Chapter (1)
Basic Concepts

Objectives
Introducing the concept of database system
Some examples
Advantages of using DBMS
Implication of the DB approach
When not to use DBMS

Introduction
New applications of database systems store data in a form other than text and/or numbers:
- Multimedia database (pictures)
- Geographic information systems, GIS (maps, …)
- Data Warehouses
- On-line Analytical Processing (OLAP)
- Real time and Active Database Technology
Introduction

Database and database technology have significant impact on all areas where computers are used:

- Business (stock, merchandise distribution and control)
- Engineering (design criteria based on available data, previous data)
- Medicine (patients’ database, diagnostic cases database)
- Law (clients’ record, finger prints, face recognition, ...)
- Education (students’ record, resource distribution, ...)
- Library (books’ records, availability, and distribution, resource sharing)

What is a database system?

Introduction

A **database** is a collection of related data.

**Data** are usually the known facts that can be recorded and have implicit meaning.

Is a phone book (electronic or non-electronic) an example of database?
- It has records
- Records have implicit meaning
- The records are related

How can we tell if some data is a database?
**Introduction**

A database has some properties:

- A database represents some aspects of the real world. This may be referred to as the **miniworld** or the **Universe of Discourse (UoD)**.

  *Changes in the miniworld are reflected in the database.*

- A database is a logically coherent collection of data with some inherent meaning.

  *A random collection of some data cannot correctly be referred to as a database.*

- A database is designed, built, and populated with data for a specific purpose.

  *The users of the database are a specific group of people and the database has a specific application to these users.*

In summary: A database has some source from which data are derived, some degree of interaction with events in the real world, and an audience that is actively interested in the contents of the database.

Now, can you see these properties in a phone book?

---

**Introduction**

A database can be of any size and of varying complexity.

- Boone’s phone database may have 10,000 phone numbers but this number is much larger in a large city like Charlotte.

- A card catalog of a library may contain a million records.

- The IRS database is much much larger. Just to give you an idea of how much disk space it may take to keep 100 million records. IRS keeps about 4 years of returns. In average each return takes about 1000 bytes. Then the total amount of disk space would be: 4*100,000,000*1000 (bytes) = 400 Gb.

That brings the disk space issue up. However, a more important issue is the way the data is organized in a database that makes the search, retrieval, insert, and update faster.

Such organization is done through the Database Management System (DBMS).

What is a DBMS?
**Introduction**

A database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is hence a general-purpose software system that facilitates the processes of defining, constructing, and manipulating databases for various applications.

**Defining** a database involves specifying the data types, structures, and constraints for the data to be stored in the database.

**Constructing** the database is the process of storing the data itself on some storage medium that is controlled by the DBMS.

**Manipulating** a database includes such functions as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.

---

**Figure 01.01**

A simplified database system environment, illustrating the concepts and terminology discussed in Section 1.1.
To *define* this database, we must specify the structure of the records of each file by specifying the different types of **data elements** to be stored in each record.

**An Example: Students’ Database**

In the student database, what are the records that must be included?

- Student’s Name
- Student’s Number
- Class (freshman, sophomore, …)
- Major (Math, CS, …)
- Courses
  - Course Name, Course Number, Credit Hours, …

We must also specify a **data type** for each data element within a record.
Database manipulation involves querying and updating.

What is an example of a query in students’ database?

List courses, with their grades, taken by Smith.

Smith has student Number 17
Student Number 17 has taken section 112 and 119 and has earned a B and a C for these two courses respectively.
Course 112 is MATH2410 and it was offered by Chang in Fall 99.
Course 119 is CS1310 and it was offered by Anderson in Fall 99.

An example for Update:
Change the class of Smith to Sophomore, or
Add for Smith the database course with grade A. The course was offered in Fall 2001 by Tashakkori.

Characteristics of the Database Approach

Several characteristics distinguishes the database approach from the traditional approach of programming with files.

In traditional file processing, each user defines and implements the files needed for a specific application as part of programming the application.

Example: The Record and Registrar Office keeps the data regarding students’ courses and their grades while the Accounting Office keeps data regarding students’ fees and payments.

Do you see a problem here?

In the database approach, a single repository of data is maintained that is defined once and then is accessed by various users.
Characteristics of the Database Approach

Self-Describing Nature of a Database
Insulation between Program and Data, and Data Abstraction
Support of Multiple Views of the Data
Sharing of Data and Multiuser Transaction Processing

Self-Describing Nature of a Database
A database system contains not only the database itself but also a complete definition or description of the database structure and constraints.

The definition is stored in the system catalog.
Catalog contains information such as:
The structure of each file,
The type and storage format of each data item, and
Various constraints on the data.

Self-Describing Nature of a Database
The information stored in the catalog is called meta-data, and it describes the structure of the primary database.

What is the use of the catalog?
The catalog is used by the DBMS software and also by database users who need information about the database structure.
A DBMS refers to the catalog to find the structure of the files in a specific database, i.e., the type and format of data that it will access.
Traditional file processing works with only one specific database. In this case, the structure is declared in the application program.
Insulation between Programs and Data, and Data Abstraction

In traditional file processing, the structure of data files is embedded in the access programs. Thus, any change to the structure of a file may require changing all programs that access this file.

DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. This property is called program-data independence.

Example: Internal Storage Format for a Student record shown below

<table>
<thead>
<tr>
<th>Data Item Name</th>
<th>Starting Position in Record</th>
<th>Length in Characters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>StudentNumber</td>
<td>31</td>
<td>4</td>
</tr>
<tr>
<td>Class</td>
<td>35</td>
<td>4</td>
</tr>
<tr>
<td>Major</td>
<td>39</td>
<td>4</td>
</tr>
</tbody>
</table>

Insulation between Programs and Data, and Data Abstraction

There are times that a database is designed such that an operation on the database is defined in two parts: one part is the operation name and the other is data type of the arguments (or parameters).

The implementation of the operation is specified separately and can be changed without affecting the interface.

User application programs can operate on the data by invoking these operations through their names and arguments, regardless, of how the operations are implemented.

Example: Add $x \ y$ which is the same as: $x+y$

Select CourseName where Department = ‘CS’

This is known as program-operation independence.

The characteristics that allows program-data independence and program-operation independence is called Data Abstraction.
A DBMS provides users with a **conceptual representation** of data that includes many of the details of how the data is stored or how the operations are implemented.

A **data model** is a type of data abstraction that is used to provide this conceptual representation.

A data model hides storage and implementation details that are not of interest to most database users.

**Support of Multiple Views of the Data**

Every user of a database may have a different **view** of that database.

A view may be a subset of the database or it may contain **virtual data** that is derived from the database files but is not explicitly stored.

Examples:
- Create the student transcript view
- Create the course prerequisite view

---

**Figure 1.4 – Two views derived from the example database shown in Figure 1.2.**

(a) The student transcript view

(b) The course prerequisite view
Sharing of Data and Multiuser Transaction Processing

A multiuser DBMS must allow multiple users to access the database at the same time.

The database must include concurrency control software to ensure that several users trying to update the same data do so in a controlled manner so that the result of the updates is correct.

Think of reservation procedure for an airline flight

Applications that are used to ensure that each record is updated (accessed) by only one user is called On-line Transaction Processing (OLTP) applications.

Actors on the Scene

A small personal database is usually handled by one person. However, a large database requires the attention of many people to design and manipulate it. These people are called the Actors on the Scene by the author.

These people are:
- **Database Administrators (DBA)**
  - oversees and manages the resources
    - Primary resource: the database itself
    - Secondary resource: the DBMS and related software
  - authorizes the access to the database
  - coordinates and monitors its use
  - acquires the software and hardware resources as needed
  - ensure the security and performance of the database
- **Database Designers**
- **End Users**
Database Designers
A database designer is responsible
• to identify the data to be stored in the database, and
• to choose appropriate structures to represent and store this data
• to communicate with all prospective database users in order to understand their requirements
• to come up with a design that meets all the requirements.

End Users
End users are those people who are accessing the database for:
querying,
updating, and
generating reports.
There are several categories of end users.
Casual end users
Naive or parametric end users
Sophisticated end users
Stand-alone users.

Quiz (1)
End Users
Use the textbook to complete
Casual end users: Occasionally access the database, but they may need different info at different time.

Naive or parametric end users: A large group of end users.

Sophisticated end users: Includes engineers, scientists, business analysts,

Stand-alone users: Maintain personal databases.
Quiz (2)

Determine the End-user type in the following cases:

1) The department secretary uses a system to find out whether you have enough hours to be a junior.
2) You are hired by a company to develop a graphical interface for an existing database system that is designed in MySQL.
3) Your boss is a computer programmer/manager trying to find out more about the tool you have developed in (2).
4) The traffic officer trying to determine your tickets on a computer.
5) Your parents or friends who are having a DBMS for their properties they own in MS Access and maintain it themselves.

System Analyst and Application Programmers (Software Engineer)

A system analyst determines the requirements of end users, specially naive and parametric users and develop specifications for canned transactions that meet these requirements.

Application programmers

implement these specifications as programs, test, debug, document, and maintain

These canned transactions.
Workers behind the Scene
In addition to those who design, use, and administrate a database, others are associated with the design, development, and operation of the DBMS software and system environment. These people are not interested in the database itself. We call them the “workers behind the scene.”

They are:
- DBMS system designers and implementers
- Tool developers
- Operators and maintenance personnel

Although the above workers are instructional in making the database system available to end users, they typically do not use the database for their own purposes.

Advantages of Using a DBMS

- Controlling Redundancy
- Restricting Unauthorized Access
- Providing Persistent Storage for Program Objects and Data Structures
- Permitting Inferencing and Actions Using Rules
- Providing Multiple User Interface
- Representing Complex Relationships Among Data
- Enforcing Integrity Constraints
- Providing Backup and Recovery
Figure 01.05  The redundant storage of data items. (a) Controlled redundancy: Including StudentName and CourseNumber in the GRADE_REPORT file. (b) Uncontrolled redundancy: A GRADE_REPORT record that is inconsistent with the STUDENT records in Figure 01.02, because the Name of student number 17 is Smith, not Brown.

<table>
<thead>
<tr>
<th>GRADE REPORT</th>
<th>StudentNumber</th>
<th>StudentName</th>
<th>SectionIdentifier</th>
<th>CourseNumber</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Smith</td>
<td>112</td>
<td>MATH2410</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Smith</td>
<td>119</td>
<td>CS1310</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Brown</td>
<td>85</td>
<td>MATH2410</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Brown</td>
<td>92</td>
<td>CS1310</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Brown</td>
<td>102</td>
<td>CS3320</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Brown</td>
<td>135</td>
<td>CS3380</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

Implications of the Database Approach

- Potential for Enforcing Standards
- Reduced Application Development Time
- Flexibility
- Availability of UP-to-Date Information
- Economics of Scale

When Not to Use a DBMS

Do not use a DBMS when the overhead cost due to:

- High initial investment in hardware, software, and training
- Generality that a DBMS provides for defining and processing data
- Overhead for providing security, concurrency control, recovery, and integrity functions.

Use traditional file system if:

- The database and applications are simple, well defined, and not expected to change.
- When the real-time requirement may not be met by a DBMS
- Multiple users may not need to access the data
Summary
We identified several characteristics that distinguish the database approach from traditional file-processing applications:

- Existence of a catalog.
- Program-data independence and program-operation independence.
- Data abstraction.
- Support of multiple user views.
- Sharing of data among multiple transactions.

We then discussed the main categories of database users, or the "actors on the scene":
- Administrators.
- Designers.
- End users.
- System analysts and application programmers.

"workers behind the scene," in a database environment:
- DBMS system designers and implementers.
- Tool developers.
- Operators and maintenance personnel.

Then we presented a list of capabilities that should be provided by the DBMS software to the DBA, database designers, and users to help them design, administer, and use a database:
- Controlling redundancy.
- Restricting unauthorized access.
- Providing persistent storage for program objects and data structures.
- Permitting inferencing and actions by using rules.
- Providing multiple user interfaces.
- Representing complex relationships among data.
- Enforcing integrity constraints.
- Providing backup and recovery.
We listed some additional advantages of the database approach over traditional file-processing systems:

- Potential for enforcing standards.
- Reduced application development time.
- Flexibility.
- Availability of up-to-date information to all users.
- Economies of scale.

Finally, we discussed the overhead costs of using a DBMS and discussed some situations in which it may not be advantageous to use a DBMS.

---

**Have we met the Objectives?**

- [ ] Introducing the concept of database system
- [ ] Some examples
- [ ] Advantages of using DBMS
- [ ] Implication of the DB approach
- [ ] When not to use DBMS