Chapter 1

Database Systems

Database Systems:
Design, Implementation, and Management,
Sixth Edition, Rob and Coronel

In this chapter, you will learn:

- The difference between data and information
- What a database is, about different types of databases, and why they are valuable assets for decision making
- Why database design is important
- How modern databases evolved from files and file systems

In this chapter, you will learn (continued):

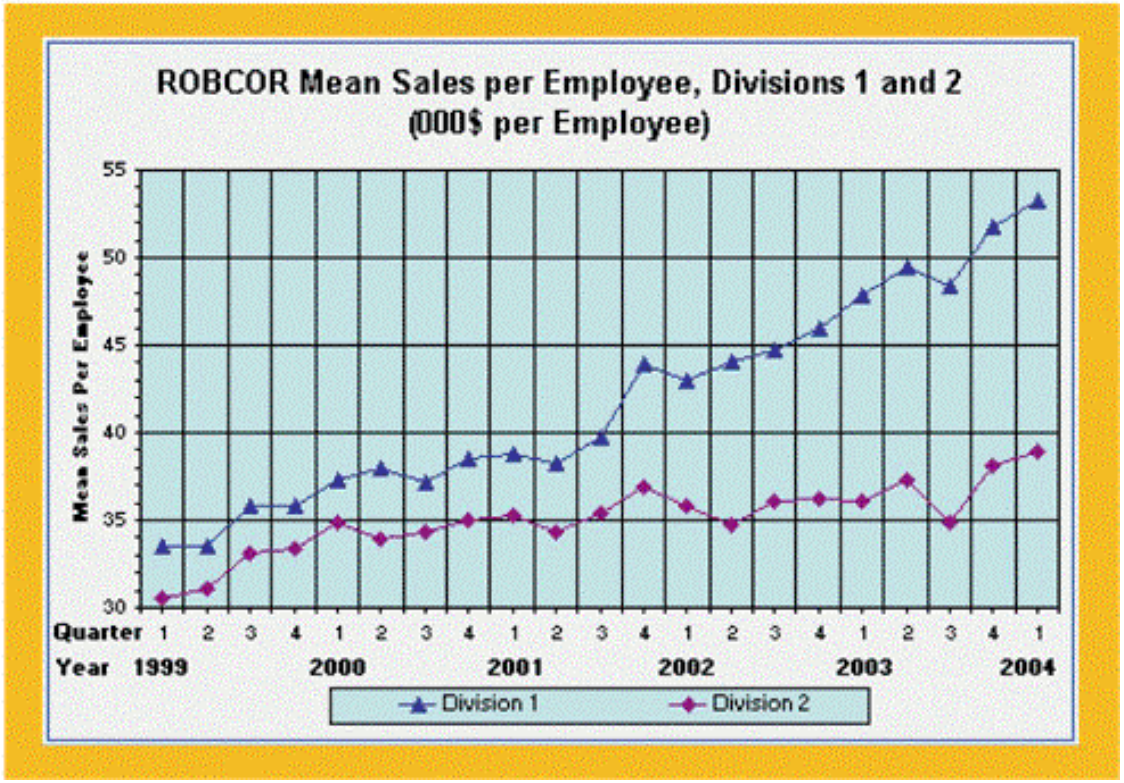
- About flaws in file system data management
- How a database system differs from a file system, and how a DBMS functions within the database system

Data vs. Information

- Data:
  - Raw facts; building blocks of information
  - Unprocessed information
- Information:
  - Data processed to reveal meaning
- Accurate, relevant, and timely information is key to good decision making
- Good decision making is key to survival in global environment

Sales per Employee for Each of ROBCOR's Two Divisions

FIGURE 1.1 SALES PER EMPLOYEE FOR EACH OF ROBCOR'S TWO DIVISIONS



Introducing the Database and the DBMS

- Database—shared, integrated computer structure that houses:
  - End user data (raw facts)
  - Metadata (data about data)

Introducing the Database and the DBMS (continued)

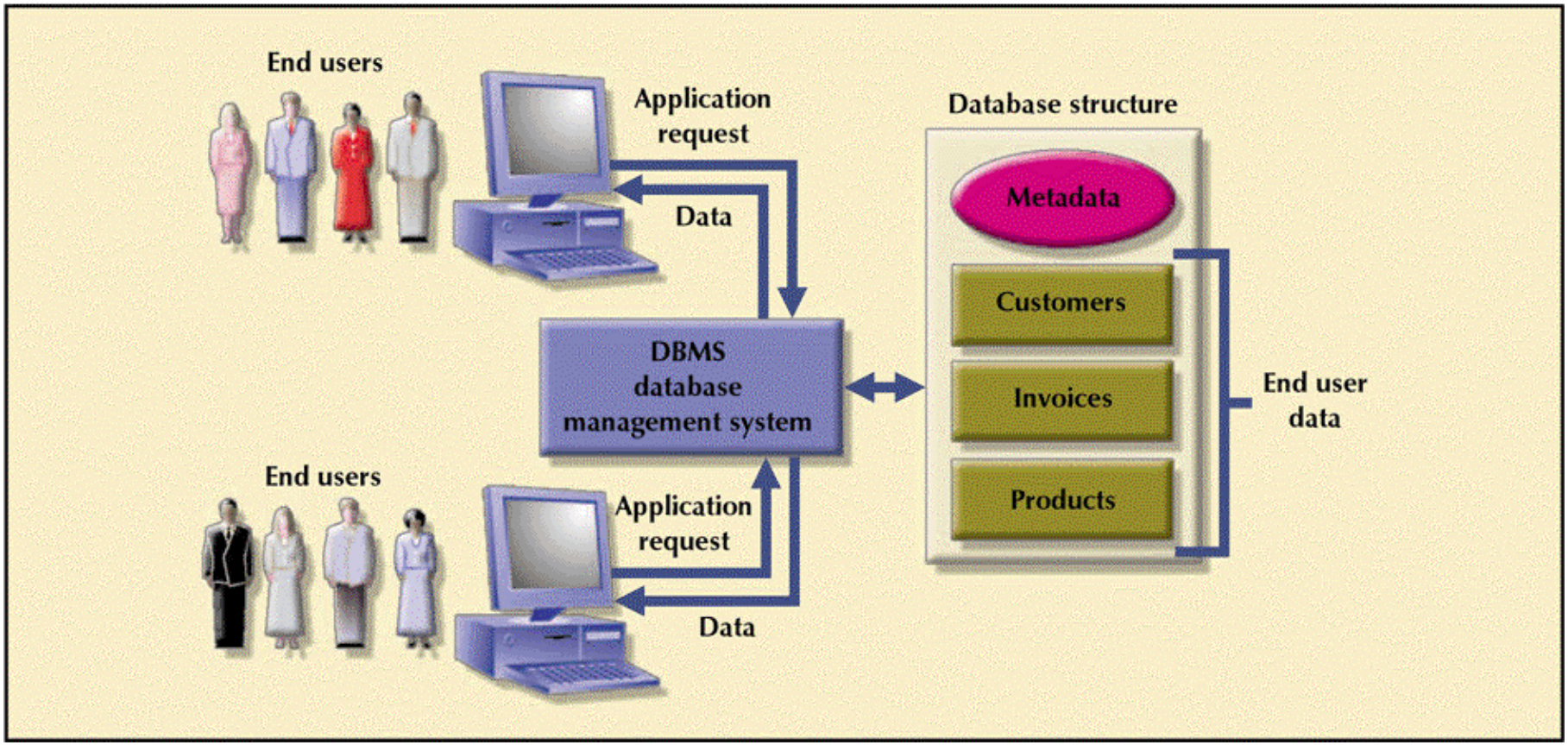
- DBMS (database management system):
  - Collection of programs that manages database structure and controls access to data
  - Possible to share data among multiple applications or users
  - Makes data management more efficient and effective

DBMS Makes Data Management More Efficient and Effective

- End users have better access to more and better-managed data
  - Promotes integrated view of organization's operations
  - Probability of data inconsistency is greatly reduced
  - Possible to produce quick answers to ad hoc queries

The DBMS Manages the Interaction Between the End User and the Database

FIGURE 1.2 THE DBMS MANAGES THE INTERACTION BETWEEN THE END USER AND THE DATABASE



Types of Databases

- Single-user:
  - Supports only one user at a time
- Desktop:
  - Single-user database running on a personal computer
- Multi-user:
  - Supports multiple users at the same time

Types of Databases (continued)

- Workgroup:
  - Multi-user database that supports a small group of users or a single department
- Enterprise:
  - Multi-user database that supports a large group of users or an entire organization

Location of Databases

- Centralized:
  - Supports data located at a single site
- Distributed:
  - Supports data distributed across several sites

Uses of Databases

- Transactional (or production):
  - Supports a company's day-to-day operations
- Data warehouse:
  - Stores data used to generate information required to make tactical or strategic decisions
    - Such decisions typically require "data massaging"
  - Often used to store historical data
  - Structure is quite different

Why Database Design is Important

- Defines the database's expected use
- Different approach needed for different types of databases
- Avoid redundant data (unnecessarily duplicated)
- Poorly designed database generates errors → leads to bad decisions → can lead to failure of organization

The Historical Roots of Database: Files and File Systems

- Although managing data through file systems is largely obsolete
  - Understanding relatively simple characteristics of file systems makes complexity of database design easier to understand
  - Awareness of problems that plagued file systems can help prevent similar problems in DBMS
  - Knowledge of file systems is helpful if you plan to convert an obsolete file system to a DBMS

Manual File Systems

- Traditionally composed of collection of file folders kept in file cabinet
- Organization within folders was based on data's expected use (ideally logically related)
- System was adequate for small amounts of data with few reporting requirements
- Finding and using data in growing collections of file folders became time-consuming and cumbersome

Conversion from Manual File System to Computer File System

- Could be technically complex, requiring hiring of data processing (DP) specialists
- DP specialists created file structures, wrote software, and designed application programs
- Resulted in numerous “home-grown” systems being created
- Initially, computer files were similar in design to manual files (see Figure 1.3)

Contents of Customer File

FIGURE 1.3 CONTENTS OF THE CUSTOMER FILE

C_NAME	C_PHONE	C_ADDRESS	C_ZIP	A_NAME	A_PHONE	TP	AMT	REN
Alfred A. Ramas	615-844-2573	218 Fork Rd., Babs, TN	36123	Leah F. Hahn	615-882-1244	T1	$100.00	05-Apr-2004
Leona K. Dunne	713-894-1238	Box 12A, Fox, KY	25246	Alex B. Alby	713-228-1249	T1	$250.00	16-Jun-2004
Kathy W. Smith	615-894-2285	125 Oak Ln, Babs, TN	36123	Leah F. Hahn	615-882-2144	S2	$150.00	29-Jan-2005
Paul F. Olowski	615-894-2180	217 Lee Ln., Babs, TN	36123	Leah F. Hahn	615-882-1244	S1	$300.00	14-Oct-2004
Myron Orlando	615-222-1672	Box 111, New, TN	36155	Alex B. Alby	713-228-1249	T1	$100.00	28-Dec-2004
Amy B. O'Brian	713-442-3381	387 Troll Dr., Fox, KY	25246	John T. Okon	615-123-5589	T2	$850.00	22-Sep-2004
James G. Brown	615-297-1228	21 Tye Rd., Nash, TN	37118	Leah F. Hahn	615-882-1244	S1	$120.00	25-Mar-2004
George Williams	615-290-2556	155 Maple, Nash, TN	37119	John T. Okon	615-123-5589	S1	$250.00	17-Jul-2004
Anne G. Farriss	713-382-7185	2119 Elm, Crew, KY	25432	Alex B. Alby	713-228-1249	T2	$100.00	03-Dec-2004
Olette K. Smith	615-297-3809	2782 Main, Nash, TN	37118	John T. Okon	615-123-5589	S2	$500.00	14-Mar-2004

C_NAME = Customer name
C_PHONE = Customer phone
C_ADDRESS = Customer address
C_ZIP = Customer ZIP code

A_NAME = Agent name
A_PHONE = Agent phone
TP = Insurance type
AMT = Insurance policy amount, in thousands of $
REN = Insurance renewal date

Basic File Terminology

TABLE 1.1 BASIC FILE TERMINOLOGY

TERM	DEFINITION
Data	"Raw" facts, such as a telephone number, a birth date, a customer name, and a year-to-date (YTD) sales value. Data have little meaning unless they have been organized in some logical manner. The smallest piece of data that can be "recognized" by the computer is a single character, such as the letter A, the number 5, or a symbol such as /. A single character requires one byte of computer storage.
Field	A character or group of characters (alphabetic or numeric) that has a specific meaning. A field is used to define and store data.
Record	A logically connected set of one or more fields that describes a person, place, or thing. For example, the fields that constitute a record for a customer named J.D. Rudd might consist of J.D. Rudd's name, address, phone number, date of birth, credit limit, and unpaid balance.
File	A collection of related records. For example, a file might contain data about vendors of ROBCOR Company; or a file might contain the records for the students currently enrolled at Gigantic University.

Example of Early Database Design

- DP specialist wrote programs for reports:
  - Monthly summaries of types and amounts of insurance sold by agents
  - Monthly reports about which customers should be contacted for renewal
  - Reports that analyzed ratios of insurance types sold by agent
  - Customer contact letters summarizing coverage
- Additional reports were written as required

Example of Early Database Design (continued)

- Other departments requested databases be written for them
  - SALES database created for sales department
  - AGENT database created for personnel department

Contents of the Agent File

FIGURE 1.4 CONTENTS OF THE AGENT FILE

	A_NAME	A_PHONE	A_ADDRESS	ZIP	HIRED	YTD_PAY	YTD_FIT	YTD_FICA	YTD_SLS	DEP
▶	Alex B. Alby	713-228-1249	123 Toll, Nash, TN	37119	01-Nov-1998	$26,566.24	$6,641.56	$2,125.30	$132,735.75	3
	Leah F. Hahn	615-882-1244	334 Main, Fox, KY	25246	23-May-1984	$32,213.76	$8,053.44	$2,577.10	$138,967.35	0
	John T. Okon	615-123-5589	452 Elm, New, TN	36155	15-Jun-2003	$23,198.29	$5,799.57	$1,855.86	$127,093.45	2

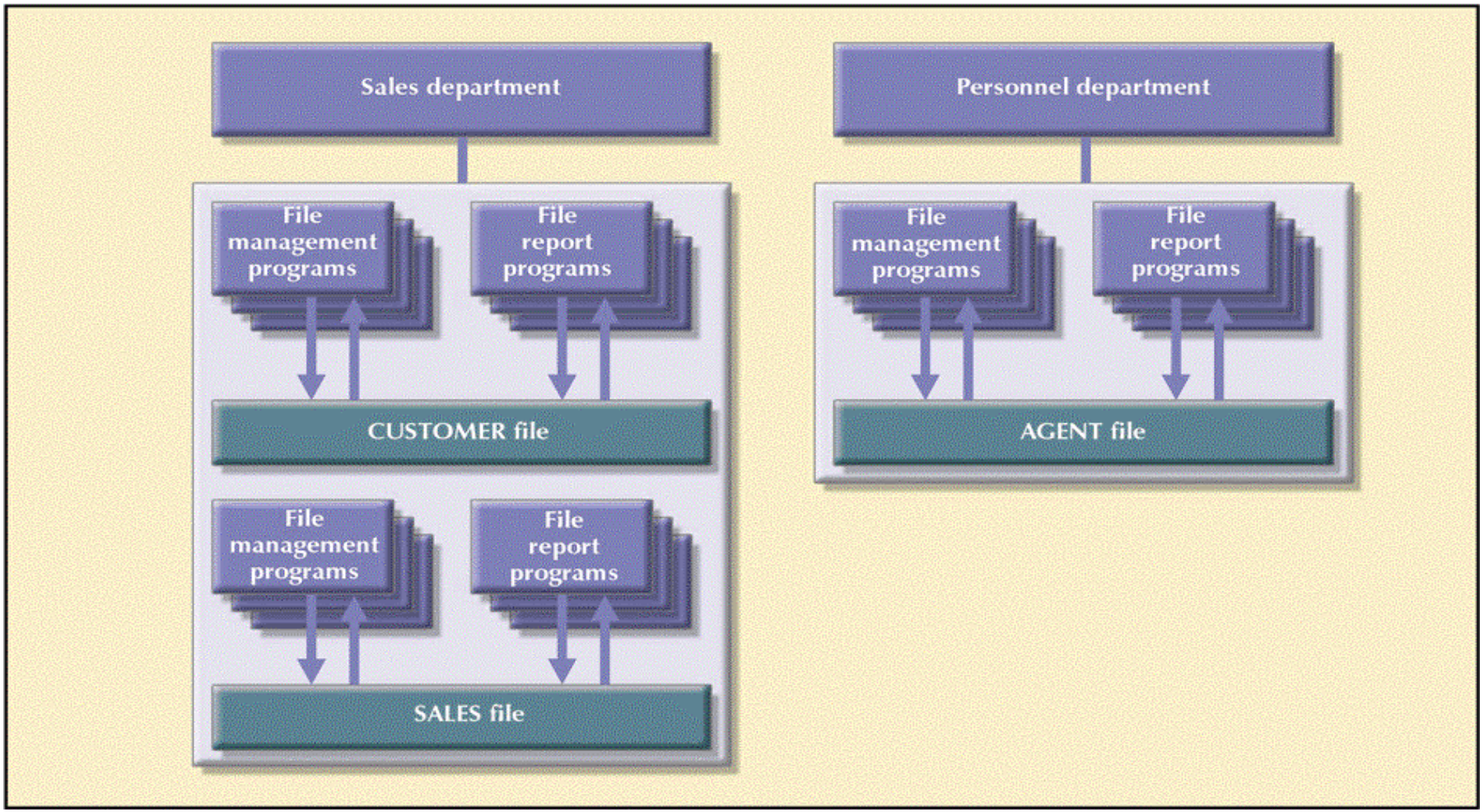
A_NAME = Agent name
A_PHONE = Agent phone
A_ADDRESS = Agent address
ZIP = Agent zip code
HIRED = Agent date of hire
YTD_PAY = Year-to-date pay
YTD_FIT = Year-to-date federal income tax paid
YTD_FICA = Year-to-date Social Security taxes paid
YTD_SLS = Year-to-date sales
DEP = Number of dependents

Evolution of Simple File System

- As number of databases increased, small file system evolved
- Each file used its own application programs
- Each file was owned by individual or department who commissioned its creation

A Simple File System

FIGURE 1.5 A SIMPLE FILE SYSTEM



Example of Early Database Design (continued)

- As system grew, demand for DP's programming skills grew
- Additional programmers hired
- DP specialist evolved into DP manager, supervising a DP department
- Primary activity of department (and DP manager) remained programming

Problems with File System Data Management

- Every task requires extensive programming in a third-generation language (3GL)
  - Programmer must specify task and *how* it must be done
- Modern databases use fourth-generation language (4GL)
  - Allows user to specify what must be done *without specifying how* it is to be done

Programming in 3GL

- Time-consuming, high-level activity
- Programmer must be familiar with physical file structure
- As system becomes complex, access paths become difficult to manage and tend to produce malfunctions
- Complex coding establishes precise location of files and system components and data characteristics

Programming in 3GL (continued)

- Ad hoc queries are impossible
- Writing programs to design new reports is time consuming
- As number of files increases, system administration becomes difficult
- Making changes in existing file structure is difficult
- File structure changes require modifications in all programs that use data in that file

Programming in 3GL (continued)

- Modifications are likely to produce errors, requiring additional time to “debug” the program
- Security features hard to program and therefore often omitted

Structural and Data Dependence

- Structural dependence
  - Access to a file depends on its structure
- Data dependence
  - Changes in database structure affect program's ability to access data
  - Logical data format
    - How a human being views the data
  - Physical data format
    - How the computer "sees" the data

Field Definitions and Naming Conventions

- Flexible record definition anticipates reporting requirements by breaking up fields into their component parts

Sample Customer File Fields

TABLE 1.2 SAMPLE CUSTOMER FILE FIELDS

FIELD	CONTENTS	SAMPLE ENTRY
CUS_LNAME	Customer last name	Ramas
CUS_FNAME	Customer first name	Alfred
CUS_INITIAL	Customer initial	A
CUS_AREACODE	Customer area code	615
CUS_PHONE	Customer phone	234-5678
CUS_ADDRESS	Customer street address or box number	123 Green Meadow Lane
CUS_CITY	Customer city	Murfreesboro
CUS_STATE	Customer state	TN
CUS_ZIP	Customer zip code	37130

Data Redundancy

- Data redundancy results in data inconsistency
  - Different and conflicting versions of the same data appear in different places
- Errors more likely to occur when complex entries are made in several different files and recur frequently in one or more files
- Data anomalies develop when required changes in redundant data are not made successfully

Data Anomalies

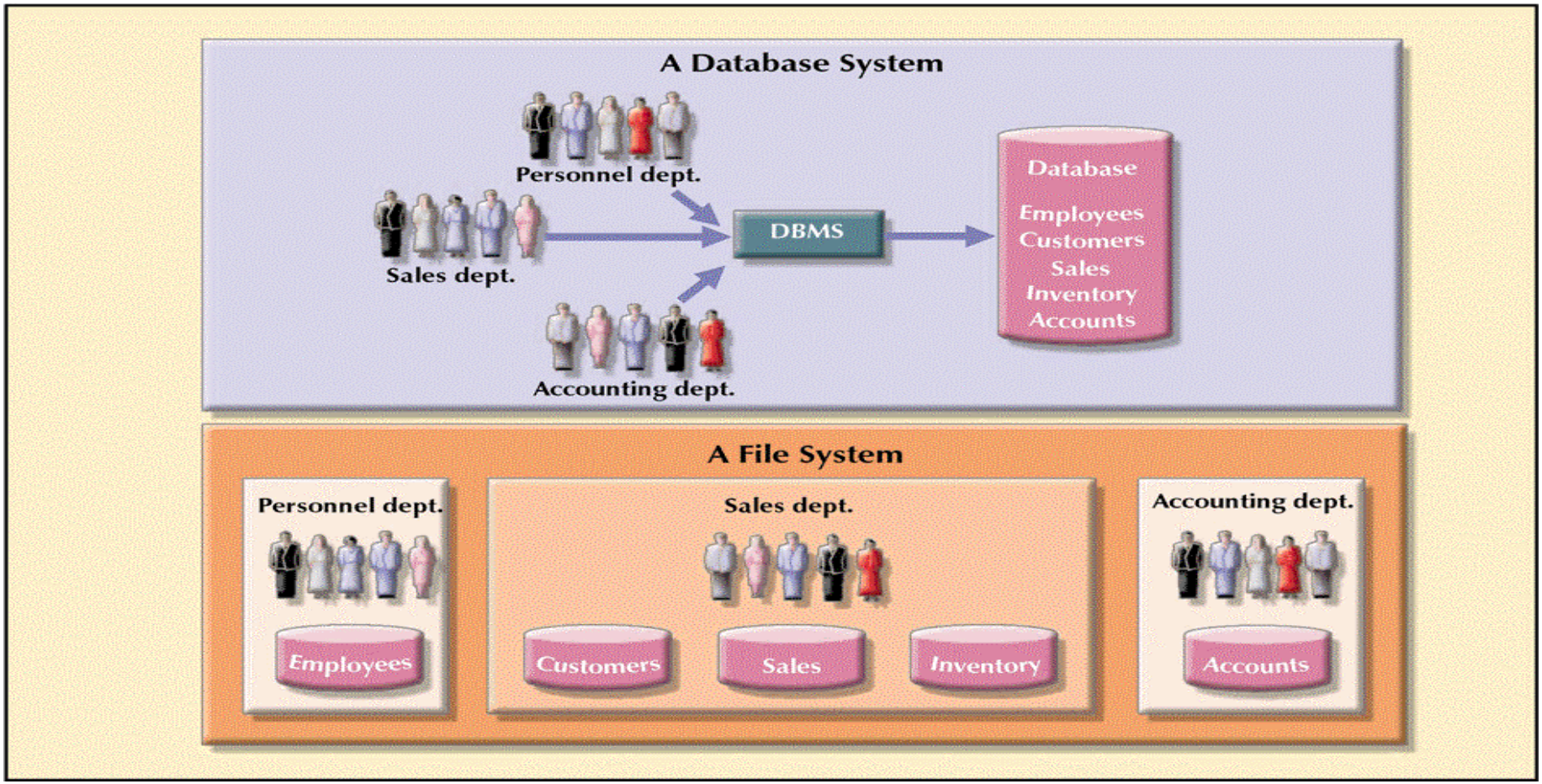
- Modification anomalies
  - Occur when changes must be made to existing records
- Insertion anomalies
  - Occur when entering new records
- Deletion anomalies
  - Occur when deleting records

Database vs. File System

- Problems inherent in file systems make using a database system desirable
- File system
  - Many separate and unrelated files
- Database
  - Logically related data stored in a single logical data repository

Contrasting Database and File Systems

FIGURE 1.6 CONTRASTING DATABASE AND FILE SYSTEMS

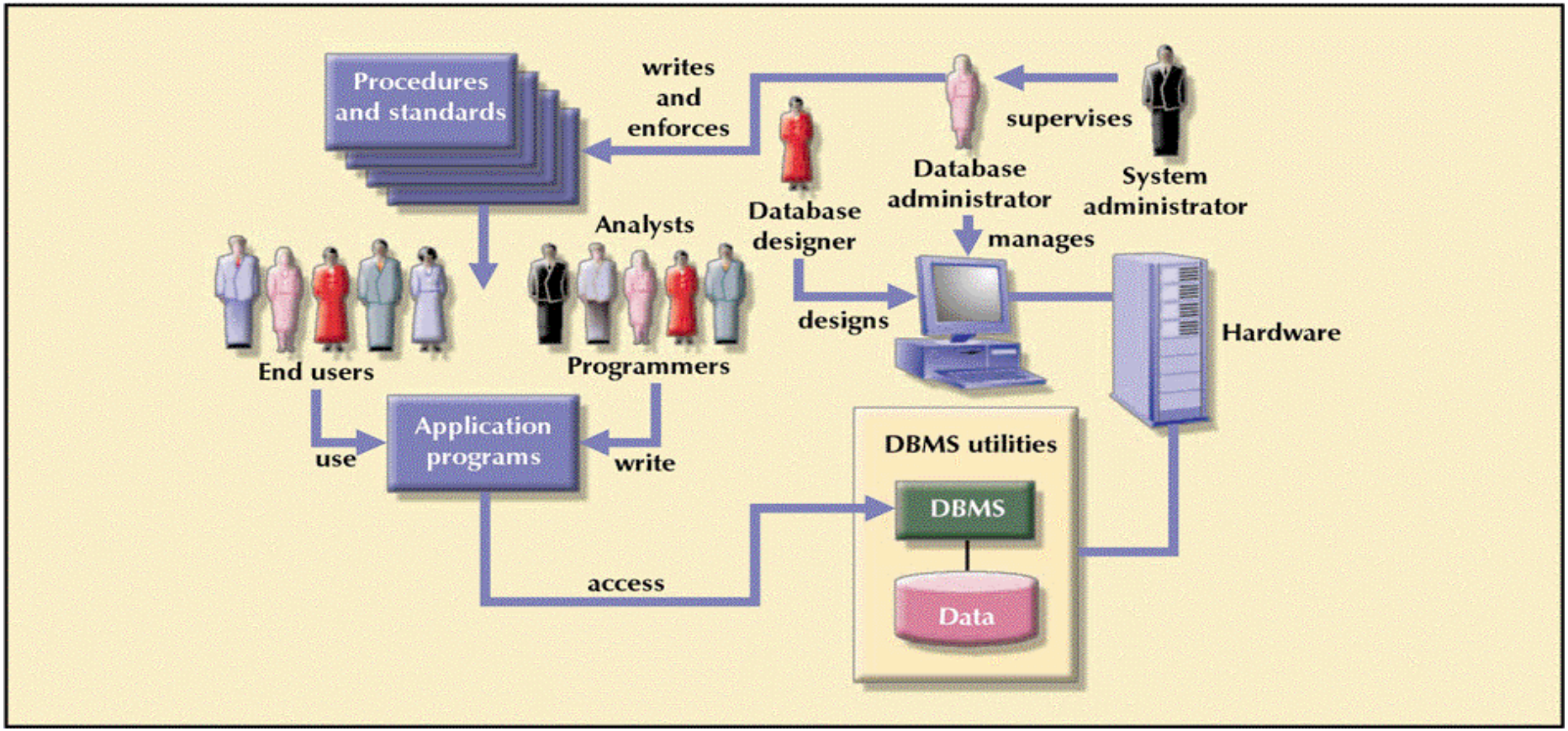


The Database System Environment

- Database system is composed of 5 main parts:
  1. Hardware
  2. Software
     - Operating system software
     - DBMS software
     - Application programs and utility software
  3. People
  4. Procedures
  5. Data

The Database System Environment (continued)

FIGURE 1.7 THE DATABASE SYSTEM ENVIRONMENT



DBMS Functions

- Performs functions that guarantee integrity and consistency of data
  - Data dictionary management
    - defines data elements and their relationships
  - Data storage management
    - stores data and related data entry forms, report definitions, etc.
  - Data transformation and presentation
    - translates logical requests into commands to physically locate and retrieve the requested data

DBMS Functions (continued)

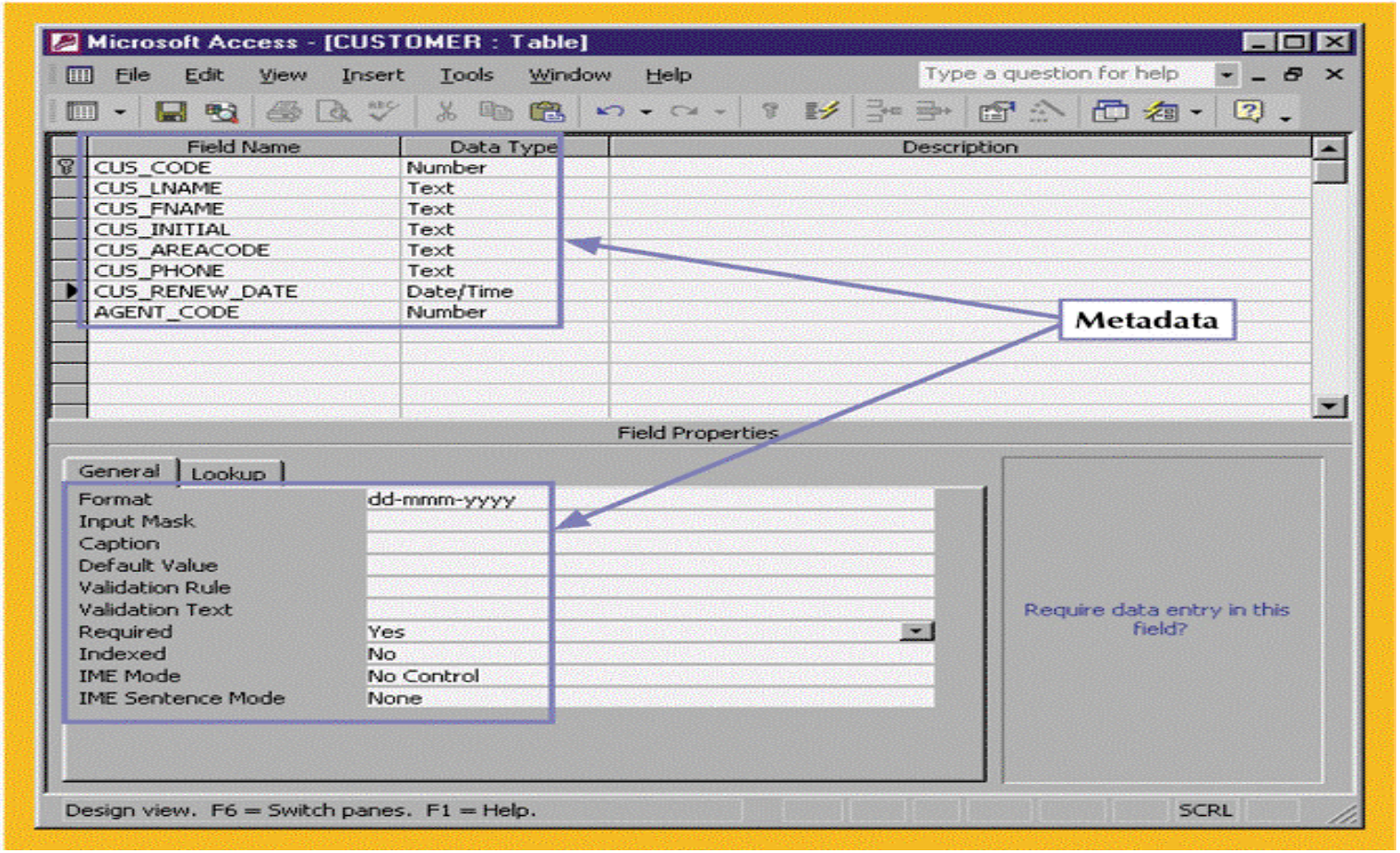
- Security management
  - enforces user security and data privacy within database
- Multi-user access control
  - creates structures that allow multiple users to access the data
- Backup and recovery management
  - provides backup and data recovery procedures

DBMS Functions (continued)

- Data integrity management
  - promotes and enforces integrity rules to eliminate data integrity problems
- Database access languages and application programming interfaces
  - provides data access through a query language
- Database communication interfaces
  - allows database to accept end-user requests within a computer network environment

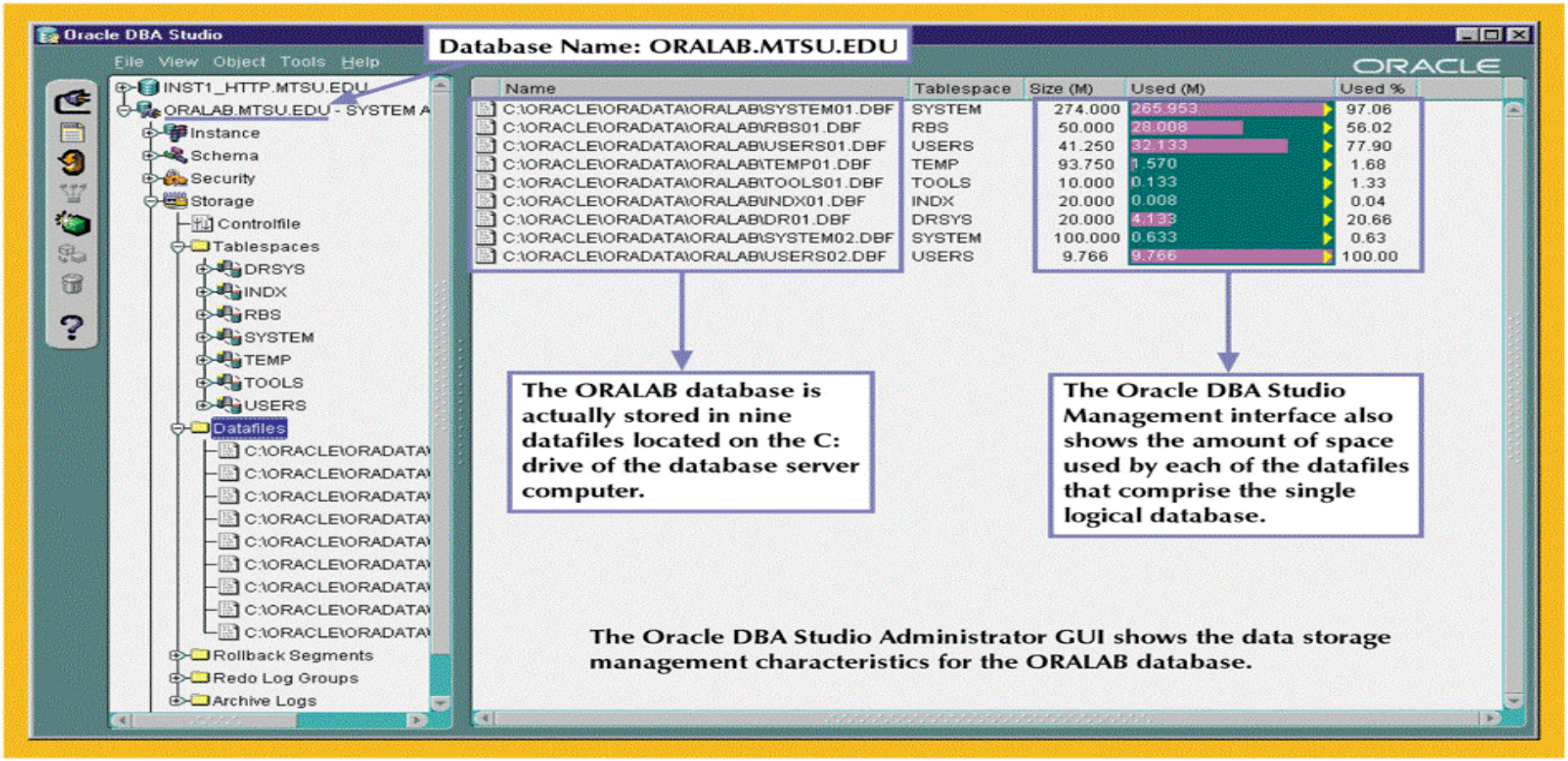
Illustrating Metadata with Microsoft Access

FIGURE 1.8 ILLUSTRATING METADATA WITH MICROSOFT ACCESS



Illustrating Data Storage Management with Oracle

FIGURE 1.9 ILLUSTRATING DATA STORAGE MANAGEMENT WITH ORACLE



Summary

- Information is derived from data, which is stored in a database
- To implement and manage a database, use a DBMS
- Database design defines its structure
- Good design is important

Summary (continued)

- Databases were preceded by file systems
- Because file systems lack a DBMS, file management becomes difficult as a file system grows
- DBMS were developed to address file systems' inherent weaknesses