



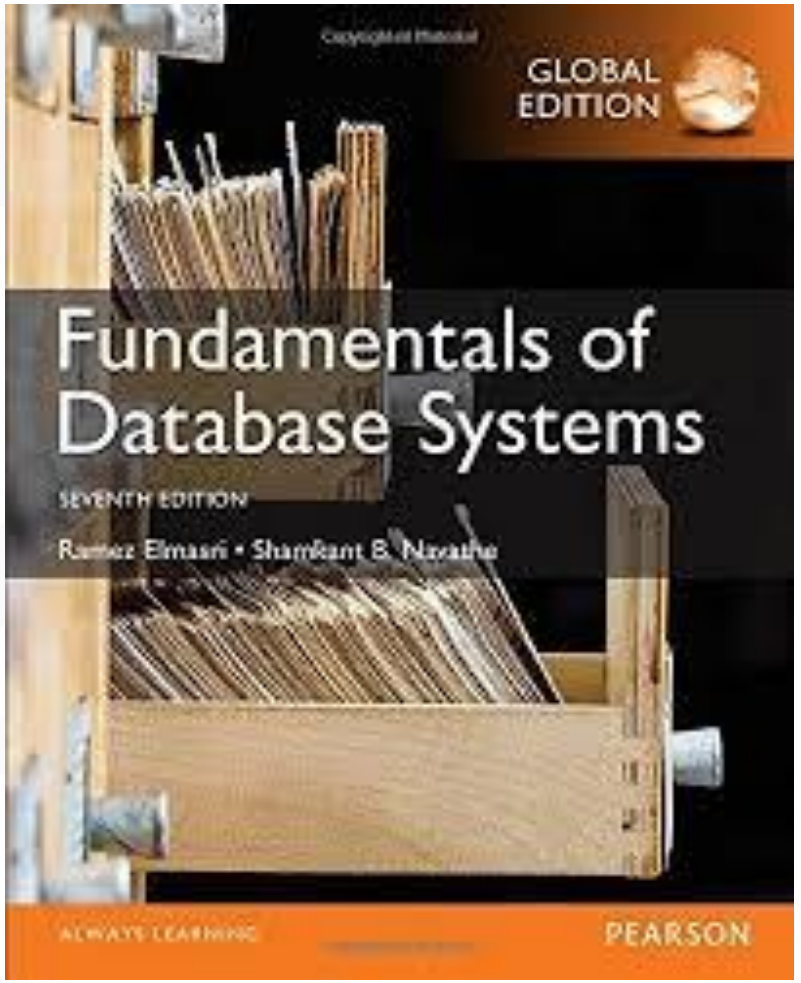
# Database Systems

Program in Computer Engineering
School of Engineering

King Mongkut's Institute of Technology Ladkrabang

# Text

- Ramez Elmasri and Shamkant B. Navathe. "Fundamentals of Database Systems" 7th Edition., Pearson, 2017



# Common ground of how people communicate

- ???

# Common ground of how people communicate

- Mutual knowledge
- Mutual beliefs
- Mutual assumptions


Clark, Herbert H.; Brennan, Susan E. (1991), Resnick, L. B.; Levine, J. M. (eds.), *Perspectives on socially shared cognition*, American Psychological Association, ISBN 1-55798-376-3

KMITL
พระจอมเกล้าลาดกระบัง
pakorn.wa@kmitl.ac.th

# Databases and Database Users

# Types of Databases and Database Applications

- Traditional Applications:
  - Numeric and Textual Databases
- More Recent Applications:
  - Multimedia Databases
  - Geographic Information Systems (GIS)
  - Biological and Genome Databases
  - Data Warehouses
  - Mobile databases
  - Real-time and Active Databases

# Recent Developments (1)

- Social Networks started capturing a lot of information about people and about communications among people-posts, tweets, photos, videos in systems such as:
  - Facebook
  - Twitter
  - Linked-In
- All of the above constitutes data
- Search Engines- Google, Bing, Yahoo : collect their own repository of web pages for searching purposes

# Recent Developments (2)

- New Technologies are emerging from the so-called non-database software vendors to manage vast amounts of data generated on the web:
- Big Data storage systems involving large clusters of distributed computers
- NOSQL (Not Only SQL) systems
- A large amount of data now resides on the "cloud" which means it is in huge data centers using thousands of machines.

# Basic Definitions

- **Data**:
  - Known facts that can be recorded and have an implicit meaning.
- **Database**:
  - A collection of related data.
- **Mini-world**:
  - Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.
- **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.

# Simplified database system environment

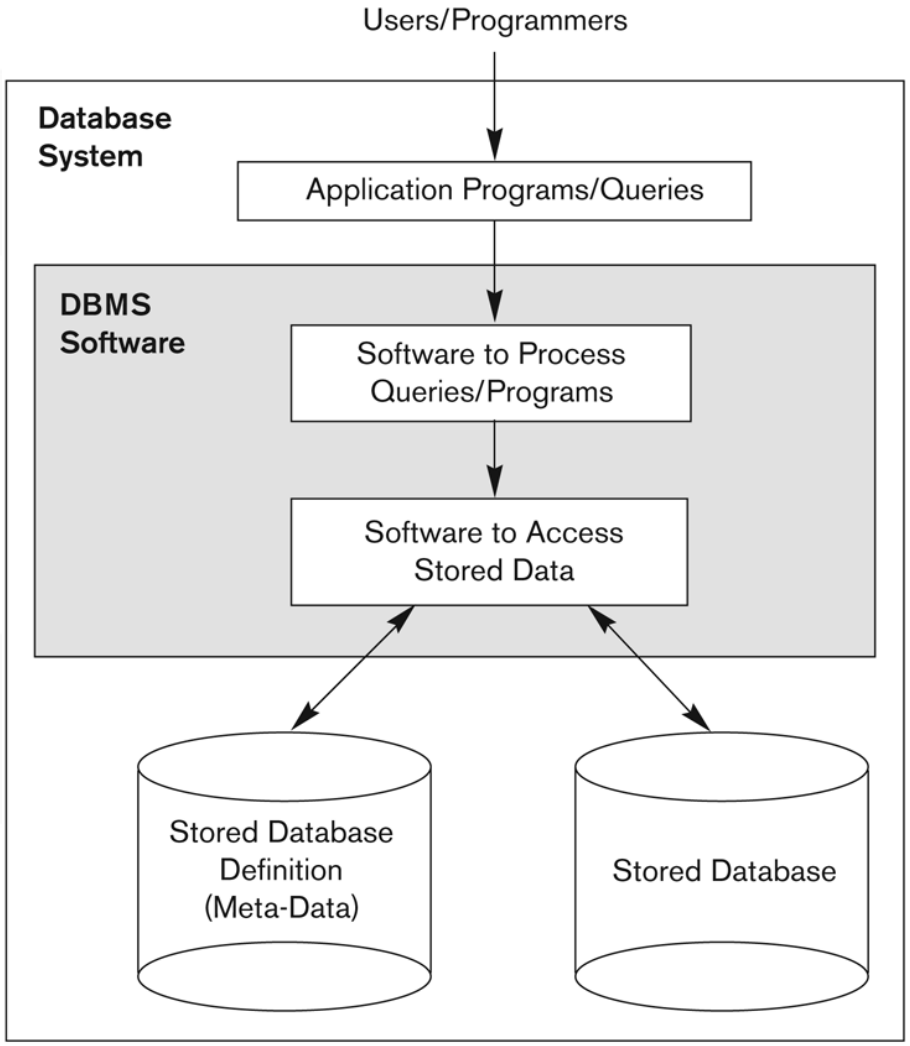



**Figure 1.1**
A simplified database system environment.

# 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

# Application Activities Against a Database

- Applications interact with a database by generating
  - Queries: access different parts of data and formulate the result of a request
  - Transactions: that may read some data and "update" certain values or generate new data and store that in the database
- Applications must not allow unauthorized users to access data
- Applications must keep up with changing user requirements against the database

# Additional DBMS Functionality

- DBMS may additionally provide:
  - Protection or Security measures to prevent unauthorized access
  - "Active" processing to take internal actions on data
  - Presentation and Visualization of data
  - Maintenance of the database and associated programs over the lifetime of the database application
    - Called database, software, and system maintenance

# 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)
  - (academic) 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

# Example of a simple database

**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 |

**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 |

**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 |

**PREREQUISITE**

| Course_number | Prerequisite_number |
|---|---|
| CS3380 | CS3320 |
| CS3380 | MATH2410 |
| CS3320 | CS1310 |

**Figure 1.2**
A database that stores student and course information.

# 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.

---

* Some newer systems such as a few NOSQL systems need no meta-data: they store the data definition within its structure making it self describing

# Example of a simplified database catalog

**RELATIONS**

| Relation_name | No_of_columns |
|---|---|
| STUDENT | 4 |
| COURSE | 4 |
| SECTION | 5 |
| GRADE_REPORT | 3 |
| PREREQUISITE | 2 |

**COLUMNS**

| Column_name | Data_type | Belongs_to_relation |
|---|---|---|
| Name | Character (30) | STUDENT |
| Student_number | Character (4) | STUDENT |
| Class | Integer (1) | STUDENT |
| Major | Major_type | STUDENT |
| Course_name | Character (10) | COURSE |
| Course_number | XXXXNNNN | COURSE |
| …. | …. | ….. |
| …. | …. | ….. |
| …. | …. | ….. |
| Prerequisite_number | XXXXNNNN | PREREQUISITE |

*Note*: Major_type is defined as an enumerared type with all known majors. XXXXNNNN is used to define a type with four alpha characters followed by four digits

**Figure 1.3**
An example of a database catalog for the database in Figure 1.2.

# Main Characteristics of the Database Approach (continued)

- **Data Abstraction:**
  - A **data model** is used to hide storage details and present the users with a conceptual view of the database.
  - Programs refer to the data model constructs rather than data storage details
- **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
  - **OLTP** (Online Transaction Processing) is a major part of database applications. This allows hundreds of concurrent transactions to execute per second.

# Database Users

- Users may be divided into
  - Those who actually use and control the database content, and those who design, develop and maintain database applications (called "Actors on the Scene"), and
  - Those who design and develop the DBMS software and related tools, and the computer systems operators (called "Workers Behind the Scene").

# Database Users – Actors on the Scene

- 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.

# Database End Users

- Actors on the scene (continued)
  - **End-users**: They use the data for queries, reports and some of them update the database content. End-users can be categorized into:
    - **Casual**: access database occasionally when needed
    - **Naïve** or **Parametric**: they make up a large section of the end-user population.
      - They use previously well-defined functions in the form of "canned transactions" against the database.
      - Users of Mobile Apps mostly fall in this category
      - Bank-tellers or reservation clerks are parametric users who do this activity for an entire shift of operations.
      - Social Media Users post and read information from websites

# Database End Users (continued)

- **Sophisticated**:
  - These include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities.
  - Many use tools in the form of software packages that work closely with the stored database.
- **Stand-alone**:
  - Mostly maintain personal databases using ready-to-use packaged applications.
  - An example is the user of a tax program that creates its own internal database.
  - Another example is a user that maintains a database of personal photos and videos.

# Database Users – Actors on the Scene (continued)

- **System Analysts and Application Developers**

  This category currently accounts for a very large proportion of the IT work force.

  - System Analysts: They understand the user requirements of naïve and sophisticated users and design applications including canned transactions to meet those requirements.
  - Application Programmers: Implement the specifications developed by analysts and test and debug them before deployment.
  - Business Analysts: There is an increasing need for such people who can analyze vast amounts of business data and real-time data ("Big Data") for better decision making related to planning, advertising, marketing etc.

# Database Users – Actors behind the Scene

- **System Designers and Implementors:** Design and implement DBMS packages in the form of modules and interfaces and test and debug them. The DBMS must interface with applications, language compilers, operating system components, etc.
- **Tool Developers:** Design and implement software systems called tools for modeling and designing databases, performance monitoring, prototyping, test data generation, user interface creation, simulation etc. that facilitate building of applications and allow using database effectively.
- **Operators and Maintenance Personnel:** They manage the actual running and maintenance of the database system hardware and software environment.

# Historical Development of Database Technology

- Early Database Applications:
  - The Hierarchical and Network Models were introduced in mid 1960s and dominated during the seventies.
  - A bulk of the worldwide database processing still occurs using these models, particularly, the hierarchical model using IBM's IMS system.
- Relational Model based Systems:
  - Relational model was originally introduced in 1970, was heavily researched and experimented within IBM Research and several universities.
  - Relational DBMS Products emerged in the early 1980s.

# Historical Development of Database Technology (continued)

- Object-oriented and emerging applications:
  - Object-Oriented Database Management Systems (OODBMSs) were introduced in late 1980s and early 1990s to cater to the need of complex data processing in CAD and other applications.
    - Their use has not taken off much.
  - Many relational DBMSs have incorporated object database concepts, leading to a new category called *object-relational* DBMSs (ORDBMSs)
  - Extended relational systems add further capabilities (e.g. for multimedia data, text, XML, and other data types)

# When not to use a DBMS

- Small and simple data:
  - Do not need the overhead of a DBMS
  - Use a spreadsheet or flat file to store and manage your data
- Temporary data:
  - Do not need to be stored for a long time
  - Use a temporary file or an in-memory data store
- Unstructured data:
  - E.g., text documents, images, or videos
  - Use a NoSQL database or a specialized data store
- High performance requirements:
  - Use an in-memory data store or a distributed database system

# When not to use a DBMS (cont'd)

- Lack of expertise:
  - do not have the expertise to administer a DBMS
  - Managing a DBMS can be complex and time-consuming
- Cost:
  - DBMS software can be expensive, and you may not need all of the features that it provides
  - There are open-source DBMS options available, but you will still need to invest time and resources to manage them
- Security:
  - If your data is sensitive, you may need to use a specialized DBMS that is designed for security.
  - A general-purpose DBMS may not be sufficient to protect your data from unauthorized access.

# When to use a DBMS

- Need to store, manage, and retrieve data
  - Large and complex
  - Structured
  - Long-term
  - Shared by multiple users
  - Subject by frequent changes


?
Q
&
A
!