |                          |
| 15 | 19912 | Lee, Mattan           | Programmer          | 9/29/2014 |                              |           |                                |           |                          |

- The structure of the database will have to be changed by adding more columns to the table when an employee is certified in a fourth skill. It will have to be modified again if an employee is certified in a fifth skill.

Contrast this poor design with that shown in Figure 1.6 where the design has been improved by decomposing the data into three related tables. These tables contain all of the same data that was represented in Figure 1.5, but the tables are structured so that you can easily manipulate the data to view it in different ways and answer simple questions.

With the improved structure in Figure 1.6, you can use simple commands in a standard data manipulation language to do the following:

- Produce an alphabetical listing of employees by last name:
 

```
SELECT * FROM EMPLOYEE ORDER BY EMPLOYEE_LNAME;
```
- Determine how many employees are certified in Basic Database Manipulation:
 

```
SELECT COUNT(*)
FROM SKILL JOIN CERTIFIED ON SKILL.SKILL_ID = CERTIFIED.SKILL_ID
WHERE SKILL_NAME = 'Basic Database Manipulation';
```

You will learn more about these commands in Chapter 7, Introduction to Structured Query Language (SQL).

Note that because each skill name is stored only once, the names cannot be spelled or abbreviated differently for different employees. Also, the additional certification of an employee with a fourth or fifth skill does not require changes to the structure of the tables.

Proper database design requires the designer to identify precisely the database’s expected use. Designing a transactional database emphasizes accurate and consistent data and operational speed. Designing a data warehouse database emphasizes the use of historical and aggregated data. Designing a database to be used in a centralized, single-user environment requires a different approach from that used in the design of a distributed, multiuser database. This book emphasizes the design of transactional, centralized, single-user, and multiuser databases. Chapters 12 and 13 also examine critical issues confronting the designer of distributed and data warehouse databases.

www.EBooksWorld.ir

FIGURE 1.6 EMPLOYEE SKILLS CERTIFICATION IN A GOOD DESIGN

Table name: EMPLOYEE

| Employee_ID | Employee_FName | Employee_LName | Employee_HireDate | Employee_Title      |
|-------------|----------------|----------------|-------------------|---------------------|
| 02345       | Brian          | Oates          | 2/14/1999         | DBA                 |
| 03373       | Franklin       | Johnson        | 3/15/2006         | Purchasing Agent    |
| 04993       | Patricia       | Richards       | 6/11/2008         | DBA                 |
| 05234       | Jasmine        | Patel          | 8/10/2009         | Programmer          |
| 05273       | Marco          | Bianz          | 7/26/2010         | Analyst             |
| 09002       | Wade           | Gather         | 5/29/2014         | Clerk               |
| 09393       | Juan           | Chavez         | 7/4/2014          | Clerk               |
| 09392       | Susan          | Mathis         | 8/2/2014          | Database Programmer |
| 10382       | Amanda         | Richardson     | 4/11/2015         | Clerk               |
| 13383       | Raymond        | Matthews       | 3/12/2016         | Programmer          |
| 13657       | Robert         | Almont         | 9/20/2016         | Analyst             |
| 13992       | Megan          | Lee            | 9/29/2017         | Programmer          |
| 14311       | Lee            | Duong          | 9/1/2018          | Programmer          |

Database name: Ch01\_Text

Table name: CERTIFIED

| Employee_ID | Skill_ID | Certified_Date |
|-------------|----------|----------------|
| 02345       | 100      | 2/14/2004      |
| 02345       | 110      | 8/9/2005       |
| 02345       | 180      | 2/14/2007      |
| 03373       | 120      | 6/20/2013      |
| 04993       | 180      | 6/11/2009      |
| 04993       | 220      | 9/23/2014      |
| 06234       | 110      | 8/10/2009      |
| 06234       | 200      | 8/10/2009      |
| 06234       | 210      | 1/29/2014      |
| 06273       | 110      | 3/8/2011       |
| 06273       | 190      | 8/19/2014      |
| 09002       | 110      | 5/16/2015      |
| 09002       | 120      | 5/16/2015      |
| 09392       | 140      | 9/2/2014       |
| 09392       | 210      | 8/2/2014       |
| 09392       | 220      | 5/1/2015       |
| 13383       | 170      | 3/12/2016      |
| 13657       | 130      | 9/30/2016      |
| 13657       | 140      | 5/23/2017      |
| 14311       | 110      | 9/1/2018       |

Table name: SKILL

| Skill_ID | Skill_Name                     | Skill_Description   |
|----------|--------------------------------|---|
| 100      | Basic Database Management      | Create and manage database user accounts.   |
| 110      | Basic Web Design               | Create and maintain HTML and CSS documents.   |
| 120      | Advanced Spreadsheets          | Use of advanced functions, user-defined functions, and macroing.  |
| 130      | Basic Process Modeling         | Create core business process models using standard libraries.   |
| 140      | Basic Database Design          | Create simple data models.  |
| 150      | Master Database Programming    | Create integrated trigger and procedure packages for a distributed environment.                               |
| 160      | Basic Spreadsheets             | Create single tab worksheets with basic formulas.   |
| 170      | Basic C# Programming           | Create single-tier data aware modules.  |
| 180      | Advanced Database Management   | Manage Database Server Clusters.  |
| 190      | Advance Process Modeling       | Evaluate and Redesign cross-functional internal and external business processes.                              |
| 200      | Advanced C# Programming        | Create multi-tier applications using multi-threading.   |
| 210      | Basic Database Manipulation    | Create simple data retrieval and manipulation statements in SQL.  |
| 220      | Advanced Database Manipulation | Use of advanced data manipulation methods for multi-table inserts, set operations, and correlated subqueries. |

Designing appropriate data repositories of integrated information using the two-dimensional table structures found in most databases is a process of decomposition. The integrated data must be decomposed properly into its constituent parts, with each part stored in its own table. Further, the relationships between these tables must be carefully considered and implemented so the integrated view of the data can be recreated later as information for the end user. A well-designed database facilitates data management and generates accurate and valuable information. A poorly designed database is likely to become a breeding ground for difficult-to-trace errors that may lead to poor decision making—and poor decision making can lead to the failure of an organization. Database design is simply too important to be left to luck. That's why college students study database design, why organizations of all types and sizes send personnel to database design seminars, and why database design consultants often make an excellent living.

www.EBooksWorld.ir

## 1-5 Evolution of File System Data Processing

Understanding what a database is, what it does, and the proper way to use it can be clarified by considering what a database is not. A brief explanation of the evolution of file system data processing can be helpful in understanding the data access limitations that databases attempt to overcome. Understanding these limitations is relevant to database designers and developers because database technologies do not make these problems magically disappear—database technologies simply make it easier to create solutions that avoid these problems. Creating database designs that avoid the pitfalls of earlier systems requires that the designer understand these problems and how to avoid them; otherwise, the database technologies are no better (and are potentially even worse!) than the technologies and techniques they have replaced.

### 1-5a Manual File Systems

To be successful, an organization must develop systems for handling core business tasks. Historically, such systems were often manual, paper-and-pencil systems. The papers within these systems were organized to facilitate the expected use of the data. Typically, this was accomplished through a system of file folders and filing cabinets. As long as a collection of data was relatively small and an organization's business users had few reporting requirements, the manual system served its role well as a data repository. However, as organizations grew and as reporting requirements became more complex, keeping track of data in a manual file system became more difficult. Therefore, companies looked to computer technology for help.

### 1-5b Computerized File Systems

Generating reports from manual file systems was slow and cumbersome. In fact, some business managers faced government-imposed reporting requirements that led to weeks of intensive effort each quarter, even when a well-designed manual system was used. Therefore, a **data processing (DP) specialist** was hired to create a computer-based system that would track data and produce required reports.

Initially, the computer files within the file system were similar to the manual files. A simple example of a customer data file for a small insurance company is shown in Figure 1.7. (You will discover later that the file structure shown in Figure 1.7, although typically found in early file systems, is unsatisfactory for a database.)

#### data processing (DP) specialist

The person responsible for developing and managing a computerized file processing system.

FIGURE 1.7 CONTENTS OF THE CUSTOMER FILE

| Database name: Ch01_Text |              |                        |       |              |              |    |        |             |  |
|--------------------------|--------------|------------------------|-------|--------------|--------------|----|--------|-------------|--|
| C_NAME                   | C_PHONE      | C_ADDRESS              | C_ZIP | A_NAME       | A_PHONE      | TP | AMT    | REN         |  |
| Alfred A. Ramas          | 615-844-2573 | 218 Fork Rd., Babs, TN | 36123 | Leah F. Hahn | 615-882-1244 | T1 | 100.00 | 05-Apr-2016 |  |
| Leona K. Dunne           | 713-894-1238 | Box 124, Fox, KY       | 25246 | Alex B. Alby | 713-228-1249 | T1 | 250.00 | 16-Jun-2016 |  |
| Kathy W. Smith           | 615-894-2285 | 125 Oak Ln, Babs, TN   | 36123 | Leah F. Hahn | 615-882-2144 | S2 | 150.00 | 29-Jan-2015 |  |
| Paul F. Olowski          | 615-894-2180 | 217 Lee Ln., Babs, TN  | 36123 | Leah F. Hahn | 615-882-1244 | S1 | 300.00 | 14-Oct-2016 |  |
| Myron Orlando            | 615-222-1672 | Box 111, New, TN       | 36155 | Alex B. Alby | 713-228-1249 | T1 | 100.00 | 28-Dec-2016 |  |
| Amy B. O'Brian           | 713-442-3381 | 387 Troll Dr., Fox, KY | 25246 | John T. Okon | 615-123-5589 | T2 | 850.00 | 22-Sep-2016 |  |
| James G. Brown           | 615-297-1228 | 21 Tye Rd., Nash, TN   | 37118 | Leah F. Hahn | 615-882-1244 | S1 | 120.00 | 25-Mar-2015 |  |
| George Williams          | 615-290-2556 | 155 Maple, Nash, TN    | 37119 | John T. Okon | 615-123-5589 | S1 | 250.00 | 17-Jul-2016 |  |
| Anne G. Farriss          | 713-382-7185 | 2119 Elm, Crew, KY     | 25432 | Alex B. Alby | 713-228-1249 | T2 | 100.00 | 03-Dec-2016 |  |
| Olette K. Smith          | 615-297-3809 | 2782 Main, Nash, TN    | 37118 | John T. Okon | 615-123-5589 | S2 | 500.00 | 14-Mar-2015 |  |


C\_NAME = Customer name  
C\_PHONE = Customer phone  
C\_ADDRESS = Customer address  
C\_ZIP = Customer zip code

A\_NAME = Agent name  
A\_PHONE = Agent phone  
TP = Insurance type  
AMT = Insurance policy amount, in thousands of \$  
REN = Insurance renewal date

www.EBooksWorld.ir

The description of computer files requires a specialized vocabulary. Every discipline develops its own terminology to enable its practitioners to communicate clearly. The basic file vocabulary shown in Table 1.2 will help you to understand subsequent discussions more easily.

| TABLE 1.2              |   |
|------------------------|---|
| BASIC FILE TERMINOLOGY |   |
| TERM                   | DEFINITION  |
| Data                   | Raw facts, such as a telephone number, a birth date, a customer name, and a year-to-date (YTD) sales value. Data has little meaning unless it has been organized in some logical manner.  |
| Field                  | A character or group of characters (alphabetic or numeric) that has a specific meaning. A field is used to define and store data.   |
| Record                 | A logically connected set of one or more fields that describes a person, place, or thing. For example, the fields that constitute a record for a customer might consist of the customer's name, address, phone number, date of birth, credit limit, and unpaid balance. |
| File                   | A collection of related records. For example, a file might contain data about the students currently enrolled at Gigantic University.   |



**Online Content**

The databases used in each chapter are available at [www.cengagebrain.com](http://www.cengagebrain.com). Throughout the book, Online Content boxes highlight material related to chapter content on the website.

**field**

A character or group of characters (alphabetic or numeric) that has a specific meaning. A field is used to define and store data.

**record**

A logically connected set of one or more fields that describes a person, place, or thing.

**file**

A collection of related records. For example, a file might contain data about the students currently enrolled at Gigantic University.

FIGURE 1.8 CONTENTS OF THE AGENT FILE

| Database name: Ch01_Text |              |                    |       |             |          |         |          |           |     |
|--------------------------|--------------|--------------------|-------|-------------|----------|---------|----------|-----------|-----|
| A_NAME                   | A_PHONE      | A_ADDRESS          | ZIP   | HIRED       | YTD_PAY  | YTD_FIT | YTD_FICA | YTD_SLS   | DEP |
| Alex B. Alby             | 713-228-1249 | 123 Toll, Nash, TN | 37119 | 01-Nov-2000 | 26566.24 | 6641.56 | 2125.30  | 132737.75 | 3   |
| Leah F. Hahn             | 615-882-1244 | 334 Main, Fox, KY  | 25246 | 23-May-1996 | 32213.78 | 8053.44 | 2577.10  | 138967.35 | 0   |
| John T. Okon             | 615-123-5589 | 452 Elm, Newr, TN  | 36155 | 15-Jun-2005 | 23198.29 | 5799.57 | 1855.86  | 127093.45 | 2   |

A\_NAME = Agent name

A\_PHONE = Agent phone

A\_ADDRESS = Agent address

ZIP = Agent zip code

HIRED = Agent date of hire

YTD\_PAY = Year-to-date pay

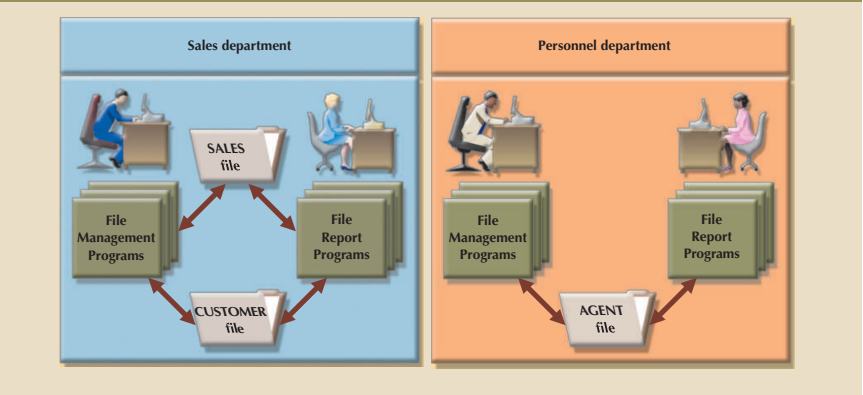
YTD\_FIT = Year-to-date federal income tax paid

YTD\_FICA = Year-to-date Social Security taxes paid

YTD\_SLS = Year-to-date sales

DEP = Number of dependents

FIGURE 1.9 A SIMPLE FILE SYSTEM



forward in a business's ability to process data. Previously, users had direct, hands-on access to all of the business data. But they didn't have the tools to convert that data into the information they needed. The creation of computerized file systems gave them improved tools for manipulating the company data that allowed them to create new information. However, it had the additional effect of introducing a schism between the end users and their data. The desire to close the gap between the end users and the data influenced the development of many types of computer technologies, system designs, and uses (and misuses) of many technologies and techniques. However, such developments also created a split between the ways DP specialists and end users viewed the data.

- From the DP specialist's perspective, the computer files within the file system were created to be similar to the manual files. Data management programs were created to add to, update, and delete data from the file.
- From the end user's perspective, the systems separated the users from the data. As the users' competitive environment pushed them to make more and more decisions in less time, users became frustrated by the delay between conceiving of a new way to create information from the data and the point when the DP specialist actually created the programs to generate that information.



### 1-5c File System Redux: Modern End-User Productivity Tools

The users' desire for direct, hands-on access to data helped to fuel the adoption of personal computers for business use. Although not directly related to file system evolution, the ubiquitous use of personal productivity tools can introduce the same problems as the old file systems.

Personal computer spreadsheet programs such as Microsoft Excel are widely used by business users, and they allow the user to enter data in a series of rows and columns so the data can be manipulated using a wide range of functions. The popularity of spreadsheet applications has enabled users to conduct sophisticated data analysis that has greatly enhanced their ability to understand the data and make better decisions. Unfortunately, as in the old adage "When the only tool you have is a hammer, every problem looks like a nail," users have become so adept at working with spreadsheets that they tend to use them to complete tasks for which spreadsheets are not appropriate.

A common misuse of spreadsheets is as a substitute for a database. Interestingly, end users often take the limited data to which they have direct access and place it in a spreadsheet format similar to that of the traditional, manual data storage systems—which is precisely what the early DP specialists did when creating computerized data files. Due to the large number of users with spreadsheets, each making separate copies of the data, the resulting "file system" of spreadsheets suffers from the same problems as the file systems created by the early DP specialists, which are outlined in the next section.

## 1-6 Problems with File System Data Processing

The file system method of organizing and managing data was a definite improvement over the manual system, and the file system served a useful purpose in data management for over two decades—a very long time in the computer era. Nonetheless, many problems and limitations became evident in this approach. A critique of the file system method serves two major purposes:

- Understanding the shortcomings of the file system enables you to understand the development of modern databases.
- Failure to understand such problems is likely to lead to their duplication in a database environment, even though database technology makes it easy to avoid them.

The following problems associated with file systems, whether created by DP specialists or through a series of spreadsheets, severely challenge the types of information that can be created from the data as well as the accuracy of the information:

- *Lengthy development times.* The first and most glaring problem with the file system approach is that even the simplest data-retrieval task requires extensive programming. With the older file systems, programmers had to specify what must be done and how to do it. As you will learn in upcoming chapters, modern databases use a nonprocedural data manipulation language that allows the user to specify what must be done without specifying how.
- *Difficulty of getting quick answers.* The need to write programs to produce even the simplest reports makes ad hoc queries impossible. Harried DP specialists who worked with mature file systems often received numerous requests for new reports. They were often forced to say that the report will be ready "next week" or even "next month." If you need the information now, getting it next week or next month will not serve your information needs.
- *Complex system administration.* System administration becomes more difficult as the number of files in the system expands. Even a simple file system with a few files requires creating and maintaining several file management programs. Each file must

have its own file management programs that allow the user to add, modify, and delete records; to list the file contents; and to generate reports. Because ad hoc queries are not possible, the file reporting programs can multiply quickly. The problem is compounded by the fact that each department in the organization "owns" its data by creating its own files.

- *Lack of security and limited data sharing.* Another fault of a file system data repository is a lack of security and limited data sharing. Data sharing and security are closely related. Sharing data among multiple geographically dispersed users introduces a lot of security risks. In terms of spreadsheet data, while many spreadsheet programs provide rudimentary security options, they are not always used, and even when they are, they are insufficient for robust data sharing among users. In terms of creating data management and reporting programs, security and data-sharing features are difficult to program and consequently are often omitted from a file system environment. Such features include effective password protection, the ability to lock out parts of files or parts of the system itself, and other measures designed to safeguard data confidentiality. Even when an attempt is made to improve system and data security, the security devices tend to be limited in scope and effectiveness.
- *Extensive programming.* Making changes to an existing file structure can be difficult in a file system environment. For example, changing just one field in the original CUSTOMER file would require a program that:
  1. reads a record from the original file
  2. transforms the original data to conform to the new structure's storage requirements
  3. writes the transformed data into the new file structure
  4. repeats the preceding steps for each record in the original file.

In fact, any change to a file structure, no matter how minor, forces modifications in all of the programs that use the data in that file. Modifications are likely to produce errors (bugs), and additional time is spent using a debugging process to find those errors. Those limitations, in turn, lead to problems of structural and data dependence.

### 1-6a Structural and Data Dependence

A file system exhibits **structural dependence**, which means that access to a file is dependent on its structure. For example, adding a customer date-of-birth field to the CUSTOMER file shown in Figure 1.7 would require the four steps described in the previous section. Given this change, none of the previous programs will work with the new CUSTOMER file structure. Therefore, all of the file system programs must be modified to conform to the new file structure. In short, because the file system application programs are affected by changes in the file structure, they exhibit structural dependence. Conversely, **structural independence** exists when you can change the file structure without affecting the application's ability to access the data.

Even changes in the characteristics of data, such as changing a field from integer to decimal, require changes in all the programs that access the file. Because all data access programs are subject to change when any of the file's data storage characteristics change (that is, changing the **data type**), the file system is said to exhibit **data dependence**. Conversely, **data independence** exists when you can change the data storage characteristics without affecting the program's ability to access the data.

The practical significance of data dependence is the difference between the **logical data format** (how the human being views the data) and the **physical data format** (how the computer must work with the data). Any program that accesses a file system's file must tell the computer not only what to do but also how to do it. Consequently, each

#### structural dependence

A data characteristic in which a change in the database schema affects data access, thus requiring changes in all access programs.

#### structural independence

A data characteristic in which changes in the database schema do not affect data access.

#### data type

Defines the kind of values that can be used or stored. Also, used in programming languages and database systems to determine the operations that can be applied to such data.

#### data dependence

A data condition in which data representation and manipulation are dependent on the physical data storage characteristics.

#### data independence

A condition in which data access is unaffected by changes in the physical data storage characteristics.

#### logical data format

The way a person views data within the context of a problem domain.

#### physical data format

The way a computer "sees" (stores) data.

program must contain lines that specify the opening of a specific file type, its record specification, and its field definitions. Data dependence makes the file system extremely cumbersome from the point of view of a programmer and database manager.

### 1-6b Data Redundancy

The file system's structure makes it difficult to combine data from multiple sources, and its lack of security renders the file system vulnerable to security breaches. The organizational structure promotes the storage of the same basic data in different locations. (Database professionals use the term **islands of information** for such scattered data locations.) The dispersion of data is exacerbated by the use of spreadsheets to store data. In a file system, the entire sales department would share access to the SALES data file through the data management and reporting programs created by the DP specialist. With the use of spreadsheets, each member of the sales department can create his or her own copy of the sales data. Because data stored in different locations will probably not be updated consistently, the islands of information often contain different versions of the same data. For example, in Figures 1.7 and 1.8, the agent names and phone numbers occur in both the CUSTOMER and the AGENT files. You only need one correct copy of the agent names and phone numbers. Having them occur in more than one place produces data redundancy. **Data redundancy** exists when the same data is stored unnecessarily at different places.

Uncontrolled data redundancy sets the stage for the following:

- **Poor data security.** Having multiple copies of data increases the chances for a copy of the data to be susceptible to unauthorized access. Chapter 16, Database Administration and Security, explores the issues and techniques associated with securing data.
- **Data inconsistency.** Data inconsistency exists when different and conflicting versions of the same data appear in different places. For example, suppose you change an agent's phone number in the AGENT file. If you forget to make the corresponding change in the CUSTOMER file, the files contain different data for the same agent. Reports will yield inconsistent results that depend on which version of the data is used.
- **Data-entry errors.** Data-entry errors are more likely to occur when complex entries (such as 10-digit phone numbers) are made in several different files or recur frequently in one or more files. In fact, the CUSTOMER file shown in Figure 1.7 contains just such an entry error: the third record in the CUSTOMER file has transposed digits in the agent's phone number (615-882-2144 rather than 615-882-1244).
- **Data integrity problems.** It is possible to enter a nonexistent sales agent's name and phone number into the CUSTOMER file, but customers are not likely to be impressed if the insurance agency supplies the name and phone number of an agent who does not exist. Should the personnel manager allow a nonexistent agent to accrue bonuses and benefits? In fact, a data-entry error such as an incorrectly spelled name or an incorrect phone number yields the same kind of data integrity problems.

#### islands of information

In the old file system environment, pools of independent, often duplicated, and inconsistent data created and managed by different departments.

#### data redundancy

Exists when the same data is stored unnecessarily at different places.

#### data integrity

In a relational database, a condition in which the data in the database complies with all entity and referential integrity constraints.



#### Note

Data that displays data inconsistency is also referred to as data that lacks data integrity. **Data integrity** is defined as the condition in which all of the data in the database is consistent with the real-world events and conditions. In other words, data integrity means the following:

- Data is *accurate*—there are no data inconsistencies.
- Data is *verifiable*—the data will always yield consistent results.

### 1-6c Data Anomalies

The dictionary defines *anomaly* as “an abnormality.” Ideally, a field value change should be made in only a single place. Data redundancy, however, fosters an abnormal condition by forcing field value changes in many different locations. Look at the CUSTOMER file in Figure 1.7. If agent Leah F. Hahn decides to get married and move, the agent's name, address, and phone number are likely to change. Instead of making these changes in a single file (AGENT), you must also make the change each time that agent's name and phone number occur in the CUSTOMER file. You could be faced with the prospect of making hundreds of corrections, one for each of the customers served by that agent! The same problem occurs when an agent decides to quit. Each customer served by that agent must be assigned a new agent. Any change in any field value must be correctly made in many places to maintain data integrity. A **data anomaly** develops when not all of the required changes in the redundant data are made successfully. The data anomalies found in Figure 1.7 are commonly defined as follows:

- **Update anomalies.** If agent Leah F. Hahn has a new phone number, it must be entered in each of the CUSTOMER file records in which Ms. Hahn's phone number is shown. In this case, only four changes must be made. In a large file system, such a change might occur in hundreds or even thousands of records. Clearly, the potential for data inconsistencies is great.
- **Insertion anomalies.** If only the CUSTOMER file existed and you needed to add a new agent, you would also add a dummy customer data entry to reflect the new agent's addition. Again, the potential for creating data inconsistencies would be great.
- **Deletion anomalies.** If you delete the customers Amy B. O'Brian, George Williams, and Olette K. Smith, you will also delete John T. Okon's agent data. Clearly, this is not desirable.

On a positive note, however, this book will help you develop the skills needed to design and model a successful database that avoids the problems listed in this section.

## 1-7 Database Systems

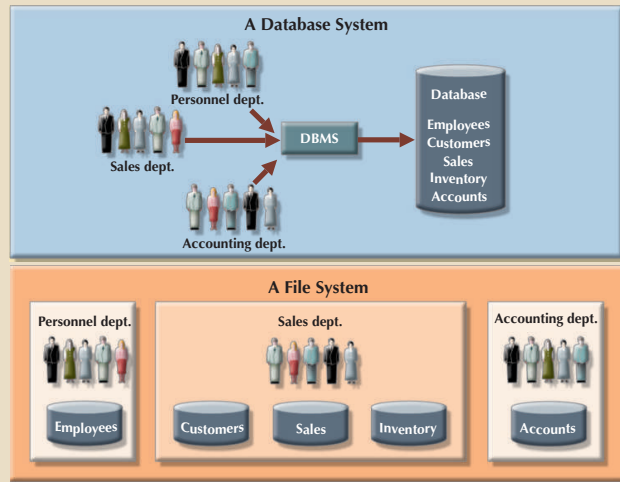
The problems inherent in file systems make using a database system very desirable. Unlike the file system, with its many separate and unrelated files, the database system consists of logically related data stored in a single logical data repository. (The “logical” label reflects the fact that the data repository appears to be a single unit to the end user, even though data might be physically distributed among multiple storage facilities and locations.) Because the database's data repository is a single logical unit, the database represents a major change in the way end-user data is stored, accessed, and managed. The database's DBMS, shown in Figure 1.10, provides numerous advantages over file system management, shown in Figure 1.9, by making it possible to eliminate most of the file system's data inconsistency, data anomaly, data dependence, and structural dependence problems. Better yet, the current generation of DBMS software stores not only the data structures but also the relationships between those structures and the access paths to those structures—all in a central location. The current generation of DBMS software also takes care of defining, storing, and managing all required access paths to those components.

Remember that the DBMS is just one of several crucial components of a database system. The DBMS may even be referred to as the database system's heart. However, just as it takes more than a heart to make a human being function, it takes more than a

#### data anomaly

A data abnormality in which inconsistent changes have been made to a database. For example, an employee moves, but the address change is not corrected in all files in the database.

FIGURE 1.10 CONTRASTING DATABASE AND FILE SYSTEMS



DBMS to make a database system function. In the sections that follow, you'll learn what a database system is, what its components are, and how the DBMS fits into the picture.

### 1-7a The Database System Environment

The term **database system** refers to an organization of components that define and regulate the collection, storage, management, and use of data within a database environment. From a general management point of view, the database system is composed of the five major parts shown in Figure 1.11: hardware, software, people, procedures, and data.

Let's take a closer look at the five components shown in Figure 1.11:

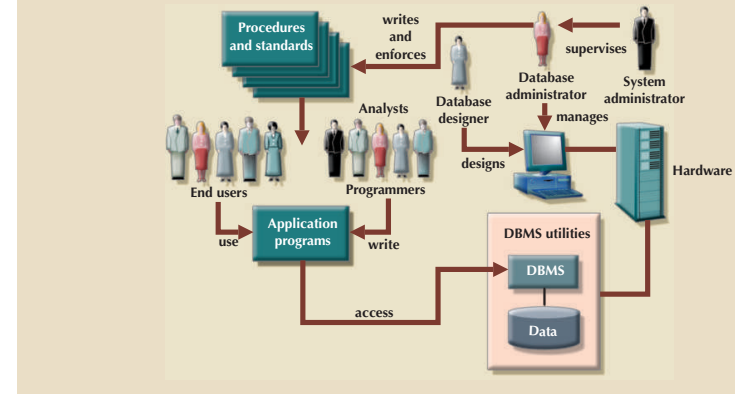
- **Hardware.** Hardware refers to all of the system's physical devices, including computers (PCs, tablets, workstations, servers, and supercomputers), storage devices, printers, network devices (hubs, switches, routers, fiber optics), and other devices (automated teller machines, ID readers, and so on).
- **Software.** Although the most readily identified software is the DBMS itself, three types of software are needed to make the database system function fully: operating system software, DBMS software, and application programs and utilities.
  - *Operating system software* manages all hardware components and makes it possible for all other software to run on the computers. Examples of operating system software are Microsoft Windows, Linux, Mac OS, UNIX, and MVS.
  - *DBMS software* manages the database within the database system. Some examples of DBMS software are Microsoft's SQL Server, Oracle Corporation's Oracle, Oracle's MySQL, and IBM's DB2.

#### database system

An organization of components that defines and regulates the collection, storage, management, and use of data in a database environment.

www.EBooksWorld.ir

FIGURE 1.11 THE DATABASE SYSTEM ENVIRONMENT



- *Application programs and utility software* are used to access and manipulate data in the DBMS and to manage the computer environment in which data access and manipulation take place. Application programs are most commonly used to access data within the database to generate reports, tabulations, and other information to facilitate decision making. Utilities are the software tools used to help manage the database system's computer components. For example, all of the major DBMS vendors now provide graphical user interfaces (GUIs) to help create database structures, control database access, and monitor database operations.
- **People.** This component includes all users of the database system. On the basis of primary job functions, five types of users can be identified in a database system: system administrators, database administrators, database designers, system analysts and programmers, and end users. Each user type, described next, performs both unique and complementary functions.
  - *System administrators* oversee the database system's general operations.
  - *Database administrators*, also known as DBAs, manage the DBMS and ensure that the database is functioning properly. The DBA's role is sufficiently important to warrant a detailed exploration in Chapter 16, Database Administration and Security.
  - *Database designers* design the database structure. They are, in effect, the database architects. If the database design is poor, even the best application programmers and the most dedicated DBAs cannot produce a useful database environment. Because organizations strive to optimize their data resources, the database designer's job description has expanded to cover new dimensions and growing responsibilities.
  - *System analysts and programmers* design and implement the application programs. They design and create the data-entry screens, reports, and procedures through which end users access and manipulate the database's data.
  - *End users* are the people who use the application programs to run the organization's daily operations. For example, sales clerks, supervisors, managers, and directors are all classified as end users. High-level end users employ the information obtained from the database to make tactical and strategic business decisions.

www.EBooksWorld.ir

- **Procedures.** Procedures are the instructions and rules that govern the design and use of the database system. Procedures are a critical, although occasionally forgotten, component of the system. Procedures play an important role in a company because they enforce the standards by which business is conducted within the organization and with customers. Procedures also help to ensure that companies have an organized way to monitor and audit the data that enter the database and the information generated from those data.
- **Data.** The word *data* covers the collection of facts stored in the database. Because data is the raw material from which *information* is generated, determining which data to enter into the database and how to organize that data is a vital part of the database designer's job.

A database system adds a new dimension to an organization's management structure. The complexity of this managerial structure depends on the organization's size, its functions, and its corporate culture. Therefore, database systems can be created and managed at different levels of complexity and with varying adherence to precise standards. For example, compare a local convenience store system with a national insurance claims system. The convenience store system may be managed by two people, the hardware used is probably a single computer, the procedures are probably simple, and the data volume tends to be low. The national insurance claims system is likely to have at least one systems administrator, several full-time DBAs, and many designers and programmers; the hardware probably includes several servers at multiple locations throughout the United States; the procedures are likely to be numerous, complex, and rigorous; and the data volume tends to be high.

In addition to the different levels of database system complexity, managers must also take another important fact into account: database solutions must be cost-effective as well as tactically and strategically effective. Producing a million-dollar solution to a thousand-dollar problem is hardly an example of good database system selection or of good database design and management. Finally, the database technology already in use is likely to affect the selection of a database system.

### 1-7b DBMS Functions

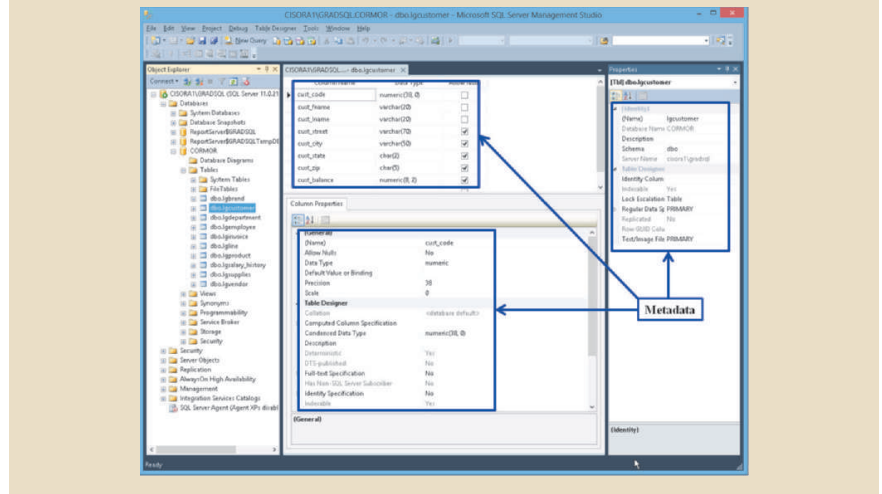
A DBMS performs several important functions that guarantee the integrity and consistency of the data in the database. Most of those functions are transparent to end users, and most can be achieved only through the use of a DBMS. They include data dictionary management, data storage management, data transformation and presentation, security management, multiuser access control, backup and recovery management, data integrity management, database access languages and application programming interfaces, and database communication interfaces. Each of these functions is explained as follows:

- **Data dictionary management.** The DBMS stores definitions of the data elements and their relationships (metadata) in a data dictionary. In turn, all programs that access the data in the database work through the DBMS. The DBMS uses the **data dictionary** to look up the required data component structures and relationships, thus relieving you from having to code such complex relationships in each program. Additionally, any changes made in a database structure are automatically recorded in the data dictionary, thereby freeing you from having to modify all of the programs that access the changed structure. In other words, the DBMS provides data abstraction, and it removes structural and data dependence from the system. For example, Figure 1.12 shows how Microsoft SQL Server Express presents the data definition for the CUSTOMER table.

#### data dictionary

A DBMS component that stores metadata—data about data. The data dictionary contains data definitions as well as data characteristics and relationships. May also include data that is external to the DBMS.

FIGURE 1.12 ILLUSTRATING METADATA WITH MICROSOFT SQL SERVER EXPRESS



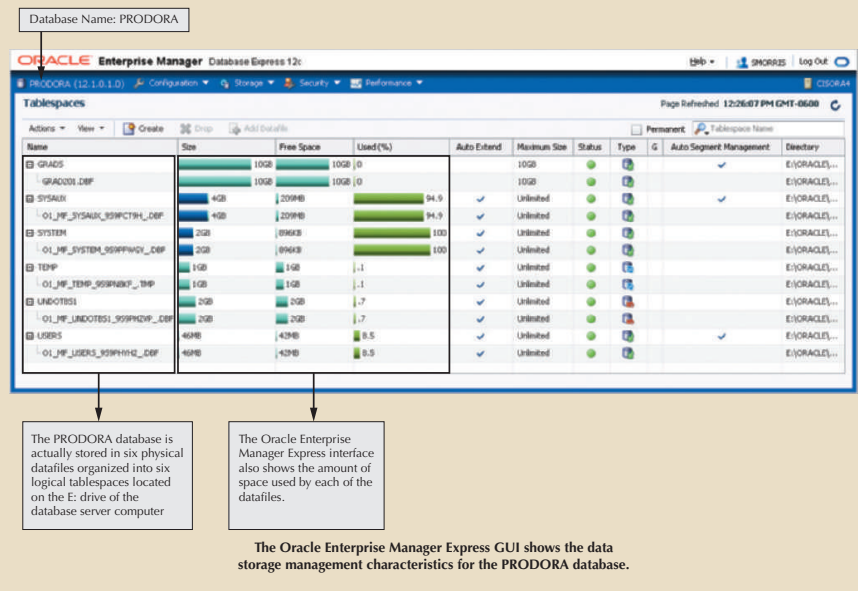
- **Data storage management.** The DBMS creates and manages the complex structures required for data storage, thus relieving you from the difficult task of defining and programming the physical data characteristics. A modern DBMS provides storage not only for the data but also for related data-entry forms or screen definitions, report definitions, data validation rules, procedural code, structures to handle video and picture formats, and so on. Data storage management is also important for database performance tuning. **Performance tuning** relates to the activities that make the database perform more efficiently in terms of storage and access speed. Although the user sees the database as a single data storage unit, the DBMS actually stores the database in multiple physical data files (see Figure 1.13). Such data files may even be stored on different storage media. Therefore, the DBMS doesn't have to wait for one disk request to finish before the next one starts. In other words, the DBMS can fulfill database requests concurrently. Data storage management and performance tuning issues are addressed in Chapter 11, Database Performance Tuning and Query Optimization.
- **Data transformation and presentation.** The DBMS transforms entered data to conform to required data structures. The DBMS relieves you of the chore of distinguishing between the logical data format and the physical data format. That is, the DBMS formats the physically retrieved data to make it conform to the user's logical expectations. For example, imagine an enterprise database used by a multinational company. An end user in England would expect to enter the date July 11, 2017, as "11/07/2017." In contrast, the same date would be entered in the United States as "07/11/2017." Regardless of the data presentation format, the DBMS must manage the date in the proper format for each country.
- **Security management.** The DBMS creates a security system that enforces user security and data privacy. Security rules determine which users can access the

#### performance tuning

Activities that make a database perform more efficiently in terms of storage and access speed.



FIGURE 1.13 ILLUSTRATING DATA STORAGE MANAGEMENT WITH ORACLE



database, which data items each user can access, and which data operations (read, add, delete, or modify) the user can perform. This is especially important in multiuser database systems. Chapter 16, Database Administration and Security, examines data security and privacy issues in greater detail. All database users may be authenticated to the DBMS through a username and password or through biometric authentication such as a fingerprint scan. The DBMS uses this information to assign access privileges to various database components such as queries and reports.

- **Multiuser access control.** To provide data integrity and data consistency, the DBMS uses sophisticated algorithms to ensure that multiple users can access the database concurrently without compromising its integrity. Chapter 10, Transaction Management and Concurrency Control, covers the details of multiuser access control.
- **Backup and recovery management.** The DBMS provides backup and data recovery to ensure data safety and integrity. Current DBMS systems provide special utilities that allow the DBA to perform routine and special backup and restore procedures. Recovery management deals with the recovery of the database after a failure, such as a bad sector in the disk or a power failure. Such capability is critical to preserving the database's integrity. Chapter 16 covers backup and recovery issues.
- **Data integrity management.** The DBMS promotes and enforces integrity rules, thus minimizing data redundancy and maximizing data consistency. The data

relationships stored in the data dictionary are used to enforce data integrity. Ensuring data integrity is especially important in transaction-oriented database systems. Data integrity and transaction management issues are addressed in Chapter 7, Introduction to Structured Query Language (SQL), and Chapter 10.

- **Database access languages and application programming interfaces.** The DBMS provides data access through a query language. A **query language** is a nonprocedural language—one that lets the user specify what must be done without having to specify how. **Structured Query Language (SQL)** is the de facto query language and data access standard supported by the majority of DBMS vendors. Chapter 7, Introduction to Structured Query Language (SQL), and Chapter 8, Advanced SQL, address the use of SQL. The DBMS also provides application programming interfaces to procedural languages such as COBOL, C, Java, Visual Basic.NET, and C#. In addition, the DBMS provides administrative utilities used by the DBA and the database designer to create, implement, monitor, and maintain the database.
- **Database communication interfaces.** A current-generation DBMS accepts end-user requests via multiple, different network environments. For example, the DBMS might provide access to the database via the Internet through the use of web browsers such as Mozilla Firefox, Google Chrome, Microsoft Edge, or Microsoft Internet Explorer. In this environment, communications can be accomplished in several ways:

- End users can generate answers to queries by filling in screen forms through their preferred web browser.
- The DBMS can automatically publish predefined reports on a website.
- The DBMS can connect to third-party systems to distribute information via email or other productivity applications.

Database communication interfaces are examined in greater detail in Chapter 12, Distributed Database Management Systems; in Chapter 15, Database Connectivity and Web Technologies; and in Appendix I, Databases in Electronic Commerce. (Appendixes are available at [www.cengagebrain.com](http://www.cengagebrain.com).)



## Note

### Why a Spreadsheet Is Not a Database

While a spreadsheet allows for the manipulation of data in a tabular format, it does not support even the most basic database functionality such as support for self-documentation through metadata, enforcement of data types or domains to ensure consistency of data within a column, defined relationships among tables, or constraints to ensure consistency of data across related tables. Most users lack the necessary training to recognize the limitations of spreadsheets for these types of tasks.

## 1-7c Managing the Database System: A Shift in Focus

The introduction of a database system over the file system provides a framework in which strict procedures and standards can be enforced. Consequently, the role of the human component changes from an emphasis on programming (in the file system) to a focus on the broader aspects of managing the organization's data resources and on the administration of the complex database software itself.

The database system makes it possible to tackle far more sophisticated uses of the data resources, as long as the database is designed to make use of that power. The

### query language

A nonprocedural language that is used by a DBMS to manipulate its data. An example of a query language is SQL.

### Structured Query Language (SQL)

A powerful and flexible relational database language composed of commands that enable users to create database and table structures, perform various types of data manipulation and data administration, and query the database to extract useful information.

kinds of data structures created within the database and the extent of the relationships among them play a powerful role in determining the effectiveness of the database system.

Although the database system yields considerable advantages over previous data management approaches, database systems do carry significant disadvantages:

- *Increased costs.* Database systems require sophisticated hardware and software and highly skilled personnel. The cost of maintaining the hardware, software, and personnel required to operate and manage a database system can be substantial. Training, licensing, and regulation compliance costs are often overlooked when database systems are implemented.
- *Management complexity.* Database systems interface with many different technologies and have a significant impact on a company's resources and culture. The changes introduced by the adoption of a database system must be properly managed to ensure that they help advance the company's objectives. Because database systems hold crucial company data that are accessed from multiple sources, security issues must be assessed constantly.
- *Maintaining currency.* To maximize the efficiency of the database system, you must keep your system current. Therefore, you must perform frequent updates and apply the latest patches and security measures to all components. Because database technology advances rapidly, personnel training costs tend to be significant.
- *Vendor dependence.* Given the heavy investment in technology and personnel training, companies might be reluctant to change database vendors. As a consequence, vendors are less likely to offer pricing point advantages to existing customers, and those customers might be limited in their choice of database system components.
- *Frequent upgrade/replacement cycles.* DBMS vendors frequently upgrade their products by adding new functionality. Such new features often come bundled in new upgrade versions of the software. Some of these versions require hardware upgrades. Not only do the upgrades themselves cost money but it also costs money to train database users and administrators to properly use and manage the new features.

Now that you know what a database and DBMS are, and why they are necessary, you are ready to begin developing your career as a database professional.

### 1-8 Preparing for Your Database Professional Career

In this chapter, you were introduced to the concepts of data, information, databases, and DBMSs. You also learned that, regardless of what type of database you use (OLTP, OLAP, or NoSQL), or what type of database environment you are working in (e.g., Oracle, Microsoft, IBM, or Hadoop), the success of a database system greatly depends on how well the database structure is designed.

Throughout this book, you will learn the building blocks that lay the foundation for your career as a database professional. Understanding these building blocks and developing the skills to use them effectively will prepare you to work with databases at many different levels within an organization. A small sample of such career opportunities is shown in Table 1.3.

TABLE 1.3

| DATABASE CAREER OPPORTUNITIES  |  |  |
|--------------------------------|--|--|
| JOB TITLE                      | DESCRIPTION  | SAMPLE SKILLS REQUIRED   |
| Database Developer             | Create and maintain database-based applications  | Programming, database fundamentals, SQL  |
| Database Designer              | Design and maintain databases  | Systems design, database design, SQL   |
| Database Administrator         | Manage and maintain DBMS and databases   | Database fundamentals, SQL, vendor courses   |
| Database Analyst               | Develop databases for decision support reporting   | SQL, query optimization, data warehouses   |
| Database Architect             | Design and implementation of database environments (conceptual, logical, and physical)                 | DBMS fundamentals, data modeling, SQL, hardware knowledge, etc.  |
| Database Consultant            | Help companies leverage database technologies to improve business processes and achieve specific goals | Database fundamentals, data modeling, database design, SQL, DBMS, hardware, vendor-specific technologies, etc.       |
| Database Security Officer      | Implement security policies for data administration  | DBMS fundamentals, database administration, SQL, data security technologies, etc.                                    |
| Cloud Computing Data Architect | Design and implement the infrastructure for next-generation cloud database systems                     | Internet technologies, cloud storage technologies, data security, performance tuning, large databases, etc.          |
| Data Scientist                 | Analyze large amounts of varied data to generate insights, relationships, and predictable behaviors    | Data analysis, statistics, advanced mathematics, SQL, programming, data mining, machine learning, data visualization |

As you also learned in this chapter, database technologies are constantly evolving to address new challenges such as large databases, semistructured and unstructured data, increasing processing speed, and lowering costs. While database technologies can change quickly, the fundamental concepts and skills do not. It is our goal that after you learn the database essentials in this book, you will be ready to apply your knowledge and skills to work with traditional OLTP and OLAP systems as well as cutting-edge, complex database technologies such as the following:

- *Very large databases (VLDB).* Many vendors are addressing the need for databases that support large amounts of data, usually in the petabyte range. (A petabyte is more than 1,000 terabytes.) VLDB vendors include Oracle Exadata, IBM's Netezza, HP's Vertica, and Teradata. VLDBs are now being superseded by Big Data databases.
- *Big Data databases.* Products such as Cassandra (Facebook) and BigTable (Google) are using "columnar-database" technologies to support the needs of database applications that manage large amounts of "nontabular" data. See more about this topic in Chapter 2.
- *In-memory databases.* Most major database vendors also offer some type of in-memory database support to address the need for faster database processing. In-memory databases store most of their data in primary memory (RAM) rather than in slower secondary storage (hard disks). In-memory databases include IBM's solidDB and Oracle's TimesTen.
- *Cloud databases.* Companies can now use cloud database services to quickly add database systems to their environment while simultaneously lowering the total cost of ownership of a new DBMS. A cloud database offers all the advantages of a local DBMS, but instead of residing within your organization's network infrastructure, it resides on the Internet. See more about this topic in Chapter 15.

We address some of these topics in this book, but not all—no single book can cover the entire realm of database technologies. This book's primary focus is to help you learn database fundamentals, develop your database design skills, and master your SQL skills so you will have a head start in becoming a successful database professional. However, you first must learn about the tools at your disposal. In the next chapter, you will learn different approaches to data management and how these approaches influence your designs.

## Summary

- Data consists of raw facts. Information is the result of processing data to reveal its meaning. Accurate, relevant, and timely information is the key to good decision making, and good decision making is the key to organizational survival in a global environment.
- Data is usually stored in a database. To implement a database and to manage its contents, you need a database management system (DBMS). The DBMS serves as the intermediary between the user and the database. The database contains the data you have collected and “data about data,” known as metadata.
- Database design defines the database structure. A well-designed database facilitates data management and generates accurate and valuable information. A poorly designed database can lead to poor decision making, and poor decision making can lead to the failure of an organization.
- Databases can be classified according to the number of users supported, where the data is located, the type of data stored, the intended data usage, and the degree to which the data is structured.
- Databases evolved from manual and then computerized file systems. In a file system, data is stored in independent files, each requiring its own data management programs. Although this method of data management is largely outmoded, understanding its characteristics makes database design easier to comprehend.
- Some limitations of file system data management are that it requires extensive programming, system administration can be complex and difficult, making changes to existing structures is difficult, and security features are likely to be inadequate. Also, independent files tend to contain redundant data, leading to problems of structural and data dependence.
- DBMSs were developed to address the file system's inherent weaknesses. Rather than depositing data in independent files, a DBMS presents the database to the end user as a single data repository. This arrangement promotes data sharing, thus eliminating the potential problem of islands of information. In addition, the DBMS enforces data integrity, eliminates redundancy, and promotes data security.
- Knowledge of database technologies leads to many career opportunities in the ever-expanding IT industry. There is a variety of specialization within the database arena for a wide range of skills and expertise.

## Key Terms

|                                   |   |                                 |
|-----------------------------------|---|---------------------------------|
| ad hoc query                      | database system                               | performance tuning              |
| analytical database               | desktop database                              | physical data format            |
| business intelligence             | discipline-specific database                  | production database             |
| centralized database              | distributed database                          | query                           |
| cloud database                    | enterprise database                           | query language                  |
| data                              | Extensible Markup Language (XML)              | query result set                |
| data anomaly                      | field   | record                          |
| data dependence                   | file  | semistructured data             |
| data dictionary                   | general-purpose database                      | single-user database            |
| data inconsistency                | information                                   | social media                    |
| data independence                 | islands of information                        | structural dependence           |
| data integrity                    | knowledge                                     | structural independence         |
| data management                   | logical data format                           | structured data                 |
| data processing (DP) specialist   | metadata                                      | Structured Query Language (SQL) |
| data quality                      | multiuser database                            | transactional database          |
| data redundancy                   | NoSQL   | unstructured data               |
| data type                         | online analytical processing (OLAP)           | workgroup database              |
| data warehouse                    | online transaction processing (OLTP) database | XML database                    |
| database                          |   |                                 |
| database design                   |   |                                 |
| database management system (DBMS) | operational database                          |                                 |

## Review Questions

1. Define each of the following terms:
  - a. data
  - b. field
  - c. record
  - d. file
2. What is data redundancy, and which characteristics of the file system can lead to it?
3. What is data independence, and why is it lacking in file systems?
4. What is a DBMS, and what are its functions?
5. What is structural independence, and why is it important?
6. Explain the differences among data, information, and a database.
7. What is the role of a DBMS, and what are its advantages? What are its disadvantages?

8. List and describe the different types of databases.
9. What are the main components of a database system?
10. What is metadata?
11. Explain why database design is important.
12. What are the potential costs of implementing a database system?
13. Use examples to compare and contrast unstructured and structured data. Which type is more prevalent in a typical business environment?
14. What are some basic database functions that a spreadsheet cannot perform?
15. What common problems do a collection of spreadsheets created by end users share with the typical file system?
16. Explain the significance of the loss of direct, hands-on access to business data that end users experienced with the advent of computerized data repositories.
17. Explain why the cost of ownership may be lower with a cloud database than with a traditional, company database.

## Problems



The file structures you see in this problem set are simulated in a Microsoft Access database named Ch01\_Problems, which is available at [www.cengagebrain.com](http://www.cengagebrain.com).

Given the file structure shown in Figure P1.1, answer Problems 1–4.

FIGURE P1.1 THE FILE STRUCTURE FOR PROBLEMS 1–4

| PROJECT_CODE | PROJECT_MANAGER | MANAGER_PHONE | MANAGER_ADDRESS                      | PROJECT_ID_PRICE |
|--------------|-----------------|---------------|--------------------------------------|------------------|
| 21-52        | Holly B. Parker | 904-338-3416  | 3334 Lee Rd., Gainesville, FL 37123  | 18033660.00      |
| 25-2D        | Jane D. Grant   | 615-098-9909  | 210 Clark Blvd., Nashville, TN 36362 | 12500000.00      |
| 25-5A        | George F. Doets | 615-227-1245  | 124 River Dr., Franklin, TN 29185    | 32812420.00      |
| 25-9T        | Holly B. Parker | 904-338-3416  | 3334 Lee Rd., Gainesville, FL 37123  | 21953234.00      |
| 27-4G        | George F. Doets | 615-227-1245  | 124 River Dr., Franklin, TN 29185    | 10294840.00      |
| 29-2D        | Holly B. Parker | 904-338-3416  | 3334 Lee Rd., Gainesville, FL 37123  | 25559999.00      |
| 31-7P        | William K. Moor | 904-445-2719  | 216 Morton Rd., Stetson, FL 30155    | 56850000.00      |

1. How many records does the file contain? How many fields are there per record?
2. What problem would you encounter if you wanted to produce a listing by city? How would you solve this problem by altering the file structure?
3. If you wanted to produce a listing of the file contents by last name, area code, city, state, or zip code, how would you alter the file structure?
4. What data redundancies do you detect? How could those redundancies lead to anomalies?
5. Identify and discuss the serious data redundancy problems exhibited by the file structure shown in Figure P1.5.

FIGURE P1.5 THE FILE STRUCTURE FOR PROBLEMS 5–8

| PROJ_NAME   | PROJ_MANAGER          | EMP_NUM | EMP_NAME | JOB_CODE | JOB_CHG_HOUR | PROJ_HOURS   | EMP_PHONE |
|-------------|-----------------------|---------|----------|----------|--------------|--------------|-----------|
| 1 Hurricane | 101 John D. Newson    | EE      |          | 65.00    | 13.3         | 653-234-3245 |           |
| 1 Hurricane | 105 David F. Schwartz | CT      |          | 60.00    | 18.2         | 653-234-1123 |           |
| 1 Hurricane | 110 Anne R. Ratonas   | CT      |          | 60.00    | 14.3         | 615-233-5569 |           |
| 2 Coast     | 101 John D. Newson    | EE      |          | 65.00    | 19.8         | 653-234-3254 |           |
| 2 Coast     | 109 June H. Sattelmel | EE      |          | 65.00    | 17.5         | 905-554-7812 |           |
| 3 Satellite | 110 Anne R. Ratonas   | CT      |          | 62.00    | 11.6         | 615-233-5569 |           |
| 3 Satellite | 105 David F. Schwartz | CT      |          | 26.00    | 23.4         | 653-234-1123 |           |
| 3 Satellite | 123 Mary D. Chen      | EE      |          | 65.00    | 19.1         | 615-233-5432 |           |
| 3 Satellite | 112 Alicia R. Smith   | EE      |          | 65.00    | 20.7         | 615-678-6679 |           |

6. Looking at the EMP\_NAME and EMP\_PHONE contents in Figure P1.5, what change(s) would you recommend?
7. Identify the various data sources in the file you examined in Problem 5.
8. Given your answer to Problem 7, what new files should you create to help eliminate the data redundancies found in the file shown in Figure P1.5?
9. Identify and discuss the serious data redundancy problems exhibited by the file structure shown in Figure P1.9. (The file is meant to be used as a teacher class assignment schedule. One of the many problems with data redundancy is the likely occurrence of data inconsistencies—two different initials have been entered for the teacher named Maria Cordoza.)

FIGURE P1.9 THE FILE STRUCTURE FOR PROBLEMS 9–10

| BUILDING_CODE | ROOM_CODE | TEACHER_LASTNAME | TEACHER_FNAME | TEACHER_INITIAL | DAYTIME          |
|---------------|-----------|------------------|---------------|-----------------|------------------|
| KOM           | 204E      | vWilton          | Horace        | G               | MMWF 8:00-8:50   |
| KOM           | 123       | Cordoza          | Maria         | L               | MMWF 8:00-8:50   |
| LDB           | 504       | Petroski         | Donald        | J               | TTh 1:00-2:15    |
| KOM           | 34        | Hawkins          | Anne          | vW              | MMWF 10:00-10:50 |
| AP            | 225B      | Rivet            | James         |                 | TTh 9:00-10:15   |
| LDB           | 301       | Robertson        | Jeanette      | P               | TTh 9:00-10:15   |
| KOM           | 204E      | Cordoza          | Maria         | I               | MMWF 9:00-9:50   |
| LDB           | 504       | vWilton          | Horace        | G               | TTh 1:00-2:15    |
| KOM           | 34        | Cordoza          | Maria         | L               | MMWF 11:00-11:50 |
| LDB           | 504       | Petroski         | Donald        | J               | MMWF 2:00-2:50   |

10. Given the file structure shown in Figure P1.9, what problem(s) might you encounter if building KOM were deleted?
11. Using your school's student information system, print your class schedule. The schedule probably would contain the student identification number, student name, class code, class name, class credit hours, class instructor name, the class meeting days and times, and the class room number. Use Figure P1.11 as a template to complete the following actions.

FIGURE P1.11 STUDENT SCHEDULE DATA FORMAT

| STU_ID | STU_NAME | CLASS_CODE | CLASS_NAME | CLASS_CREDHRS | INSTR_NAME | CLASS_DAYS | CLASS_TIMES | ROOM |
|--------|----------|------------|------------|---------------|------------|------------|-------------|------|
|        |          |            |            |               |            |            |             |      |
|        |          |            |            |               |            |            |             |      |

- a) Create a spreadsheet using the template shown in Figure P1.11 and enter your current class schedule.
- b) Enter the class schedule of two of your classmates into the same spreadsheet.
- c) Discuss the redundancies and anomalies caused by this design.