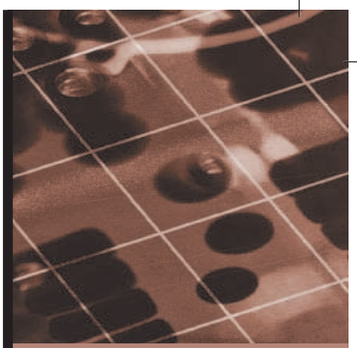


PART

VI

Database Administration

Business Vignette

Oreck Revises Disaster Recovery Plan After Katrina

Because companies design disaster recovery plans during normal business operations, holes in the plan often become apparent only during crises. Oreck Corporation, the vacuum manufacturer, had a decent disaster recovery plan. The company, headquartered in New Orleans, had arranged with IBM to host its AS/400-based applications in a data center in Boulder, CO. The staff in New Orleans would relocate to Long Beach, Mississippi, where the company had a large manufacturing and distribution center. On Sunday, August 28, two days before Hurricane Katrina hit New Orleans, the IT staff put the plan into operation.

"It took 12 hours to make two backups," remembers Michael Evanson, Oreck's vice president of IT. As roads out of the city were closing or clogged, his staff was still backing up the AS/400 data. Finally, on Monday morning, CEO Tom Oreck grabbed the tapes and his family, and they flew a private plane to Houston, where he overnighted the tapes to Boulder. By the time Mr. Oreck arrived in Houston, however, a major oversight in their plan had become apparent.

"We had two facilities, and we assumed that at least one would survive the hurricane. They're about 80 miles apart, but Katrina went right through the middle of the two," Evanson explains. The hurricane had flooded the Long Beach factory, and many of its 900 employees had evacuated the area. The company's Intel-based data which were needed for its facilities elsewhere had been left behind — in New Orleans. Like most backup and recovery programs, Oreck's plan did not cover all components of its IT system, and in establishing its priorities, it had failed to prepare for a disaster that could take down both its headquarters and its backup center.

The company scrambled to recover quickly. It established temporary headquarters in Dallas, located workers by setting up an 800 number, bought RVs for its homeless employees and generators for the plant, and brought in contractors to repair physical damage. Two weeks later, the Long Beach facility was up and running.

Today, the company has revised its plan to provide for better access to and protection of the hardware, software, and data it needs in the event of an emergency. Oreck now runs backups every two hours instead of every eight hours. The company tests its data recovery and contingency plans regularly. Recently, the company opened a new plant further away in Cookeville, Tennessee.

"The plans that we had in place before Katrina, which I think served us well, have been highly modified," says Oreck. "Honestly, the plan hadn't been updated in a long time," says Oreck. "We've obviously changed our view on that."

15 DATABASE ADMINISTRATION AND SECURITY

FIFTEEN

In this chapter, you will learn:

- That data are a valuable business asset requiring careful management
- How a database plays a critical role in an organization
- That the introduction of a DBMS has important technological, managerial, and cultural organizational consequences
- What the database administrator's managerial and technical roles are
- About data security, database security, and the information security framework
- About several database administration tools and strategies
- How various database administration technical tasks are performed with Oracle

Preview

This chapter shows you the basis for a successful database administration strategy. Such a strategy requires that data be considered important and valuable resources to be treated and managed as corporate assets.

The chapter explores how a database fits within an organization, what the data views and requirements are at various management levels, and how the DBMS supports those views and requirements. Database administration must be fully understood and accepted within an organization before a sound data administration strategy can be implemented. In this chapter, you learn about important data management issues by looking at the managerial and technical roles of the database administrator (DBA). This chapter also explores database security issues, such as the confidentiality, integrity, and availability of data. In our information-based society, one of the key aspects of data management is to ensure that the data are protected against intentional or unintentional access by unauthorized personnel. It is also essential to ensure that the data are available when and where needed, even in the face of natural disaster or hardware failure, and to maintain the integrity of the data in the database.

The technical aspects of database administration are augmented by a discussion of database administration tools and the corporate-wide data architectural framework. The managerial aspects of database administration are explained by showing you how the database administration function fits within classical organizational structures. Because Oracle is the current leader in mid- to high-level corporate database markets, you learn how a DBA performs some typical database management functions in Oracle.

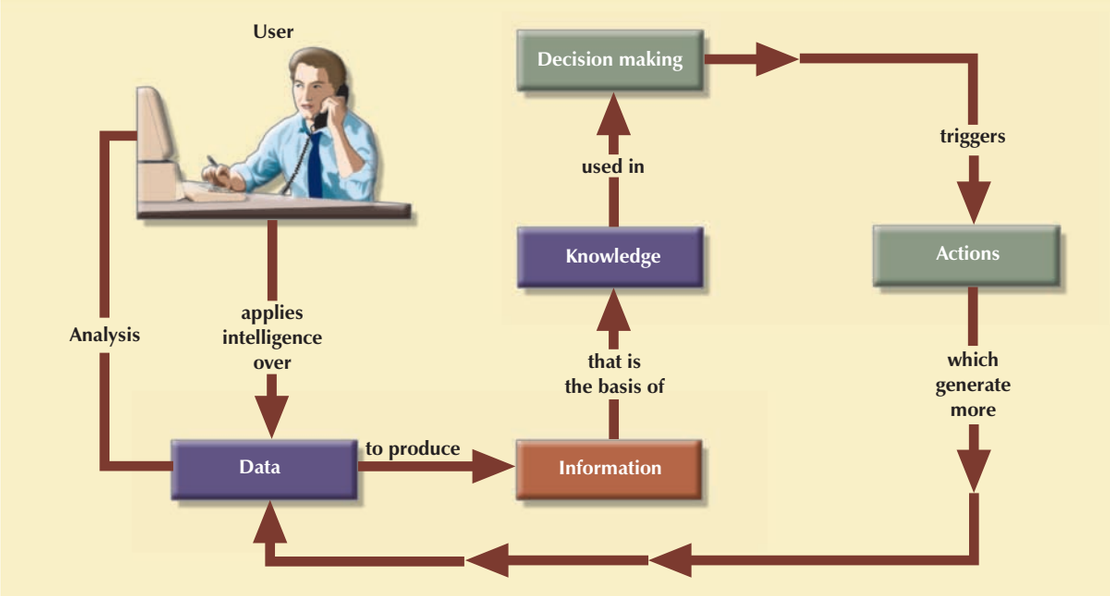
15.1 DATA AS A CORPORATE ASSET

In Chapter 1, Database Systems, you learned that data are the raw material from which information is produced. Therefore, it is not surprising that in today's information-driven environment, data are a valuable asset that requires careful management.

To assess data's monetary value, take a look at what's stored in a company database: data about customers, suppliers, inventory, operations, and so on. How many opportunities are lost if the data are lost? What is the actual cost of data loss? For example, an accounting firm whose entire database is lost would incur significant direct and indirect costs. The accounting firm's problems would be magnified if the data loss occurred during tax season. Data loss puts any company in a difficult position. The company might be unable to handle daily operations effectively, it might be faced with the loss of customers who require quick and efficient service, and it might lose the opportunity to gain new customers.

Data are a valuable *resource* that can translate into *information*. If the information is accurate and timely, it is likely to trigger actions that enhance the company's competitive position and generate wealth. In effect, an organization is subject to a *data-information-decision cycle*; that is, the data *user* applies intelligence to *data* to produce *information* that is the basis of *knowledge* used in *decision making* by the user. This cycle is illustrated in Figure 15.1.

FIGURE 15.1 The data-information-decision-making cycle



Note in Figure 15.1 that the decisions made by high-level managers trigger actions within the organization's lower levels. Such actions produce additional data to be used for monitoring company performance. In turn, the additional data must be recycled within the data/information/decision framework. Thus, data form the basis for decision making, strategic planning, control, and operations monitoring.

A critical success factor of an organization is efficient asset management. To manage data as a corporate asset, managers must understand the value of information—that is, processed data. In fact, there are companies (for example, those that provide credit reports) whose only product is information and whose success is solely a function of information management.

15.2 THE NEED FOR AND ROLE OF A DATABASE IN AN ORGANIZATION

Data are used by different people in different departments for different reasons. Therefore, data management must address the concept of shared data. Chapter 1 showed how the need for data sharing made the DBMS almost inevitable. Used properly, the DBMS facilitates:

- *Interpretation* and *presentation* of data in useful formats by transforming raw data into information.
- *Distribution* of data and information to the right people at the right time.
- Data *preservation* and *monitoring* the data usage for adequate periods of time.
- *Control* over data duplication and use, both internally and externally.

Whatever the type of organization, the database's predominant role is *to support managerial decision making at all levels in the organization while preserving data privacy and security*.

An organization's managerial structure might be divided into three levels: top, middle, and operational. Top-level management makes strategic decisions, middle management makes tactical decisions, and operational management makes daily operational decisions. Operational decisions are short term and affect only daily operations; for example, deciding to change the price of a product to clear it from inventory. Tactical decisions involve a longer time frame and affect larger-scale operations; for example, changing the price of a product in response to competitive pressures. Strategic decisions are those that affect the long-term well-being of the company or even its survival; for example, changing pricing strategy across product lines to capture market share.

The DBMS must provide tools that give each level of management a useful view of the data and that support the required level of decision making. The following activities are typical of each management level.

At the *top management* level, the database must be able to:

- Provide the information necessary for strategic decision making, strategic planning, policy formulation, and goals definition.
- Provide access to external and internal data to identify growth opportunities and to chart the direction of such growth. (Direction refers to the nature of the operations: Will a company become a service organization, a manufacturing organization, or some combination of the two?)
- Provide a framework for defining and enforcing organizational policies. (Remember that such polices are translated into business rules at lower levels in the organization.)
- Improve the likelihood of a positive return on investment for the company by searching for new ways to reduce costs and/or by boosting productivity.
- Provide feedback to monitor whether the company is achieving its goals.

At the *middle management* level, the database must be able to:

- Deliver the data necessary for tactical decisions and planning.
- Monitor and control the allocation and use of company resources and evaluate the performance of the various departments.
- Provide a framework for enforcing and ensuring the security and privacy of the data in the database. **Security** means protecting the data against accidental or intentional use by unauthorized users. **Privacy** deals with the rights of individuals and the organization to determine the "who, what, when, where, and how" of data usage.

At the *operational management* level, the database must be able to:

- Represent and support the company operations as closely as possible. The data model must be flexible enough to incorporate all required present and expected data.

- Produce query results within specified performance levels. Keep in mind that the performance requirements increase for lower levels of management and operations. Thus, the database must support fast responses to a greater number of transactions at the operational management level.
- Enhance the company's short-term operational ability by providing timely information for customer support and for application development and computer operations.

A general objective for any database is to provide a seamless flow of information throughout the company.

The company's database is also known as the corporate or enterprise database. The **enterprise database** might be defined as "the company's data representation that provides support for all present and expected future operations." Most of today's successful organizations depend on the enterprise database to provide support for all of their operations—from design to implementation, from sales to services, and from daily decision making to strategic planning.

15.3 INTRODUCTION OF A DATABASE: SPECIAL CONSIDERATIONS

Having a computerized database management system does not guarantee that the data will be properly used to provide the best solutions required by managers. A DBMS is a tool for managing data; like any tool, it must be used effectively to produce the desired results. Consider this analogy: in the hands of a carpenter, a hammer can help produce furniture; in the hands of a child, it might do damage. The solution to company problems is not the mere existence of a computer system or its database, but, rather, its effective management and use.

The introduction of a DBMS represents a big change and challenge; throughout the organization, the DBMS is likely to have a profound impact, which might be positive or negative depending on how it is administered. For example, one key consideration is adapting the DBMS to the organization rather than forcing the organization to adapt to the DBMS. The main issue should be the organization's needs rather than the DBMS's technical capabilities. However, the introduction of a DBMS cannot be accomplished without affecting the organization. The flood of new DBMS-generated information has a profound effect on the way the organization functions, and therefore, on its corporate culture.

The introduction of a DBMS into an organization has been described as a process that includes three important aspects:[1]

- *Technological:* DBMS software and hardware.
- *Managerial:* Administrative functions.
- *Cultural:* Corporate resistance to change.

The *technological* aspect includes selecting, installing, configuring, and monitoring the DBMS to make sure that it efficiently handles data storage, access, and security. The person or people in charge of addressing the technological aspect of the DBMS installation must have the technical skills necessary to provide or secure adequate support for the various users of the DBMS: programmers, managers, and end users. Therefore, database administration staffing is a key technological consideration in the DBMS introduction. The selected personnel must exhibit the right mix of technical and managerial skills to provide a smooth transition to the new shared-data environment.

The *managerial* aspect of the DBMS introduction should not be taken lightly. A high-quality DBMS does not guarantee a high-quality information system, just as having the best race car does not guarantee winning a race.

The introduction of a DBMS into an organization requires careful planning to create an appropriate organizational structure to accommodate the person or people responsible for administering the DBMS. The organizational structure must also be subject to well-developed monitoring and controlling functions. The administrative personnel must have excellent interpersonal and communications skills combined with broad organizational and business understanding.

[1] Murray, John P. "The Managerial and Cultural Issues of a DBMS," *370/390 Database Management* 1(8), September 1991, pp. 32–33.

Top management must be committed to the new system and must define and support the data administration functions, goals, and roles within the organization.

The *cultural* impact of the introduction of a database system must be assessed carefully. The DBMS's existence is likely to have an effect on people, functions, and interactions. For example, additional personnel might be added, new roles might be allocated to existing personnel, and employee performance might be evaluated using new standards.

A cultural impact is likely because the database approach creates a more controlled and structured information flow. Department managers who are used to handling their own data must surrender their subjective ownership to the data administration function and must share their data with the rest of the company. Application programmers must learn and follow new design and development standards. Managers might be faced with what they consider to be an information overload and might require some time to adjust to the new environment.

When the new database comes online, people might be reluctant to use the information provided by the system and might question its value or accuracy. (Many will be surprised and possibly chagrined to discover that the information does not fit their preconceived notions and strongly held beliefs.) The database administration department must be prepared to open its doors to end users, listen to their concerns, act on those concerns when possible, and educate end users about the system's uses and benefits.

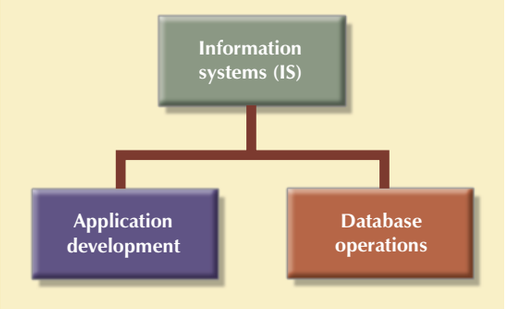
15.4 THE EVOLUTION OF THE DATABASE ADMINISTRATION FUNCTION

Data administration has its roots in the old, decentralized world of the file system. The cost of data and managerial duplication in such file systems gave rise to a centralized data administration function known as the electronic data processing (EDP) or data processing (DP) department. The DP department's task was to pool all computer resources to support all departments *at the operational level*. The DP administration function was given the authority to manage all existing company file systems as well as resolve data and managerial conflicts created by the duplication and/or misuse of data.

The advent of the DBMS and its shared view of data produced a new level of data management sophistication and led the DP department to evolve into an **information systems (IS) department**. The responsibilities of the IS department were broadened to include:

- A *service* function to provide end users with active data management support.
- A *production* function to provide end users with specific solutions for their information needs through integrated application or management information systems.

FIGURE 15.2 The IS department internal organization



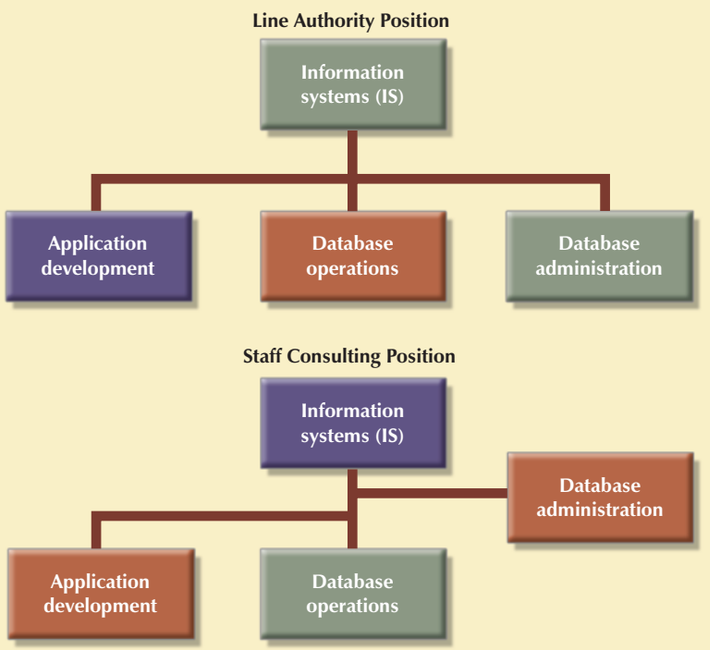
The functional orientation of the IS department was reflected in its internal organizational structure. IS departments typically were structured as shown in Figure 15.2. As the demand for application development grew, the IS application development segment was subdivided by the type of supported system: accounting, inventory, marketing, and so on. However, this development meant that the database administration responsibilities were divided. The application development segment was in charge of gathering database requirements and logical database design, whereas the database operations segment took charge of implementing, monitoring, and controlling the DBMS operations.

As the number of database applications grew, data management became an increasingly complex job, thus leading to the development of the database administration function.

The person responsible for the control of the centralized and shared database became known as the **database administrator** (**DBA**).

The size and role of the DBA function varies from company to company, as does its placement within a company's organizational structure. On the organization chart, the DBA function might be defined as either a staff or line position. Placing the DBA function in a staff position often creates a consulting environment in which the DBA is able to devise the data administration strategy but does not have the authority to enforce it or to resolve possible conflicts.[2] The DBA function in a line position has both the responsibility and the authority to plan, define, implement, and enforce the policies, standards, and procedures used in the data administration activity. The two possible DBA function placements are illustrated in Figure 15.3.

FIGURE 15.3 **The placement of the DBA function**



There is no standard for how the DBA function fits in an organization's structure. In part, that is because the DBA function itself is probably the most dynamic of any organization's functions. In fact, the fast-paced changes in DBMS technology dictate changing organizational styles. For example:

- The development of distributed databases can force an organization to decentralize the data administration function further. The distributed database requires the system DBA to define and delegate the responsibilities of each local DBA, thus imposing new and more complex *coordinating* activities on the system DBA.
- The growing use of Internet-accessible data and the growing number of data warehousing applications are likely to add to the DBA's data modeling and design activities, thus expanding and diversifying the DBA's job.
- The increasing sophistication and power of microcomputer-based DBMS packages provide an easy platform for the development of user-friendly, cost-effective, and efficient solutions to the needs of specific departments.

[2]For a historical perspective on the development of the DBA function and a broader coverage of its organizational placement alternatives, refer to Jay-Louise Weldon's classic *Data Base Administration* (New York, Plenum Press, 1981). Although you might think that the book's publication date renders it obsolete, a surprising number of its topics are returning to the current operational database scene.

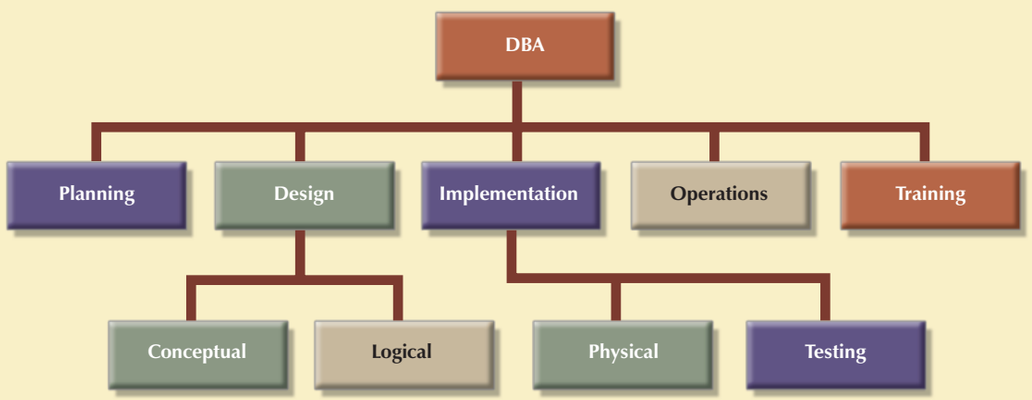
But such an environment also invites data duplication, not to mention the problems created by people who lack the technical qualifications to produce good database designs. In short, the new microcomputer environment requires the DBA to develop a new set of technical and managerial skills.

It is common practice to define the DBA function by dividing the DBA operations according to the Database Life Cycle (DBLC) phases. If that approach is used, the DBA function requires personnel to cover the following activities:

- Database planning, including the definition of standards, procedures, and enforcement.
- Database requirements gathering and conceptual design.
- Database logical and transaction design.
- Database physical design and implementation.
- Database testing and debugging.
- Database operations and maintenance, including installation, conversion, and migration.
- Database training and support.

Figure 15.4 represents an appropriate DBA functional organization according to that model.

FIGURE 15.4 **A DBA functional organization**

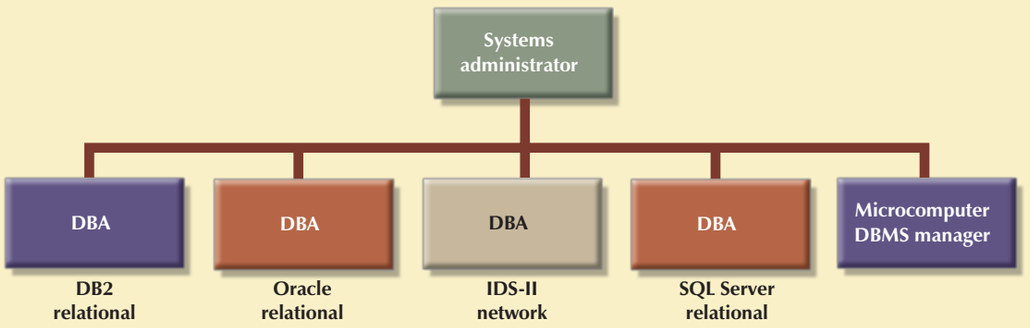


Keep in mind that a company might have several different and incompatible DBMSs installed to support different operations. For example, it is not uncommon to find corporations with a hierarchical DBMS to support the daily transactions at the operational level and a relational database to support middle and top management's ad hoc information needs. There may also be a variety of microcomputer DBMSs installed in the different departments. In such an environment, the company might have one DBA assigned for each DBMS. The general coordinator of all DBAs is sometimes known as the **systems administrator**; that position is illustrated in Figure 15.5.

There is a growing trend toward specialization in the data management function. For example, the organization charts used by some of the larger corporations make a distinction between a DBA and the **data administrator** (**DA**). The DA, also known as the **information resource manager** (**IRM**), usually reports directly to top management and is given a higher degree of responsibility and authority than the DBA, although the two roles overlap some.

The DA is responsible for controlling the overall corporate data resources, both computerized and manual. Thus, the DA's job description covers a larger area of operations than that of the DBA because the DA is in charge of controlling not only the computerized data, but also the data outside the scope of the DBMS. The placement of the DBA within the expanded organizational structure may vary from company to company. Depending on the structure's components, the DBA might report to the DA, the IRM, the IS manager, or directly to the company's CEO.

FIGURE 15.5 Multiple database administrators in an organization



15.5 THE DATABASE ENVIRONMENT'S HUMAN COMPONENT

A substantial portion of this book is devoted to relational database design and implementation and to examining DBMS features and characteristics. Thus far the book has focused on the very important technical aspects of the database. However, there is another important side of the database coin: even the most carefully crafted database system cannot operate without the human component. So in this section, you will explore how people perform the data administration activities that make a good database design useful.

Effective data administration requires both technical and managerial skills. For example, the DA's job typically has a strong managerial orientation with company-wide scope. In contrast, the DBA's job tends to be more technically oriented and has a narrower DBMS-specific scope. However, the DBA, too, must have a considerable store of people skills. After all, both the DA and the DBA perform "people" functions common to all departments in an organization. For example, both the DA and DBA direct and control personnel staffing and training within their respective departments.

Table 15.1 contrasts the general characteristics of both positions by summarizing the typical DA and DBA activities. All activities flowing from the characteristics shown in Table 15.1 are invested in the DBA if the organization does not employ both a DA and a DBA.

TABLE 15.1 Contrasting DA and DBA Activities and Characteristics

DATA ADMINISTRATOR (DA)	DATABASE ADMINISTRATOR (DBA)
Does strategic planning	Controls and supervises
Sets long-term goals	Executes plans to reach goals
Sets policies and standards	Enforces policies and procedures Enforces programming standards
Is broad in scope	Is narrow in scope
Focuses on the long term	Focuses on the short term (daily operations)
Has a managerial orientation	Has a technical orientation
Is DBMS-independent	Is DBMS-specific

Note that the DA is responsible for providing a global and comprehensive administrative strategy for all of the organization's data. In other words, the DA's plans must consider the entire data spectrum. Thus, the DA is responsible for the consolidation and consistency of both manual and computerized data.

The DA also must set data administration goals. Those goals are defined by issues such as:

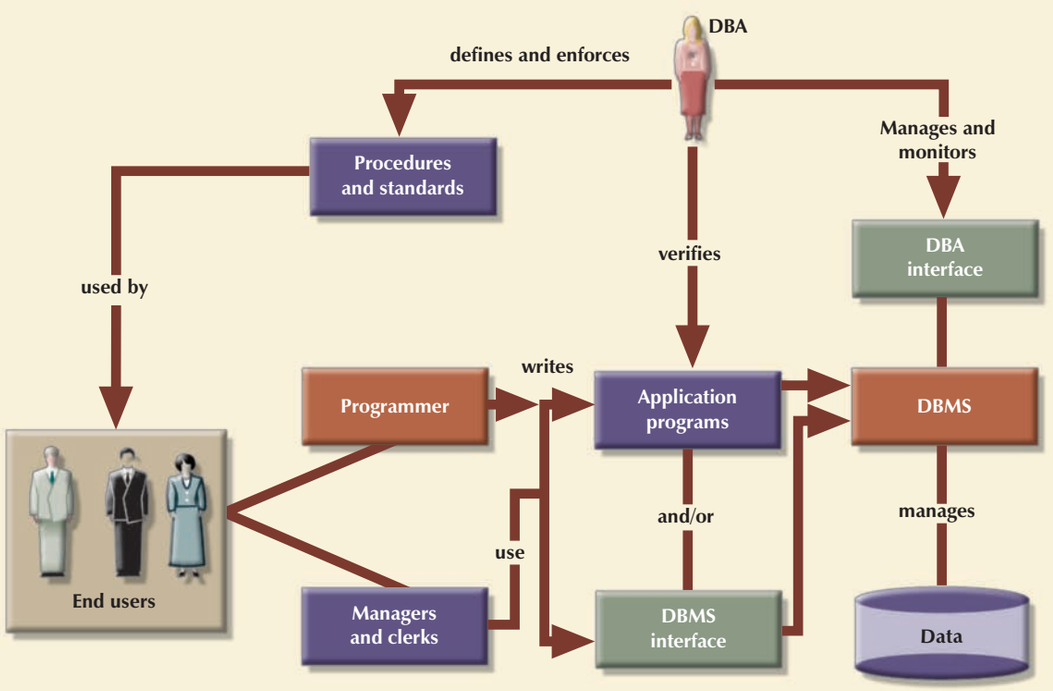
- Data "sharability" and time availability.
- Data consistency and integrity.
- Data security and privacy.
- Data quality standards.
- Extent and type of data use.

Naturally, that list can be expanded to fit the organization's specific data needs. Regardless of how data management is conducted—and despite the fact that much authority is invested in the DA or DBA to define and control the way company data are used—the DA and DBA do not own the data. Instead, DA and DBA functions are defined to emphasize that data are a shared company asset.

The preceding discussion should not lead you to believe that there are universally accepted DA and DBA administrative standards. As a matter of fact, the style, duties, organizational placement, and internal structure of both functions vary from company to company. For example, many companies distribute DA duties between the DBA and the manager of information systems. For simplicity and to avoid confusion, the label DBA is used here as a general title that encompasses all appropriate data administration functions. Having made that point, let's move on to the DBA's role as an arbitrator between data and users.

The arbitration of interactions between the two most important assets of any organization, people and data, places the DBA in the dynamic environment portrayed in Figure 15.6.

FIGURE 15.6 **A summary of DBA activities**



As you examine Figure 15.6, note that the DBA is the focal point for data/user interaction. The DBA defines and enforces the procedures and standards to be used by programmers and end users during their work with the DBMS. The DBA also verifies that programmer and end-user access meets the required quality and security standards.

Database users might be classified by the:

- Type of decision-making support required (operational, tactical, or strategic).
- Degree of computer knowledge (novice, proficient, or expert).
- Frequency of access (casual, periodic, or frequent).

Those classifications are not exclusive and usually overlap. For example, an operational user can be an expert with casual database access. Nevertheless, a typical top-level manager might be a strategic novice user with periodic database access. On the other hand, a database application programmer is an operational expert and frequent database user. Thus, each organization employs people whose levels of database expertise span an entire spectrum. The DBA must be able to interact with all of those people, understand their different needs, answer questions at all levels of the expertise scale, and communicate effectively.

The DBA activities portrayed in Figure 15.6 suggest the need for a diverse mix of skills. In large companies, such skills are likely to be distributed among several people who work within the DBA function. In small companies, the skills might be the domain of just one individual. The skills can be divided into two categories—managerial and technical—as summarized in Table 15.2.

TABLE 15.2 Desired DBA Skills

MANAGERIAL	TECHNICAL
Broad business understanding	Broad data-processing background
Coordination skills	Systems development life cycle knowledge
Analytical skills	Structured methodologies: Data flow diagrams Structure charts Programming languages
Conflict resolution skills	Database life cycle knowledge
Communications skills (oral and written)	Database modeling and design skills Conceptual Logical Physical
Negotiation skills	Operational skills: database implementation, data dictionary management, security, and so on
Experience: 10 years in a large DP department	

As you examine Table 15.2, keep in mind that the DBA must perform two distinct roles. The DBA's managerial role is focused on personnel management and on interactions with the end-user community. The DBA's technical role involves the use of the DBMS—database design, development, and implementation—as well as the production, development, and use of application programs. The DBA's managerial and technical roles will be examined in greater detail in the following sections.

15.5.1 The DBA's Managerial Role

As a manager, the DBA must concentrate on the control and planning dimensions of database administration. Therefore, the DBA is responsible for:

- Coordinating, monitoring, and allocating database administration resources: people and data.
- Defining goals and formulating strategic plans for the database administration function.

More specifically, the DBA's responsibilities are shown in Table 15.3.

TABLE 15.3 DBA Activities and Services

DBA ACTIVITY		DBA SERVICE
Planning		End-user support
Organizing		Policies, procedures, and standards
Testing	of	Data security, privacy, and integrity
Monitoring		Data backup and recovery
Delivering		Data distribution and use

Table 15.3 illustrates that the DBA is generally responsible for planning, organizing, testing, monitoring, and delivering quite a few services. Those services might be performed by the DBA or, more likely, by the DBA's personnel. Let's examine the services in greater detail.

End-User Support

The DBA interacts with the end user by providing data and information support services to the organization's departments. Because end users usually have dissimilar computer backgrounds, end-user support services include:

- *Gathering user requirements.* The DBA must work within the end-user community to help gather the data required to identify and describe the end users' problems. The DBA's communications skills are very important at this stage because the DBA works closely with people who have varying computer backgrounds and communication styles. The gathering of user requirements requires the DBA to develop a precise understanding of the users' views and needs and to identify present and future information needs.
- *Building end-user confidence.* Finding adequate solutions to end users' problems increases end-user trust and confidence in the DBA function. The DBA function is also to educate the end-user in the services provided and how those services enhance data stewardship and data security.
- *Resolving conflicts and problems.* Finding solutions to end users' problems in one department might trigger conflicts with other departments. End users are typically concerned with their own specific data needs rather than with those of others, and they are not likely to consider how their data affect other departments within the organization. When data/information conflicts arise, the DBA function has the authority and responsibility to resolve them.
- *Finding solutions to information needs.* The ability and authority to resolve data conflicts enables the DBA to develop solutions that will properly fit within the existing data management framework. The DBA's primary objective is to provide solutions to the end users' information needs. Given the growing importance of the Internet, those solutions are likely to require the development and management of Web servers to interface with the databases. In fact, the explosive growth of e-commerce requires the use of *dynamic* interfaces to facilitate interactive product queries and product sales.
- *Ensuring quality and integrity of data and applications.* Once the right solution has been found, it must be properly implemented and used. Therefore, the DBA must work with both application programmers and end users to teach them the database standards and procedures required for data quality, access, and manipulation. The DBA must also make sure that the database transactions do not adversely affect the quality of the data. Likewise, certifying the quality of the application programs that access the database is a crucial DBA function. Special attention must be given to the DBMS Internet interfaces because those interfaces are prone to security issues.
- *Managing the training and support of DBMS users.* One of the most time-consuming DBA activities is teaching end users how to use the database properly. The DBA must ensure that all users accessing the database have a basic understanding of the functions and use of the DBMS software. The DBA coordinates and monitors all activities concerning end-user education.

Policies, Procedures, and Standards

A prime component of a successful data administration strategy is the continuous enforcement of the policies, procedures, and standards for correct data creation, usage, distribution, and deletion within the database. The DBA must define, document, and communicate the policies, procedures, and standards before they can be enforced. Basically:

- **Policies** are general statements of direction or action that communicate and support DBA goals.
- **Standards** describe the minimum requirements of a given DBA activity; they are more detailed and specific than policies. In effect, standards are rules that are used to evaluate the quality of the activity. For example, standards define the structure of application programs and the naming conventions programmers must use.
- **Procedures** are written instructions that describe a series of steps to be followed during the performance of a given activity. Procedures must be developed within existing working conditions, and they must support and enhance that environment.

To illustrate the distinctions among policies, standards, and procedures, look at the following examples:

Policies

All users must have passwords.

Passwords must be changed every six months.

Standards

A password must have a minimum of five characters.

A password must have a maximum of 12 characters.

Social Security numbers, names, and birth dates cannot be used as passwords.

Procedures

To create a password, (1) the end user sends to the DBA a written request for the creation of an account; (2) the DBA approves the request and forwards it to the computer operator; (3) the computer operator creates the account, assigns a temporary password, and sends the account information to the end user; (4) a copy of the account information is sent to the DBA; and (5) the user changes the temporary password to a permanent one.

Standards and procedures defined by the DBA are used by all end users who want to benefit from the database. Standards and procedures must complement each other and must constitute an extension of data administration policies. Procedures must facilitate the work of end users and the DBA. The DBA must define, communicate, and enforce procedures that cover areas such as:

- *End-user database requirements gathering.* What documentation is required? What forms must be used?
- *Database design and modeling.* What database design methodology is to be used (normalization or object-oriented methodology)? What tools are to be used (CASE tools, data dictionaries, UML or ER diagrams)?
- *Documentation and naming conventions.* What documentation must be used in the definition of all data elements, sets, and programs that access the database?
- *Design, coding, and testing of database application programs.* The DBA must define the standards for application program coding, documentation, and testing. The DBA standards and procedures are given to the application programmers, and the DBA must enforce those standards.
- *Database software selection.* The selection of the DBMS package and any other software related to the database must be properly managed. For example, the DBA might require that software be properly interfaced with existing software, that it have the features needed by the organization, and that it provide a positive return on investment. In today's Internet environment, the DBA must also work with Web administrators to implement efficient and secure Web-to-database connectivity.

- *Database security and integrity.* The DBA must define the policies governing security and integrity. Database security is especially crucial. Security standards must be clearly defined and strictly enforced. Security procedures must be designed to handle a multitude of security scenarios to ensure that security problems are minimized. Although no system can ever be completely secure, security procedures must be designed to meet critical standards. The growing use of Internet interfaces to databases opens the door to new security threats that are far more complex and difficult to manage than those encountered with more traditional internally generated and controlled interfaces. Therefore, the DBA must work closely with Internet security specialists to ensure that the databases are properly protected from attacks launched inadvertently or deliberately.
- *Database backup and recovery.* Database backup and recovery procedures must include the information necessary to guarantee proper execution and management of the backups.
- *Database maintenance and operation.* The DBMS's daily operations must be clearly documented. Operators must keep job logs, and they must write operator instructions and notes. Such notes are helpful in pinpointing the causes and solutions of problems. Operational procedures must also include precise information concerning backup and recovery procedures.
- *End-user training.* A full-featured training program must be established within the organization, and procedures governing the training must be clearly specified. The objective is to indicate clearly who does what, when, and how. Each end user must be aware of the type and extent of the available training methodology.

Procedures and standards must be revised at least annually to keep them up to date and to ensure that the organization can adapt quickly to changes in the work environment. Naturally, the introduction of new DBMS software, the discovery of security or integrity violations, the reorganization of the company, and similar changes require revision of the procedures and standards.

Data Security, Privacy, and Integrity

The security, privacy, and integrity of the data in the database are of great concern to DBAs who manage current DBMS installations. Technology has pointed the way to greater productivity through information management. Technology has also resulted in the distribution of data across multiple sites, thus making it more difficult to maintain data control, security, and integrity. The multiple-site data configuration has made it imperative that the DBA use the security and integrity mechanisms provided by the DBMS to enforce the database administration policies defined in the previous section. In addition, DBAs must team up with Internet security experts to build security mechanisms to safeguard data from possible attacks or unauthorized access. Section 15.6 covers security issues in more detail.

Data Backup and Recovery

When data are not readily available, companies face potentially ruinous losses. Therefore, data backup and recovery procedures are critical in all database installations. The DBA also must ensure that the data in the database can be fully recovered in case of physical data loss or loss of database integrity.

Data loss can be partial or total. A partial loss is caused by a physical loss of part of the database or when part of the database has lost integrity. A total loss might mean that the database continues to exist but its integrity is entirely lost or that the entire database is physically lost. In any case, backup and recovery procedures are the cheapest database insurance you can buy.

The management of database security, integrity, backup, and recovery is so critical that many DBA departments have created a position called the **database security officer** (**DSO**). The DSO's sole job is to ensure database security and integrity. In large organizations, the DSO's activities are often classified as *disaster management*.

Disaster management includes all of the DBA activities designed to secure data availability following a physical disaster or a database integrity failure. Disaster management includes all planning, organizing, and testing of database contingency plans and recovery procedures. The backup and recovery measures must include at least:

- *Periodic data and applications backups.* Some DBMSs include tools to ensure backup and recovery of the data in the database. The DBA should use those tools to render the backup and recovery tasks automatic. Products such as IBM's DB2 allow the creation of different backup types: full, incremental, and concurrent. A **full backup**, also known as a **database dump**, produces a complete copy of the entire database. An **incremental backup** produces a backup of all data since the last backup date; a **concurrent backup** takes place while the user is working on the database.
- *Proper backup identification.* Backups must be clearly identified through detailed descriptions and date information, thus enabling the DBA to ensure that the correct backups are used to recover the database. The most common backup medium is tape; the storage and labeling of tapes must be done diligently by the computer operators, and the DBA must keep track of tape currency and location. However, organizations that are large enough to hire a DBA do not typically use CDs and DVDs for enterprise backup. Other emerging backup solutions include optical and disk-based backup devices. Such backup solutions include online storage based on Network Attached Storage (NAS) and Storage Area Networks (SAN). Enterprise backup solutions use a layered backup approach in which the data are first backed up to fast disk media for intermediate storage and fast restoration. Later, the data is transferred to tape for archival storage.
- *Convenient and safe backup storage.* There must be multiple backups of the same data, and each backup copy must be stored in a different location. The storage locations must include sites inside and outside the organization. (Keeping different backups in the same place defeats the purpose of having multiple backups in the first place.) The storage locations must be properly prepared and may include fire-safe and quakeproof vaults, as well as humidity and temperature controls. The DBA must establish a policy to respond to two questions: (1) Where are the backups to be stored? (2) How long are backups to be stored?
- *Physical protection of both hardware and software.* Protection might include the use of closed installations with restricted access, as well as preparation of the computer sites to provide air conditioning, backup power, and fire protection. Physical protection also includes the provision of a backup computer and DBMS to be used in case of emergency. For example, when Hurricane Katrina hit the U.S. Gulf Coast in 2005, New Orleans suffered almost total destruction of its communications infrastructure. Many organizations and educational institutions did not have adequate disaster recovery plans for such an extreme level of service interruption (see the Part VI Business Vignette at the beginning of this chapter for one example).[3]
- *Personal access control to the software of a database installation.* Multilevel passwords and privileges and hardware and software challenge/response tokens can be used to properly identify authorized users of resources.
- *Insurance coverage for the data in the database.* The DBA or security officer must secure an insurance policy to provide financial protection in the event of a database failure. The insurance might be expensive, but it is less expensive than the disaster created by massive data loss.

Two additional points are worth making.

- Data recovery and contingency plans must be thoroughly tested and evaluated, and they must be practiced frequently. So-called fire drills are not to be disparaged, and they require top-level management's support and enforcement.
- A backup and recovery program is not likely to cover all components of an information system. Therefore, it is appropriate to establish priorities concerning the nature and extent of the data recovery process.

[3]See "AAUP Responds to Katrina's Impact on New Orleans Universities," http://www.aaup.org/AAUP/pubsres/academe/2006/MA/AW/kat.htm.

Data Distribution and Use

Data are useful only when they reach the right users in a timely fashion. The DBA is responsible for ensuring that the data are distributed to the right people, at the right time, and in the right format. The DBA's data distribution and use tasks can become very time-consuming, especially when the data delivery capacity is based on a typical applications programming environment, where users depend on programmers to deliver the programs to access the data in the database. Although the Internet and its intranet and extranet extensions have opened databases to corporate users, their use has also created a new set of challenges for the DBA.

Current data distribution philosophy makes it easy for *authorized* end users to access the database. One way to accomplish that task is to facilitate the use of a new generation of more sophisticated query tools and the new Internet Web front ends. They enable the DBA to educate end users to produce the required information without being dependent on applications programmers. Naturally, the DBA must ensure that all users adhere to appropriate standards and procedures.

This data-sharing philosophy is common today, and it is likely that it will become more common as database technology marches on. Such an environment is more flexible for the end user. Clearly, enabling end users to become relatively self-sufficient in the acquisition and use of data can lead to more efficient use of data in the decision process. Yet this "data democracy" can also produce some troublesome side effects. Letting end users micromanage their data subsets could inadvertently sever the connection between those users and the data administration function. The DBA's job under those circumstances might become sufficiently complicated to compromise the efficiency of the data administration function. Data duplication might flourish again without checks at the organizational level to ensure the uniqueness of data elements. Thus, end users who do not completely understand the nature and sources of data might make improper use of the data elements.

15.5.2 The DBA's Technical Role

The DBA's technical role requires a broad understanding of DBMS functions, configuration, programming languages, data modeling and design methodologies, and so on. For example, the DBA's technical activities include the selection, installation, operation, maintenance, and upgrading of the DBMS and utility software, as well as the design, development, implementation, and maintenance of the application programs that interact with the database.

Many of the DBA's technical activities are a logical extension of the DBA's managerial activities. For example, the DBA deals with database security and integrity, backup and recovery, and training and support. Thus, the DBA's dual role might be conceptualized as a capsule whose technical core is covered by a clear managerial shell.

The technical aspects of the DBA's job are rooted in the following areas of operation:

- Evaluating, selecting, and installing the DBMS and related utilities.
- Designing and implementing databases and applications.
- Testing and evaluating databases and applications.
- Operating the DBMS, utilities, and applications.
- Training and supporting users.
- Maintaining the DBMS, utilities, and applications.

The following sections will explore the details of those operational areas.

Evaluating, Selecting, and Installing the DBMS and Utilities

One of the DBA's first and most important technical responsibilities is selecting the database management system, utility software, and supporting hardware to be used in the organization. Therefore, the DBA must develop and execute a plan for evaluating and selecting the DBMS, utilities, and hardware. That plan must be based primarily on the organization's needs rather than on specific software and hardware features. The DBA must recognize that the search is for solutions to problems rather than for a computer or DBMS software. Put simply, a DBMS is a management tool and not a technological toy.

The first and most important step of the evaluation and acquisition plan is to determine company needs. To establish a clear picture of those needs, the DBA must make sure that the entire end-user community, including top- and mid-level managers, is involved in the process. Once the needs are identified, the objectives of the data administration function can be clearly established and the DBMS features and selection criteria can be defined.

To match DBMS capability to the organization's needs, the DBA would be wise to develop a checklist of desired DBMS features. That DBMS checklist should address at least these issues:

- *DBMS model.* Are the company's needs better served by a relational, object-oriented, or object/relational DBMS? If a data warehouse application is required, should a relational or multidimensional DBMS be used? Does the DBMS support star schemas?
- *DBMS storage capacity.* What maximum disk and database size is required? How many disk packages must be supported? How many tape units are needed? What are other storage needs?
- *Application development support.* Which programming languages are supported? What application development tools (database schema design, data dictionary, performance monitoring, and screen and menu painters) are available? Are end-user query tools provided? Does the DBMS provide Web front-end access?
- *Security and integrity.* Does the DBMS support referential and entity integrity rules, access rights, and so on? Does the DBMS support the use of audit trails to spot errors and security violations? Can the audit trail size be modified?
- *Backup and recovery.* Does the DBMS provide some automated backup and recovery tools? Does the DBMS support tape, optical disc, or network-based backups? Does the DBMS automatically back up the transaction logs?
- *Concurrency control.* Does the DBMS support multiple users? What levels of isolation (table, page, row) does the DBMS offer? How much manual coding is needed in the application programs?
- *Performance.* How many transactions per second does the DBMS support? Are additional transaction processors needed?
- *Database administration tools.* Does the DBMS offer some type of DBA management interface? What type of information does the DBA interface provide? Does the DBMS provide alerts to the DBA when errors or security violations occur?
- *Interoperability and data distribution.* Can the DBMS work with other DBMS types in the same environment? What coexistence or interoperability level is achieved? Does the DBMS support READ and WRITE operations to and from other DBMS packages? Does the DBMS support a client/server architecture?
- *Portability and standards.* Can the DBMS run on different operating systems and platforms? Can the DBMS run on mainframes, midrange computers, and personal computers? Can the DBMS applications run without modification on all platforms? What national and industry standards does the DBMS follow?
- *Hardware.* What hardware does the DBMS require?
- *Data dictionary.* Does the DBMS have a data dictionary? If so, what information is kept in it? Does the DBMS interface with any data dictionary tool? Does the DBMS support any CASE tools?
- *Vendor training and support.* Does the vendor offer in-house training? What type and level of support does the vendor provide? Is the DBMS documentation easy to read and helpful? What is the vendor's upgrade policy?
- *Available third-party tools.* What additional tools are offered by third-party vendors (query tools, data dictionary, access management and control, storage allocation management tools)?
- *Cost.* What costs are involved in the acquisition of the software and hardware? How many additional personnel are required, and what level of expertise is required of them? What are the recurring costs? What is the expected payback period?

Pros and cons of several alternative solutions must be evaluated during the selection process. Available alternatives are often restricted because software must be compatible with the organization's existing computer system. Remember that a DBMS is just part of a solution; it requires support from collateral hardware, application software, and utility programs. For example, the DBMS's use is likely to be constrained by the available CPU(s), front-end processor(s), auxiliary storage devices, data communication devices, the operating system, a transaction processor system, and so on. The costs associated with the hardware and software components must be included in the estimations.

The selection process must also consider the site's preparation costs. For example, the DBA must include both one-time and recurring expenditures involved in the preparation and maintenance of the computer room installations.

The DBA must supervise the installation of all software and hardware designated to support the data administration strategy; must have a thorough understanding of the components being installed; and must be familiar with the installation, configuration, and startup procedures of such components. The installation procedures include details such as the location of backup and transaction log files, network configuration information, and physical storage details.

Keep in mind that installation and configuration details are DBMS-dependent. Therefore, such details cannot be addressed in this book. Consult the installation and configuration sections of your system's DBMS administration guide for those details.

Designing and Implementing Databases and Applications

The DBA function also provides data modeling and design services to end-users. Such services are often coordinated with an application development group within the data-processing department. Therefore, one of the primary activities of a DBA is to determine and enforce standards and procedures to be used. Once the appropriate standards and procedures framework are in place, the DBA must ensure that the database modeling and design activities are performed within the framework. The DBA then provides the necessary assistance and support during the design of the database at the conceptual, logical, and physical levels. (Remember that the conceptual design is both DBMS- and hardware-independent, the logical design is DBMS-dependent and hardware-independent, and the physical design is both DBMS- and hardware-dependent.)

The DBA function usually requires that several people be dedicated to database modeling and design activities. Those people might be grouped according to the organizational areas covered by the application. For example, database modeling and design personnel may be assigned to production systems, financial and managerial systems, or executive and decision support systems. The DBA schedules the design jobs to coordinate the data design and modeling activities. That coordination may require reassignment of available resources based on externally determined priorities.

The DBA also works with applications programmers to ensure the quality and integrity of database design and transactions. Such support services include reviewing the database application design to ensure that transactions are:

- *Correct:* The transactions mirror real-world events.
- *Efficient:* The transactions do not overload the DBMS.
- *Compliant*. Complies with integrity rules and standards.

These activities require personnel with broad database design and programming skills.

The implementation of the applications requires the implementation of the physical database. Therefore, the DBA must provide assistance and oversight during the physical design, including storage space determination and creation, data loading, conversion, and database migration services. The DBA's implementation tasks also include the generation, compilation, and storage of the application's access plan. An **access plan** is a set of instructions generated at application completion time that predetermines how the application will access the database at run time. To be able to create and validate the access plan, the user must have the required rights to access the database (see Chapter 11, Database Performance Tuning and Query Optimization).

Before an application comes online, the DBA must develop, test, and implement the operational procedures required by the new system. Such operational procedures include utilizing training, security, and backup and recovery plans, as well as assigning responsibility for database control and maintenance. Finally, the DBA must authorize application users to access the database from which the applications draw the required data.

The addition of a new database might require the fine-tuning and/or reconfiguring of the DBMS. Remember that the DBMS assists all applications by managing the shared corporate data repository. Therefore, when data structures are added or modified, the DBMS might require the assignment of additional resources to service the new and original users with equal efficiency (see Chapter 11, Database Performance Tuning and Query Optimization).

Testing and Evaluating Databases and Applications

The DBA must also provide testing and evaluation services for all of the database and end-user applications. Those services are the logical extension of the design, development, and implementation services described in the preceding section. Clearly, testing procedures and standards must already be in place before any application program can be approved for use in the company.

Although testing and evaluation services are closely related to database design and implementation services, they usually are maintained independently. The reason for the separation is that application programmers and designers often are too close to the problem being studied to detect errors and omissions.

Testing usually starts with the loading of the testbed database. That database contains test data for the applications, and its purpose is to check the data definition and integrity rules of the database and application programs.

The testing and evaluation of a database application cover all aspects of the system—from the simple collection and creation of data to its use and retirement. The evaluation process covers:

- Technical aspects of both the applications and the database. Backup and recovery, security and integrity, use of SQL, and application performance must be evaluated.
- Evaluation of the written documentation to ensure that the documentation and procedures are accurate and easy to follow.
- Observance of standards for naming, documenting, and coding.
- Data duplication conflicts with existing data.
- The enforcement of all data validation rules.

Following the thorough testing of all applications, the database, and the procedures, the system is declared operational and can be made available to end users.

Operating the DBMS, Utilities, and Applications

DBMS operations can be divided into four main areas:

- System support.
- Performance monitoring and tuning.
- Backup and recovery.
- Security auditing and monitoring.

System support activities cover all tasks directly related to the day-to-day operations of the DBMS and its applications. These activities include filling out job logs, changing tape, and verifying the status of computer hardware, disk packages, and emergency power sources. System-related activities include periodic, occasional tasks such as running special programs and resource configurations for new and/or upgraded versions of database applications.

Performance monitoring and tuning require much of the DBA's attention and time. These activities are designed to ensure that the DBMS, utilities, and applications maintain satisfactory performance levels. To carry out the performance monitoring and tuning tasks, the DBA must:

- Establish DBMS performance goals.
- Monitor the DBMS to evaluate whether the performance objectives are being met.
- Isolate the problem and find solutions (if performance objectives are not met).
- Implement the selected performance solutions.

DBMSs often include performance-monitoring tools that allow the DBA to query database usage information. Performance-monitoring tools are also available from many different sources: DBMS utilities are provided by third-party vendors, or they might be included in operating system utilities or transaction processor facilities. Most of the performance-monitoring tools allow the DBA to focus on selected system bottlenecks. The most common bottlenecks in DBMS performance tuning are related to the use of indexes, query-optimization algorithms, and management of storage resources.

Because improper index selection can have a deleterious effect on system performance, most DBMS installations adhere to a carefully defined index creation and usage plan. Such a plan is especially important in a relational database environment.

To produce satisfactory performance, the DBA is likely to spend much time trying to educate programmers and end users on the proper use of SQL statements. Typically, DBMS programmers' manuals and administration manuals contain useful performance guidelines and examples that demonstrate the proper use of SQL statements, both in the command-line mode and within application programs. Because relational systems do not give the user an index choice within a query, the DBMS makes the index selection for the user. Therefore, the DBA should create indexes that can be used to improve system performance. (For examples of database performance tuning, see Chapter 11, Database Performance Tuning and Query Optimization.)

Query-optimization routines are usually integrated into the DBMS package, thereby allowing few tuning options. Query-optimization routines are oriented to improve concurrent access to the database. Several database packages let the DBA specify parameters for determining the desired level of concurrency. Concurrency is also affected by the types of locks used by the DBMS and requested by the applications. Because the concurrency issue is important to the efficient operation of the system, the DBA must be familiar with the factors that influence concurrency. (See Chapter 10, Transaction Management and Concurrency Control, for more information on that subject.)

During DBMS performance tuning, the DBA must also consider available storage resources in terms of both primary and secondary memory. The allocation of storage resources is determined when the DBMS is configured. Storage configuration parameters can be used to determine:

- The number of databases that may be opened concurrently.
- The number of application programs or users supported concurrently.
- The amount of primary memory (buffer pool size) assigned to each database and each database process.
- The size and location of the log files. (Remember that these files are used to recover the database. The log files can be located in a separate volume to reduce the disk's head movement and to increase performance.)

Performance-monitoring issues are DBMS-specific. Therefore, the DBA must become familiar with the DBMS manuals to learn the technical details involved in the performance-monitoring task (see Chapter 11).

Because data loss is likely to be devastating to the organization, *backup and recovery activities* are of primary concern during the DBMS operation. The DBA must establish a schedule for backing up database and log files at appropriate intervals. Backup frequency is dependent on the application type and on the relative importance of the data. All critical system components—the database, the database applications, and the transaction logs—must be backed up periodically.

Most DBMS packages include utilities that schedule automated database backups, be they full or incremental. Although incremental backups are faster than full backups, an incremental backup requires the existence of a periodic full backup to be useful for recovery purposes.

Database recovery after a media or systems failure requires application of the transaction log to the correct database copy. The DBA must plan, implement, test, and enforce a "bulletproof" backup and recovery procedure.

Security auditing and monitoring assumes the appropriate assignment of access rights and the proper use of access privileges by programmers and end users. The technical aspects of security auditing and monitoring involve creating users, assigning access rights, using SQL commands to grant and revoke access rights to users and database objects, and creating audit trails to discover security violations or attempted violations. The DBA must periodically generate an audit trail report to determine whether there have been actual or attempted security violations—and, if so, from what locations, and if possible, by whom.

Training and Supporting Users

Training people to use the DBMS and its tools is included in the DBA's technical activities. In addition, the DBA provides or secures technical training in the use of the DBMS and its utilities for the applications programmers. Applications programmer training covers the use of the DBMS tools as well as the procedures and standards required for database programming.

Unscheduled, on-demand technical support for end users and programmers is also included in the DBA's activities. A technical troubleshooting procedure can be developed to facilitate such support. The technical procedure might include the development of a technical database used to find solutions to common technical problems.

Part of the support is provided by interaction with DBMS vendors. Establishing good relationships with software suppliers is one way to ensure that the company has a good external support source. Vendors are the source for up-to-date information concerning new products and personnel retraining. Good vendor–company relations also are likely to give organizations an edge in determining the future direction of database development.

Maintaining the DBMS, Utilities, and Applications

The maintenance activities of the DBA are an extension of the operational activities. Maintenance activities are dedicated to the preservation of the DBMS environment.

Periodic DBMS maintenance includes management of the physical or secondary storage devices. One of the most common maintenance activities is reorganizing the physical location of data in the database. (That is usually done as part of the DBMS fine-tuning activities.) The reorganization of a database might be designed to allocate contiguous disk-page locations to the DBMS to increase performance. The reorganization process also might free the space allocated to deleted data, thus providing more disk space for new data.

Maintenance activities also include upgrading the DBMS and utility software. The upgrade might require the installation of a new version of the DBMS software or an Internet front-end tool. Or it might create an additional DBMS gateway to allow access to a host DBMS running on a different host computer. DBMS gateway services are very common in distributed DBMS applications running in a client/server environment. Also, new-generation databases include features such as spatial data support, data warehousing and star query support, and support for Java programming interfaces for Internet access (see Chapter 14, Database Connectivity and Web Technologies).

Quite often companies are faced with the need to exchange data in dissimilar formats or between databases. The maintenance efforts of the DBA include migration and conversion services for data in incompatible formats or for different DBMS software. Such conditions are common when the system is upgraded from one version to another or when the existing DBMS is replaced by an entirely new DBMS. Database conversion services also include downloading data from the host DBMS (mainframe-based) to an end user's personal computer to allow that user to perform a variety of activities—spreadsheet analysis, charting, statistical modeling, and so on. Migration and conversion services can be

done at the logical level (DBMS- or software-specific) or at the physical level (storage media or operating-system-specific). Current generation DBMSs support XML as a standard format for data exchange among database systems and applications (see Chapter 14).

15.6 SECURITY

Security refers to activities and measures to ensure the confidentiality, integrity, and availability of an information system and its main asset, data.[4] It is important to understand that securing data requires a comprehensive, company-wide approach. That is, you cannot secure data if you do not secure all the processes and systems around it. Indeed, securing data entails securing the overall information system architecture, including hardware systems, software applications, the network and its devices, people (internal and external users), procedures, and the data itself. To understand the scope of data security, let's discuss each of the three security goals in more detail:

- **Confidentiality** deals with ensuring that data is protected against unauthorized access, and if the data are accessed by an authorized user, that the data are used only for an authorized purpose. In other words, confidentiality entails safeguarding data against disclosure of any information that would violate the privacy rights of a person or organization. Data must be evaluated and classified according to the level of confidentiality: highly restricted (very few people have access), confidential (only certain groups have access), and unrestricted (can be accessed by all users). The data security officer spends a great amount of time ensuring that the organization is in compliance with the desired levels of confidentiality. **Compliance** refers to activities undertaken to meet data privacy and security reporting guidelines. These reporting guidelines are either part of internal procedures or are imposed by external regulatory agencies such as the federal government. Examples of U.S. legislation enacted with the purpose of ensuring data privacy and confidentiality include the Health Insurance Portability and Accountability Act (HIPAA), Gramm-Leach-Bliley Act (GLBA), and Sarbanes-Oxley Act (SOX).[5]
- **Integrity**, within the data security framework, is concerned with keeping data consistent, free of errors or anomalies. Integrity focuses on maintaining the data free of inconsistencies and anomalies (see Chapter 1, Database Systems, to review the concepts of data inconsistencies and data anomalies). The DBMS plays a pivotal role in ensuring the integrity of the data in the database. However, from the security point of view, integrity deals not only with the data in the database, but also with ensuring that organizational processes, users, and usage patterns maintain such integrity. For example, a work-at-home employee using the Internet to access product costing could be considered an acceptable use; however, security standards might require the employee to use a secure connection and follow strict procedures to manage the data at home (shredding printed reports, using encryption to copy data to the local hard drive, etc.). Maintaining the integrity of the data is a process that starts with data collection and continues with data storage, processing, usage, and archival (see Chapter 13, Business Intelligence and Data Warehouses). The rationale behind integrity is to treat data as the most valuable asset in the organization and therefore to ensure that rigorous data validation is carried out at all levels within the organization.
- **Availability** refers to the accessibility of data whenever required by authorized users and for authorized purposes. To ensure data availability, the entire system (not only the data component) must be protected from service degradation or interruption caused by any source (internal or external). Service interruptions could be very costly for companies and users alike--recall the JetBlue[6] case in the Part V Business Vignette of this book, and, more recently the case of SKYPE, the voice over IP (VoIP) telephone service provider who suffered a 48-hour worldwide service interruption.[7] System availability is an important goal of security.

[4]The National Security Telecommunications and Information Systems Security Committee (NSTISSC) defines the CIA framework. See *http://www.nsa.gov/snac/wireless/I332-005R-2005.pdf*.

[5]To find additional information about these various laws, please visit *http://library.uis.edu/findinfo/govinfo/federal/law.html*.

[6]"JetBlue's C.E.O. Is 'Mortified' After Fliers Are Stranded," Jeff Baily, February 19, 2007, New York Times, *http://www.nytimes.com/2007/02/19/business/19jetblue.html?ex=1189051200&en=d63f3b54a602bf0d&ei=5070*.

[7]"Skype protection is limited," Andrew Garcia, *eWeek*, p. 59, August 27, 2007.

15.6.1 Security Policies

Normally, the tasks of securing the system and its main asset, the data, are performed by the database security officer and the database administrator(s), who work together to establish a cohesive data security strategy. Such security strategy begins with defining a sound and comprehensive security policy. A **security policy** is a collection of standards, policies and procedures created to guarantee the security of a system and ensure auditing and compliance. The security audit process starts by identifying the security vulnerabilities in the organization's information system infrastructure and identifying measures to protect the system and data against those vulnerabilities.

15.6.2 Security Vulnerabilities

A **security vulnerability** is a weakness in a system component that could be exploited to allow unauthorized access or cause service disruptions. The nature of such vulnerabilities could be of multiple types: technical (such as a flaw in the operating system or Web browser), managerial (for example, not educating users about critical security issues), cultural (hiding passwords under the keyboard or not shredding confidential reports), procedural (not requiring complex passwords or not checking user IDs), and so on. Whatever the case, when a security vulnerability is left unchecked, it could become a security threat. A **security threat** is an imminent security violation that could occur at any time due to unchecked security vulnerability.

A **security breach** occurs when a security threat is exploited to negatively affect the integrity, confidentiality, or availability of the system. Security breaches can yield a database whose integrity is either preserved or corrupted:

- *Preserved:* Action is required to avoid the repetition of similar security problems, but data recovery may not be necessary. As a matter of fact, most security violations are produced by unauthorized and unnoticed access for information purposes, but such snooping does not disrupt the database.
- *Corrupted:* Action is required to avoid the repetition of similar security problems, and the database must be recovered to a consistent state. Corrupting security breaches include database access by computer viruses and by hackers whose actions are intended to destroy or alter data.

Table 15.4 illustrates some security vulnerabilities that systems components are exposed to and some measures typically taken to protect against them.

TABLE 15.4 Sample Security Vulnerabilities and Related Measures

SYSTEM COMPONENT	SECURITY VULNERABILITY	SECURITY MEASURES
People	• User sets a blank password. • Password is short or includes birth date. • User leaves office door open all the time. • User leaves payroll information on screen for long periods of time.	• Enforce complex password policies. • Use multilevel authentication. • Use security screens and screen savers. • Educate users about sensitive data. • Install security cameras. • Use automatic door locks.
Workstations and Servers	• User copies data to flash drive. • Workstation is used by multiple users. • Power failure crashes computer. • Unauthorized personnel can use computer. • Sensitive data stored in laptop computer. • Data lost due to stolen hard disk/laptop. • Natural disasters—earthquake, flood, etc.	• Use group policies to restrict use of flash drives. • Assign user access rights to workstations. • Install Uninterrupted Power Supplies (UPS). • Add security lock devices to computers. • Implement a "kill" switch for stolen laptops. • Create and test data backup and recovery plans. • Protect system against natural disasters—use co-location strategies.

TABLE 15. 4 Sample Security Vulnerabilities and Related Measures (continued)

SYSTEM COMPONENT	SECURITY VULNERABILITY	SECURITY MEASURES
Operating System	• Buffer overflow attacks. • Virus attacks. • Root kits and worm attacks. • Denial of service attacks. • Trojan horses. • Spyware applications. • Password crackers.	• Apply OS security patches and updates. • Apply application server patches. • Install antivirus and antispyware software. • Enforce audit trails on the computers. • Perform periodic system backups. • Install only authorized applications. • Use group policies to prevent unauthorized installs.
Applications	• Application bugs–buffer overflow. • SQL injection, session hijacking, etc. • Application vulnerabilities—cross site scripting, nonvalidated inputs. • E-mail attacks: spamming, phishing, etc. • Social engineering e-mails.	• Test application programs extensively. • Built safeguards in code. • Do extensive vulnerability testing in applications. • Install spam filter/antivirus for e-mail system. • Use secure coding techniques (see *www.owasp.org*). • Educate users about social engineering attacks.
Network	• IP spoofing. • Packet sniffers. • Hacker attacks. • Clear passwords on network.	• Install firewalls. • Virtual Private Networks (VPN). • Intrusion Detection Systems (IDS). • Network Access Control (NAC). • Network activity monitoring.
Data	• Data shares are open to all users. • Data can be accessed remotely. • Data can be deleted from shared resource.	• Implement file system security. • Implement share access security. • Use access permission. • Encrypt data at the file system or database level.

15.6.3 Database Security

Database security refers to the use of the DBMS features and other related measures to comply with the security requirements of the organization. From the DBA's point of view, security measures should be implemented to protect the DBMS against service degradation and the database against loss, corruption, or mishandling. In short, the DBA should secure the DBMS from the point of installation through operation and maintenance.

NOTE

James Martin provides an excellent enumeration and description of the desirable attributes of a database security strategy that remains relevant today (James Martin, *Managing the Database Environment,* Englewood Cliffs, NJ: Prentice-Hall, 1977). Martin's security strategy is based on the seven essentials of database security and may be summarized as one in which:

Data are	**Users are**
Protected	Identifiable
Reconstructable	Authorized
Auditable	Monitored
Tamperproof	

To protect the DBMS against service degradation there are certain minimum recommended security safeguards. For example: change default system passwords, change default installation paths, apply the latest patches, secure installation folders with proper access rights, make sure only required services are running, set up auditing logs, set up

session logging, and require session encryption. Furthermore, the DBA should work closely with the network administrator to implement network security to protect the DBMS and all services running on the network. In current organizations, one of the most critical components in the information architecture is the network.

Protecting the data in the database is a function of authorization management. **Authorization management** defines procedures to protect and guarantee database security and integrity. Those procedures include, but are not limited to, user access management, view definition, DBMS access control, and DBMS usage monitoring.

- *User access management.* This function is designed to limit access to the database and likely includes at least the following procedures:
  - *Define each user to the database.* This is achieved at the operating system level and at the DBMS level. At the operating system level, the DBA can request the creation of a logon user ID that allows the end user to log on to the computer system. At the DBMS level, the DBA can either create a different user ID or employ the same user ID to authorize the end user to access the DBMS.
  - *Assign passwords to each user.* This, too, can be done at both operating system and DBMS levels. The database passwords can be assigned with predetermined expiration dates. The use of expiration dates enables the DBA to screen end users periodically and to remind users to change their passwords periodically, thus making unauthorized access less probable.
  - *Define user groups.* Classifying users into user groups according to common access needs facilitates the DBA's job of controlling and managing the access privileges of individual users. Also, the DBA can use database roles and resource limits to minimize the impact of rogue users in the system (see Section 15.9.6 for more information about these topics).
  - *Assign access privileges.* The DBA assigns access privileges or access rights to specific users to access specified databases. An access privilege describes the type of authorized access. For example, access rights may be limited to read-only, or the authorized access might include READ, WRITE, and DELETE privileges. Access privileges in relational databases are assigned through SQL GRANT and REVOKE commands.
  - *Control physical access.* Physical security can prevent unauthorized users from directly accessing the DBMS installation and facilities. Some common physical security practices found in large database installations include secured entrances, password-protected workstations, electronic personnel badges, closed-circuit video, voice recognition, and biometric technology.
- *View definition.* The DBA must define data views to protect and control the scope of the data that are accessible to an authorized user. The DBMS must provide the tools that allow the definition of views that are composed of one or more tables and the assignment of access rights to a user or a group of users. The SQL command CREATE VIEW is used in relational databases to define views. Oracle DBMS offers Virtual Private Database (VPD), which allows the DBA to create customized views of the data for multiple different users. With this feature, the DBA could restrict a regular user querying a payroll database to see only the rows and columns necessary, while the department manager would see only the rows and columns pertinent to that department.
- *DBMS access control.* Database access can be controlled by placing limits on the use of DBMS query and reporting tools. The DBA must make sure that those tools are used properly and only by authorized personnel.
- *DBMS usage monitoring.* The DBA must also audit the use of the data in the database. Several DBMS packages contain features that allow the creation of an **audit log**, which automatically records a brief description of the database operations performed by all users. Such audit trails enable the DBA to pinpoint access violations. The audit trails can be tailored to record all database accesses or just failed database accesses.

The integrity of a database could be lost because of external factors beyond the DBA's control. For example, the database might be damaged or destroyed by an explosion, a fire, or an earthquake. Whatever the reason, the specter of database corruption or destruction makes backup and recovery procedures crucial to any DBA.

15.7 DATABASE ADMINISTRATION TOOLS

The importance of the data dictionary as a prime DBA tool cannot be overstated. This section will examine the data dictionary as a data administration tool, as well as the DBA's use of computer-aided software engineering (CASE) tools to support database analysis and design.

15.7.1 THE DATA DICTIONARY

In Chapter 1, a *data dictionary* was defined as "a DBMS component that stores the definition of data characteristics and relationships." You may recall that such "data about data" are called *metadata*. The DBMS data dictionary provides the DBMS with its self-describing characteristic. In effect, the data dictionary resembles an x-ray of the company's entire data set, and it is a crucial element in data administration.

Two main types of data dictionaries exist: *integrated* and *standalone*. An integrated data dictionary is included with the DBMS. For example, all relational DBMSs include a built-in data dictionary or system catalog that is frequently accessed and updated by the RDBMS. Other DBMSs, especially older types, do not have a built-in data dictionary; instead, the DBA may use third-party *standalone data dictionary* systems.

Data dictionaries can also be classified as *active* or *passive*. An **active data dictionary** is automatically updated by the DBMS with every database access, thereby keeping its access information up to date. A **passive data dictionary** is not updated automatically and usually requires running a batch process. Data dictionary access information is normally used by the DBMS for query optimization purposes.

The data dictionary's main function is to store the description of all objects that interact with the database. Integrated data dictionaries tend to limit their metadata to the data managed by the DBMS. Standalone data dictionary systems are usually more flexible and allow the DBA to describe and manage all of the organization's data, whether or not they are computerized. Whatever the data dictionary's format, its existence provides database designers and end users with a much-improved ability to communicate. In addition, the data dictionary is the tool that helps the DBA resolve data conflicts.

Although there is no standard format for the information stored in the data dictionary, several features are common. For example, the data dictionary typically stores descriptions of all:

- *Data elements that are defined in all tables of all databases*. Specifically, the data dictionary stores the names, data types, display format, internal storage format, and validation rules. The data dictionary tells where an element is used, by whom it is used, and so on.
- *Tables defined in all databases*. For example, the data dictionary is likely to store the name of the table creator, the date of creation, access authorizations, and the number of columns.
- *Indexes defined for each database table*. For each index, the DBMS stores at least the index name, the attributes used, the location, specific index characteristics, and the creation date.
- *Defined databases*. This includes who created each database, when the database was created, where the database is located, who the DBA is, and so on.
- *End users and administrators of the database*.
- *Programs that access the database*. This includes screen formats, report formats, application programs, and SQL queries.
- *Access authorizations for all users of all databases*.
- *Relationships among data elements*. This includes which elements are involved, whether the relationships are mandatory or optional, and what the connectivity and cardinality requirements are.

If the data dictionary can be organized to include data external to the DBMS itself, it becomes an especially flexible tool for more general corporate resource management. The management of such an extensive data dictionary thus makes it possible to manage the use and allocation of all of the organization's information, regardless of whether the

information has its roots in the database data. That is why some managers consider the data dictionary to be a key element of information resource management. And that is also why the data dictionary might be described as the *information resource dictionary*.

The metadata stored in the data dictionary are often the basis for monitoring database use and for assigning access rights to the database users. The information stored in the data dictionary is usually based on a relational table format, thus enabling the DBA to query the database with SQL commands. For example, SQL commands can be used to extract information about the users of a specific table or about the access rights of a particular user. In the following example, the IBM DB2 system catalog tables will be used as the basis for several examples of how a data dictionary is used to derive information.

SYSTABLES stores one row for each table or view.
SYSCOLUMNS stores one row for each column of each table or view.
SYSTABAUTH stores one row for each authorization given to a user for a table or view in a database.

Examples of Data Dictionary Usage

Example 1

List the names and creation dates of all tables created by the user JONESVI in the current database.

```
SELECT      NAME, CTIME
FROM        SYSTABLES
WHERE       CREATOR = 'JONESVI';
```

Example 2

List the names of the columns for all tables created by JONESVI in the current database.

```
SELECT      NAME
FROM        SYSCOLUMNS
WHERE       TBCREATOR = 'JONESVI';
```

Example 3

List the names of all tables for which the user JONESVI has DELETE authorization.

```
SELECT      TTNAME
FROM        SYSTABAUTH
WHERE       GRANTEE = 'JONESVI' AND DELETEAUTH = 'Y';
```

Example 4

List the names of all users who have some type of authority over the INVENTORY table.

```
SELECT      DISTINCT GRANTEE
FROM        SYSTABAUTH
WHERE       TTNAME = 'INVENTORY';
```

Example 5

List the user and table names for all users who can alter the database structure for any table in the database.

```
SELECT      GRANTEE, TTNAME
FROM        SYSTABAUTH
WHERE       ALTERAUTH = 'Y'
ORDER BY    GRANTEE, TTNAME;
```

As you can see in the preceding examples, the data dictionary can be a tool for monitoring the security of data in the database by checking the assignment of data access privileges. Although the preceding examples targeted database tables and users, information about the application programs that access the database can also be drawn from the data dictionary.

The DBA can use the data dictionary to support data analysis and design. For example, the DBA can create a report that lists all data elements to be used in a particular application; a list of all users who access a particular program; a report that checks for data redundancies, duplications, and the use of homonyms and synonyms; and a number of other reports that describe data users, data access, and data structure. The data dictionary can also be used to ensure that applications programmers have met all of the naming standards for the data elements in the database and that the data validation rules are correct. Thus, the data dictionary can be used to support a wide range of data administration activities and to facilitate the design and implementation of information systems. Integrated data dictionaries are also essential to the use of computer-aided software engineering tools.

15.7.2 CASE Tools

CASE is the acronym for **computer-aided systems engineering**. A CASE tool provides an automated framework for the Systems Development Life Cycle (SDLC). CASE uses structured methodologies and powerful graphical interfaces. Because they automate many tedious system design and implementation activities, CASE tools play an increasingly important role in information systems development.

CASE tools are usually classified according to the extent of support they provide for the SDLC. For example, **front-end CASE tools** provide support for the planning, analysis, and design phases; **back-end CASE tools** provide support for the coding and implementation phases. The benefits associated with CASE tools include:

- A reduction in development time and costs.
- Automation of the SDLC.
- Standardization of systems development methodologies.
- Easier maintenance of application systems developed with CASE tools.

One of the CASE tools' most important components is an extensive data dictionary, which keeps track of all objects created by the systems designer. For example, the CASE data dictionary stores data flow diagrams, structure charts, descriptions of all external and internal entities, data stores, data items, report formats, and screen formats. A CASE data dictionary also describes the relationships among the components of the system.

Several CASE tools provide interfaces that interact with the DBMS. Those interfaces allow the CASE tool to store its data dictionary information by using the DBMS. Such CASE/DBMS interaction demonstrates the interdependence that exists between systems development and database development, and it helps create a fully integrated development environment.

In a CASE development environment, the database and application designers use the CASE tool to store the description of the database schema, data elements, application processes, screens, reports, and other data relevant to the development process. The CASE tool integrates all systems development information in a common repository, which can be checked by the DBA for consistency and accuracy.

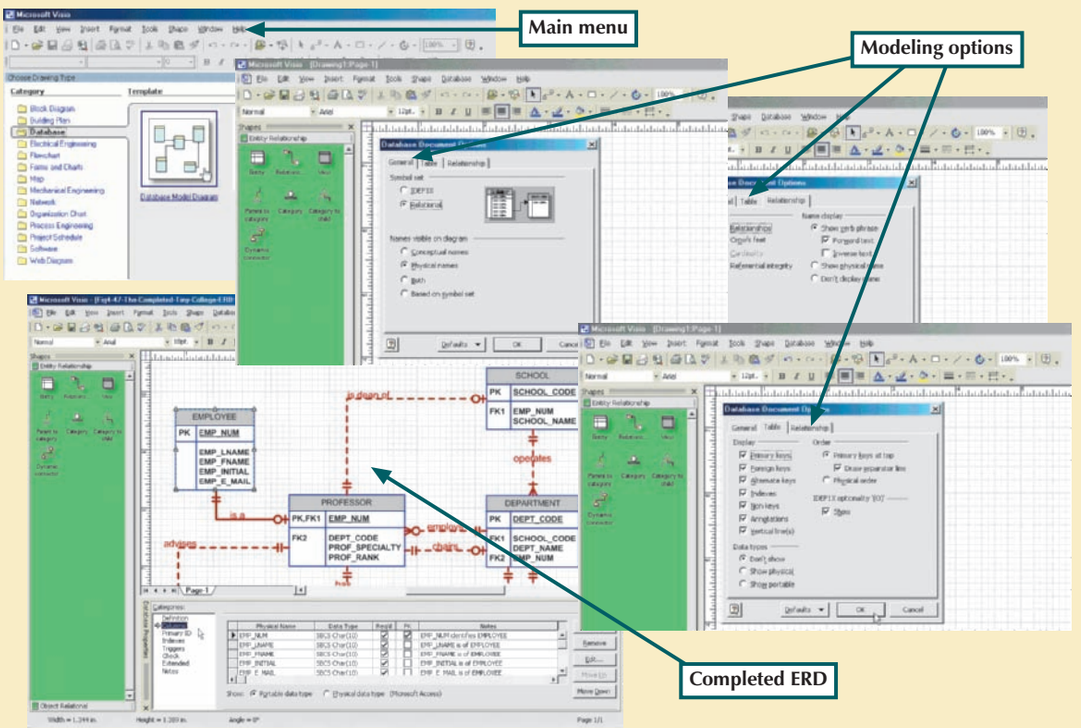
As an additional benefit, a CASE environment tends to improve the extent and quality of communication among the DBA, the application designers, and the end users. The DBA can interact with the CASE tool to check the definition of the data schema for the application, the observance of naming conventions, the duplication of data elements, the validation rules for the data elements, and a host of other developmental and managerial variables. When the CASE tool indicates conflicts, rule(s) violations, and inconsistencies, it facilitates making corrections. Better yet, a correction is transported by the CASE tool to cascade its effects throughout the applications environment, thus greatly simplifying the job of the DBA and the application designer.

A typical CASE tool provides five components:

- Graphics designed to produce structured diagrams such as data flow diagrams, ER diagrams, class diagrams, and object diagrams.
- Screen painters and report generators to produce the information system's input/output formats (for example, the end-user interface).
- An integrated repository for storing and cross-referencing the system design data. This repository includes a comprehensive data dictionary.
- An analysis segment to provide a fully automated check on system consistency, syntax, and completeness.
- A program documentation generator.

Figure 15.7 illustrates how Microsoft Visio Professional can be used to produce an ER diagram.

FIGURE 15.7 **An example of a CASE tool: Visio Professional**



One CASE tool, ERwin Data Modeler by Computer Associates, produces fully documented ER diagrams that can be displayed at different abstraction levels. In addition, ERwin can produce detailed relational designs. The user specifies the attributes and primary keys for each entity and describes the relations. ERwin then assigns foreign keys based on the specified relationships among the entities. Changes in primary keys are always updated automatically throughout the system. Table 15.5 shows a short list of the many available CASE tool vendors.

TABLE 15.5 CASE Tools

COMPANY	PRODUCT	WEB SITE
Casewise	Corporate Modeler Suite	*www.casewise.com*
Computer Associates	ERwin	*www3.ca.com/Solutions/Product.asp?ID=260*
Embarcadero Technologies	ER/Studio	*www.embarcadero.com/products/erstudio*
Microsoft	Visio	*office.microsoft.com/en-us/FX010857981033.aspx*
Oracle	Designer	*www.oracle.com/technology/products/designer*
Telelogic	System Architect	*www.telelogic.com/products/system_architect/sa*
Sybase	Power Designer	*www.sybase.com/products/developmentintegration/powerdesigner*
Visible	Visible Analyst	*www.visible.com/Products/Analyst*

Major relational DBMS vendors, such as Oracle, now provide fully integrated CASE tools for their own DBMS software as well as for RDBMSs supplied by other vendors. For example, Oracle's CASE tools can be used with IBM's DB2, SQL/DS, and Microsoft's SQL Server to produce fully documented database designs. Some vendors even take nonrelational DBMSs, develop their schemas, and produce the equivalent relational designs automatically.

There is no doubt that CASE has enhanced the database designer's and the applications programmer's efficiency. But no matter how sophisticated the CASE tool, its users must be well versed in conceptual design ideas. In the hands of database novices, CASE tools simply produce impressive-looking but bad designs.

15.8 DEVELOPING A DATA ADMINISTRATION STRATEGY

For a company to succeed, its activities must be committed to its main objectives or mission. Therefore, regardless of a company's size, a critical step for any organization is to ensure that its information system supports its strategic plans for each of its business areas.

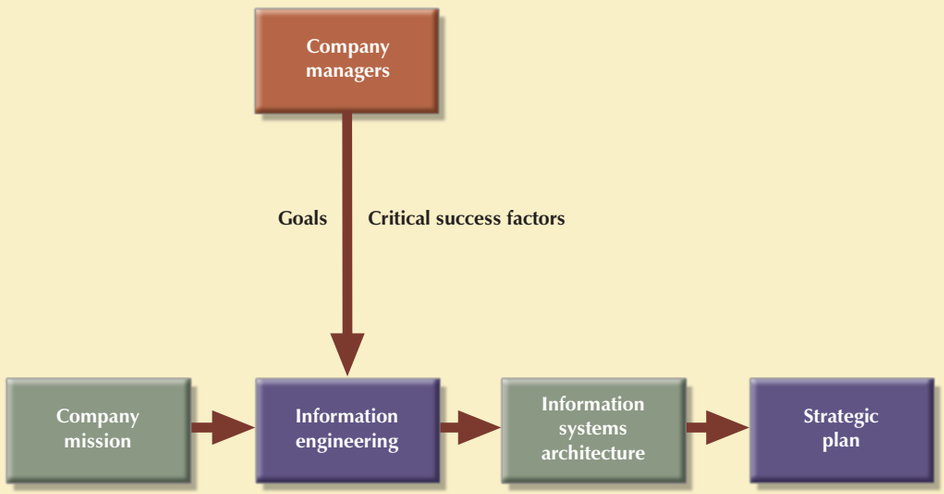
The database administration strategy must not conflict with the information systems plans. After all, the information systems plans are derived from a detailed analysis of the company's goals, its condition or situation, and its business needs. Several methodologies are available to ensure the compatibility of data administration and information systems plans and to guide the strategic plan development. The most commonly used methodology is known as information engineering.

Information engineering (**IE**) allows for the translation of the company's strategic goals into the data and applications that will help the company achieve those goals. IE focuses on the description of the corporate data instead of the processes. The IE rationale is simple: business data types tend to remain fairly stable. In contrast, processes change often and thus require the frequent modification of existing systems. By placing the emphasis on data, IE helps decrease the impact on systems when processes change.

The output of the IE process is an **information systems architecture** (**ISA**) that serves as the basis for planning, development, and control of future information systems. Figure 15.8 shows the forces that affect ISA development.

Implementing IE methodologies in an organization is a costly process that involves planning, a commitment of resources, management liability, well-defined objectives, identification of critical factors, and control. An ISA provides a framework that includes the use of computerized, automated, and integrated tools such as a DBMS and CASE tools.

FIGURE 15.8 Forces affecting the development of the ISA



The success of the overall information systems strategy, and therefore, of the data administration strategy depends on several critical success factors. Understanding the critical success factors helps the DBA develop a successful corporate data administration strategy. Critical success factors include managerial, technological, and corporate culture issues, such as:

- *Management commitment.* Top-level management commitment is necessary to enforce the use of standards, procedures, planning, and controls. The example must be set at the top.
- *Thorough company situation analysis.* The current situation of the corporate data administration must be analyzed to understand the company's position and to have a clear vision of what must be done. For example, how are database analysis, design, documentation, implementation, standards, codification, and other issues handled? Needs and problems should be identified first; then prioritized.
- *End-user involvement.* End-user involvement is another aspect critical to the success of the data administration strategy. What is the degree of organizational change involved? Successful organizational change requires that people be able to adapt to the change. Users should be given an open communication channel to upper-level management to ensure success of the implementation. Good communication is key to the overall process.
- *Defined standards.* Analysts and programmers must be familiar with appropriate methodologies, procedures, and standards. If analysts and programmers lack familiarity, they might need to be trained in the use of the procedures and standards.
- *Training.* The vendor must train the DBA personnel in the use of the DBMS and other tools. End users must be trained to use the tools, standards, and procedures to obtain and demonstrate the maximum benefit, thereby increasing end-user confidence. Key personnel should be trained first so they can train others.
- *A small pilot project.* A small project is recommended to ensure that the DBMS will work in the company, that the output is what was expected, and that the personnel have been trained properly.

That list of factors is not and cannot be comprehensive. Nevertheless, it does provide the initial framework for the development of a successful strategy. Remember that no matter how comprehensive the list of success factors is, it must be based on the notion that development and implementation of a successful data administration strategy are tightly integrated with the overall information systems planning activity of the organization.

15.9 THE DBA AT WORK: USING ORACLE FOR DATABASE ADMINISTRATION

Thus far you've learned about the DBA's work environment and responsibilities in general terms. In this section, you will get a more detailed look at how a DBA might handle the following technical tasks in a specific DBMS:

- Creating and expanding database storage structures.
- Managing database objects such as tables, indexes, triggers, and procedures.
- Managing the end-user database environment, including the type and extent of database access.
- Customizing database initialization parameters.

Many of those tasks require the DBA to use software tools and utilities that are commonly provided by the database vendor. In fact, all DBMS vendors provide a set of programs to interface with the database and to perform a wide range of database administrative tasks.

We chose Oracle 10g for Windows to illustrate the selected DBA tasks because it is typically found in organizations that are sufficiently large and have a sufficiently complex database environment to require (and afford) the use of a DBA, it has good market presence, and it is also often found in small colleges and universities.

NOTE

Although Microsoft Access is a superb DBMS, it is typically used in smaller organizations or in organizations and departments with relatively simple data environments. Access yields a superior database prototyping environment, and given its easy-to-use GUI tools, rapid front-end application development is a snap. Also, Access is one of the components in the MS Office suite, thus making end-user applications integration relatively simple and seamless. Finally, Access does provide some important database administration tools. However, an Access-based database environment does not typically require the services of a DBA. Therefore, MS Access does not fit this section's mission.

Keep in mind that most of the tasks described in this section are encountered by DBAs regardless of their DBMS or their operating system. However, the *execution* of those tasks tends to be specific to the DBMS and the operating system. Therefore, if you use IBM DB2 Universal Database or Microsoft SQL Server, you must adapt the procedures shown here to your DBMS. And because these examples run under the Windows operating system, if you use some other OS, you must adapt the procedures shown in this section to your OS.

This section will not serve as a database administration manual. Instead, it will offer a brief introduction to the way some typical DBA tasks would be performed in Oracle. Before learning how to use Oracle to accomplish specific database administration tasks, you should become familiar with the tools Oracle offers for database administration and with the procedures for logging on, which will be discussed in the next two sections.

15.9.1 Oracle Database Administration Tools

All database vendors supply a set of database administration tools. In Oracle, you perform most DBA tasks via the Oracle Enterprise Manager interface. See Figure 15.9.

In Figure 15.9, note that it shows the status of the current database. (This section uses the ORALAB database.) In the following sections, you will examine the tasks most commonly encountered by a DBA.

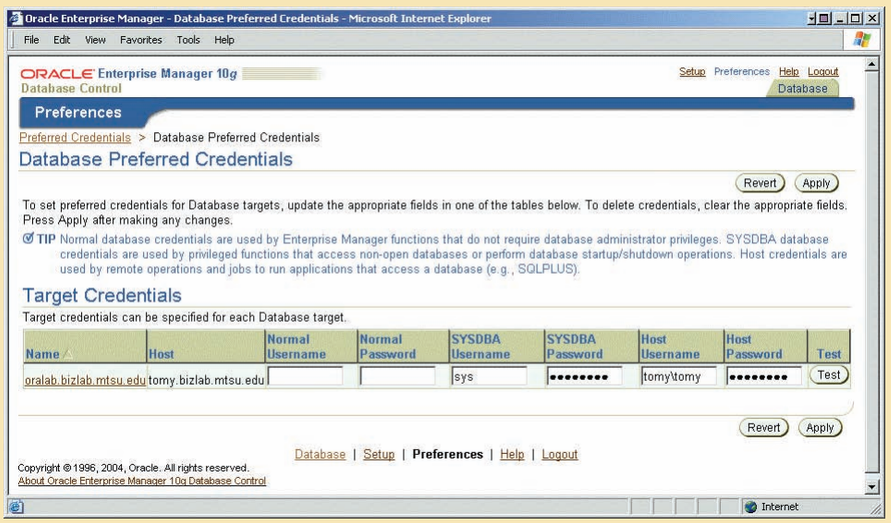
FIGURE 15.9 The Oracle Enterprise Manager interface

15.9.2 THE DEFAULT LOGIN

To perform any administrative task, you must connect to the database, using a username with administrative (DBA) privileges. By default, Oracle automatically creates SYSTEM and SYS user IDs that have administrative privileges with every new database you create. You can define the preferred credentials for each database by clicking on the **Preferences** link at the top of the page, then click on **Preferred Credentials**. Finally, choose your target username under **Set Credentials**. Figure 15.10 shows the Edit Local Preferred Credentials page that defines the user ID (SYS) used to log on to the ORALAB database.

Keep in mind that usernames and passwords are database-specific. Therefore, each database can have different usernames and passwords. One of the first things you must do is change the password for the SYSTEM and SYS users. Immediately after doing that, you can start defining your users and assigning them database privileges.

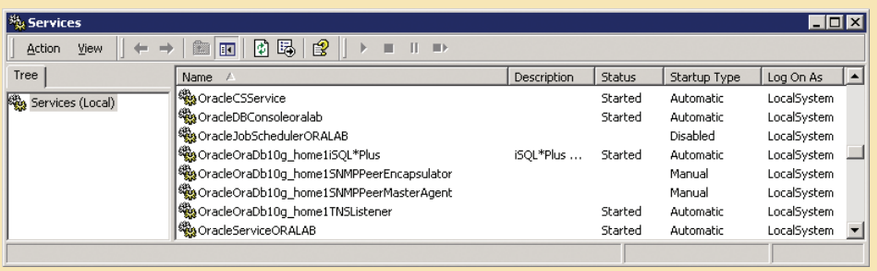
FIGURE 15.10 The Oracle Edit Local Preferred Credentials page



15.9.3 Ensuring an Automatic RDBMS Start

One of the basic DBA tasks is to ensure that your database access is automatically started when you turn on the computer. Startup procedures will be different for each operating system. Because Oracle is used for this section's examples, you would need to identify the required services to ensure automatic database startup. (A *service* is the Windows system name for a special program that runs automatically as part of the operating system. This program ensures the availability of required services to the system and to end users on the local computer or over the network.) Figure 15.11 shows the required Oracle services that are started automatically when Windows starts up.

FIGURE 15.11 Oracle RDBMS services



As you examine Figure 15.11, note the following Oracle services:

- *OracleOraDb10g_home1TNSListener* is the process that "listens to" and processes the end-user connection requests over the network. For example, when a SQL connection request such as "connect userid/password@ORALAB" is sent over the network, the listener service will take the request, validate it, and establish the connection.
- *OracleServiceORALAB* refers to the Oracle processes running in memory that are associated with the ORALAB database instance. You can think of a **database instance** as a separate location in memory that is reserved to run your database. Because you can have several databases (and, therefore, several instances) running in memory at the same time, you need to identify each database instance uniquely, using a different suffix for each one.

15.9.4 Creating Tablespaces and Datafiles

Each DBMS manages data storage differently. In this example, the Oracle RDBMS will be used to illustrate how the database manages data storage at the logical and the physical levels. In Oracle:

A database is *logically* composed of one or more tablespaces. A **tablespace** is a logical storage space. Tablespaces are used primarily to group related data logically.

The tablespace data are *physically* stored in one or more datafiles. A **datafile** physically stores the database's data. Each datafile is associated with one and only one tablespace, but each datafile can reside in a different directory on the hard disk or even on one or more different hard disks.

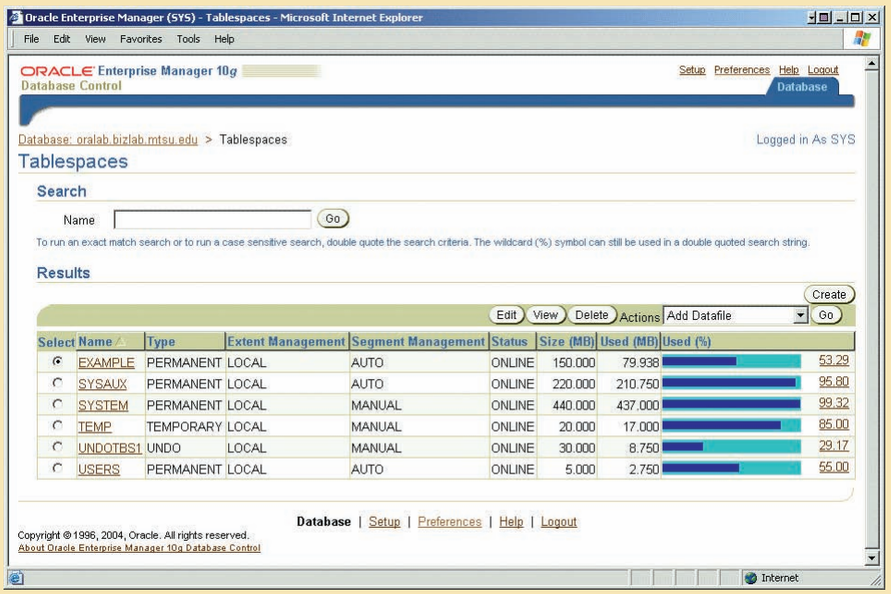
Given the preceding description of tablespaces and datafiles, you can conclude that a database has a one-to-many relationship with tablespaces and that a tablespace has a one-to-many relationship with datafiles. This set of 1:M hierarchical relationships isolates the end user from any physical details of the data storage. However, *the DBA must be aware of these details in order to properly manage the database*.

To perform database storage management tasks such as creating and managing tablespaces and datafiles, the DBA uses Enterprise Manager, Administration, Storage option. See Figure 15.12.

When the DBA creates a database, Oracle automatically creates the tablespaces and datafiles shown in Figure 15.12. A few of them are described here.

- The *SYSTEM* tablespace is used to store the data dictionary data.
- The *USERS* tablespace is used to store the table data created by the end users.
- The *TEMP* tablespace is used to store the temporary tables and indexes created during the execution of SQL statements. For example, temporary tables are created when your SQL statement contains an ORDER BY, GROUP BY, or HAVING clause.
- The *UNDOTBS1* tablespace is used to store database transaction recovery information. If for any reason a transaction must be rolled back (usually to preserve database integrity), the UNDOTBS1 tablespace is used to store the undo information.

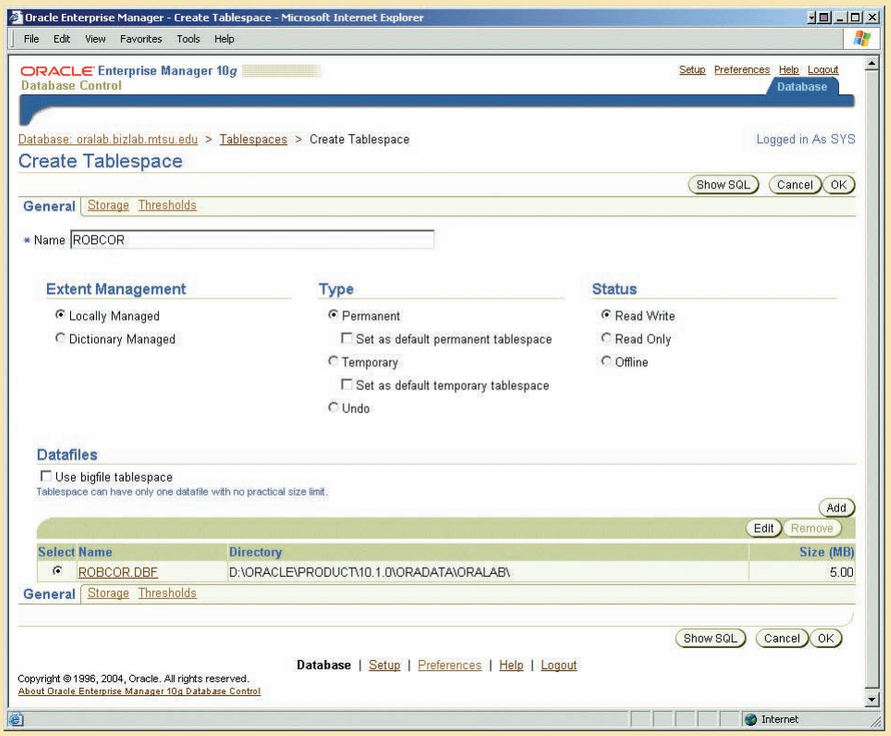
FIGURE 15.12 The Oracle Storage Manager



Using the Storage Manager, the DBA can:

- Create additional tablespaces to organize the data in the database. Therefore, if you have a database with several hundred users, you can create several user tablespaces to segment the data storage for different types of users. For example, you might create a teacher tablespace and a student tablespace.
- Create additional tablespaces to organize the various subsystems that exist within the database. For example, you might create different tablespaces for human resources data, payroll data, accounting data, and manufacturing data. Figure 15.13 shows the page used to create a new tablespace called ROBCOR to hold the tables used in this book. This tablespace will be stored in the datafile named D:\ORACLE\PRODUCT\10.1.0\ORADATA\ORALAB\ROBCOR.DBF and its initial size is 5 megabytes. Note in Figure 15.13 that the tablespace will be put online immediately so it is available to users for data storage purposes. Note also the "Show SQL" button at the top of the page. You can use this button to see the SQL code generated by Oracle to create the tablespace. (Actually, all DBA tasks can also be accomplished through the direct use of SQL commands. In fact, some die-hard DBAs prefer writing their own SQL code rather than using the "easy-way-out" GUI.)
- Expand the tablespace storage capacity by creating additional datafiles. Remember that the datafiles can be stored in the same directory or on different hard disks to increase access performance. For example, you could increase storage and access performance to the USERS tablespace by creating a new datafile in a different drive.

FIGURE 15.13 Creating a new tablespace



15.9.5 MANAGING THE DATABASE OBJECTS: TABLES, VIEWS, TRIGGERS, AND PROCEDURES

Another important aspect of managing a database is monitoring the database objects that were created in the database. The Oracle Enterprise Manager gives the DBA a graphical user interface to create, edit, view, and delete database objects in the database. A **database object** is basically any object created by end users; for example, tables, views, indexes, stored procedures, and triggers. Figure 15.14 shows some of the different types of objects listed in the Oracle Schema Manager.

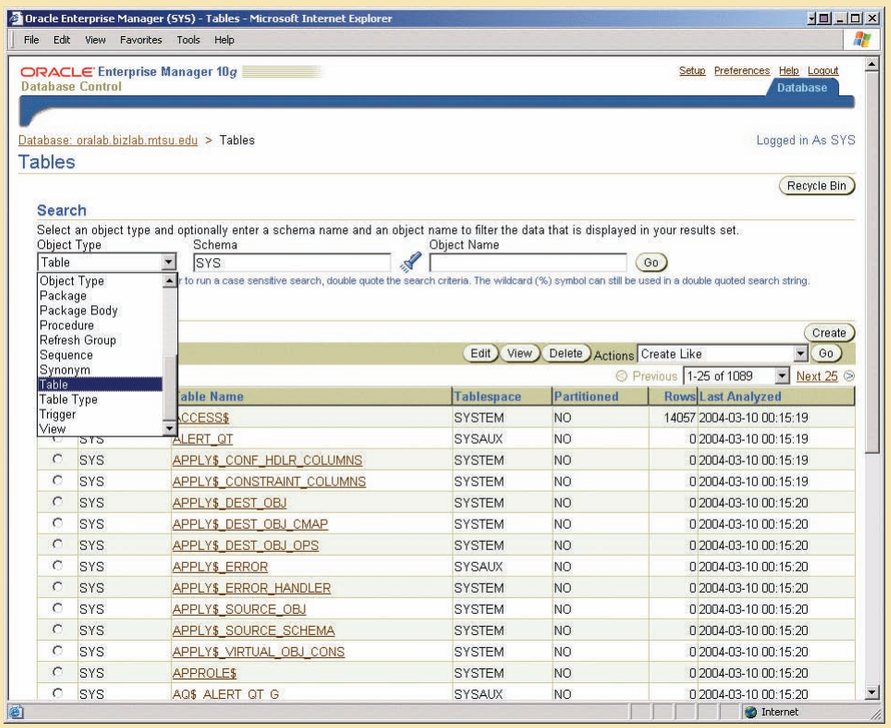
An Oracle **schema** is a logical section of the database that belongs to a given user, and that schema is identified by the username. For example, if the user named SYSTEM creates a VENDOR table, the table will belong to the SYSTEM schema. Oracle prefixes the table name with the username. Therefore, the SYSTEM's VENDOR table name will be named SYSTEM.VENDOR by Oracle. Similarly, if the user PEROB creates a VENDOR table, that table will be created in the PEROB schema and will be named PEROB.VENDOR.

Within the schema, users can create their own tables and other objects. The database can contain as many different schemas as there are users. Because users see only their own object(s), each user might gain the impression that there are no other users of the database.

Normally, users are authorized to access only the objects that belong to their own schemas. Users could, of course, give other users access to their data by changing access rights. In fact, all users with DBA authorization have access to all objects in all schemas in the database.

As you can see in Figure 15.14, the Schema Manager presents an organized view of all of the objects in the database schema. With this program, the DBA can create, edit, view, and delete tables, indexes, views, functions, triggers, procedures, and other specialized objects.

FIGURE 15.14 The Oracle Schema Manager



15.9.6 Managing Users and Establishing Security

One of the most common database administration activities is creating and managing database users. (Actually, the creation of user IDs is just the first component of any well-planned database security function. As was indicated earlier in this chapter, database security is one of the most important database administration tasks.)

The Security section of the Oracle Enterprise Manager's Administration page enables the DBA to create users, roles, and profiles.

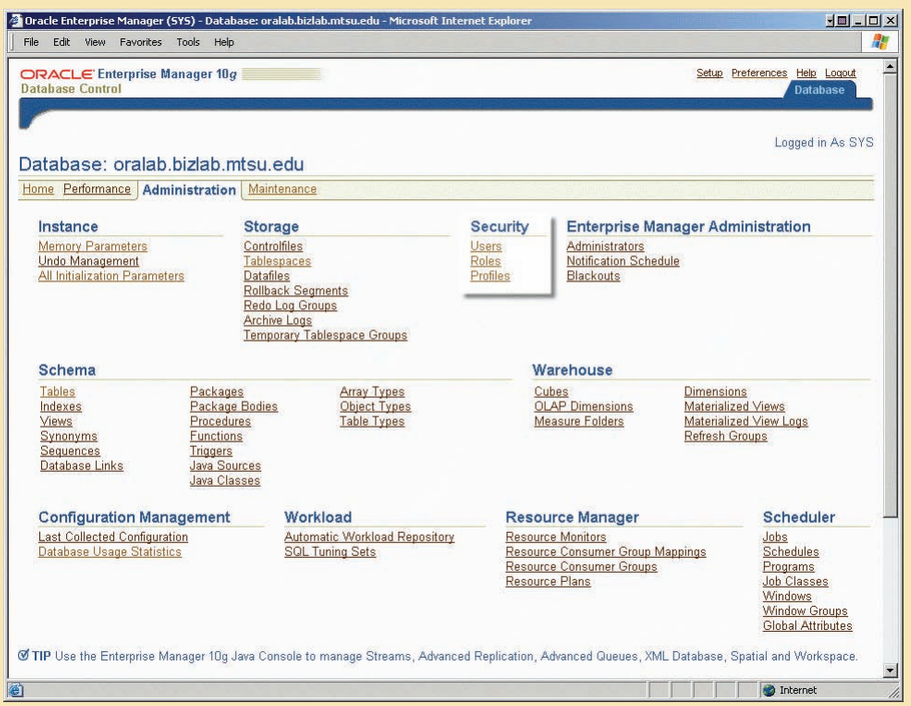
- A **user** is a uniquely identifiable object that allows a given person to log on to the database. The DBA assigns privileges for accessing the objects in the database. Within the privilege assignment, the DBA may specify a set of limits that define how many of the database's resources the user can use.
- A **role** is a named collection of database access privileges that authorize a user to connect to the database and use the database system resources. Examples of roles are as follows:
  - *CONNECT* allows a user to connect to the database and create and modify tables, views, and other data-related objects.

- *RESOURCE* allows a user to create triggers, procedures, and other data management objects.
- *DBA* gives the user database administration privileges.

- A **profile** is a named collection of settings that control how much of the database resource a given user can use. (If you consider the possibility that a runaway query could cause the database to lock up or to stop responding to the user's commands, you'll understand why it is important to limit access to the database resource.) By specifying profiles, the DBA can limit how much storage space a user can use, how long a user can be connected, how much idle time may be used before the user is disconnected, and so on. In an ideal world, all users would have unlimited access to all resources at all times, but in the real world, such access is neither possible nor desirable.

Figure 15.15 shows the Oracle Enterprise Manager Administration page. From here, the DBA can manage the database and create security objects (users, roles, and profiles).

FIGURE 15.15 **The Oracle Enterprise Manager Administration page**

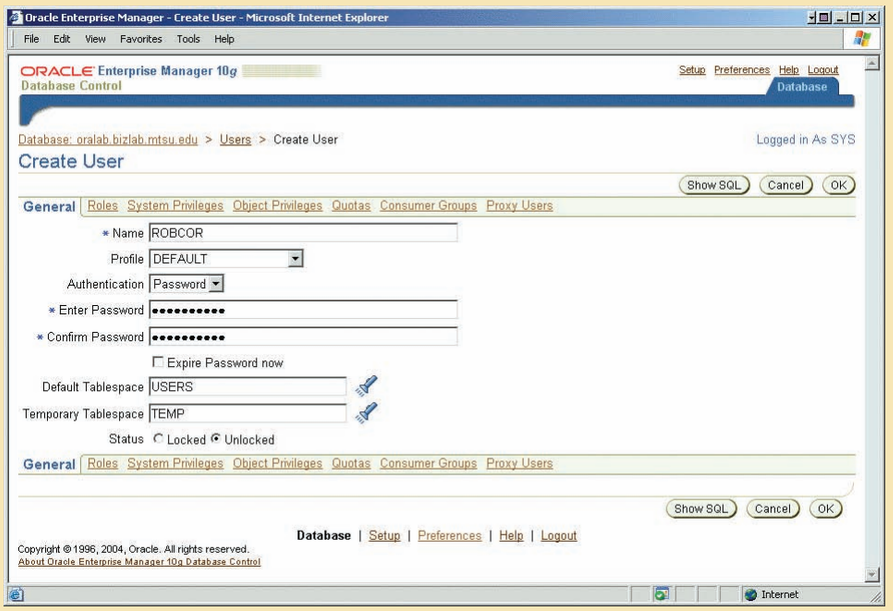


To create a new user, the DBA uses the Create User page, shown in Figure 15.16.

The Create User page contains many links; the most important ones are as follows:

- The *General* link allows the DBA to assign the name, profile, and password to the new user. Also in this page, the DBA defines the default tablespace used to store table data and the temporary tablespace for temporary data.
- The *Roles* link allows the DBA to assign the roles for a user.
- The *Object Privileges* link is used by the DBA to assign specific access rights to other database objects.
- The *Quotas* link allows the DBA to specify the maximum amount of storage that the user can have in each assigned tablespace.

FIGURE 15.16 The Create User page



15.9.7 Customizing the Database Initialization Parameters

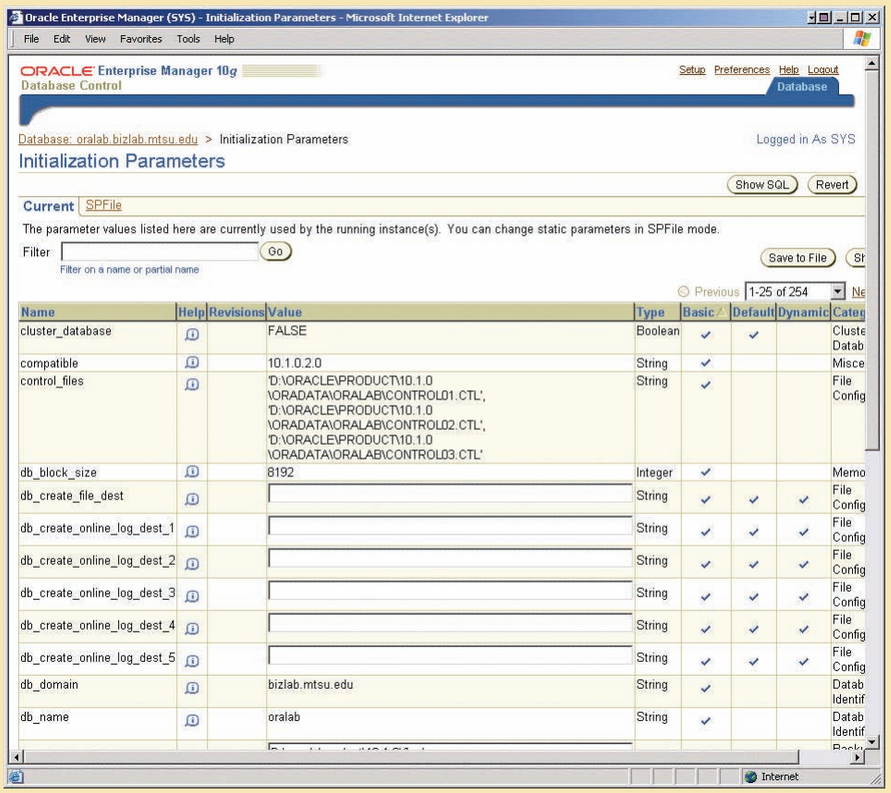
Fine-tuning a database is another important DBA task. This task usually requires the modification of database configuration parameters, some of which can be changed in real time, using SQL commands. Others require the database to be shut down and restarted. Also, some parameters may affect only the database instance, while others affect the entire RDBMS and all instances running. So it is very important that the DBA become familiar with database configuration parameters, especially those that affect performance.

Each database has an associated database initialization file that stores its run-time configuration parameters. The initialization file is read at instance startup and is used to set the working environment for the database. Oracle's Enterprise Manager allows the DBA to start up, shut down, and view/edit the database configuration parameters (stored in the initialization file) of a database instance. The Oracle Enterprise Manager interface provides a GUI to modify that text file, shown in Figure 15.17.

One of the important functions provided by the initialization parameters is to reserve the resources that must be used by the database at run time. One of those resources is the primary memory to be reserved for database caching. Such caching is used to fine-tune database performance. For example, the "db_cache_size" parameter sets the amount of memory reserved for database caching. This parameter should be set to a value that is large enough to support all concurrent transactions.

Once you modify the initialization parameters, you may be required to restart the database. As you have seen in this brief section, the DBA is responsible for a wide range of tasks. The quality and completeness of the administration tools available to the DBA go a long way toward making the DBA job easier. Even so, the DBA must become familiar with the tools and technical details of the RDBMS to perform the DBA tasks properly and efficiently.

FIGURE 15.17 The Oracle Enterprise Manager – Initialization Parameters page



15.9.8 CREATING A NEW DATABASE

Although the general database creation format tends to be generic, its execution tends to be DBMS-specific. The leading RDBMS vendors offer the DBA the option to create databases manually, using SQL commands or using a GUI-based process. Which option is selected depends on the DBA's sense of control and style.

Using the Oracle Database Configuration Assistant, it is simple to create a database. The DBA uses a wizard interface to answer a series of questions to establish the parameters for the database to be created. Figures 15.18 through 15.30 show you how to create a database with the help of the Oracle Database Configuration Assistant.

FIGURE 15.18 Creating a new database with the Database Configuration Assistant

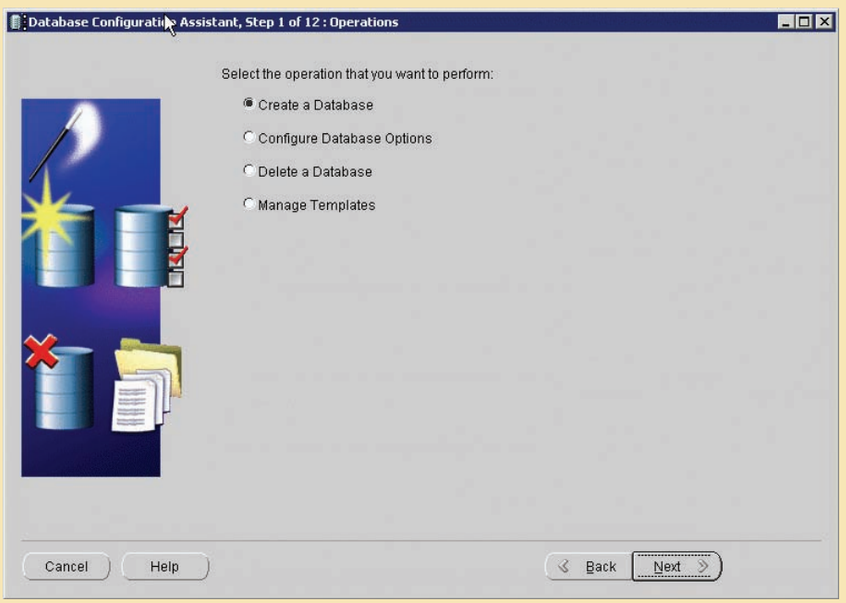


FIGURE 15.19 Selecting the new database template

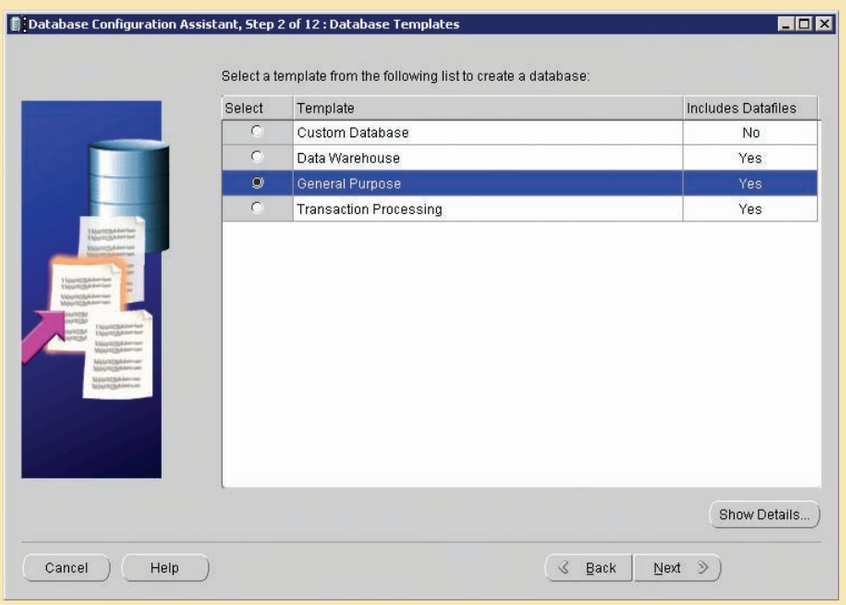


FIGURE 15.20 Naming the database

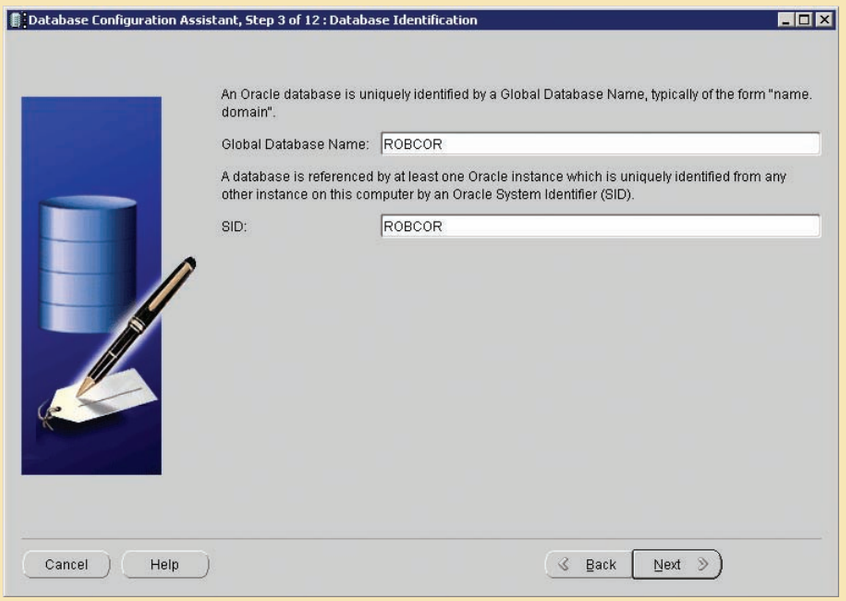


FIGURE 15.21 Selecting management options

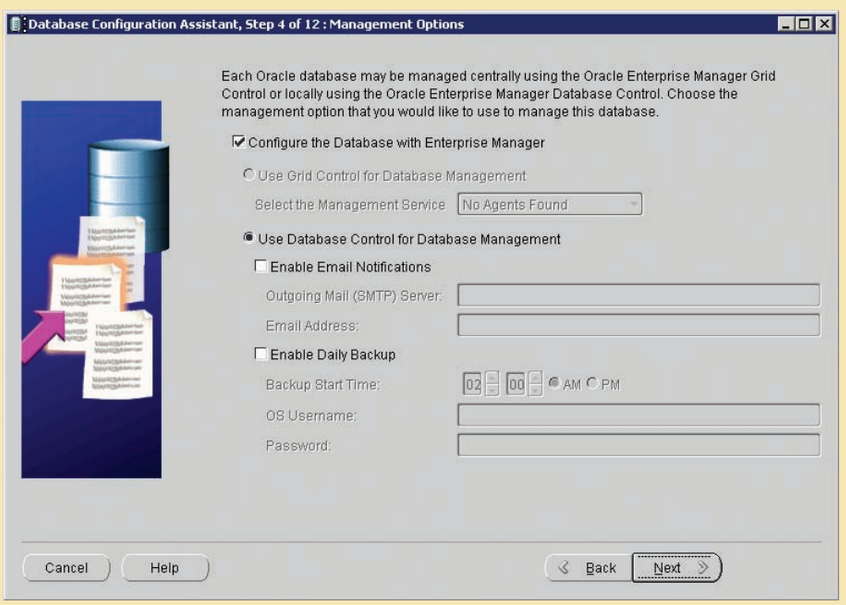


FIGURE 15.22 Specifying database credentials

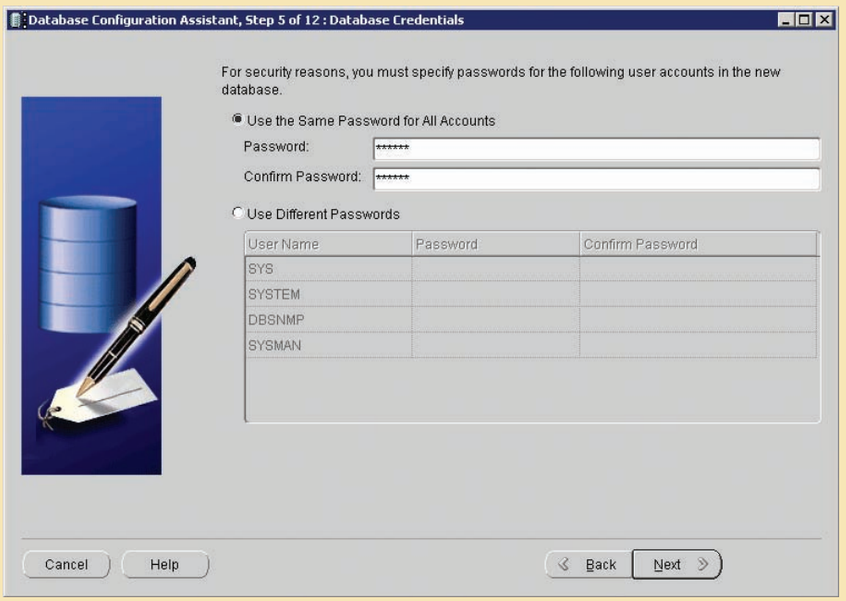


FIGURE 15.23 Selecting storage options

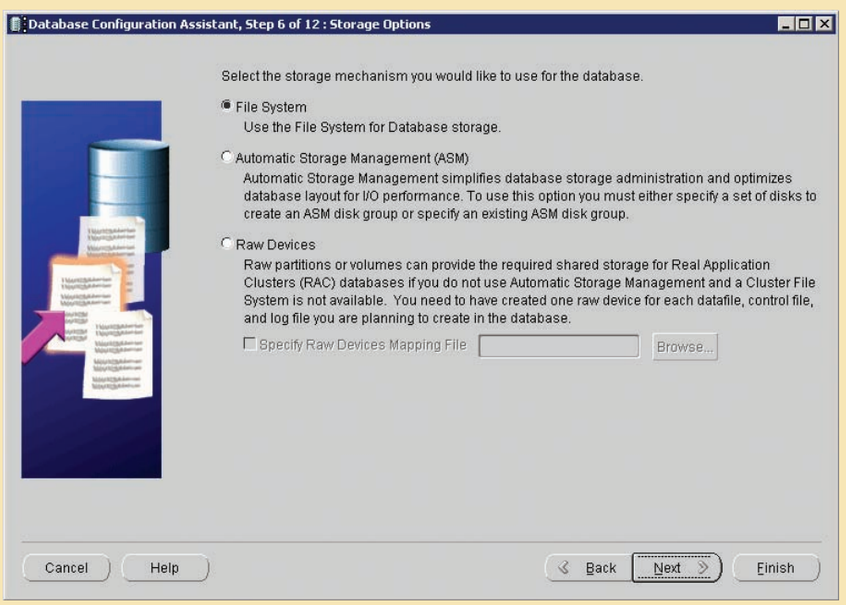


FIGURE 15.24 Specifying database file locations

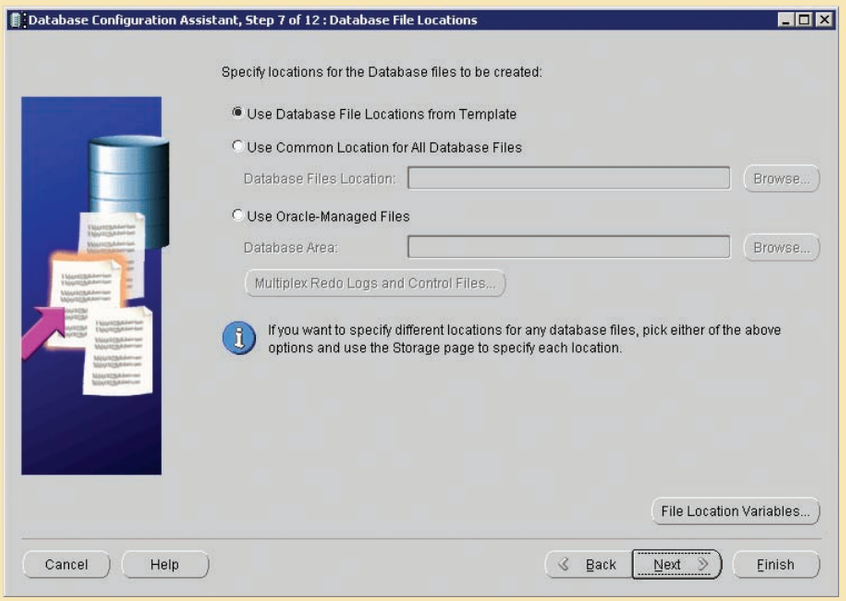


FIGURE 15.25 Specifying database recovery configuration

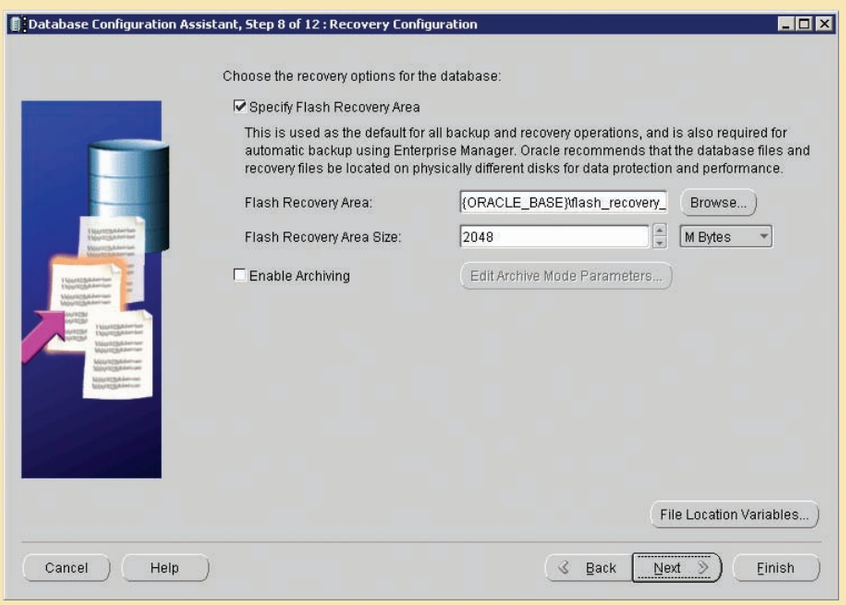


FIGURE 15.26 Selecting database sample content

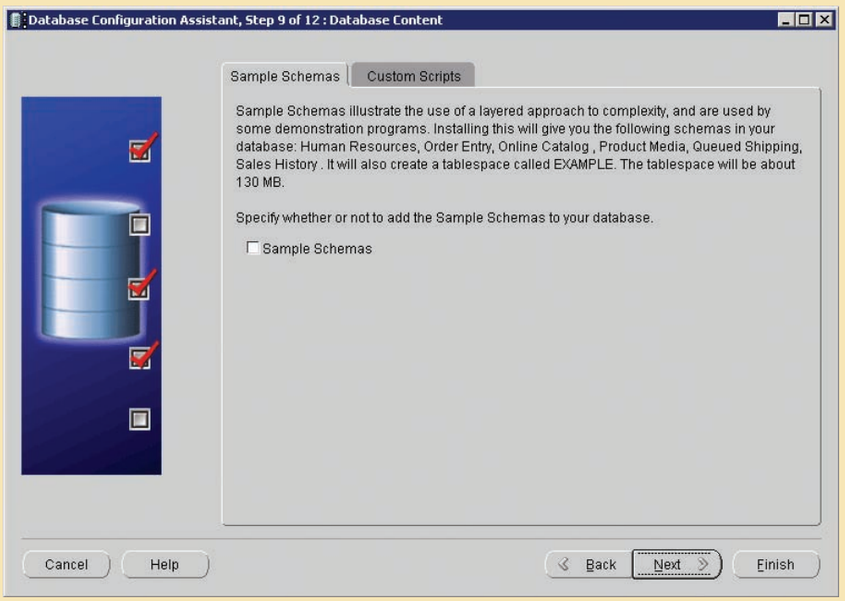


FIGURE 15.27 Selecting database initialization parameters

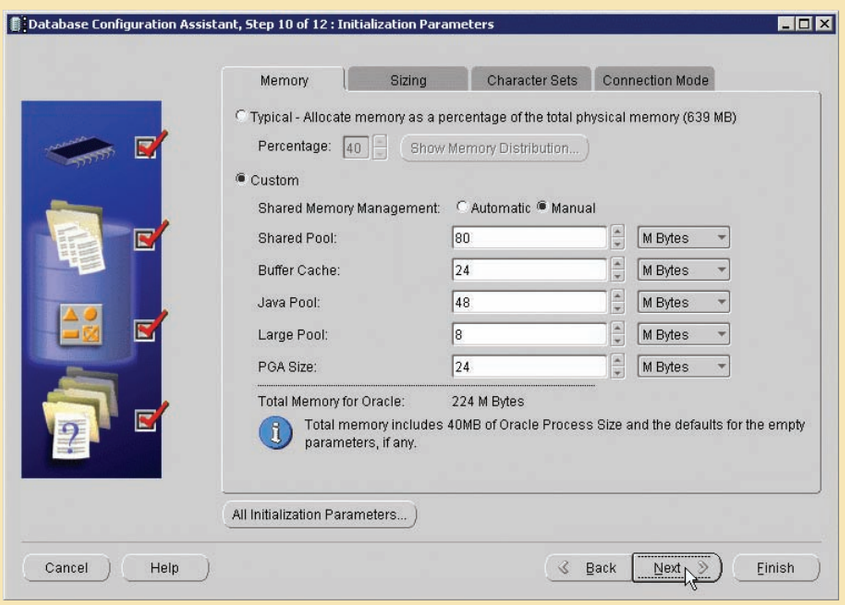


FIGURE 15.28 Confirming database storage parameters

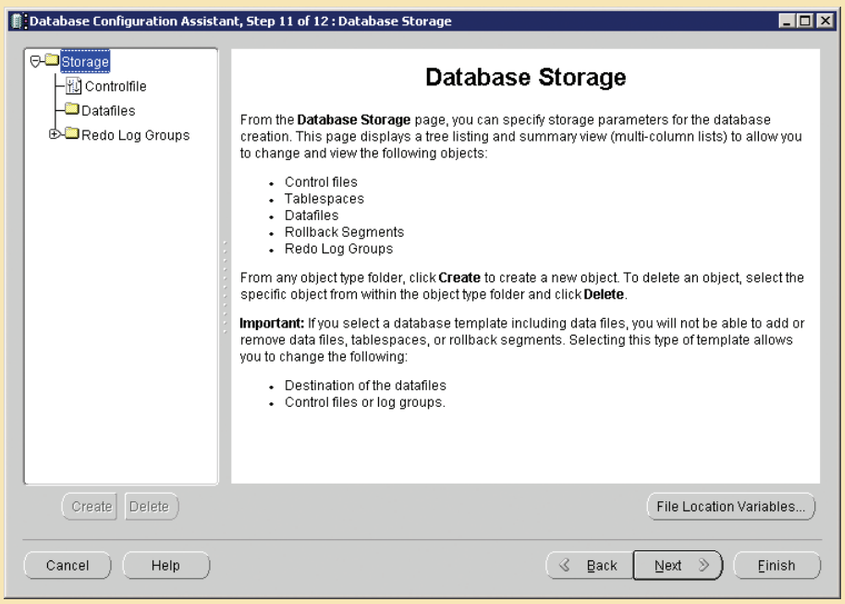


FIGURE 15.29 Confirming database creation options

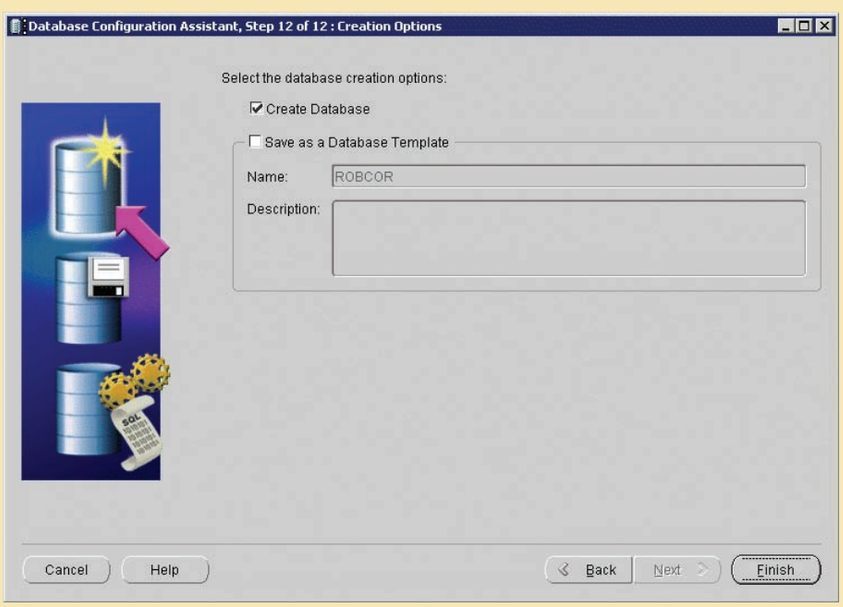
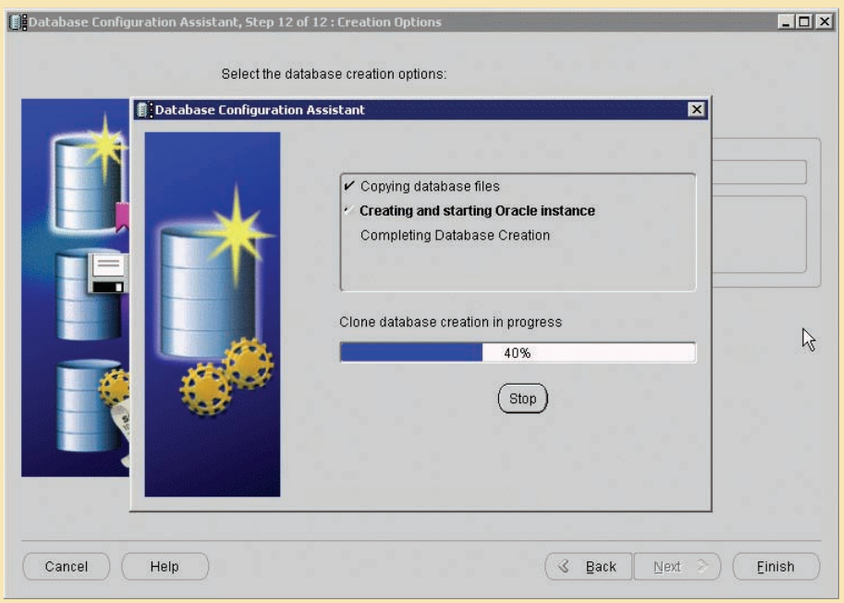


FIGURE 15.30 Database creation progress



Finally, after confirming all of the database options selected in Figures 15.18 through 15.29, the database creation process starts. This process creates the database structure, including the necessary data dictionary tables, the administrator user accounts, and other supporting processes required by the DBMS to manage the database. Figure 15.30 shows the rate at which the database creation process proceeds.

One of the disadvantages of using this graphical interface to create databases is that there are no records in the form of SQL scripts to document the steps. Even given the GUI's help, the database creation process requires a solid understanding of the database's underlying structures and components.

SUMMARY

- Data management is a critical activity for any organization. Data must be treated as a corporate asset. The value of a data set is measured by the utility of the information derived from it. Good data management is likely to produce good information, which is the basis for better decision making.
- The DBMS is the most commonly used electronic tool for corporate data management. The DBMS supports strategic, tactical, and operational decision making at all levels of the organization. The company data that are managed by the DBMS are stored in the corporate or enterprise database.
- The introduction of a DBMS into an organization is a very delicate job. In addition to managing the technical details of DBMS introduction, the impact of the DBMS on the organization's managerial and cultural framework must be carefully examined.
- Development of the data administration function is based on the evolution from departmental data processing to the more centralized electronic data processing (EDP) department to the more formal "data as a corporate asset" information systems (IS) department. Typical file systems were characterized by applications that tended to behave as distinct "islands of information." As applications began to share a common data repository, the need for centralized data management to control such data became clear.
- The database administrator (DBA) is responsible for managing the corporate database. The internal organization of the database administration function varies from company to company. Although no standard exists, it is common practice to divide DBA operations according to the database life cycle phases. Some companies have created a position with a broader data management mandate to manage computerized and other data within the organization. This broader data management activity is handled by the data administrator (DA).
- The DA and the DBA functions tend to overlap. Generally speaking, the DA is more managerially oriented than the more technically-oriented DBA. Compared to the DBA function, the DA function is DBMS-independent, with a broader and longer-term focus. However, when the organization chart does not include a DA position, the DBA executes all of the DA's functions. Because the DBA has both technical and managerial responsibilities, the DBA must have a diverse mix of skills.
- The managerial services of the DBA function include at least: supporting the end-user community; defining and enforcing policies, procedures, and standards for the database function; ensuring data security, privacy, and integrity; providing data backup and recovery services; and monitoring the distribution and use of the data in the database.
- The technical role requires the DBA to be involved in at least these activities: evaluating, selecting, and installing the DBMS; designing and implementing databases and applications; testing and evaluating databases and applications; operating the DBMS, utilities, and applications; training and supporting users; and maintaining the DBMS, utilities, and applications.
- Security refers to activities and measures to ensure the confidentiality, integrity, and availability of an information system and its main asset, data. A security policy is a collection of standards, policies, and practices created to guarantee the security of a system and ensure auditing and compliance.
- A security vulnerability is weakness in a system component that could be exploited to allow unauthorized access or service disruption. A security threat is an imminent security violation caused by an unchecked security vulnerability. Security vulnerabilities exist in all components of an information system: people, hardware, software, network, procedures, and data. Therefore, it is critical to have robust database security. Database security refers to the use of DBMS features and related measures to comply with the security requirements of the organization.

- The development of the data administration strategy is closely related to the company's mission and objectives. Therefore, the development of an organization's strategic plan corresponds to that of data administration, requiring a detailed analysis of company goals, situation, and business needs. To guide the development of this overall plan, an integrating methodology is required. The most commonly used integrating methodology is known as information engineering (IE).
- To help translate strategic plans into operational plans, the DBA has access to an arsenal of database administration tools. These tools include the data dictionary and computer-aided software engineering (CASE) tools.

KEY TERMS

access plan, 622
active data dictionary, 630
audit log, 629
authorization management, 629
back-end CASE tools, 632
CASE (computer-aided systems engineering), 632
concurrent backup, 619
data administrator (DA), 612
database administrator (DBA), 611
database instance (Oracle), 639
database object (Oracle), 641
database security, 628
database security officer (DSO), 618
datafile (Oracle), 639
disaster management, 619
enterprise database, 609
front-end CASE tools, 632
full backup (database dump), 619
incremental backup, 619
information engineering (IE), 634
information resource dictionary, 631
information resource manager (IRM), 612
information systems architecture (ISA), 634
information systems (IS) department, 610
passive data dictionary, 630
policies, 617
privacy, 608
procedures, 617
profile (Oracle), 643
role (Oracle), 642
schema (Oracle), 641
security, 627
security vulnerability, 627
security threat, 627
security breach, 627
standards, 0
systems administrator , 612
tablespace (Oracle), 639
user (Oracle), 642

ONLINE CONTENT

Answers to selected Review Questions for this chapter are contained in the Student Online Companion for this book.

REVIEW QUESTIONS

1. Explain the difference between data and information. Give some examples of raw data and information.
2. Explain the interactions among end user, data, information, and decision making. Draw a diagram and explain the interactions.
3. Suppose you are a DBA staff member. What data dimensions would you describe to top-level managers to obtain their support for the data administration function?
4. How and why did database management systems become the organizational data management standard? Discuss some advantages of the database approach over the file-system approach.
5. Using a single sentence, explain the role of databases in organizations. Then explain your answer.
6. Define *security* and *privacy*. How are those two concepts related?
7. Describe and contrast the information needs at the strategic, tactical, and operational levels in an organization. Use examples to explain your answer.

8. What special considerations must you take into account when contemplating the introduction of a DBMS into an organization?
9. Describe the DBA's responsibilities.
10. How can the DBA function be placed within the organization chart? What effect(s) will that placement have on the DBA function?
11. Why and how are new technological advances in computers and databases changing the DBA's role?
12. Explain the DBA department's internal organization, based on the DBLC approach.
13. Explain and contrast the differences and similarities between the DBA and DA.
14. Explain how the DBA plays an arbitration role between an organization's two main assets. Draw a diagram to facilitate your explanation.
15. Describe and characterize the skills desired for a DBA.
16. What are the DBA's managerial roles? Describe the managerial activities and services provided by the DBA.
17. What DBA activities are used to support the end-user community?
18. Explain the DBA's managerial role in the definition and enforcement of policies, procedures, and standards.
19. Protecting data security, privacy, and integrity are important database functions. What activities are required in the DBA's managerial role of enforcing those functions?
20. Discuss the importance and characteristics of database backup and recovery procedures. Then describe the actions that must be detailed in backup and recovery plans.
21. Assume that your company assigned you the responsibility of selecting the corporate DBMS. Develop a checklist for the technical and other aspects involved in the selection process.
22. Describe the activities that are typically associated with the design and implementation services of the DBA technical function. What technical skills are desirable in the DBA's personnel?
23. Why are testing and evaluation of the database and applications not done by the same people who are responsible for design and implementation? What minimum standards must be met during the testing and evaluation process?
24. Identify some bottlenecks in DBMS performance. Then propose some solutions used in DBMS performance tuning.
25. What are typical activities involved in the maintenance of the DBMS, utilities, and applications? Would you consider application performance tuning to be part of the maintenance activities? Explain your answer.
26. How do you normally define security? How is your definition of security similar to or different from the definition of database security in this chapter?
27. What are the levels of data confidentiality?
28. What are security vulnerabilities? What is a security threat? Give some examples of security vulnerabilities that exist in different IS components.
29. Define the concept of a data dictionary. Discuss the different types of data dictionaries. If you were to manage an organization's entire data set, what characteristics would you look for in the data dictionary?
30. Using SQL statements, give some examples of how you would use the data dictionary to monitor the security of the database.

NOTE

If you use IBM DB2, the names of the main tables are SYSTABLES, SYSCOLUMNS, and SYSTABAUTH.

31. What characteristics do a CASE tool and a DBMS have in common? How can those characteristics be used to enhance the data administration function?

32. Briefly explain the concepts of information engineering (IE) and information systems architecture (ISA). How do those concepts affect the data administration strategy?
33. Identify and explain some of the critical success factors in the development and implementation of a successful data administration strategy.
34. What is the tool used by Oracle to create users?
35. In Oracle, what is a tablespace?
36. In Oracle, what is a database role?
37. In Oracle, what is a datafile? How does it differ from a file systems file?
38. In Oracle, what is a database profile?
39. In Oracle, what is a database schema?
40. In Oracle, what role is required to create triggers and procedures?