Chapter 1
Databases and Database Users
Chapter 1 Outline

- Introduction
- An Example
- Characteristics of the Database Approach
- Actors on the Scene
- Workers behind the Scene
- Advantages of Using the DBMS Approach
- A Brief History of Database Applications
- When Not to Use a DBMS
Overview

- Traditional database applications
  - Store textual or numeric information

- Multimedia databases
  - Store images, audio clips, and video streams digitally

- Geographic information systems (GIS)
  - Store and analyze maps, weather data, and satellite images
Overview (cont'd.)

- Data warehouses and online analytical processing (OLAP) systems
  - Extract and analyze useful business information from very large databases
  - Support decision making
- Real-time and active database technology
  - Control industrial and manufacturing processes
Introduction

- **Database**
  - Collection of related data
  - Known facts that can be recorded and that have implicit meaning
  - **Miniworld or universe of discourse (UoD)**
  - Represents some aspect of the real world
  - Logically coherent collection of data with inherent meaning
  - Built for a specific purpose
Introduction (cont'd.)

- Example of a large commercial database
  - Amazon.com

- Database management system (DBMS)
  - Collection of programs
  - Enables users to create and maintain a database

- Defining a database
  - Specify the data types, structures, and constraints of the data to be stored
Introduction (cont'd.)

- **Meta-data**
  - Database definition or descriptive information
  - Stored by the DBMS in the form of a database catalog or dictionary

- **Manipulating a database**
  - Query and update the database miniworld
  - Generate reports
Introduction (cont'd.)

- **Sharing** a database
  - Allow multiple users and programs to access the database simultaneously

- **Application program**
  - Accesses database by sending queries to DBMS

- **Query**
  - Causes some data to be retrieved
Introduction (cont'd.)

- **Transaction**
  - May cause some data to be read and some data to be written into the database

- **Protection** includes:
  - System protection
  - Security protection

- **Maintain** the database system
  - Allow the system to evolve as requirements change over time
An Example

- UNIVERSITY database
  - Information concerning students, courses, and grades in a university environment

- Data records
  - STUDENT
  - COURSE
  - SECTION
  - GRADE_REPORT
  - PREREQUISITE
An Example (cont'd.)

- Specify structure of records of each file by specifying **data type** for each **data element**
  - String of alphabetic characters
  - Integer
  - Etc.
Figure 1.1
A simplified database system environment.
An Example (cont'd.)

- Construct UNIVERSITY database
  - Store data to represent each student, course, section, grade report, and prerequisite as a record in appropriate file
- Relationships among the records
- Manipulation involves querying and updating
An Example (cont'd.)

- Examples of queries:
  - Retrieve the transcript
  - List the names of students who took the section of the ‘Database’ course offered in fall 2008 and their grades in that section
  - List the prerequisites of the ‘Database’ course
An Example (cont'd.)

- Examples of updates:
  - Change the class of ‘Smith’ to sophomore
  - Create a new section for the ‘Database’ course for this semester
  - Enter a grade of ‘A’ for ‘Smith’ in the ‘Database’ section of last semester
An Example (cont'd.)

- Phases for designing a database:
  - Requirements specification and analysis
  - Conceptual design
  - Logical design
  - Physical design
### STUDENT

<table>
<thead>
<tr>
<th>Name</th>
<th>Student_number</th>
<th>Class</th>
<th>Major</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smith</td>
<td>17</td>
<td>1</td>
<td>CS</td>
</tr>
<tr>
<td>Brown</td>
<td>8</td>
<td>2</td>
<td>CS</td>
</tr>
</tbody>
</table>

### COURSE

<table>
<thead>
<tr>
<th>Course_name</th>
<th>Course_number</th>
<th>Credit_hours</th>
<th>Department</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro to Computer Science</td>
<td>CS1310</td>
<td>4</td>
<td>CS</td>
</tr>
<tr>
<td>Data Structures</td>
<td>CS3320</td>
<td>4</td>
<td>CS</td>
</tr>
<tr>
<td>Discrete Mathematics</td>
<td>MATH2410</td>
<td>3</td>
<td>MATH</td>
</tr>
<tr>
<td>Database</td>
<td>CS3380</td>
<td>3</td>
<td>CS</td>
</tr>
</tbody>
</table>

### SECTION

<table>
<thead>
<tr>
<th>Section_identifier</th>
<th>Course_number</th>
<th>Semester</th>
<th>Year</th>
<th>Instructor</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td>MATH2410</td>
<td>Fall</td>
<td>07</td>
<td>King</td>
</tr>
<tr>
<td>92</td>
<td>CS1310</td>
<td>Fall</td>
<td>07</td>
<td>Anderson</td>
</tr>
<tr>
<td>102</td>
<td>CS3320</td>
<td>Spring</td>
<td>08</td>
<td>Knuth</td>
</tr>
<tr>
<td>112</td>
<td>MATH2410</td>
<td>Fall</td>
<td>08</td>
<td>Chang</td>
</tr>
<tr>
<td>119</td>
<td>CS1310</td>
<td>Fall</td>
<td>08</td>
<td>Anderson</td>
</tr>
<tr>
<td>135</td>
<td>CS3380</td>
<td>Fall</td>
<td>08</td>
<td>Stone</td>
</tr>
</tbody>
</table>

### GRADE_REPORT

<table>
<thead>
<tr>
<th>Student_number</th>
<th>Section_identifier</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>112</td>
<td>B</td>
</tr>
<tr>
<td>17</td>
<td>119</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>85</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>92</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>102</td>
<td>B</td>
</tr>
<tr>
<td>8</td>
<td>135</td>
<td>A</td>
</tr>
</tbody>
</table>

### PREREQUISITE

<table>
<thead>
<tr>
<th>Course_number</th>
<th>Prerequisite_number</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS3380</td>
<td>CS3320</td>
</tr>
<tr>
<td>CS3380</td>
<td>MATH2410</td>
</tr>
<tr>
<td>CS3320</td>
<td>CS1310</td>
</tr>
</tbody>
</table>

**Figure 1.2**
A database that stores student and course information.
Characteristics of the Database Approach

- **Traditional file processing**
  - Each user defines and implements the files needed for a specific software application

- **Database approach**
  - Single repository maintains data that is defined once and then accessed by various users
Characteristics of the Database Approach (cont'd.)

- Main characteristics of database approach
  - Self-describing nature of a database system
  - Insulation between programs 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 System

- Database system contains complete definition of structure and constraints

- **Meta-data**
  - Describes structure of the database

- Database catalog used by:
  - DBMS software
  - Database users who need information about database structure
Insulation Between Programs and Data

- Program-data independence
  - Structure of data files is stored in DBMS catalog separately from access programs
- Program-operation independence
  - Operations specified in two parts:
    - Interface includes operation name and data types of its arguments
    - Implementation can be changed without affecting the interface
Data Abstraction

- **Data abstraction**
  - Allows program-data independence and program-operation independence

- **Conceptual representation** of data
  - Does not include details of how data is stored or how operations are implemented

- **Data model**
  - Type of data abstraction used to provide conceptual representation
### RELATIONS

<table>
<thead>
<tr>
<th>Relation_name</th>
<th>No_of_columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>STUDENT</td>
<td>4</td>
</tr>
<tr>
<td>COURSE</td>
<td>4</td>
</tr>
<tr>
<td>SECTION</td>
<td>5</td>
</tr>
<tr>
<td>GRADE_REPORT</td>
<td>3</td>
</tr>
<tr>
<td>PREREQUISITE</td>
<td>2</td>
</tr>
</tbody>
</table>

### COLUMNS

<table>
<thead>
<tr>
<th>Column_name</th>
<th>Data_type</th>
<th>Belongs_to_relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Character (30)</td>
<td>STUDENT</td>
</tr>
<tr>
<td>Student_number</td>
<td>Character (4)</td>
<td>STUDENT</td>
</tr>
<tr>
<td>Class</td>
<td>Integer (1)</td>
<td>STUDENT</td>
</tr>
<tr>
<td>Major</td>
<td>Major_type</td>
<td>STUDENT</td>
</tr>
<tr>
<td>Course_name</td>
<td>Character (10)</td>
<td>COURSE</td>
</tr>
<tr>
<td>Course_number</td>
<td>XXXXNNNNN</td>
<td>COURSE</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>.....</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>.....</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
<td>.....</td>
</tr>
<tr>
<td>Prerequisite_number</td>
<td>XXXXNNNNN</td>
<td>PREREQUISITE</td>
</tr>
</tbody>
</table>

*Note:* Major_type is defined as an enumerated type with all known majors. XXXXNNNNN is used to define a type with four alpha characters followed by four digits.
Support of Multiple Views of the Data

- **View**
  - Subset of the database
  - Contains *virtual data* derived from the database files but is not explicitly stored

- **Multiuser DBMS**
  - Users have a variety of distinct applications
  - Must provide facilities for defining multiple views
Sharing of Data and Multiuser Transaction Processing

- Allow multiple users to access the database at the same time

- **Concurrency control software**
  - Ensure that several users trying to update the same data do so in a controlled manner
    - Result of the updates is correct

- **Online transaction processing (OLTP) application**
Sharing of Data and Multiuser Transaction Processing (cont'd.)

- **Transaction**
  - Central to many database applications
  - Executing program or process that includes one or more database

- **Isolation** property
  - Each transaction appears to execute in isolation from other transactions

- **Atomicity** property
  - Either all the database operations in a transaction are executed or none are
Actors on the Scene

- **Database administrators (DBA)** are responsible for:
  - Authorizing access to the database
  - Coordinating and monitoring its use
  - Acquiring software and hardware resources

- **Database designers** are responsible for:
  - Identifying the data to be stored
  - Choosing appropriate structures to represent and store this data
Actors on the Scene (cont'd.)

- **End users**
  - People whose jobs require access to the database
  - **Types**
    - Casual end users
    - Naive or parametric end users
    - Sophisticated end users
    - Standalone users
Actors on the Scene (cont'd.)

- **System analysts**
  - Determine requirements of end users
- **Application programmers**
  - Implement these specifications as programs
Workers behind the Scene

- **DBMS system designers and implementers**
  - Design and implement the DBMS modules and interfaces as a software package

- **Tool developers**
  - Design and implement tools

- **Operators and maintenance personnel**
  - Responsible for running and maintenance of hardware and software environment for database system
Advantages of Using the DBMS Approach

- Controlling redundancy
  - Data normalization
  - Denormalization
    - Sometimes necessary to use controlled redundancy to improve the performance of queries

- Restricting unauthorized access
  - Security and authorization subsystem
  - Privileged software
Advantages of Using the DBMS Approach (cont'd.)

- Providing **persistent** storage for program objects
  - Complex object in C++ can be stored permanently in an object-oriented DBMS
- **Impedance mismatch problem**
  - Object-oriented database systems typically offer data structure **compatibility**
Advantages of Using the DBMS Approach (cont'd.)

- Providing storage structures and search techniques for efficient query processing
  - Indexes
  - Buffering and caching
  - Query processing and optimization
Advantages of Using the DBMS Approach (cont'd.)

- Providing backup and recovery
  - Backup and recovery subsystem of the DBMS is responsible for recovery
- Providing multiple user interfaces
  - Graphical user interfaces (GUIs)
- Representing complex relationships among data
  - May include numerous varieties of data that are interrelated in many ways
Advantages of Using the DBMS Approach (cont'd.)

- **Enforcing integrity constraints**
  - **Referential integrity** constraint
    - Every section record must be related to a course record
  - **Key or uniqueness** constraint
    - Every course record must have a unique value for Course_number
- **Business rules**
- **Inherent rules** of the data model
Advantages of Using the DBMS Approach (cont'd.)

- Permitting inferencing and actions using rules
  - **Deductive database systems**
    - Provide capabilities for defining deduction rules
    - Inferencing new information from the stored database facts
  - **Trigger**
    - Rule activated by updates to the table
  - **Stored procedures**
    - More involved procedures to enforce rules
Advantages of Using the DBMS Approach (cont'd.)

- Additional implications of using the database approach
  - Reduced application development time
  - Flexibility
  - Availability of up-to-date information
  - Economies of scale
A Brief History of Database Applications

- Early database applications using hierarchical and network systems
  - Large numbers of records of similar structure
- Providing data abstraction and application flexibility with relational databases
  - Separates physical storage of data from its conceptual representation
  - Provides a mathematical foundation for data representation and querying
A Brief History of Database Applications (cont'd.)

- Object-oriented applications and the need for more complex databases
  - Used in specialized applications: engineering design, multimedia publishing, and manufacturing systems

- Interchanging data on the Web for e-commerce using XML
  - Extended markup language (XML) primary standard for interchanging data among various types of databases and Web pages
A Brief History of Database Applications (cont'd.)

- Extending database capabilities for new applications
  - Extensions to better support specialized requirements for applications
  - Enterprise resource planning (ERP)
  - Customer relationship management (CRM)
- Databases versus information retrieval
  - Information retrieval (IR)
    - Deals with books, manuscripts, and various forms of library-based articles
When Not to Use a DBMS

More desirable to use regular files for:

- Simple, well-defined database applications not expected to change at all
- Stringent, real-time requirements that may not be met because of DBMS overhead
- Embedded systems with limited storage capacity
- No multiple-user access to data
Summary

- **Database**
  - Collection of related data (recorded facts)

- **DBMS**
  - Generalized software package for implementing and maintaining a computerized database

- Several categories of database users

- Database applications have evolved
  - Current trends: IR, Web