

Database Administration

Chapter Outline

- Introduction, 480
- Two-Minute Chapter, 481
- Data Administrator, 482
- Database Administrator, 483
- Database Structure, 485
- Metadata, 486
- Database Tasks by Development Stages, 488
 - Database Planning, 488*
 - Database Design, 489*
 - Database Implementation, 489*
 - Database Operation and Maintenance, 490*
- Backup and Recovery, 492
- Physical Configuration, 494
- Security and Privacy, 496
 - Data Privacy, 497*
 - Threats, 498*
 - Physical Security, 499*
 - Managerial Controls, 500*
 - Logical Security, 500*
- Division of Duties, 506*
- Software Updates, 507*
- Encryption, 507
- Sally's Pet Store, 510
- Summary, 512
- Key Terms, 514
- Review Questions, 514
- Exercises, 515
- Web Site References, 519
- Additional Reading, 519

What You Will Learn in This Chapter

- What administrative tasks need to be performed with a database application?
- How do you ensure data is consistent across multiple databases?
- What are the basic tasks of a database administrator?
- How does a DBMS support multiple databases?
- How does a DBA find out what is stored in each database?
- What DBA tasks need to be performed as an application is developed?
- How do you back up data that is constantly changing?
- How should computers be configured for DBMS software and database files?
- What security techniques are used to protect databases?
- How do you prevent eavesdroppers or hackers from reading data?
- What security conditions would be needed at Sally's Pet Store?

A Developer's View

Miranda: Finally, everything seems to be running well.

Ariel: Does that mean you finally got paid?

Miranda: Yes. They gave me the check yesterday. They even liked my work so well, they offered me a job.

Ariel: That's great. Are you going to take it? What job is it?

Miranda: I think so. They want me to be a database administrator. They said they need me to keep the database running properly. They also hinted

that they want me to help their existing programmers learn to build database applications.

Ariel: Wow! That means you'll get more money than the programmers.

Miranda: Probably. But I'll have to learn some new material. I'm really starting to worry about security. The accounting manager talked to me yesterday and gave me some idea of the problems that I can expect with the sales application.

Getting Started

Someone has to perform several administrative tasks to keep a database running. The DBMS software has to be installed, monitored, and updated. Databases have to be backed up. Security permissions have to be assigned, tested, and revised. Applications and queries need to be optimized.

Introduction

What administrative tasks need to be performed with a database application?

The power of a DBMS comes from its ability to share data. Data can be shared across many users, departments, and applications. Most organizations build more than one application and more than one database. Large organizations might use more than one DBMS. Most companies have several projects being developed or revised at the same time by different teams. Imagine what happens if you just turn developers loose to create databases, tables, and applications anyway they want to. It is highly unlikely the applications would work together. Just using a DBMS is not enough. An organization that wants to build integrated applications must have someone in charge of the data and the databases.

Data administration consists of the planning and coordination required to define data consistently throughout the company. Some person or group should have the responsibility for determining what data should be collected, how it should be stored, and promoting ways in which it can be used. This person or group is responsible for the integrity of the data.

Database administration consists of technical aspects of creating and running the database. The basic tasks are performance monitoring, backup and recovery, and assigning and controlling security. Database administrators are trained in the details of installing, configuring, and operating the DBMS. Smaller organiza-

tions might have a single person responsible for the data administration and DBA roles. Larger companies tend to have multiple DBAs, but only one or two data administrators.

Computer security is increasingly important to organizations. Large organizations usually have a head computer security officer responsible for setting policies, identifying major threats, monitoring compliance, and organizing responses to threats. However, security topics need to be studied by all application developers and database managers. Security is not a separate set of topics that can be added on at the end of a project. Security issues need to be integrated into the design and development process. Some of the more critical issues have been mentioned in earlier chapters. In particular, the SQL injection attack is best solved by developers adding code to test all input values. Likewise, developers can use selection boxes and radio buttons to limit user input. They can also create menu options that are visible only to selected groups of users.

Database security is a subset of computer security topics, and it is important to build security at multiple levels. Some of these levels are best handled centrally by the DBA. If database security is assigned properly, it has the ability to reduce many types of fraud. If database security is ignored or performed poorly, major assets of the company could be manipulated or stolen from any computer in the world. It pays to understand the security issues and to handle security properly. This chapter presents an introduction to DBA and security tasks. Both of these roles require considerable additional learning and practice. This chapter focuses on the general tasks that need to be performed with any DBMS, but many DBA tasks rely on specific features or a particular DBMS. If you choose to become a DBA, you will have to study a particular system in detail.

Two-Minute Chapter

Database systems are powerful tools and they easily handle several complex tasks. But because data is so critical, someone has to monitor and manage the databases. Database design and consistency are critical to being able to integrate data. But performance, security, and backup and recovery are also critical operations to ensure the long-term value of the data. A database administrator (DBA) is in charge of keeping the database system running, planning upgrades, and monitoring performance. The larger systems including SQL Server, Oracle, and DB2 are complex tools with many internal controls and options that can be configured. It takes several months of study and practical experience to become a good DBA.

Computer security concepts are important in database design and operations. Typically, roles are assigned to cover specific tasks, such as an Order Entry Clerk which needs the ability to lookup Customer and Product data and create new Orders. Permissions to access tables have to be given to each role. Then the roles are assigned to individual employees or groups of employees. The permissions have to be thoroughly tested so that the individuals have enough access to complete their tasks.

Physical security has to be established through physical locks, fire safety, and other standard precautions. Backup facilities are critical to any company. If time is critical, companies will run duplicate data centers and share the workload and data. If one center fails, the other can immediately pick up the load and later be expanded to handle more of the operations. Cloud-based systems are useful for backup because they can be expanded or contracted without incurring huge fixed costs.

Some data needs to be encrypted both for data transmission and for safety in storing the data. Common examples include credit card numbers and taxpayer IDs. Encryption tools are built into the higher-end systems but they usually require configuration. An important step is securing the encryption keys so that even if someone steals the database, they will not be able to decrypt the data.

Data Administrator

How do you ensure data is consistent across multiple databases? Data is an important asset to companies. Think about how long a modern company would survive if its computers were suddenly destroyed or all the data lost. Some organizations might survive as long as a few days or a week. Many, like banks, would be out of business immediately. A company should not have to lose any data before it recognizes the value of the information contained in the data. As indicated by Figure 10.1, companies have many databases for different purposes. Over time, organizations build different databases and applications to support decisions for operations, tactics, and strategies. Each application is important by itself, but when the applications and databases can coordinate and exchange data, managers receive a complete picture of the entire organization.

Despite the power and flexibility of database systems, applications built at different times by different people do not automatically share data. The key to integrating data is to put someone in charge of the data resources of the company. In most companies the **data administrator (DA)** fills this position.

As summarized in Figure 10.2, the primary role of the DA is to provide centralized control over the data for the entire organization. The DA sets data definition standards to ensure that all applications use consistent formats and naming conventions. The DA coordinates applications and teams to ensure that data from individual projects can be integrated into a corporatewide information system. If disputes occur among developers or managers, the DA serves as the judge, mak-

Figure 10.1

Data administration. With many projects and developers, a data administrator coordinates the projects so data can be integrated across applications.



Provide centralized control over the data.
 Data definition: format and naming convention.
 Data integration.
 DBMS selection.

Act as data and database advocate.
 Application ideas.
 Decision support.
 Strategic uses.

Coordinate data integrity, security, privacy, and control.

Figure 10.2

Data administrator roles. The DA is responsible for maintaining the quality of the data and for integrating data across the organization. The DA also advocates the use of databases and is often in charge of security.

ing decisions to ensure compatibility across the organization. The DA also monitors the database industry and watches trends and technologies to advise the company on which database systems and tools to consider for long-term benefits.

The DA plays a crucial role as an advocate. Most managers and many developers are not aware of the power and capabilities of modern database systems. By understanding the managerial tasks and the database capabilities, the DA is in a position to suggest new applications and expanded uses of the existing data.

Ultimately, the DA is also responsible for the integrity of the data: Does the data contained in the DBMS represent a true picture of the firm? Does the firm have the proper systems and controls in place to ensure the accuracy and timeliness of the data?

The DA position is largely a management job. The DA tasks consist of organizing and controlling the design aspects of application development. Control is maintained by setting standards, monitoring ongoing development and changes, and providing assistance in database design as needed. The DA also spends time with business managers to evaluate current systems, monitor business trends, and identify future needs. The person hired for this position usually has several years of experience in designing databases and needs a detailed knowledge of the company. The DA also needs technical database skills to understand the various storage implications of the decisions. The DA must also be able to communicate easily with technical managers and business managers.

Database Administrator

What are the basic tasks of a database administrator? A DBMS is a complex software package. Installing, running, and upgrading a DBMS are not trivial tasks. Even with personal computer-based systems, these tasks can require the services of a full-time person. Every database requires the services of a **database administrator (DBA)**. The DBA position is generally staffed by a specialist who is trained in the administration of a particular DBMS. In smaller companies, instead of hiring a specialist, one of the lead developers may be asked to perform DBA duties.

The DBA role is relatively technical. As highlighted in Figure 10.3, the DBA's responsibilities include installing and upgrading the DBMS. Additional tasks include creating user accounts and monitoring security. The DBA is also responsible for managing backups. Although the actual backup task may be performed by a

- Install and upgrade DBMS.
- Create user accounts and monitor security.
- Backup and recovery of the database.
- Monitor and tune the database performance.
- Coordinate with DBMS vendor and plan for changes.
- Maintain DBMS-specific information for developers.

Figure 10.3

Database administrator roles. The DBA tasks are fairly technical and require daily monitoring and changes to the DBMS.

system operator, the DBA is responsible for setting schedules and making sure the data backups are safe. The DBA also monitors the performance of the databases and plans upgrades and additional capacity. The DBA must stay in contact with the DBMS vendor to track system problems and to be notified of changes. As new utilities, tools, or information are provided, the DBA functions as a liaison to gather this knowledge and make it available to developers. The DBA has complete access to the data in the application. In many organizations the DBA is in charge of security for each database. Larger companies might appoint a special security officer to specify policies and procedures and to help with the monitoring. However, the DBA is generally in charge of carrying out the technical details of assigning security privileges for the database.

Data allocation and storage are an important part of the daily tasks of the DBA. Some large database systems require the DBA to preassign a space on the disk drive for each database. Many systems allocate physical space by creating datafiles and **tablespaces**, which are logical collections of space where data can be stored.

Separate space is usually allocated for the data tables, the indexes, and the transaction logs, and the DBA must estimate the size of each component. If the DBA allocates too little space, performance will suffer; on the other hand, allocating too much space means that the company will waste money on unneeded disk drive capacity. Most systems provide tools to add space later, but it is best to get good estimates up front. The data volume estimates from Chapter 3 provide crucial information in determining the space requirements. For tables, the main concept is to determine the size of an average row (in bytes) and multiply by the expected number of rows in the table. Note that each DBMS stores data slightly differently and some add bytes per row of storage. The documentation will provide details for each DBMS. A more accurate solution is to set up a temporary database, create a few rows of data in each table, and then use the actual average space to estimate future needs. Space required for the indexes and rollback log depend on the specific DBMS and the computer system. If you need highly accurate estimates, you will have to consult the documentation and support tools for your specific DBMS. Space for indexes and logs also depends on the number and length of transactions defined in the applications. For example, the transaction log in a database used for transaction processing will have to be substantially larger than the log in a database used primarily for decision support and data retrieval. The larger DBMSs provide tools to help estimate and monitor storage space.

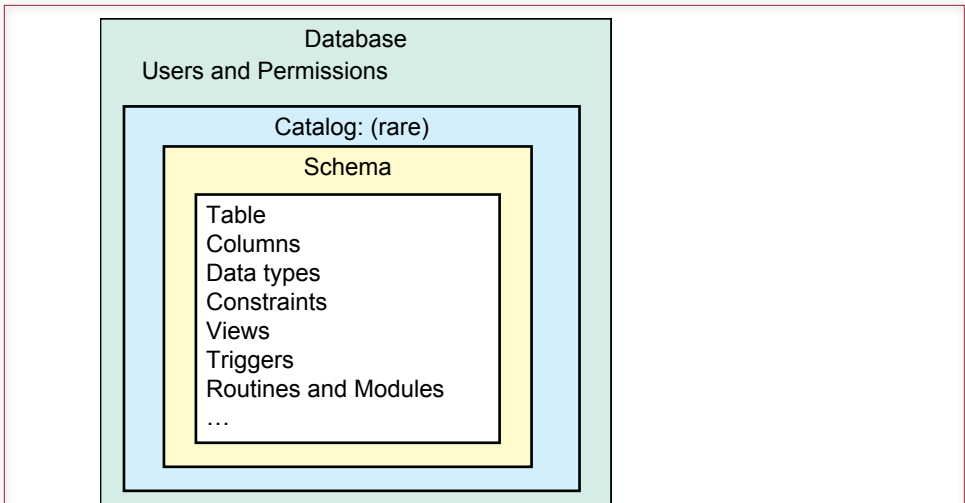


Figure 10.4

Database structure. The schema serves as a container for other elements to minimize potential naming conflicts.

Database Structure

How does a DBMS support multiple databases? The DBA works on a daily basis with the structure of the database. Although each DBMS has slightly different characteristics, Figure 10.4 shows the overall structure of a database as defined by the SQL standard. Users are defined within the individual database instance and granted permissions by the DBA. The **schema** is a container that serves as a namespace so that duplicate table names can be avoided. Originally, it was defined so that each user would have a separate space to create tables. Two users could each create a table named `Employee` without causing a problem. Today, schemas can be created for any purpose, not just for each user. The catalog was proposed in the SQL 99 standard, primarily to make it easier to find and access related schemas by placing them into one container. At this point, it is not likely that any DBMS supports the catalog element. However, the schema approach is relatively common. Users and applications are assigned to a default schema, and tables and views within that schema are directly accessible (depending on the security permissions, of course). But sometimes you need to access tables or views stored in a different schema. In these situations, you need to use the full name of the item. The full name includes the schema name (and eventually the catalog name). For example, if you want to access the `Employee` table in the `Corporate` schema, you would use `SELECT * FROM Corporate.Employee` to indicate the full name of the table. If you need to specify the catalog (e.g., `Main`), you would use `Main.Corporate.Employee` as the full name of the table. More commonly, the name of the database is used instead of a catalog name. The standard database elements such as tables, views, and triggers reside within each schema. One of the tasks of the DBA (and the DA) is to identify when to create new schemas. Although there are no specific rules, keep in mind that the purpose of a schema is to isolate and compartmentalize applications.

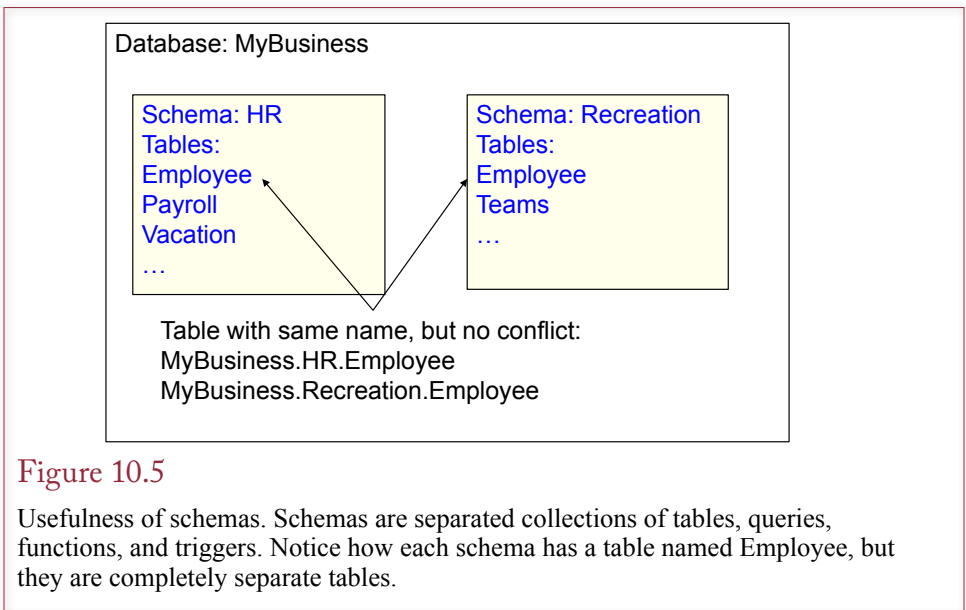


Figure 10.5 gives an example of using a schema to support two different applications within the same database. The database name is *MyBusiness* and the two schemas are *HR* and *Recreation*. Note that both schemas contain a table named *Employee*. These two tables are completely separate, can contain different columns and have different security permissions. This approach is particularly useful when an application is purchased from an outside vendor and it is not possible to rename the tables. Placing the application definition within its own schema keeps it separated from everything else. But all of the tables can be shared across schemas as necessary.

The other option would be to place all applications into separate databases. What is the difference? The two big differences are (1) backup/files and (2) security permissions. Each database uses different files for data storage and rollback. Plus, security objects are defined separately within each database. So every time a new database is created, procedures and people need to be assigned to handle backup, and assign and test security permissions. Sometimes the increased separation is worth the effort, other times it is easier and faster to simply create a new schema within an existing database. This is just one of the decisions that needs to be made by the DBA.

Metadata

How does a DBA find out what is stored in each database? Each vendor provides tools to help DBAs accomplish common tasks. Most have a graphically oriented approach to make them easier to use. On the other hand, DBAs often choose to perform tasks using SQL by building specific procedures. The SQL commands provide detailed control over an operation and can be written to handle dozens or hundreds of operations at one time. For example, the graphical approach is easy to use for adding one user, but if you need to add 100 users, it is easier to write an SQL procedure that pulls the list of users from a file or a temporary table.

In terms of administration, one of the powerful aspects of relational database systems is that even the administrative data is stored in tables. This **metadata**

Schemata	
Tables	
Domains	
Views	
Table_Privileges	SELECT Table_Name, Table_Type
Referential_Constraints	FROM Information_Schema.Tables
Check_Constraints	WHERE table_name LIKE 'Emp%'
Triggers	
Trigger_Table_Usage	
Parameters	
Routines	

Figure 10.6

Information Schema. A few of the 61 views in the standard are listed on the left. The sample query shows how DBAs can query the metadata views to quickly find a specific item.

is data about the data. For example, a system table contains a list of all the user tables. The SQL 99 standard describes the `Information_Schema` which consists of a set of views that provide documentation on the database. Technically, the `Information_Schema` views retrieve data from the `Definition_Schema` tables; however, DBMS vendors might choose not to implement the `Definition_Schema`. DBMS vendors have already developed proprietary system tables to hold the metadata. The drawback to this approach has been that there is no consistency across products, so DBAs have to learn different commands for each DBMS. As vendors implement the newer standards, DBAs should find it easier to work with products from multiple vendors. As of 2013, only some vendors have implemented the `Information_Schema` views. Currently, most DBAs use vendor-specific queries, such as the `Describe` command or the `USER_TABLES` view in Oracle; or the `MSysObjects` system table in Access; or the `sys.xxx` views in SQL Server; or the `syscat` views (e.g., `syscat.Tables`) in IBM's DB2. It would be relatively easy to create an `Information_Schema` in most of these systems and add your own view definitions for the standard metadata. Essentially, the standard views extract the information from the underlying metadata tables. Note that Microsoft's SQL Server does support the `Information_Schema` views (along with proprietary `sys.` views). Technically, Oracle does not support the `Information_Schema`, but search the Web and you will find downloadable files that can be installed in an Oracle DBMS to provide most of `Information_Schema` definitions.

Figure 10.6 shows some of the common elements of the `Information_Schema`. The SQL command illustrates how to obtain a partial list of the tables, based on the name. Commands of this type are useful when a database has hundreds of tables and views. Instead of scrolling through dozens of pages looking for a specific table, you can use the power of SQL to quickly find the exact table needed. In the example, note that you should always retrieve the `Table_Type` as well as the name. Tables can be base types (that actually hold the data), views, or derived tables. You can use these standard views to get lists of tables, views, triggers, and other procedures. You can also use them to get detailed information about tables and views, such as a list of columns, data types, and SQL statements for the views and functions.

SELECT MSysObjects.Name, MSysObjects.Type FROM MSysObjects WHERE MSysObjects.Name Like "EMP*";	Access
SELECT * FROM sys.tables WHERE name Like N'Emp%';	SQL Server
SELECT * FROM ALL_TABLES WHERE TABLE_NAME Like 'Emp%';	Oracle
SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE N'Emp%';	SQL Standard

Figure 10.7

Usefulness of schemas. Schemas are separated collections of tables, queries, functions, and triggers. Notice how each schema has a table named Employee, but they are completely separate tables.

Figure 10.7 shows examples of queries used to obtain metadata in four different systems. The structure of the queries is the same in each case, but the table/query names are different as are the columns and the data retrieved. In particular, the Type values are different in each case. Still, it is useful to have the power of SQL to find objects by partial name if necessary.

Database Tasks by Development Stages

What DBA tasks need to be performed as an application is developed? Which-ever development methodology you follow (e.g., traditional systems development life cycle, rapid development, or prototyping), certain database tasks are required at each step. Most tasks are performed by the application developers, but some involve the DA. Many require communication with the DBA, both to get advice and to provide information to help the DBA set up the databases.

Database Planning

During the feasibility and planning stages, you will have to make an estimate of the data storage requirements. These initial estimates will be rough, but they will help determine the size and capacity of the hardware needed to support the application. For example, if you are building a simple database to track materials that will be used by five people, the database might require less than 100 megabytes of storage and run on a desktop computer. If the initial size estimates start to exceed a few hundred megabytes of storage, a file server with high-speed disk drives might be more appropriate, and the system can probably run on the “free” copies of DBMS software. As the database estimates approach gigabytes or terabytes, you should investigate special database hardware and parallel-processing systems.

The initial investigation should also provide some idea of the number of forms and reports that will be needed, as well as their complexity. These numbers will be used to estimate the time and cost required to develop the system. An experienced DBA can provide estimates of space requirements from similar projects. Company records on other projects can provide estimates of the average time to develop forms and reports.

Teamwork

- Data standards.
- Data repository.
- Reusable objects.
- CASE tools.
- Networks and communication.

Subdividing projects

- Delivering in stages: versions.
- Normalization by user views.
- Assigning forms and reports.

Figure 10.8

Managing database design. Database design requires teamwork and standards to ensure that individual components can be integrated into a complete application. CASE tools and networks improve communication through a centralized repository of design data.

Database Design

The basic goal of the design stage is to identify the user needs and design the appropriate data tables. Data normalization is the primary database-related activity in this stage. The final table definitions will also provide better estimates of the storage requirements.

Teamwork coordination and project management are important administrative tasks at this stage. As highlighted in Figure 10.8, teamwork is supported with data standards as defined by the DA. Projects can be split into pieces and assigned to each team member. The ability to integrate the pieces into a complete application is provided through standards and communication. Communication is enhanced through a shared data repository, networked tools, e-mail, and **computer-aided software engineering (CASE)** tools. Leading CASE tools include Oracle Designer/2000, Rational Rose, IEF, and IBM's Visual Age. These tools provide a centralized repository for all project work, including diagrams, data definitions, and programming code. As team members work on their portion of the project, they can see the rest of the project. In an OO project, they can use the objects created by other teams.

From the perspective of data design or normalization, the project is often split by assigning forms and reports to individual team members. Each person is then responsible for identifying the business assumptions and defining the normalized tables needed for the assigned forms. Periodically, the individuals combine their work and create a centralized list of the tables that will be used in the database. This final list must follow the standards established by the DA.

Database Implementation

The primary database tasks required for implementation are listed in Figure 10.9. The major steps are development of the application and the user interface. Management and organizational tasks largely entail determining the overall look and feel of the application. Once the overall structure is determined, programming standards and testing procedures facilitate teamwork and ensure quality. Another important management task is to assign ownership of the various databases. Owners should be from business management. Data owners are responsible for identi-

Standards for application programming.
 User interface.
 Programming structure.
 Programming variables and objects.
 Test procedures.
Data access and ownership.
Loading databases.
Backup and recovery plans.
User and operator training.

Figure 10.9

Implementation management. The user interface must be carefully chosen. Programming standards and test procedures help ensure compatibility of the components and provide quality control. Business managers should be assigned ownership of the data, so they can make final determinations of security conditions and quality. Backup and recovery plans have to be created and tested. Training programs have to be created for operators and users.

fying primary security rules and for verifying the accuracy of the data. If the DBA has any questions about access rights or changes to the data, the DBA can obtain additional information and advice from the data owner.

Backup and recovery procedures have to be established and tested. If any component fails, the database logs should be able to fully restore the data. Backups are often handled in two forms: full backup at predefined checkpoints and incremental backups of changes that have occurred since the last full backup. Complete backups are easier to restore and provide safer recovery. However, they can be time-consuming and require large amounts of backup space. For small databases, full backups are not a problem. For large, continually changing transaction databases, it may only be possible to perform a full backup once a week or so.

Users and operators also have to be trained. No matter how carefully the user interface is designed, there should always be at least an introductory training session for users. Similarly, computer operators may have to be trained in the backup and recovery procedures.

Database Operation and Maintenance

Once the database is placed in operation, the DBA performs most of the management tasks. The primary tasks are to (1) monitor usage and security, (2) perform backups and recovery, and (3) support the user.

Monitoring performance and storage space is a critical factor in managing a database and planning for growth. All of the big DBMS vendors provide graphical tools to display a variety of performance measures. Figure 10.10 shows some of the basic measures generated in Oracle's Enterprise Manager. This example shows a database with almost no load. In a production environment, you would look for spikes in certain statistics—particularly if they occur at set times every day. You would also watch for trends over time. For instance, if you see usage rates increasing and performance declining over time, you can use the data to predict future problems and schedule upgrades.

Monitoring performance and storage space is a critical factor in managing a database and planning for growth. All of the big DBMS vendors provide graphical tools to display a variety of performance measures. Figure 10.10 shows some

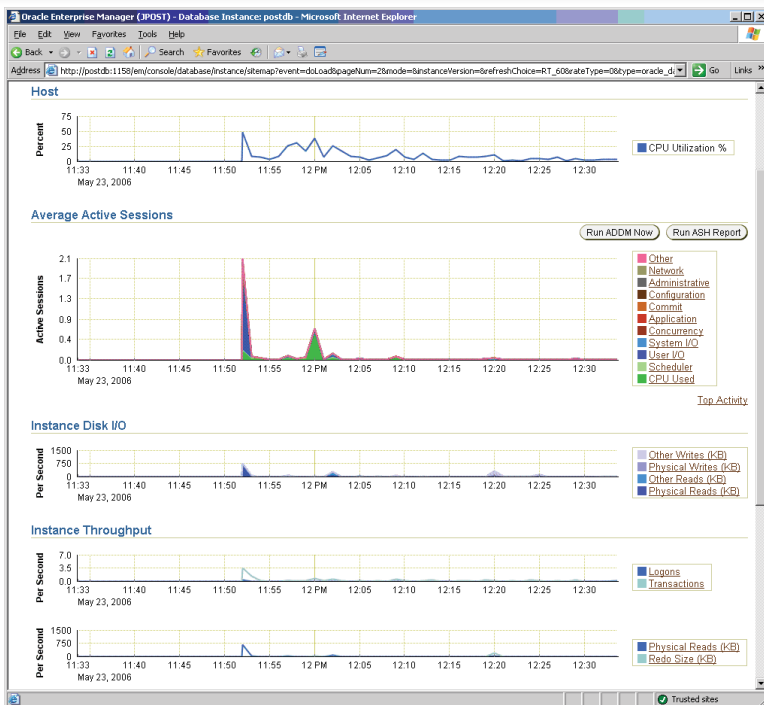


Figure 10.10

Oracle's basic performance monitoring statistics. The load on this server is minimal, but the DBA can watch these charts on a regular basis to spot problems and identify trends.

of the basic measures generated in Oracle's Enterprise Manager. This example shows a database with almost no load. In a production environment, you would look for spikes in certain statistics—particularly if they occur at set times every day. You would also watch for trends over time. For instance, if you see usage rates increasing and performance declining over time, you can use the data to predict future problems and schedule upgrades.

Monitoring is also used to fine-tune the application performance and to estimate growth and plan for future needs. Security access and changes are also monitored. Security logs can track changes to critical data. They can also be specified to track usage (both read and write) by individual users if there is a suspected problem. Monitoring user problems as well as performance provides useful feedback on the application. If users consistently have problems in certain areas, the design team should be encouraged to improve those forms. Similarly, if some users are running queries that take a long time to execute, the design team should be called in to create efficient versions of the queries. For example, do not expect a user to recognize or correct a correlated subquery. Instead, if the DBA sees users running complex queries that take too long to run, the team should add a new section to the application that stores and executes a more efficient query.

Similarly, if some people are heavily using certain sections of the database, it might be more efficient to provide them with replicated copies of the main sections. If the users do not need up-to-the-minute data, a smaller database can be set

up on a server and updated nightly. The users end up with faster response times because they have a smaller database and less communication time. The rest of the database runs faster because there are fewer heavy users.

Database vendors provide some powerful tools to help analyze queries and database performance. With these tools, you can break apart the entire query process to see exactly which step is taking the most time. With this knowledge, developers can work on alternative solutions to avoid the bottlenecks. Other tools can monitor for deadlock and transaction problems, making it relatively easy to correct problems. **Tuning** a large database to improve performance is a complex issue and depends heavily on the capabilities and tools of the specific DBMS.

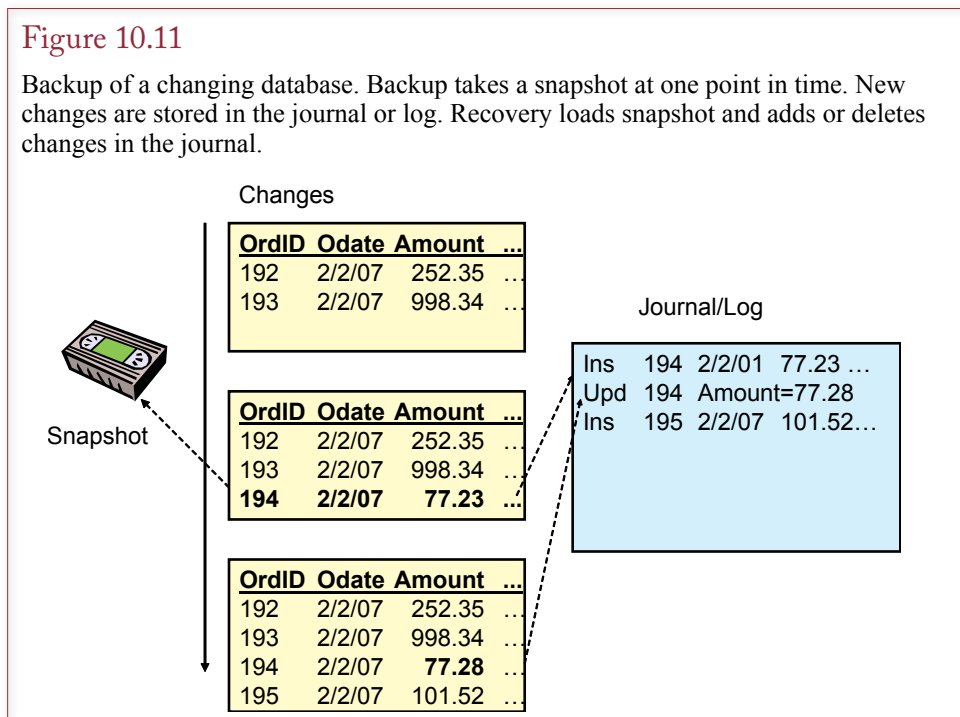
Backup and Recovery

How do you back up data that is constantly changing? Perhaps the most critical database management task is backup. No matter how well you plan, no matter how sophisticated your security system, something will go wrong. Database managers and developers have an obligation to plan for disasters. The most critical aspect of planning is to make sure that a current copy of the database is easily accessible. Any type of disaster—fire, flood, terrorist attack, power failure, computer virus, disk drive crash, or accidental deletion—requires backup data. Given the low cost of making and storing backup copies, there is no excuse for not having a current backup available at all times.

As shown in Figure 10.11, database backups provide some interesting challenges—particularly when the database must be available 24 hours a day, 7 days a week (abbreviated to **24-7**). The basic problem is that while the database is making a backup copy, changes could still be made to the data. That is, every copy of the database is immediately out of date—even while it is being made. A related

Figure 10.11

Backup of a changing database. Backup takes a snapshot at one point in time. New changes are stored in the journal or log. Recovery loads snapshot and adds or deletes changes in the journal.



issue is that the DBMS copy routines might have to wait to copy portions of the database that are currently in use (possibly creating a deadlock situation).

Fortunately, the larger database systems provide many tools to solve these problems. One approach is to take a **snapshot** of the tables. The snapshot represents the status of a table at one instant in time. Because of ongoing changes, this snapshot is likely to be inconsistent or record only portions of a transaction. To solve these problems, the **transaction log** (or journal) records all changes as they are written to the database. These log files must also be backed up. However, since the transaction system only adds new rows to the logs, it is relatively easy to back them up with no contention or deadlock issues. On restore, the system loads the snapshot data and then plays back the changes in the log files to make everything consistent, and record changes that were made after the instant the snapshot was taken.

If a problem arises with the main database files, the database has to be restored from the backup tapes. First the DBMS loads the most recent snapshot data. Then it examines the transactions. Completed transactions are rolled forward, and the changes are rewritten to the data tables. If the backup occurred in the middle of a transaction and the transaction was not completed, the DBMS will roll back or remove the initial changes and then restart the transaction. Remember that a transaction consists of a series of changes that must all succeed or fail together. The DBMS relies on the application's definition of a transaction as described in Chapter 7.

Backups have to be performed on a regular schedule. Occasionally, the schedule will have to be revised—particularly if the database records many changes. Remember that every change since the last backup is recorded in a journal or transaction log. The DBA has to watch the space on the transaction log. If it becomes too full, a backup has to be run earlier than scheduled. If these unexpected backups happen too often, the schedule should be changed. It is possible to make complete backups and incremental backups. An incremental backup saves only the data that was changed since the last backup, making it considerably faster than a full backup. However, the system has to work harder to piece everything together if you need to restore the entire database. With high-speed processors and storage, the additional time to restore the data might be minimal today, but you still face a greater risk of damage if one of the incremental backups has problems.

As a side note, be careful with your development databases on your own workstation. In particular, SQL Server default logging mode automatically extends the transaction log—potentially reaching several gigabytes in size. You have to run a full backup to clean up the transaction log. Alternatively, your development databases can be configured without transaction logging.

Backup tapes must be stored offsite. Otherwise, a fire or other disaster might destroy all data stored in the building. Snapshot and journal logs should be copied and moved offsite at least once a day. Networks make it easier to transfer data if the company is large enough to support computer facilities at more than one location. Several companies provide disaster-safe vaults for storage of data tapes and disks. In extreme situations, it might pay to have duplicate computer facilities and to program the system to automatically mirror changes from the main database onto the secondary computer in a different location. Then when something goes wrong, the secondary computer can immediately pick up the operations. However, even in this situation, you should make physical backup copies.

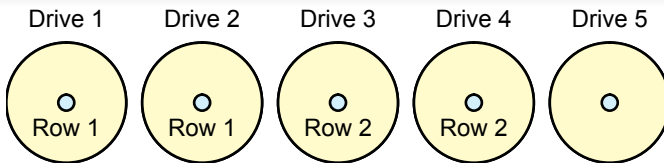


Figure 10.12

RAID drives. Pieces of data, such as a row, are written on two different drives. Other rows are spread across drives so that multiple disk reads and writes can occur at the same time to dramatically improve performance.

An increasingly popular approach to backups is to create mirrored copies of the database—using either software or hardware. With a high-speed network, you can configure the DBMS to write all changes to a second location (mirror). This server could pick up the load if the first server is slow or something crashes. You can also mirror the data in one place—by using a **redundant array of independent drives (RAID)**. With RAID (and other striping systems), each piece of data is written in two locations on different physical drives. If one drive fails, the DBA simply removes it and installs a replacement. The system automatically uses its internal copy of the data that is spread across the other drives. As indicated in Figure 10.12, the system is also considerably faster than using a single drive, since each portion of data can be simultaneously written to a different physical drive. Because disk drives are mechanical, they are the slowest and least reliable component of the computer system (not counting humans). Striping data across multiple drives means that data can be read or written in parallel at the same time—providing substantially faster throughput, as well as providing duplication to protect from the failure of a single drive.

Physical Configuration

How should computers be configured for DBMS software and database files?

Obviously, this question has multiple answers and each situation is different from others, so the final selection needs to be based on the size of the project and the amount of money available. However, a few key elements tend to be useful in many situations based on current computer and network capabilities.

It is important to recognize that computers and networks have improved dramatically in the past few years. These trends are likely to continue for at least a few more years. One of the biggest trends in computer architecture is the growing importance of parallel processing—most computer processors are being built with multiple cores, so servers can easily contain a handful to thousands of independent processors. A key consequence of having multiple processors is the creation of a **virtual machine (VM)**. With a VM, the computer running tasks is essentially hosted on a parent computer.

Figure 10.13 shows the logical structure of a VM. A single physical machine runs a base operating system that contains a VM hypervisor program that is used to configure and control VMs. A VM is loaded with its own operating system (with its own license). The VM operating system thinks it is running on an independent machine, but it actually shares processors and memory on the base computer. One of the main strengths of this approach is that more RAM and more processing power can be given to the VM when it is needed. Also, the VM definition,

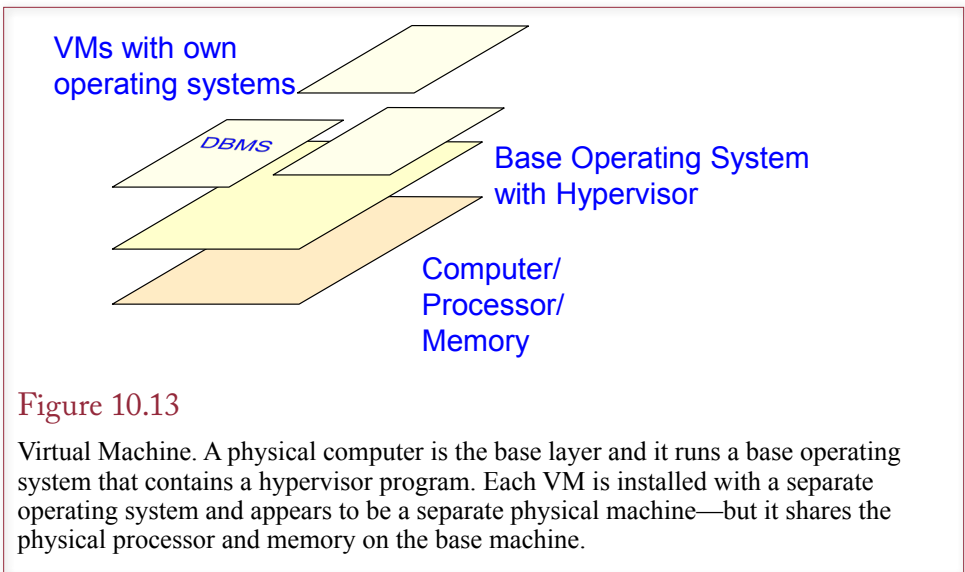


Figure 10.13

Virtual Machine. A physical computer is the base layer and it runs a base operating system that contains a hypervisor program. Each VM is installed with a separate operating system and appears to be a separate physical machine—but it shares the physical processor and memory on the base machine.

including the operating system and the software configuration is stored as a file on the base computer. It is straightforward to back up this VM definition and restore it on a different computer if the base machine crashes or is destroyed in a fire. Spend several hours configuring a new OS, installing, and configuring Oracle and you will quickly appreciate the value of a simple VM backup and restore facility.

Figure 10.14 shows the Microsoft Hyper-V tool to manage VMs in Server 2012. The base server consists of a single computer with an 8-core processor and 32 GB of actual RAM. The Hyper-V instance is running on this physical server. Four virtual machines are defined (two running versions of LINUX, two with Windows). Disk space is allocated on the main server to hold the operating systems in VHD files. This space is allocated to an individual VM which sees it as its own disk drive. Shared space can also be allocated using the virtual storage area network (SAN) manager. Using VMs makes it easier to set up systems for testing. It is also straightforward to clone a system, so if anything goes wrong with an upgrade or expansion, the original version can be restored quickly.

A second major trend is the increasing speed of local area networks. Most networks can easily handle 1 gigabits per second (gps) transfer speeds. This speed is already faster than most hard-drive transfer speeds; and network speeds are increasing every few years. Consequently, as shown in Figure 10.15, hard drives no longer need to be located in the same box as the server. Several vendors now sell **network attached storage (NAS)** or **storage area network (SAN)** devices. The devices usually have RAID configurations—both for speed and for internal backup. Several independent vendors sell these devices, and even provide separate, automatic backup systems. The DBMS sees the storage device as a giant disk drive, but the device automatically writes striped copies of the data, and writes a cached copy that is backed up to tape at regular intervals. The system provides immediate online copies of all data, as well as long term copies. Essentially, the systems transfer the responsibility for backup from software to hardware, and they store and retrieve data very rapidly. Separating the server from the data improves reliability and makes it easier to restore operations if either the server crashes or the SAN fails.

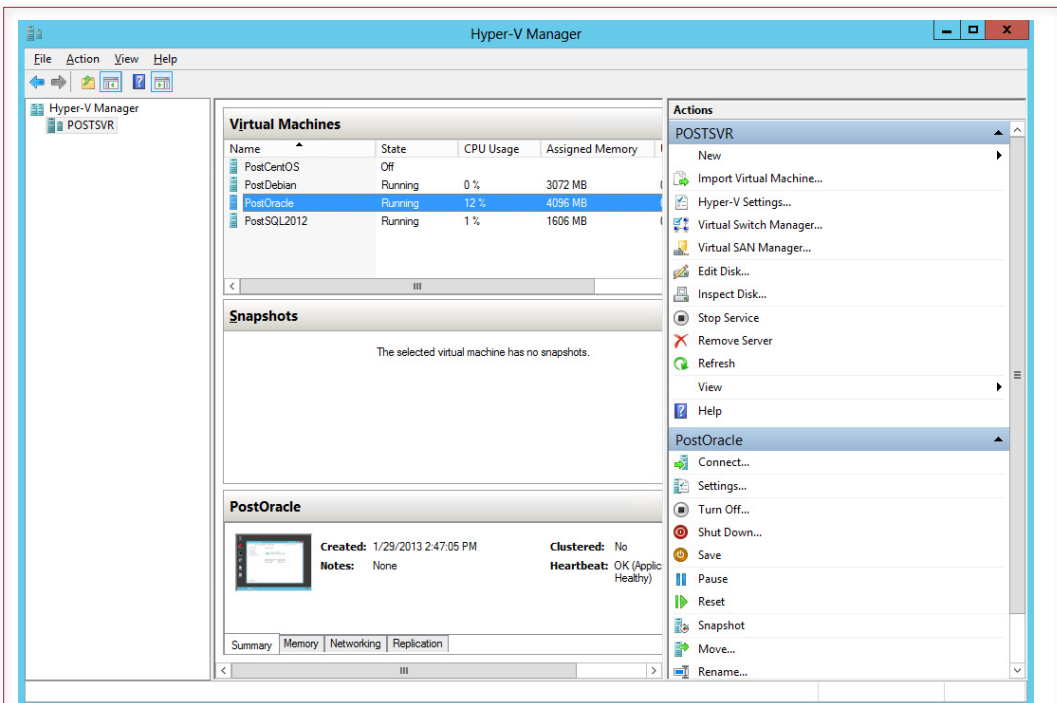


Figure 10.14

Hyper-V Management. Windows Server 2012 includes the Hyper-V tool to manage virtual machines. It is used to assign physical attributes such as memory, processors, and network access.

When using a VM, it is always a good idea to store the actual database on an external drive. Except for small systems for development and testing, the data should not be stored on a virtual drive associated with a VM. The intervening layers of software on a VM virtual drive will slow down all data operations. It is faster to write directly to the drive hardware, and SAN devices can be optimized for transaction performance.

Security and Privacy

What security techniques are used to protect databases? Computer security is an issue with every company today, and any computer application faces security problems. A database collects a large amount of data in one location and makes it easy for people to retrieve and change data. In other words, a database is a critical resource that must be protected, and it is a tempting target for attackers. Yet the same factors that make a database so useful also make it more difficult to secure. In particular, the purpose of a database is to share data. In a security context, you want to control who can share the data and what those users can do with it.

Computer security is often split into three categories: (1) physical security, (2) logical security, and (3) behavioral security. **Physical security** is concerned with physically protecting the computing resources and preparing for physical disasters that might damage equipment or data. **Logical security** consists of protecting the data and controlling access to the data. **Behavioral security** is trickier because it emphasizes the role of people or employees. It is related to logical security but involves interesting problems because it deals with mistakes that people make.

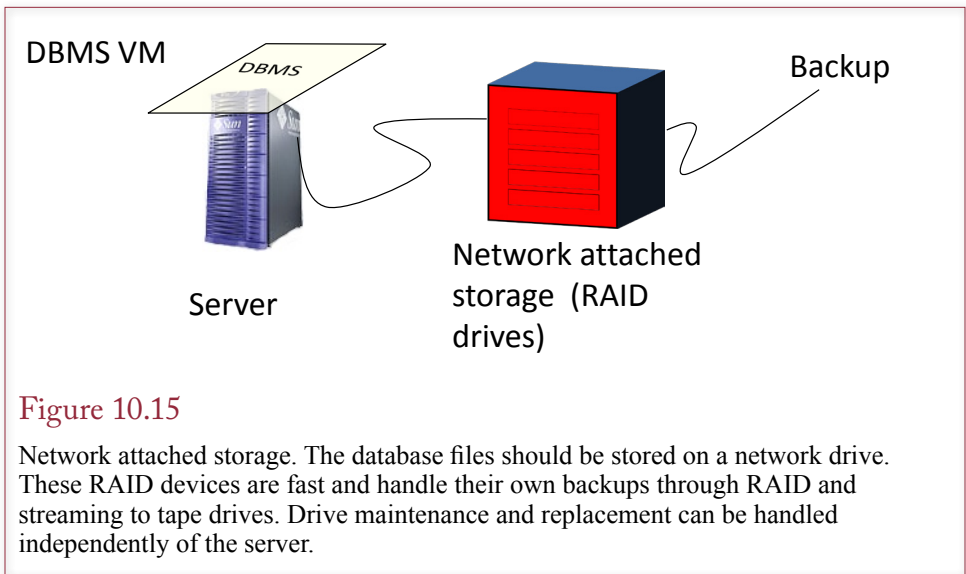


Figure 10.15

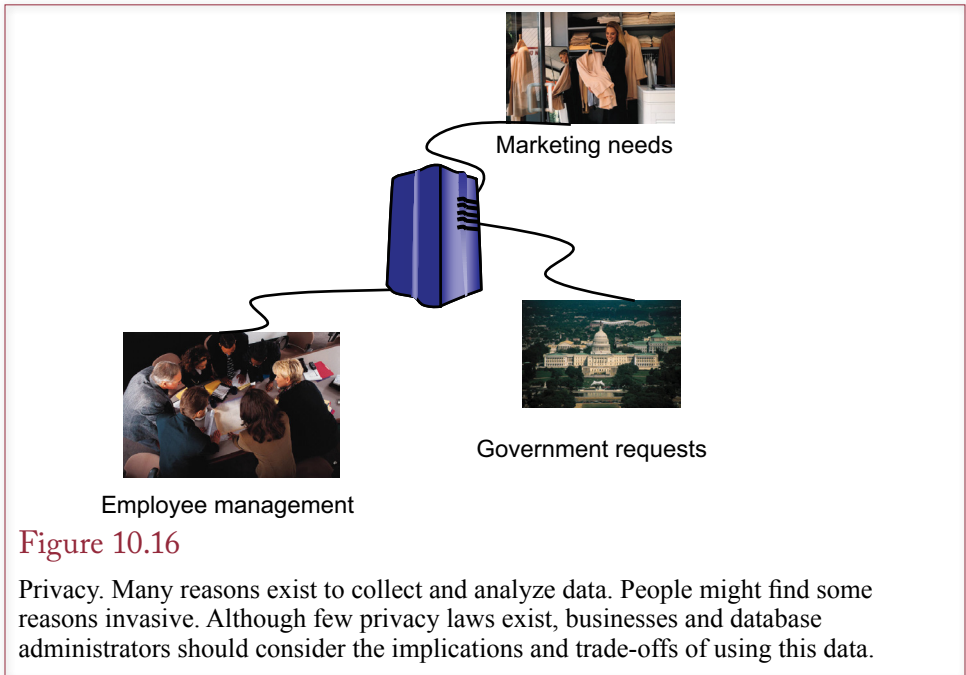
Network attached storage. The database files should be stored on a network drive. These RAID devices are fast and handle their own backups through RAID and streaming to tape drives. Drive maintenance and replacement can be handled independently of the server.

Data Privacy

Privacy is related to security but with a slight twist. Companies and governmental agencies collect huge amounts of data on customers, suppliers, and employees. Privacy means controlling the distribution of this data and respecting the wishes of these external people. Figure 10.16 shows some of the demands placed on business data in terms of marketing, employee management, and governmental requests. The concepts of keeping data accurate and limiting who has access to it are the same for security and for privacy. The differences lie in the objectives and motivation. In terms of security, every company has a self-interest in keeping its data safe and protected. In terms of privacy—at least in the United States—there are few regulations or limitations on what a company can do with personal data. Whereas customers and employees may want a company to keep personal data private, companies may have a financial incentive to trade or sell the data to other companies.

In terms of data privacy, the most important question is, Who owns the data? In most cases the answer is the company or individual that collects the data. Some people, particularly in Europe, have suggested that the individuals should be considered to be the owners. Then companies would have to get permission—or pay for permission—to use or trade personal data. So far, technical limitations have prevented most of the suggested payment schemes from being implemented. However, companies must pay attention to changes in the laws regarding privacy.

Database workers have an ethical obligation in terms of data privacy. Many times, you will have access to personal data regarding customers and other employees. You have an obligation to maintain the privacy of that data: You cannot reveal the data to other people. In fact, you should avoid even reading the data. You should also not tolerate abuses by other workers within the organization. If you detect privacy (or security) violations by others, you should report the problems and issues to the appropriate supervisors.



Threats

What are the primary threats to computer security? What possible events cause nightmares for database administrators? Is it the outside hackers or crackers that you see in the movies? Is it tornadoes, hurricanes, or earthquakes (also popular in movies)?

No. The primary threat to any company comes from “insiders.” Companies can plan for all of the other threats, and various tools exist to help minimize problems. However, you have to trust your employees, consultants, and business partners. For them to do their jobs, they need physical access to computers and logical access to the databases. Once you are committed to granting access, it becomes more difficult to control what they do. Not impossible, just more difficult. Even when employees are honest and cautious, they are still going to make mistakes because they are human. Some behavioral issues are easier to deal with than others. Writing down passwords or giving them out over the phone are risky behaviors. Picking up a USB drive in the parking lot and inserting it into a computer is flat out dangerous—but hard for people to resist. Yes, the volume of attacks from the Internet is high, but in many ways they are easier to stop.

Another, more insidious threat comes from programmers who intentionally damage data. One technique is to embed a time bomb in a program. A time bomb requires the programmer to enter a secret code every day. If the programmer leaves (or is fired) and cannot enter the code, the program begins deleting files. In other cases programmers have created programs that deliberately alter data or transfer funds to their own accounts. These examples illustrate the heart of the problem. Companies must trust their programmers, but this trust carries a potential for considerable damage or fraud. It is one of the reasons that companies are so sensitive about MIS employee misconduct. As a developer, you must always project an image of trust.

- Backup data
- Backup hardware
- Disaster planning and testing
- Prevention
 - Location
 - Fire monitoring and control
 - Control physical access

Figure 10.17

Physical security controls. Backup data is the most important step. Having a place to move to is a second step. Disaster plans and prevention help prevent problems and make recovery faster.

Physical Security

In terms of physically protecting the computer system, the most important task is to make sure you always have current backups. This policy of maintaining backups also applies to hardware. In case of a fire or other physical disaster, you need to collect the data tapes and then find a computer to load and run them. Instead of waiting until a disaster happens, you really need to create a disaster plan.

A **disaster plan** is a complete list of the steps that the IS department will take if a disaster hits the information system. The plan details who is in charge, describes what steps everyone will take, lists contact numbers, and tells how you will get the systems up and running. One popular method of finding an alternative computer is to lease a hot site from a disaster planning company. A **hot site** consists of a computer facility that has power, terminals, communication systems, and a computer. You pay a monthly fee for the right to use the facility if a disaster occurs. If there is a disaster, you activate the disaster plan, collect the data tapes, load the system, load your backup tapes, and run the system from the hot site. A slightly cheaper alternative is to lease a cold site. A **cold site** or **shell site** is similar to a hot site, but it does not have the computer and telecommunications equipment. If a disaster occurs, you call your hardware vendor and beg for a new computer. Actually, vendors have been very cooperative. The catch is that it can still take several days to receive and install a new computer. Can your company survive for several days without a computer system? To replace smaller computers, some of the disaster recovery companies can deliver a truck to your site and run the system from your parking lot.

As summarized in Figure 10.17, prevention is another important step in providing physical security. Computer facilities should have fire detection and protection systems. Similarly, computer facilities should be located away from flood plains, earthquake faults, tidal areas, and other locations subject to known disasters. Physical access to computers, network equipment, and personal computers should be limited. Most companies have instituted company badges with electronic locks. Access by visitors, delivery people, and temporary employees should be controlled.

Today, with the need for continuous online systems, the role of backup facilities has changed. Many companies now have multiple data centers in separate locations. Each center can run all of the operations but normally share the load and the data. If one system goes down, the other can immediately pick up the transactions and operations. Cloud-based operations can also be used to increase the scale of a system if something goes wrong in one location.

Managerial Controls

Because the major threats to data security come from company insiders, traditional managerial controls play an important role in enhancing security. For example, one of the most important controls begins with the hiring process. Some firms perform background checks to verify the character and trustworthiness of the employees. Even simple verification of references will help to minimize problems. Similarly, firms have become more cautious when terminating employees—particularly MIS employees with wide access to databases. Even for routine layoffs, access rights and passwords are revoked immediately.

Sensitive jobs are segmented. For example, several employees are required to complete financial transactions. Transactions involving larger amounts of money are routed to higher level employees. Similarly, outside institutions like banks often call back to designated supervisors to verify large transactions. Transactions are often monitored and recorded in terms of the time, location, and person performing the operations.

In some cases, security can be enhanced through physical control over the hardware. Centralized computers are placed in locked and guarded rooms. Employees are often tracked through video monitors. Security badges are also used to track employee access to locations and computer hardware.

Consultants and business alliances also raise security concerns. Generally, you have less control over the selection of the consultant and any partnership employees. Although you have control in the selection of a consulting firm, you have little control over the specific employees assigned to your location. These risks can be controlled by limiting their access to the data and restricting their access to physical locations. In some situations, you may also want to pair an internal employee with each consultant.

Logical Security

The essence of logical security is that you want to allow each user to have some access to the data, but you want to control exactly what type of access the user will have. You also want to monitor access to the data to identify potential problems. Figure 10.18 notes the three basic problems that you want to avoid: confidentiality (unauthorized disclosure), integrity (unauthorized modification), and accessibility (unauthorized withholding of information or denial of service).

Confidentiality refers to information that needs to be protected so that only a select group of users can retrieve it. For example, the company's strategic marketing plans need to be protected so that no competitor can retrieve the data. To be safe, only a few top people in the company would have access to the plans.

Integrity applies to information that is safe to display to users, but you do not want the users to change it. For example, an employee should be able to check the human resource files to verify his or her salary, remaining vacation days, or merit

Figure 10.18

Logical security problems. Each situation can cause problems for the company, including financial loss, lost time, lost sales, or destruction of the company.

C onfidentiality	Unauthorized disclosure
I ntegrity	Unauthorized modification
A ccessibility	Unauthorized withholding

Do not use words in a dictionary.
Do not use personal (or pet) names.
Include nonalphabetic characters.
Use at least eight characters.
Change it often.

Figure 10.19

Password suggestions. Pick passwords that are not in a dictionary and are hard to guess. The catch is, Can you remember a convoluted password?

evaluations. But it would be a mistake to allow the employee to change any of this data. No matter how honest your employees are, it would be a dangerous temptation to allow them to alter their salary, for example.

The third problem is subtle but just as dangerous. Consider what would happen if the chief financial officer needs to retrieve data to finalize a bank loan. However, the security system is set incorrectly and refuses to provide the data needed. If the data is not delivered to the bank by the end of the day, the company will default on several payments, receive negative publicity, lose stock value, and potentially risk going under. The point is that withholding data from authorized users can be just as dangerous as allowing access to the wrong people.

Assuming you have a sophisticated computer system and a DBMS that supports security controls, two steps are needed to prevent these problems. First, the computer system must be able to identify each user. Second, the owner of the data must assign the proper access rights to every piece of data. The DBA (or a security officer) is responsible for assigning and managing user accounts to uniquely identify users. The application designer and data owners are jointly responsible for identifying the necessary security controls and access rights for each user.

User Identification

One of the major difficulties of logical computer security is identifying the user. Humans recognize other people with sophisticated pattern-recognition techniques applied to appearance, voice, handwriting, and so on. Yet even people can be fooled. Computers are weak at pattern recognition, so other techniques are required.

The most common method of identifying users is by accounts and passwords. Each person has a unique account name and chooses a password. In theory, the password is known only to the individual user and the computer system. When the user enters the correct name and matching password, the computer accepts the identity of the person.

The problem is that computers are better than people at remembering passwords. Consequently, people make poor choices for passwords. Some of the basic rules for creating passwords are outlined in Figure 10.19. The best passwords are long, contain nonalphabetic characters, have no relationship to the user, and are changed often. Fine, but a user today can easily have 30, 40, or more different accounts and passwords. Almost no one can remember every account and the convoluted passwords required for security. So there is a natural inclination either to write the passwords in a convenient location (where they can be found by others) or choose simple passwords (which can be guessed).

Passwords are the easiest (and least expensive) system to implement at this time. Within a given company, it is possible to implement a central security server (e.g., Kerberos used in Microsoft's Active Directory), where a user logs into the main server and all other software verifies users with that server. Another approach is to use password generator cards. Each user carries a small card that generates a new password every minute. At login, the computer generates a password that is synchronized to the card. Once the password is used, it is invalidated, so if an interloper observes a password, it has no value. The system still requires users to memorize a short password just in case a thief steals the password card. Of course, if a user loses the card, that user cannot get access to the computers. Encrypted software variations can be loaded onto laptop computers, which then provide access to the corporate network.

Other alternatives are being developed to get away from the need to memorize passwords. Biometric systems that measure physical characteristics already exist and are becoming less expensive. For example, fingerprint, handprint, iris pattern, voice recognition, and thermal imaging systems now work relatively well. The advantage to biometric approaches is that the user does not have to memorize anything or carry around devices that could be lost or stolen. The main drawback is cost, since the validating equipment has to be installed anywhere that employees might need access to a computer. A secondary problem is that although the devices are good at preventing unauthorized access, many of them still have relatively high failure rates and refuse access to authorized users.

After individual users are identified, most systems enable you to assign the individuals to groups. Groups make it easier to assign permissions to users. For example, by putting 100 employees into a clerical group, you can grant permissions to the group, which is much faster than assigning the same permissions 100 times.

Most DBMSs identify the user either internally (database login) or through integration with the operating system (particularly Windows Active Directory). Similarly groups or roles can be created within the DBMS or within Active Directory. The choice depends on the company and the type of application being created. Centralizing logins through the operating system (Windows) makes it easier for users to access data and makes it easier for security personnel to monitor and change groups. But it requires close cooperation between the DBA, developers, and the network security group.

Access Controls

After users are identified, they can be assigned specific permissions to any resource. From a database perspective, two levels of access must be set. First, the user must be granted access to the overall database or standalone application, using operating system commands. Second, the user or group must be granted individual permissions, using the database security commands.

The privileges listed in Figure 10.20 apply to an entire table or query. The most common privileges you will grant are read, update, and insert. Delete permission means that a user can delete an entire row: —it should be granted to only a few people in specific circumstances. Privileges that are more powerful can be granted to enable users to read and modify the design of the tables, forms, and reports. These privileges should be reserved for trusted users or DBAs. Users will rarely need to modify the design of the underlying tables. These privileges are granted to developers.

Read data
Update data
Insert data
Delete data
Open/run
Read design
Modify design
Administer

Figure 10.20

DBMS privileges. These privileges apply to the entire table or query. The first three (read, update, and insert) are commonly used. The design privileges are usually granted only to developers.

With most database systems the basic security permissions can be set with two SQL commands: **GRANT** and **REVOKE**. Figure 10.21 shows the standard syntax of the GRANT command. The REVOKE command is similar. SQL 92 provides some additional control over security by allowing you to specify columns in the GRANT and REVOKE commands, so you can grant access to just one or two columns within a table. However, the privilege applies to every row in the table or query.

Most of the database systems provide a visual security tool to help assign access rights. The SQL commands are useful for batch operations when you need to assign permissions to a large number of tables or for several users at the same time. The visual tools are easier to use for single operations. They also make it easy to see exactly which permissions have currently been assigned to individuals. However, the SQL statements can be used in scripts to automate security assignments.

The GRANT command offers an additional option that is sometimes useful. As the owner of a database element, you have the ability to pass on some of your powers when you grant access. If you add the phrase **WITH GRANT OPTION**, then whoever just received the privileges you specified can also pass those on to someone else. For example, you could grant SELECT (read) privilege on a table WITH GRANT OPTION to the head of the marketing department. That person could then give read access to the other employees in the marketing department.

Figure 10.21

SQL security commands. Most systems also provide a visual tool to assign and revoke access rights.

GRANT privileges
ON objects
TO users

REVOKE privileges
ON objects
FROM users

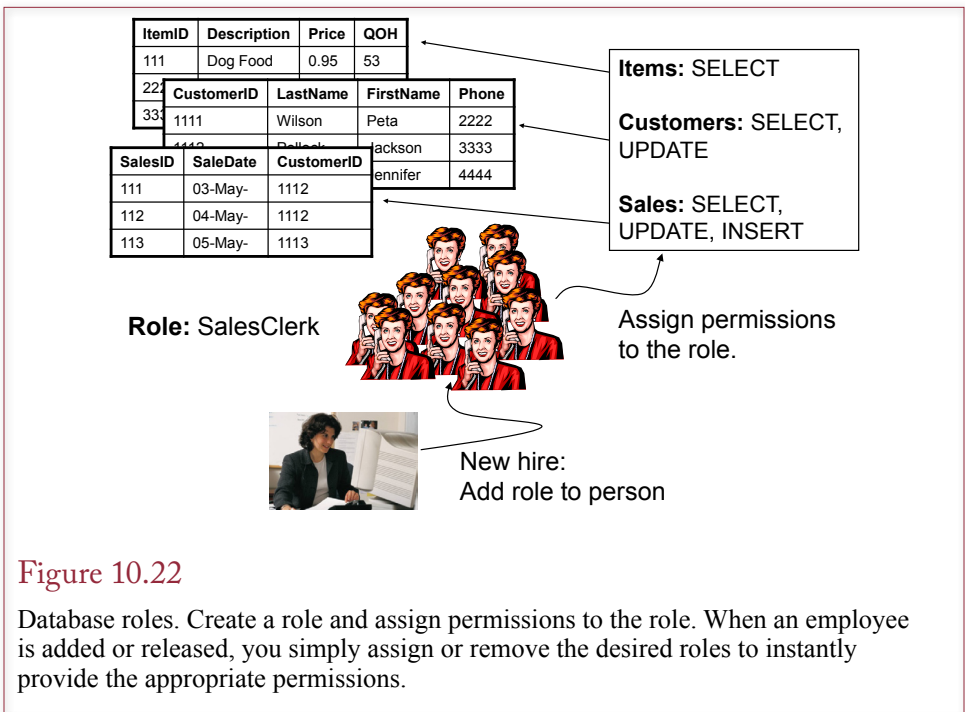


Figure 10.22

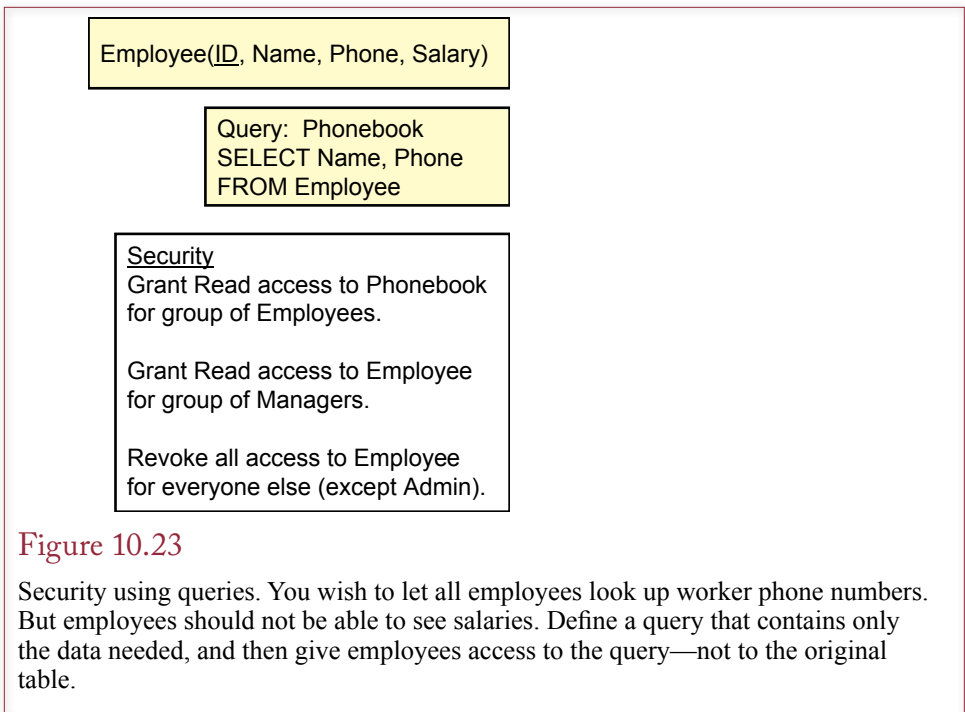
Database roles. Create a role and assign permissions to the role. When an employee is added or released, you simply assign or remove the desired roles to instantly provide the appropriate permissions.

Database Roles

Imagine the administrative hassles you would face if you have to assign security permissions for hundreds or thousands of employees. Each time an employee is hired or leaves, someone would have to assign or remove rights to possibly hundreds of tables and views. The task would be time consuming and highly error prone. Even if you build SQL procedures to help, it would require almost constant maintenance. A far more effective solution is to define database roles. A **role** is often associated with a group of users and consists of a set of permissions that are assigned together. As shown in Figure 10.22, you create a role and assign the desired permissions to the role instead of to the individual users. Then you assign the role to the specific users. When a new employee is hired, adding the role to that user provides all of the necessary permissions. The SQL 99 standard supports the ability to assign roles to other roles. For instance, you could define smaller sets of tasks as roles (sell item, add customer, update inventory) and then assign those tasks to broader roles (sales clerk, inventory clerk, and so on).

Queries as Controls

With many systems, the basic security commands are powerful but somewhat limited in their usefulness. Many systems only grant and revoke privileges to an entire table or query. Although the SQL 2003 standard supports specifying columns in the GRANT and REVOKE commands, this option might not be available in all situations. Also, queries provide detailed control through the WHERE clause. The true power of a database security system lies in the ability to assign access to individual queries. Consider the example in Figure 10.23. You have an Employee table that lists each worker's name, phone number, and salary. You want to use the table as a phone book so employees can look up phone numbers for other work-



ers. The problem is that you do not want employees to see the salary values. The solution is to create a query that contains just the name and phone number. Remember that a query does not duplicate the data—it simply retrieves the data from the other tables. Now, assign the SELECT privilege on the Phonebook query to all employees and revoke all employee privileges to the original Employee table. If you want, you can also choose specific rows from the Employee table. For example, you might not want to display the phone numbers of the senior executives.

Chapters 4 and 5 showed you the power of queries. This power can be used to create virtually any level of security that you need. Almost all user access to the database will be through queries. Avoid granting any access directly to a table. Then it will be easier to alter the security conditions as the business needs change.

The basic process is to confer with the users and to determine exactly which type of access each user needs to the data. In particular, determine which users need to add or change data. Then create the users and assign the appropriate security conditions to the queries. Be certain to test the application for each user group. If security is a critical issue, you should consider assigning a couple of programmers to “attack” the database from the perspective of different users to see whether they can delete or change important files.

As a reminder from Chapter 8, a key issue in modern databases—particularly Web-based applications—is the problem of SQL injection. Injection attacks arise when queries are created that use data entered by users—particularly uncontrolled users on the Web. The worst cases arise when SQL queries are built as strings by appending data entered on Web forms. A process should be established for reviewing all SQL statements created in applications that use Web forms. Ensure that they use parameters and remove quote and comment characters from parameter values whenever possible.

Division of Duties

For years, security experts have worried about theft and fraud by people who work for the company. Consider a classic situation that seems to arise every year. A “bad” purchasing manager sets up a fake supplier. The manager orders supplies from the fake company, and pretends that shipments arrive and authorizes payments. Of course, the manager cashes the payment checks. Some of these frauds run for several years before the perpetrators are caught. Ultimately, it is a weak scam because eventually the manager will be caught. Nonetheless, you want to stop the problem and catch the thief as quickly as possible.

The standard method to avoid this type of problem is to divide the duties of all the workers. The goal is to ensure that at least two people are involved in any major financial transaction. For example, a purchasing manager would find new suppliers and perhaps issue purchase requisitions. A different person would be in charge of receiving supplies, and a third person would authorize payments. The goal of separating duties can be challenging to implement. Companies try to reduce costs by using fewer employees. Business picks up, and someone takes advantage of the confusion. It is impossible to eliminate all fraud. However, a well-designed database application can provide some useful controls.

Consider the purchasing example shown in Figure 10.24 in which the basic tables include a Supplier table, a SupplyItem table, a PurchaseOrder table and a PurchaseItem table. In addition, financial tables authorize and record payments. The key to separation of duties is to assign permissions correctly to each table. The purchasing manager is the only user authorized to add new rows to the Supplier table. Purchasing clerks are the only users authorized to add rows to the PurchaseOrder and PurchaseItem tables. Receiving clerks are the only users authorized to record the receipt of supplies. Now if a purchasing clerk tries to create fake orders, he or she will not be able to create a new supplier. Because referential

Figure 10.24

Division of duties. The clerk cannot create a fake supplier. Referential integrity forces the clerk to enter a SupplierID from the Supplier table, and the clerk cannot add a new row to the Supplier table.

Supplier	
<u>SupplierID</u>	Name ...
673	Acme Supply
772	Basic Tools
983	Common X

Purchasing manager can add new suppliers, but cannot add new orders.



Referential integrity

Resource	Purchasing Manager	Purchasing Clerk
Supplier table	Select, Insert, Modify, Delete	Select
PurchaseOrder table PurchaseItem table	Select	Select, Insert, Modify, Delete

PurchaseOrder	
<u>OrderID</u>	<u>SupplierID</u>
8882	772
8893	673
8895	009

Clerk must use SupplierID from the Supplier table, and cannot add a new supplier.



integrity is enforced between the Order table and the Supplier table, the clerk cannot even enter a false supplier on the order form. Likewise, payments will be sent only to legitimate companies, and a purchasing manager will not be able to fake a receipt of a shipment. The power of the database security system is that it will always enforce the assigned responsibilities.

Software Updates

Modern software is large and complex. Software vendors and other researchers often find security problems after the software has been released. Typically, the company has a process to evaluate security threats, create patches to fix the problems, and notify users to update their systems. The difficulty is that announcing the security flaw to users also means that potential hackers are alerted to the problem. As long as all users patch their systems immediately, these announcements would be effective. But as a database administrator, the issue is not that simple. Even if you receive the notice, you still need to test the patch and ensure that it does not cause problems with your applications. Several major attacks have been launched against companies by attackers using known security holes that were not patched.

Consequently, a major task of database administrators is to monitor security releases for the DBMS and operating system software. These patches need to be installed on test machines and moved to the production systems as soon as possible. In the meantime, the network administrators can block specific ports and prevent outsiders from accessing features on the database. The DBAs and the network administrators also have to carefully monitor the network and the servers to see if rogue processes have been started. Most vendors have an automated system so you can discover which patches need to be installed; and even install them automatically. Of course, you must still monitor the process, and choose an update time that does not interfere with operations. You should also perform backups before updating your system.

Encryption

How do you prevent eavesdroppers or hackers from reading data? **Encryption** is a method of modifying the original information according to some code so that it can be read only if the user knows the decryption key. Encryption should be used when transmitting information from one computer to another—particularly when using the Internet. Sensitive information, particularly passwords and credit card numbers, stored within a database also can be encrypted. Without the encryption key, the files are gibberish. Encryption is critical for personal computer-based systems that do not provide user identification and access controls. Encryption is a useful technique, but you need to be aware of issues involving the keys.

Two basic types of encryption are commonly used today. Most methods use a single key to both encrypt and decrypt a message. For example, the **advanced encryption system (AES)** method uses a single key. Although AES is a U.S. standard, versions of it are available throughout the world because it is based on Rijndael created by two Belgian cryptographers. The AES algorithm is fast and supports key lengths of 128, 192, and 256 bits—protecting it from a **brute force attack** that tries all possible key values. Note that adding one bit to the key length provides twice as many keys; for example, moving from 56 to 128 bits adds 2 to the 72nd power more possibilities to test, making it virtually impossible to attack with brute force—using today's computers. Figure 10.25 shows a basic use

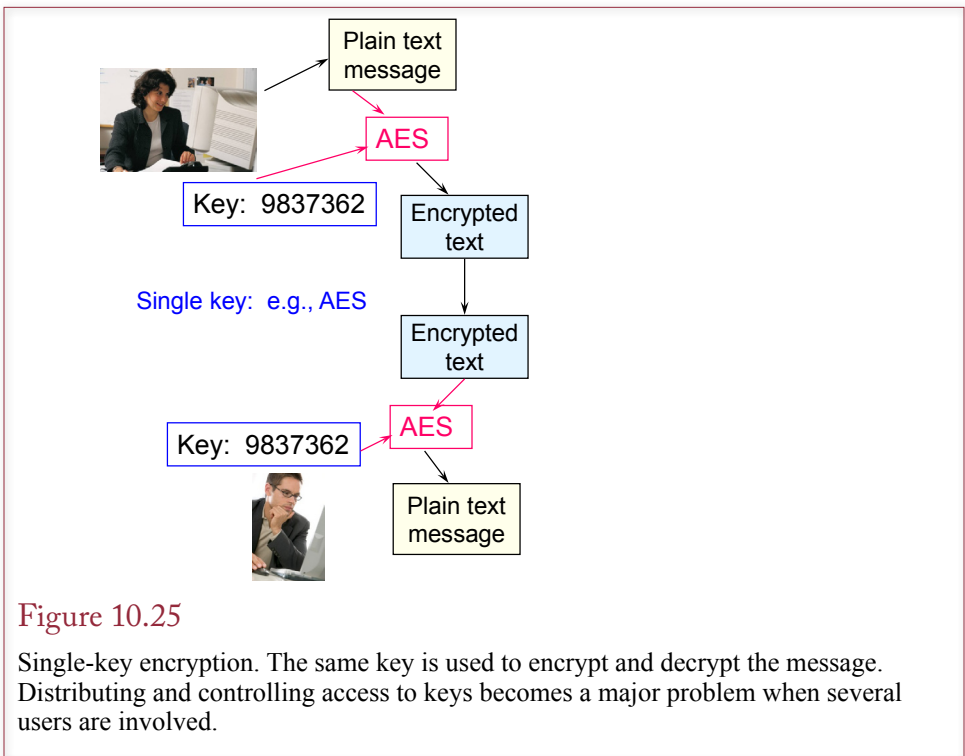


Figure 10.25

Single-key encryption. The same key is used to encrypt and decrypt the message. Distributing and controlling access to keys becomes a major problem when several users are involved.

of the AES encryption method. The primary drawback to AES is that it requires both parties to have the same value of the encryption key. Safely distributing the correct encryption key to each participant is a major problem in protecting data encrypted with a single key.

To solve the key-distribution problem, a second method was created that uses both a **private key** and a **public key**. Whichever key is used to encrypt the message, the other key must be used to decrypt it. The **Rivest-Shamir-Adelman (RSA) algorithm** is an example of a method that uses two keys. RSA protection is available on a variety of computers. RSA encryption works because of the properties of prime numbers. In particular, it is relatively easy for computers to multiply two prime numbers together. Yet it is exceedingly difficult to factor the resulting large number back into its two component parts. The security of the RSA approach relies on using huge numbers (128 digits or more), which would take many years to factor with current technology.

Methods that use two keys have some interesting uses. The trick is that everyone knows your public key, but only you know the private key. Consider the situation in Figure 10.26, where Bob wants to send a database transaction to Alice across the Internet. Bob looks up Alice's public key in a directory. Once the message is encrypted with Alice's public key, only her private key can decrypt it: No one else can read or change the transaction message. The dual-key approach solves the key-distribution problem because anyone can have access to the public key. The tradeoff is that dual-key encryption and decryption is considerably slower than single-key encryption. Consequently, many systems use dual-key encryption to establish a connection and distribute a one-time AES key to use for the session. Also, keep in mind that encryption only prevents people from reading or changing data, someone could still destroy the message before Alice receives it.

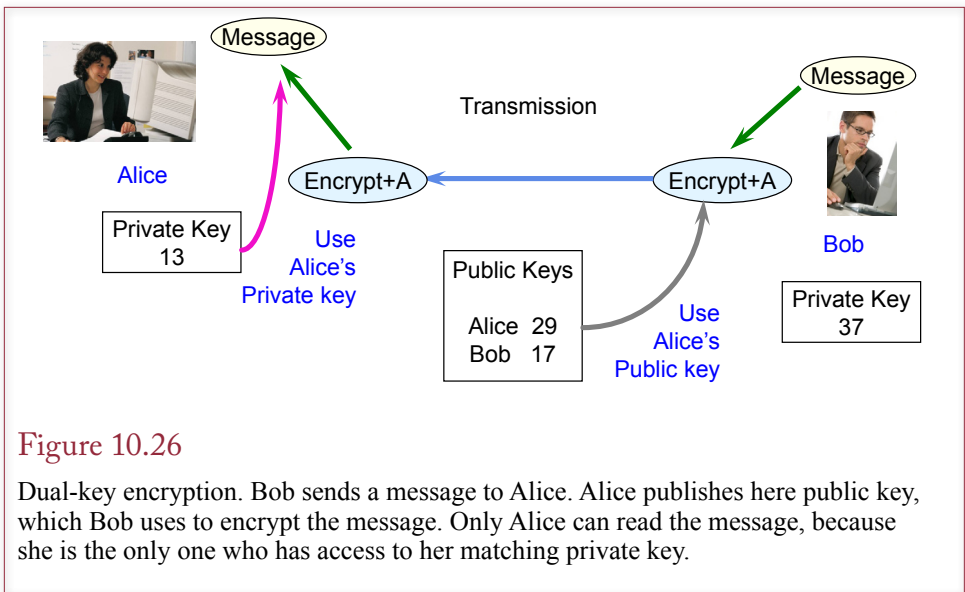


Figure 10.26

Dual-key encryption. Bob sends a message to Alice. Alice publishes here public key, which Bob uses to encrypt the message. Only Alice can read the message, because she is the only one who has access to her matching private key.

There is a second use of dual-key systems called **authentication**. Let's say that Bob wants to send a message to Alice. To make sure that only she can read it, he encrypts it with her public key. However, Bob is worried that someone has been sending false messages to Alice using his name, and he wants to make sure that Alice knows the message came from him. If Bob also encrypts the message with his private key, it can be decrypted only with Bob's public key. When Alice receives the message, she applies her private key and Bob's public key. If the message is readable, then it must have been sent by Bob.

Dual-key encryption systems are useful in all aspects of information communication. They do have one complication: The directory that lists public keys must be accurate. Think about what would happen to authentication if someone impersonated Bob and invented a private and public key for him. This interloper would then be accepted as Bob for any transaction. Hence, the public keys must be maintained by an organization that is trusted. Additionally, this organization must be careful to verify the identity of anyone (individual or corporation) who applies for a key. Several companies have begun to offer these services as a **certificate authority**. One of the leading commercial firms is Verisign, but GoDaddy is cheaper.

For internal use, it is relatively easy to set up a company server that functions as a certificate authority. With this approach, you can generate internal security certificates that will protect transmissions among employees and internal database applications. This approach is significantly less expensive than purchasing annual certificates for every employee. Of course, your certificates will probably not be accepted by anyone outside the company, so you will still need to obtain commercial certificates for applications that deal with external firms and people. Web browsers automatically use dual-key encryption on a **secure sockets layer (SSL)** connection. Check out the options in your browser and you will find a list of known certificate authorities. Certificates issued by these companies will be automatically accepted by the browser.

Encrypting data within a database has one additional twist: Where do you store the decryption key? The purpose of encrypting data within a database is to protect

it in case someone gets access to or steals the entire database file. It is particularly useful at protecting sensitive data such as credit card numbers or taxpayer IDs from browsing by employees or other insiders. However, you still have to find a place to store the decryption key so that your application can run, but someone who gets access to the database will not be able to find the decryption key. This problem must be addressed for both types of encryption. Modern operating systems are beginning to offer new options for storing decryption keys. For example, Microsoft systems now provide an encrypted data store specifically for handling encryption keys and certificates. You can find whitepapers on Microsoft's Web site that explain the options. One of the easier-to-use options is to protect the keys using special security certificates that are installed on the server. In SQL Server, you first run `CREATE MASTER KEY ENCRYPTION` to define a secure area within the database to hold certificates. Then you create certificates that are used to encrypt and decrypt the data. Oracle includes the `DBMS_CCRYPTO` package and sells a Transparent Data Encryption (TDE) system.

Another choice is to store the decryption key in a separate operating system file that has system security rights to allow only a special process to access the file. All of the methods increase the complexity of the application, but they do make it more difficult for someone to read the plaintext data. Over time, operating system vendors will likely add new features that are more secure and easier to use.

The most important rule to follow with encryption is that you should never attempt to create your own encryption method. Always use the tools that are built into the DBMS or operating system. They have been tested and verified by thousands of people.

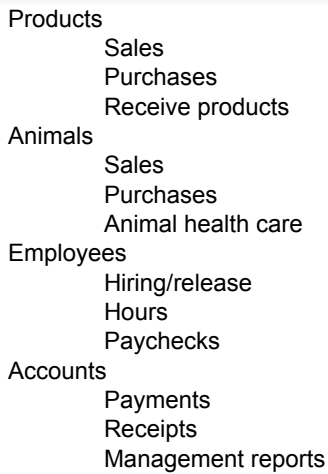
Sally's Pet Store

What security conditions would be needed at Sally's Pet Store? It is often easier to understand security issues through an example. The first step in assigning security permissions for Sally's Pet Store is to identify the various groups of users. The initial list is shown in Figure 10.27. As the company grows, there will eventually be additional categories of users. Note that these are groups and that several people might be assigned to each category.

Figure 10.27

Initial list of user groups for Sally's Pet Store.

- Management
 - Sally/CEO
- Sales Staff
 - Store manager
 - Salespeople
- Business Alliances
 - Accountant
 - Attorney
 - Suppliers
 - Customers



- Products
 - Sales
 - Purchases
 - Receive products
- Animals
 - Sales
 - Purchases
 - Animal health care
- Employees
 - Hiring/release
 - Hours
 - Paychecks
- Accounts
 - Payments
 - Receipts
 - Management reports

Figure 10.28

Primary operations at Sally's Pet Store. All of these transactions will have forms or reports built into the database.

The second step is to identify the operations that various users will perform. Separate forms will be designed to support each of these activities. Figure 10.28 contains a partial list of the major activities.

The user and group accounts need to be created within the operating system and within the DBMS. After the tables, queries, and forms are created, the DBA should make sure that only the DBA should be able to read or modify data. Go through each operation and identify the queries and tables needed to perform the operation. You should list the permissions for each user group that are required to complete the operation. Figure 10.29 presents the permissions that would be needed to purchase items from suppliers. Notice that only the store managers (and the owner) can order new merchandise. (Insert permission on the `MerchandiseOrder` and `OrderItem` tables.) Also note that only the owner can add new suppliers. Remember that a referential integrity constraint is in place that forces the `MerchandiseOrder` table to use only `Suppliers` already listed in the `Supplier` table. Therefore, a store manager will not be able to invent a fictitious supplier. Also note that you would like to permit store managers to add items to the `OrderItem` table, but they should not be able to alter the order once it has been completed. The DBMS might not support this restriction, and you probably have to give the managers `Write` permission as well. If available, the distinction would be useful. Otherwise, a manager in charge of receiving products could steal some of the items and change the original order quantity. If Sally has enough managers, this problem can be minimized by dividing the duties and having one manager place orders and another manager record the shipments. Also note that Sally wants to record the identity of the employee who placed the order. For this purpose, you need only read permission on the `EmployeeID` and `Name` columns. This privilege can be set by creating a separate `EmployeeName` query that only retrieves a minimal number of columns from the `Employee` table. Then you can use this query for purchases instead of the original `Employee` table.

Purchase	Purchase Query				PurchaseItem Query	
	Merchandise Order	Supplier	Employee	City	Order Item	Merchandise
Sally/CEO	SIUD	SIUD	SIUD	SIUD	SIUD	SIUD
Store Mgr.	SIUD	S*	SIUD	SIUD	I	SIUD
Salespeople	S	S*	S: ID, Name	S	S	S
Accountant	S	S*	S: ID, Name	S	S	S
Attorney	-	-	S: ID, Name	-	-	-
Suppliers	S	S*	-	S	S	S
Customers	-	-	-	-	-	S

S: Select, I: Insert, U: Update, D: Delete

* Basic supplier data: ID, Name, Address, Phone

Figure 10.29

Permissions for purchases. Notice that only the owner can add new suppliers, and only top-level managers can create new orders.

You follow a similar process to identify the access rights for all of the other tables. It is critical that you test all of the security conditions. Best practices dictate that you lock down the permissions so almost no one can access the data. Then, add permissions so that each group can complete assigned tasks. If the actual permissions are altered, you should go record the new values on the charts. The displays make it easier to see if some group has too much access.

Summary

Several steps are involved in managing a database. The DA performs management tasks related to design and planning. Key priorities are establishing standards to facilitate sharing data and integrating applications. The DA also works with users and business managers to identify new applications. In contrast, the DBA is responsible for installing and maintaining the DBMS software, defining databases, ensuring data is backed up, monitoring performance, and assisting developers.

Each stage of application development involves different aspects of database management. Planning entails estimating the size and approximate development costs. Project management skills and teamwork are used in the design stage to split the project and assign it to individual workers. Implementation requires establishing and enforcing development standards, testing procedures, training, and operating plans. Once the application is operational, the DBA monitors performance in terms of space and processing time. Physical storage parameters and other attributes are modified to improve the application's performance.

Backup and recovery are key administrative tasks that must be performed on a regular basis. Backup is more challenging on systems that are running continuously. The DBMS takes a snapshot and saves the data at one point in time. All changes are saved to a journal, which is also backed up on a regular basis. If the system has to be recovered, the DBMS loads the snapshot and then integrates the logged changes.

Security is an important issue in database management. Physical security consists of problems that involve the actual equipment, such as natural disasters or

physical theft of hardware. Data backup and disaster planning are the keys to providing physical security. Logical security consists of protecting data from unauthorized disclosure, unauthorized modification, and unauthorized withholding. The first step to providing logical security is to create a system that enables the computer to identify the user. Then application designers and users must determine the access rights that should be assigned to each user. Access rights should be assigned to enforce separation of duties.

Encryption is a tool that is often needed to protect databases. Encryption is particularly useful when the operating system cannot protect the database files. Encryption is also used when data must be transmitted across networks—particularly open networks like the Internet.




A Developer's View

Miranda will quickly see that the tasks of a DBA are different from those of a developer, yet the developer must work closely with the DBA. As a developer, you need to understand the importance of data standards. You also need to work with the DBA in planning, implementing, and maintaining the database application. Before implementing the application, you need to establish the database security rights and controls. For your class project, identify all users and determine their access rights. Use queries to give them access only to the data that they need. Test your work. Also, run any performance monitors or analysis tools.

Key Terms

24-7	network attached storage (NAS)
advanced encryption standard (AES)	physical security
authentication	private key
behavioral security	public key
brute force attack	redundant array of independent drives (RAID)
certificate authority	REVOKE
cold site	Rivest-Shamir-Adelman (RSA) algorithm
computer-aided software engineering (CASE)	role
data administration	schema
data administrator (DA)	secure sockets layer (SSL) connection
database administration	shell site
database administrator (DBA)	snapshot
disaster plan	storage area network (SAN)
encryption	tablespaces
GRANT	transaction log
hot site	tuning
logical security	virtual machine (VM)
metadata	WITH GRANT OPTION

Review Questions

1. What is the role and purpose of a data administrator?
2. What tasks are performed by a database administrator?
3. What tools are available to monitor database performance?
-  4. What is metadata and how do DBAs determine the structure of tables in a database?
5. How does the DA facilitate teamwork in developing database applications?
6. What are the primary security threats to a business?
7. How do DBMSs backup data that is constantly changing?
-  8. How does hardware provide real-time backup to databases?
9. What are the three problems faced by logical security systems?
10. How does a DBMS identify a user?
11. What are the basic database privileges that can be assigned to users?
12. How do queries provide detailed access controls?
13. How does a good DBMS application provide for division of duties?
-  14. How is encryption used to protect data sent to Web sites?
15. How do virtual machines and storage area networks improve administration and security?

Exercises



1. Use online resources to find three job openings for a DBA. Summarize the expected salary and the experience level required for the jobs.
2. You are working on a new project that has the following primary tables with the estimated number of rows. Provide a rough estimate of the size of the database after three years of operation. Pick a DBMS and identify the components needed and try to estimate the cost upfront and annual software licensing costs

Table	Initial size (gigabytes)	Annual Growth rate
Customers	10	10%
Employees (with photo)	12	5%
Item/Package (per year)	625	10%
Tracking (per year)	5,000	10%

3. Briefly describe how you would protect a computer system from the following problems. List steps you will take before the event and after the event has occurred.
 - a) Your local electricity provider expanded into wind and solar power and now the weather is bad and the company has insufficient power during peak days so you have rolling brownouts (no power) for up to an hour every three days.
 - b) Employees sneaking cigarette breaks behind the building somehow caused a fire that burned down the computer center.
 - c) Your CEO managed to anger a couple of hacker groups and your Web servers are now facing a huge distributed denial of service attack so no one can get access.
 - d) The North Korean government created a spyware/attack program targeted to your production databases, placed it on a USB drive and dropped a dozen of them in your parking lot. An employee plugged it in an infected your computers, destroying the databases (and some production equipment).
 - e) The FBI just appeared at your door and said the person you hired as one of your DBAs a year ago was just caught at the airport with a ticket to China, smuggling a USB drive with what looks to be 80 percent of your main database contents.
 - f) As part of a lawsuit against the company, an ex-employee is accusing an HR worker of looking up salaries in the company payroll database and telling other people the numbers.
4. Research the methods of encrypting a database column for a specific DBMS used to connect to a Web application. In particular, describe how the key is stored and comment on what it would take to obtain the key.
5. Research the costs of running SQL Server (or Oracle if you prefer) on a cloud server hosted by a third party. Assume the size of the database is about 50 GB with about 400 GB of network data traffic a month.



6. You are setting up a database for a local government agency to handle its purchases, including electronic payments to suppliers. Define the user groups and access rights to the following tables:

```
Supplier(SupplierID, Name, Address, Phone, City, State,
ZIP, BankAccount)
PurchaseOrder(OrderID, SupplierID, OrderDate, EmployeeID,
DateDue, DatePaid)
OrderItem(OrderID, ItemID, Description, Quantity, Price,
DateReceived)
Payment(PaymentID, OrderID, PaymentDate, BankConfirm,
Amount)
```

7. You are setting up a database for a company that sells products over the Web. Define the access rights to four groups of users: Customers, Shipping Clerks, Marketing Manager, and Customer Service; on the following tables:

```
Customers(CustomerID, Name, Address, Gender, Phone,
Email, ...
Items(ItemID, Description, Photo, ListPrice, Category
Sale(SaleID, SaleDate, CustomerID, IPAddress
SaleItem(SaleID, ItemID, Quantity, SalePrice,
CustomerComments
Returns(ReturnID, ReturnDate, SaleID, ItemID, Quantity,
Reason, Comments)
```

8. Employees and other insiders present the greatest security problems to companies. Outline basic policies and procedures that should be implemented to protect the computer systems. (Hint: Research employee hiring procedures.)
9. Research the current status of RAID drives versus SSDs for storing database files. What are the benefits and costs to each method?
10. Write a metadata query to retrieve a list of all stored views that use a table named Customer. Hint: Try the SQL Standard, or use your favorite DBMS to test it.



Sally's Pet Store

11. Devise a security plan for Sally's Pet Store. Identify the various classes of users and determine the level of access required by each group. Create any queries necessary to provide the desired security.
- ✓ 12. If it does not already exist, create a sales query that uses data from the Customer, Sales, SaleItems, Employee, and City tables to produce a report of all sales sorted by state. Use a query analyzer to evaluate the query and identify methods to improve its performance.
13. Create a backup and recovery plan that will be used at Sally's Pet Store. Identify the techniques used, who will be in charge, and the frequency of the backups. Explain how the process will change as the store and the database grow larger.

14. What physical security controls will be needed to protect the database and hardware?
15. Assume the database security system has user groups for SalesClerks and Managers. Write the GRANT commands to let sales clerks record a new sale but only managers can make changes to the sale.
16. Assume the Pet Store grows bigger (but still a single store), to include 10 checkout counters including a grooming salon service, handling several hundred customers and thousands of items sold each day. Where would you expect to see bottlenecks in the application? How could performance and security be improved?
17. The Pet Store owner/manager wants to be able to connect to the database while traveling or at home. What security precautions would be needed to keep this connection safe?
18. What data tables or transactions would you want to monitor on a regular basis to help indicate if a crime or attack is begin committed against the system?



Rolling Thunder Bicycles

19. Devise a security plan for Rolling Thunder Bicycles. Identify the various classes of users and determine the level of access required by each group. Create any queries necessary to provide the desired security.
20. Devise a backup and recovery plan for Rolling Thunder Bicycles. Be sure to specify what data should be backed up and how often. Outline a basic disaster plan for the company. Where are security problems likely in the existing application? How should duties be separated to improve security?
21. Use the performance analyzer tools available for your DBMS to evaluate the tables, queries, forms, and reports. Provide an explanation of the top five recommendations.
22. The company is planning to set up a Web site to enable customers to enter and track their orders using the Internet. Explain the additional security procedures that will be needed.
23. The managers want to expand the employee information in the database, such as adding sick days, vacation leave, and some benefits information. For now, these would be simple columns in the Employee table. How can security be assigned so that each employee can see his or her personal data but not that of other employees?
24. Using the existing data, estimate the size and monthly transaction volume/ data traffic. Use this data to estimate the price for hosting the entire database on Microsoft's cloud-based SQL Server database.
25. Use the metadata to find all of the queries/views that reference the Customer table.

26. Which of the elements of the Rolling Thunder Bicycle application are most likely to require updates over time—often because of changes in the database software?
27. What data could most benefit through the use of encryption in this company? How can data be encrypted in Microsoft Access?



Corner Med

28. Specify the access permissions needed at Corner Med. Focus on the two main user groups: clerical and medical staff, but also include a managerial group that reviews financial and treatment summary data—without needing access to individual patient data. The plan should address the privacy implications.
29. The company would like to give wireless devices to the medical staff to give them access to the data while talking with patients. Research the potential security issues and describe a solution that would protect the privacy and security, and remain usable.
30. Describe the physical security precautions that need to be taken. How will users be authenticated? Where should the servers and data be stored? How will backups be handled?
31. Identify the elements of the database that should be stored in encrypted form. Pick a DBMS and research the support available to encrypt individual data elements.
32. Use performance monitoring tools to evaluate the database and identify any important suggestions for improving performance.
33. The company is planning to expand to multiple locations, each with a separate manager and staff, but wants to share the database centrally. Which aspects of the database application are likely to need the most work in terms of performance as the load increases?
34. The company wants to move the entire database to a cloud-based platform (pick one). Assume the company maintains about the same patient load per physician and simply adds 19 more locations. Also, assume the company decides to store images (high-definition camera and x-ray) that average two images per patient visit. Because each file is about 10 MB, the images themselves will be stored as files in the cloud with name links stored in the database. Estimate the cost of running this database on a cloud server, storing data for five years.
35. Write the GRANT statements to let each employee see his or her own Employee records and edit simple columns such as name and phone number (but not salary).

Web Site References

http://www.dama.org	Data management organization.
http://www.aitp.org	Association for Information Technology Professionals organization.
http://www.sigsac.org/	Association for Computing Machinery: Special Interest Group on Security, Audit, and Control.
http://www.cert.org	Internet organization tracking security topics.
http://www.databasejournal.com	Database security and other database management topics.
http://www.oracle.com/pls/db112/docindex/	Oracle Administrator Books Online

Additional Reading

- Bertino, E., B. Catania, E. Ferrari, P. Perlasca, A Logical Framework for Reasoning about Access Control Models, *ACM Transactions on Information and System Security (TISSEC)*, 6(1), February 2003, 71-127. [A detailed discussion of various security access complications.]
- Bryla, B. and K. Loney, *Oracle Database 11g DBA Handbook*, New York: McGraw-Hill/Oracle Press, 2008. [One of many Oracle administrator books from Oracle Press.]
- Carpenter, T., *Microsoft SQL Server 2012 Administration: Real-World Skills for MCSA Certification and Beyond*, Indianapolis: Sybex/Wiley, 2013. [One of many books for studying for certification exams.]
- Castano, S. (ed.). *Database Security*, Reading, MA: Addison Wesley, 1994. [Collection of articles from the Association of Computing Machinery.]
- Natan, R.B., *Implementing Database Security and Auditing*, Burlington, MA: Elsevier Digital Press, 2005.
- Shamsudeen, R., "Oracle Database 12c Review: Finally, a True Cloud Database," Infoworld, June 26, 2013. <http://www.computerworld.com/s/article/9240354/>. [Overview of improvements in Oracle 12c. Almost all of them are related to distributed databases.]