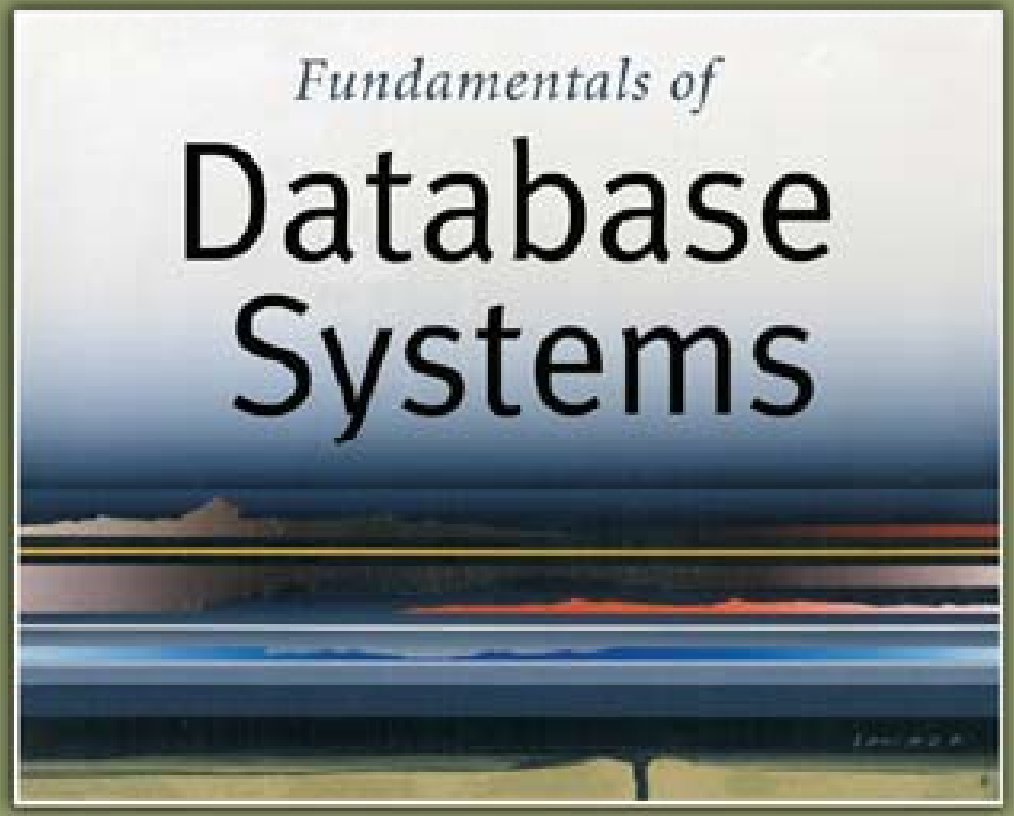
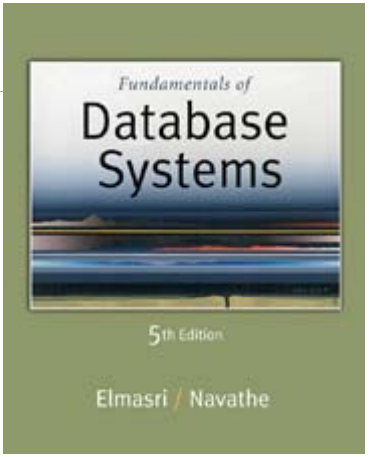


5th Edition

Elmasri / Navathe

Chapter 1

Introduction: Databases and Database Users



Outline

- Basic Definitions
- Typical DBMS Functionality
- Example of a Database (UNIVERSITY)
- Main Characteristics of the Database Approach
- Database Users
- Advantages of Using the Database Approach
- When Not to Use Databases

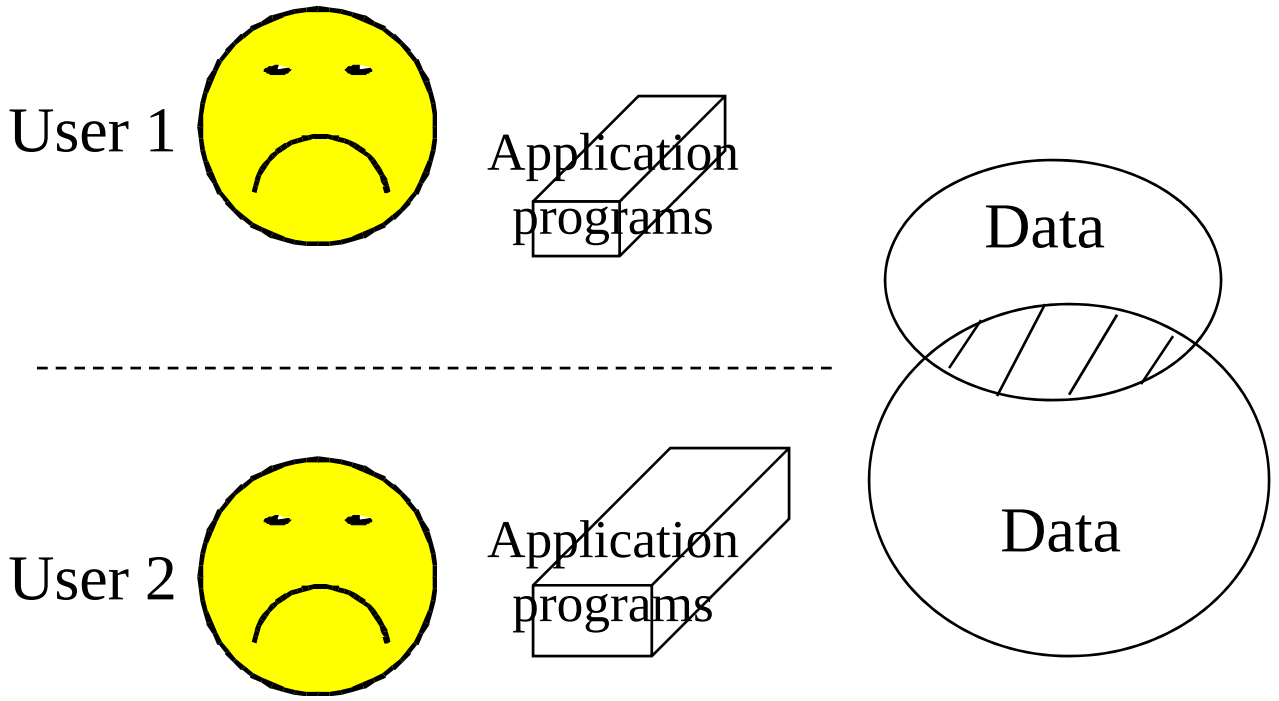
Basic Definitions

- **Data:**
  - Known facts that can be recorded and have an implicit meaning.
- **Database:**
  - A collection of related data. It has the following implicit properties:
    - A database represents some aspect of the real world, sometimes called the miniworld.
    - A database is a logically coherent collection of data with some inherent meaning.
    - A database is designed, built, and populated with data for specific purpose.
  - Examples: Airline reservation system, Students' registration system
- **Database Management System (DBMS):**
  - A software package/ system to facilitate the creation and maintenance of a computerized database.
- **Database System:**
  - The DBMS software together with the data itself. Sometimes, the applications are also included.

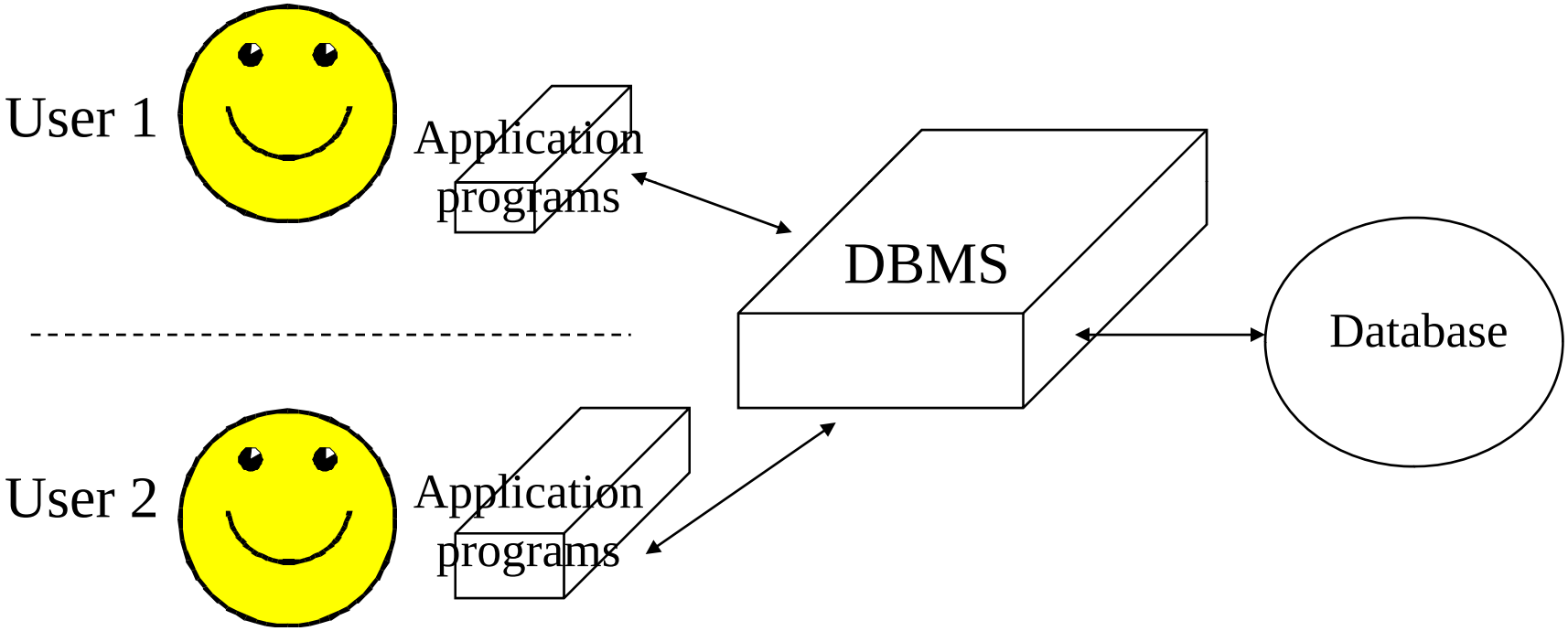
Managing Data

- There are two approaches to manage data
  - File-based approach: An approach that utilizes a collection of application programs which performs services to end-users (e.g. Reports). Each program defines and manages its own data.
  - Database approach: An approach that data is collected and manipulated using specific software called Database Management System, and many programs share this data.

File-Based Approach



Database Approach



Simplified database system environment

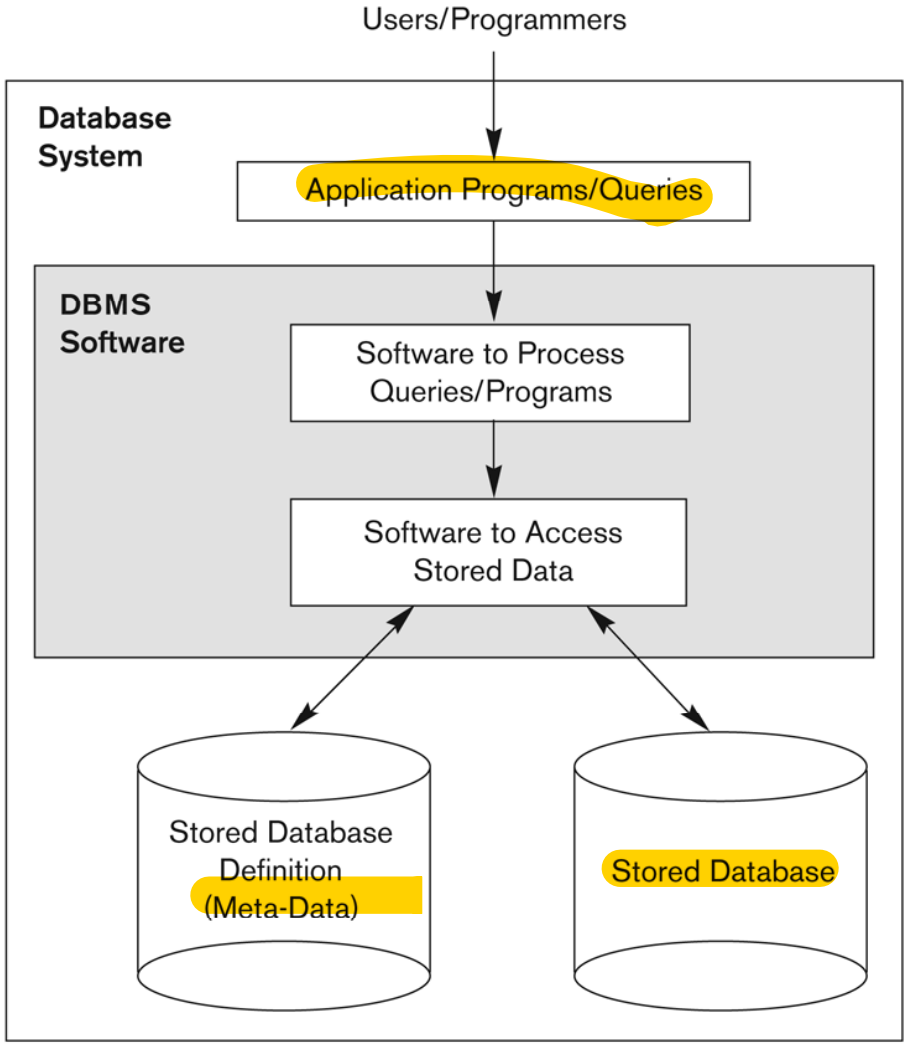


Figure 1.1
A simplified database system environment.

Typical DBMS Functionality

- *Define* a particular database in terms of its data types, structures, and constraints
- *Construct* or load the initial database contents on a secondary storage medium
- *Manipulating* the database:
  - Retrieval: Querying, generating reports
  - Modification: Insertions, deletions and updates to its content
  - Accessing the database through Web applications
- *Processing* and *Sharing* by a set of concurrent users and application programs – yet, keeping all data valid and consistent

Typical DBMS Functionality

- Other features:
  - Protection or Security measures to prevent unauthorized access
  - Maintaining the database and associated programs over the lifetime of the database application

Example of a Database (with a Conceptual Data Model)

- **Mini-world for the example:**
  - Part of a UNIVERSITY environment.
- **Some mini-world *entities*:**
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - DEPARTMENTs
  - INSTRUCTORs

Example of a Database (with a Conceptual Data Model)

- **Some mini-world *relationships*:**
  - SECTIONs *are of specific* COURSEs
  - STUDENTs *take* SECTIONs
  - COURSEs *have prerequisite* COURSEs
  - INSTRUCTORs *teach* SECTIONs
  - COURSEs *are offered by* DEPARTMENTs
  - STUDENTs *major in* DEPARTMENTs

- Note: The above entities and relationships are typically expressed in a conceptual data model, such as the ENTITY-RELATIONSHIP data model (see Chapters 3, 4)

Example of a simple database

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Main Characteristics of the Database Approach

- **Self-describing nature of a database system:**
  - A DBMS **catalog** stores the description of a particular database (e.g. data structures, types, and constraints)
  - The description is called **meta-data**.
  - This allows the DBMS software to work with different database applications.
- **Insulation between programs and data:**
  - Called **program-data independence**.
  - Allows changing data structures and storage organization without having to change the DBMS access programs.
- **Support of multiple views of the data:**
  - Each user may see a different view of the database, which describes **only** the data of interest to that user.

Main Characteristics of the Database Approach (continued)

- **Sharing of data and multi-user transaction processing:**
  - Allowing a set of **concurrent users** to retrieve from and to update the database.
  - *Concurrency control* within the DBMS guarantees that each **transaction** is correctly executed or aborted
  - *Recovery* subsystem ensures each completed transaction has its effect permanently recorded in the database. Similarly, each failed transaction is rolled back.
  - **OLTP** (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

Database Users

- **Actors on the Scene**: They actually use and control the database content; and design, develop and maintain database applications
  - Database Administrators
  - Database Designers
  - Software Engineers
  - End-users
- Workers Behind the Scene
  - Those who design and develop the DBMS software and related tools.

Database Users

- Actors on the scene
  - **Database administrators:**
    - Responsible for authorizing access to the database, for coordinating and monitoring its use, acquiring software and hardware resources, controlling its use and monitoring efficiency of operations.
  - **Database Designers:**
    - Responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

Categories of End-users

- Actors on the Scene:
  - **End-users**: Are those who require access to the database for querying, updating, and generating reports. They are categorized as:
    - **Casual end-users**: occasionally access the database, but they may need different information each time.
    - **Naive or parametric end-users**: constantly update and query databases, using standard types of queries and updates.
    - **Sophisticated end-users**: thoroughly familiarize themselves with the facilities of the DBMS so as to implement their application to meet their complex requirements.
    - **Stand-alone end-users**: maintain personal databases by using easy-to-use ready-made program packages.

Advantages of Using the Database Approach

- Controlling redundancy in data storage and in development and maintenance efforts.
  - Sharing of data among multiple users.
- Restricting unauthorized access to data.
- Providing persistent storage for program Objects
  - In Object-oriented DBMSs
- Providing Storage Structures (e.g. indexes) for efficient Query Processing

Advantages of Using the Database Approach (continued)

- Providing backup and recovery services.
- Providing multiple interfaces to different classes of users.
- Representing complex relationships among data.
- Enforcing integrity constraints on the database.
- Permitting actions using active rules
  - triggers, stored procedures

Additional Implications of Using the Database Approach

- Potential for enforcing standards:
  - for data item names, display formats, screens, report structures, Web page layouts, etc.
- Reduced application development time
- Flexibility to change data structures:
  - When requirements change
- Availability of current information:
  - Extremely important for on-line transaction systems such as airline, hotel, car reservations.
- Economies of scale:
  - Wasteful overlap of resources and personnel can be avoided by consolidating data and applications across departments.

When not to use a DBMS

- Main inhibitors (costs) of using a DBMS:
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.
- When a DBMS may be unnecessary:
  - If the database and applications are simple, well defined, and not expected to change.
  - If there are real-time requirements that may not be met because of DBMS overhead.
  - If access to data by multiple users is not required
- When no DBMS may suffice:
  - If the database system is not able to handle the complexity of data because of modeling limitations
  - If the database users need special operations not supported by the DBMS.