KMITL
พระจอมเกล้าลาดกระบัง

KMITL GO
BEYOND THE LIMIT

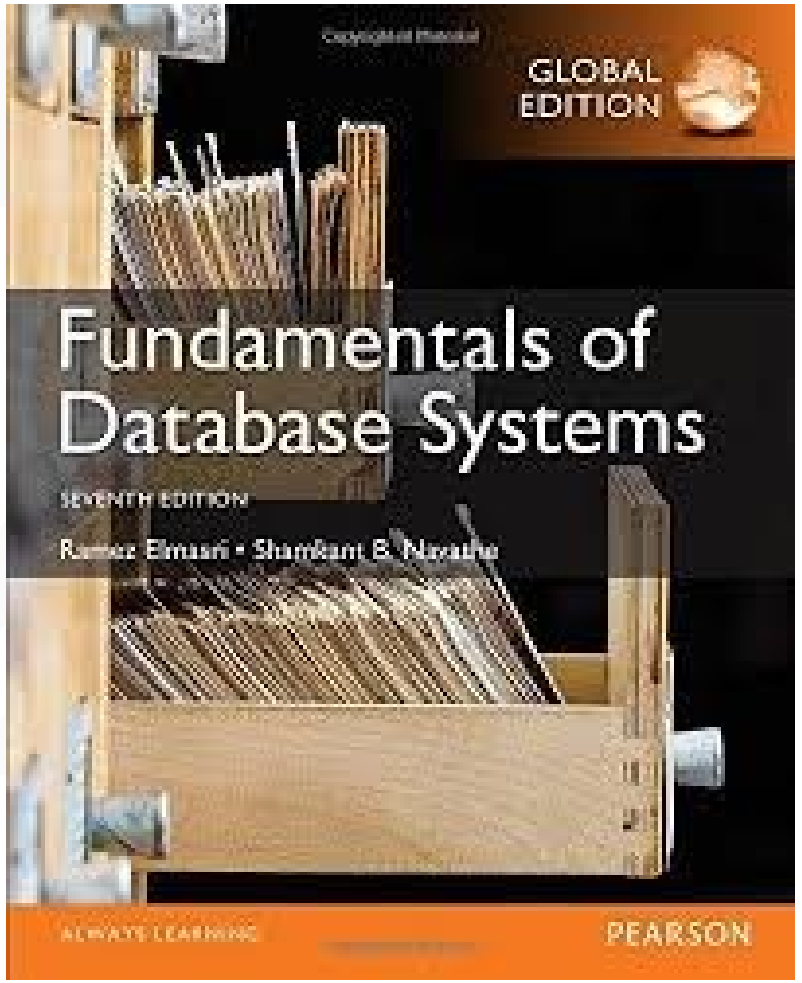
Database Systems

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



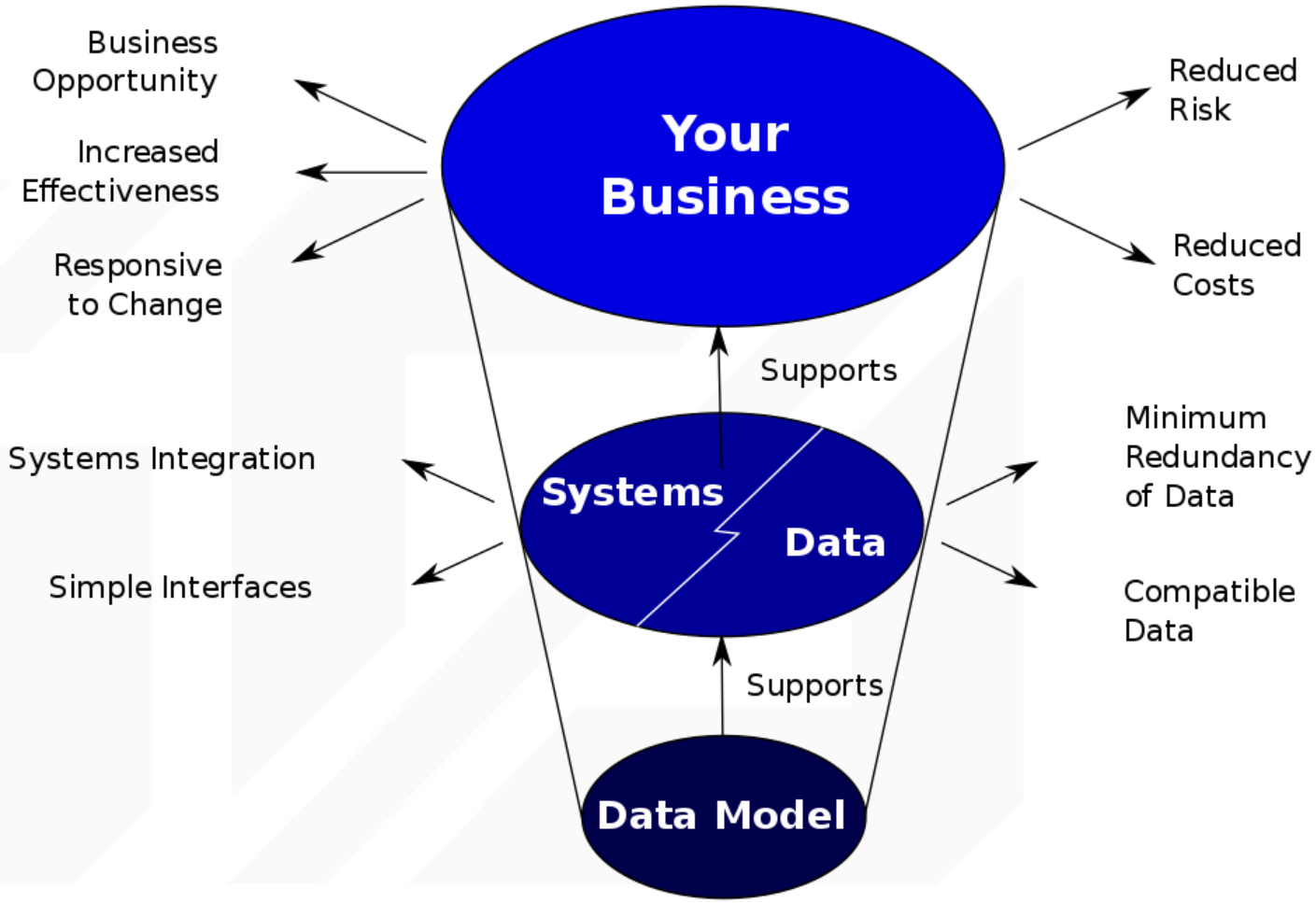
Chapter 2

Database System Concepts and Architecture

Data Models

- A set of concepts to describe the ***structure*** of a database, the ***operations*** for manipulating these structures, and certain ***constraints*** that the database should obey.

- Data models provide a **framework** for data to be used within **information systems** by providing specific definition and format



https://en.wikipedia.org/wiki/Data_modeling

Data Model Structure and Constraints

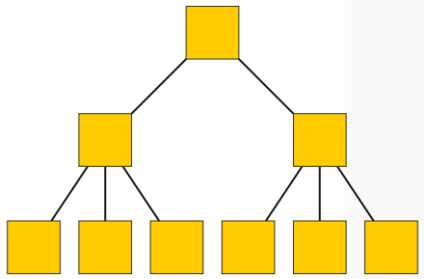
- Constructs are used to define the database structure
- Constructs typically include
  - **elements** (and their **data types**)
  - groups of elements (e.g. **entity**, **record**, **table**)
  - **relationships** among such groups
- Constraints specify some restrictions on valid data; these constraints must be enforced at all times

Data Model Operations

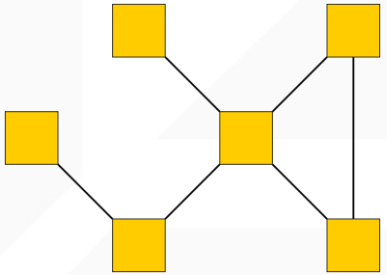
- These operations are used for specifying database **retrievals** and **updates** by referring to the constructs of the data model.
- Operations on the data model may include
  - **basic model operations** (e.g. generic insert, delete, update)
  - **user-defined operations** (e.g. compute_student_gpa, update_inventory)

Categories of Data Models

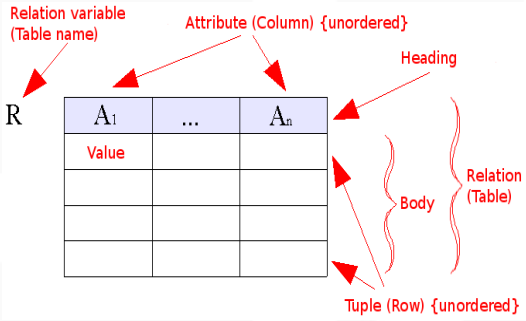
- **Conceptual (high-level, semantic) data models**:
  - Provide concepts that are close to the way many users perceive data.



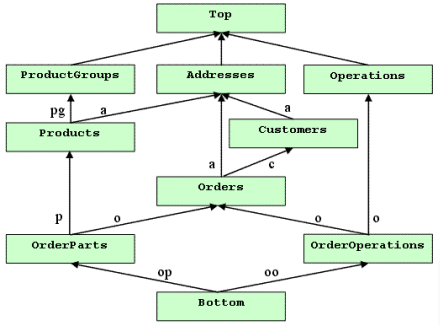
Hierarchical model



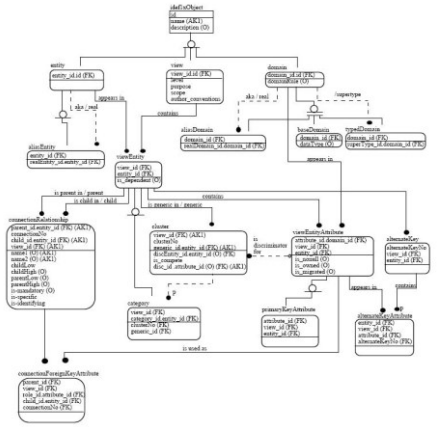
Network model



Relational model



Concept-oriented model



Object-Role model

https://en.wikipedia.org/wiki/Data_model

- **Implementation (representational) data models**:
  - A.k.a., **logical schema**
  - Provide concepts used by many commercial DBMS implementations (e.g. **relational data models** used in many commercial systems)
  - Describe the structure of some domain of information.
  - This consists of descriptions of (for example) **tables**, **columns**, **object-oriented classes**, and **XML tags**.
  - The logical schema and conceptual schema are sometimes implemented as one and the same.

https://en.wikipedia.org/wiki/Data_model

- **Physical (low-level, internal) data models**:
  - Provide concepts that describe details of how data is stored in the computer.
  - This is concerned with partitions, CPUs, tablespaces, and the like.
  - These are usually specified in an ad-hoc manner through DBMS design and administration manuals.

https://en.wikipedia.org/wiki/Data_model

Schemas

- In a data model, it is important to distinguish between the **description of the database** and the **database** itself.
- The description of a database is called the **database schema**.
- A displayed schema is called a **schema diagram**.

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Figure 2.1
Schema diagram for the database in Figure 1.2.

Database State

- The actual data stored in a database at a **particular moment** in time.
- This includes the collection of all the data in the database.
- Also called **database instance** (or **occurrence** or **snapshot**).
- The term instance is also applied to individual database components, e.g. **record instance**, **table instance**, **entity instance**

- **Database State**:
  - Refers to the content of a database at a moment in time.
- **Initial Database State**:
  - Refers to the database state when it is initially loaded into the system.
- **Valid State**:
  - A state that satisfies the structure and constraints of the database.

- Distinction
  - The ***database schema*** changes very infrequently.
  - The ***database state*** changes every time the database is updated.

- **Schema** is also called **intension**.
- **State** is also called **extension**.

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Figure 2.1
Schema diagram for the database in Figure 1.2.

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

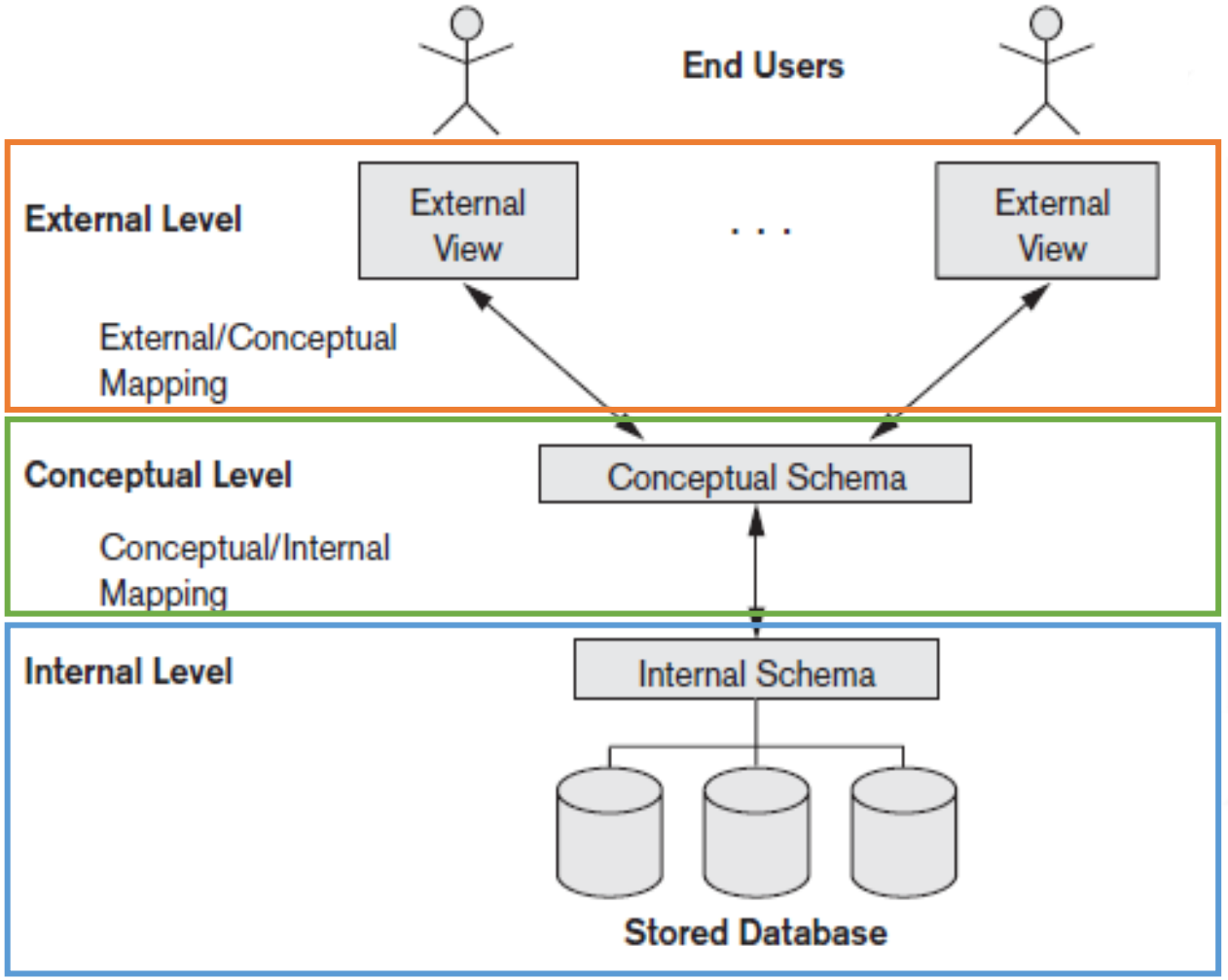
Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

The Three-Schema Architecture



- External Level
  - A number of external schemas or user views.
  - Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database.
- Conceptual Level (Logical Level)
  - Describe the structure of the whole database for a community of users.
  - Hide the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.
- Internal Level
  - Describe the physical storage structure of the database.
  - Use a physical model and describe the complete details of data storage and access paths for the database.

Data Independence

- **Logical Data Independence**:
  - The capacity to change the conceptual schema without having to change the external schemas and their associated application programs.
- **Physical Data Independence**:
  - The capacity to change the internal schema without having to change the conceptual schema.
  - For example, the internal schema may be changed when certain file structures are reorganized, or new indexes are created to improve database performance

User-Friendly DBMS Interfaces

- **Menu-based (Web-based)**
  - popular for browsing on the web
- **Forms-based**
  - designed for naïve users used to filling in entries on a form
- **Graphics-based**
  - Point and Click, Drag and Drop, etc.
  - Specifying a query on a schema diagram
- **Natural language**
  - requests in written English
- Combinations of the above
  - For example, both menus and forms used extensively in Web database interfaces

Other DBMS Interfaces

- Natural language
  - free text as a query
- Speech
  - Input query and Output response
- Web Browser with keyword search
- Parametric interfaces,
  - e.g., bank tellers using function keys.
- Interfaces for the DBA:
  - Creating user accounts, granting authorizations
  - Setting system parameters
  - Changing schemas or access paths

Database System Utilities

- To perform certain functions such as:
  - Loading data stored in files into a database. Includes data conversion tools.
  - Backing up the database periodically on tape.
  - Reorganizing database file structures.
  - Performance monitoring utilities.
  - Report generation utilities.
  - Other functions, such as sorting, user monitoring, data compression, etc.

Other Tools

- Application Development Environments and CASE (computer-aided software engineering) tools:
- Examples:
  - PowerBuilder (Sybase)
  - JBuilder (Borland)
  - JDeveloper 10G (Oracle)

Typical DBMS Component Modules

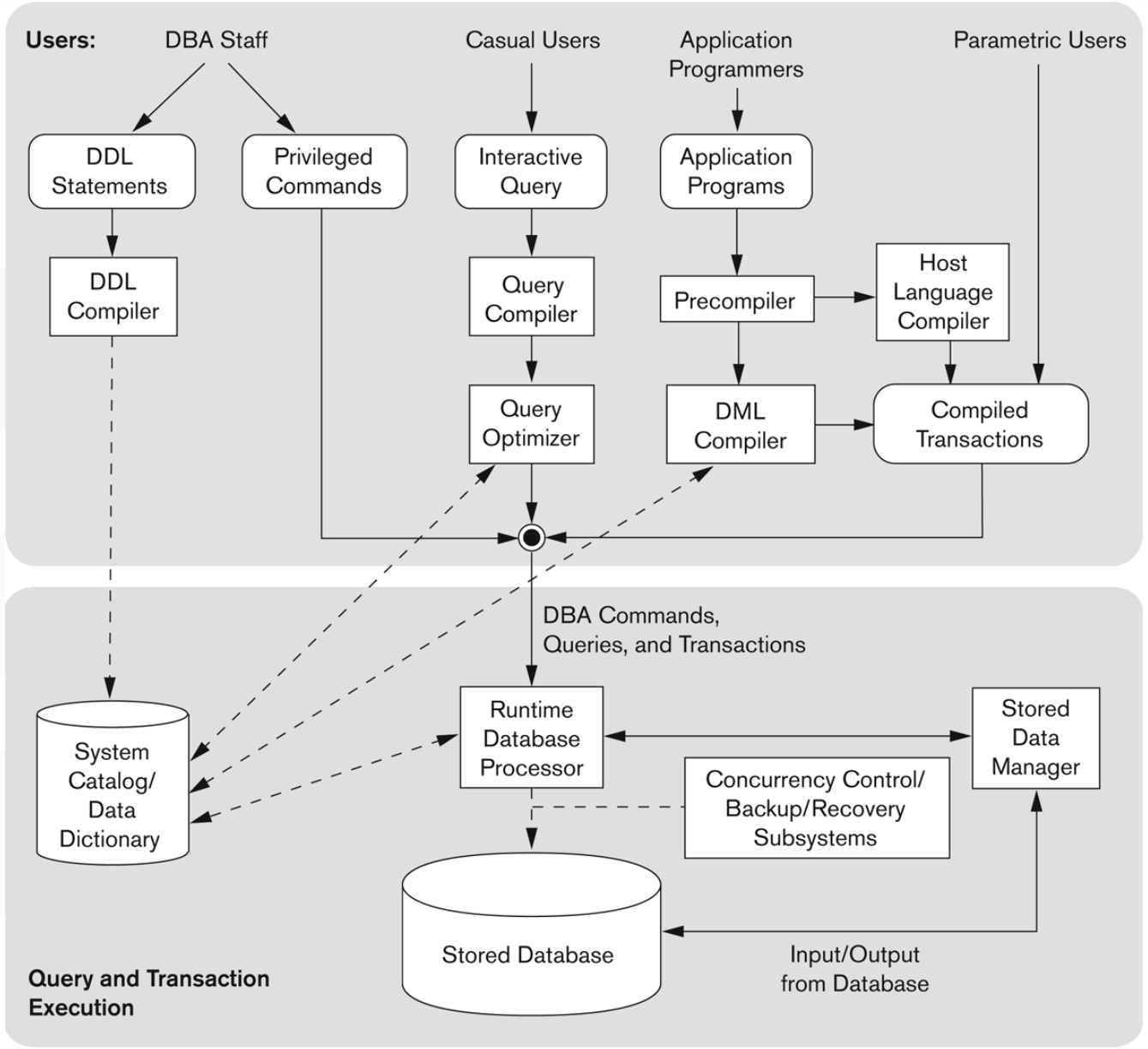


Figure 2.3
Component modules of a DBMS and their interactions.

A Physical Centralized Architecture

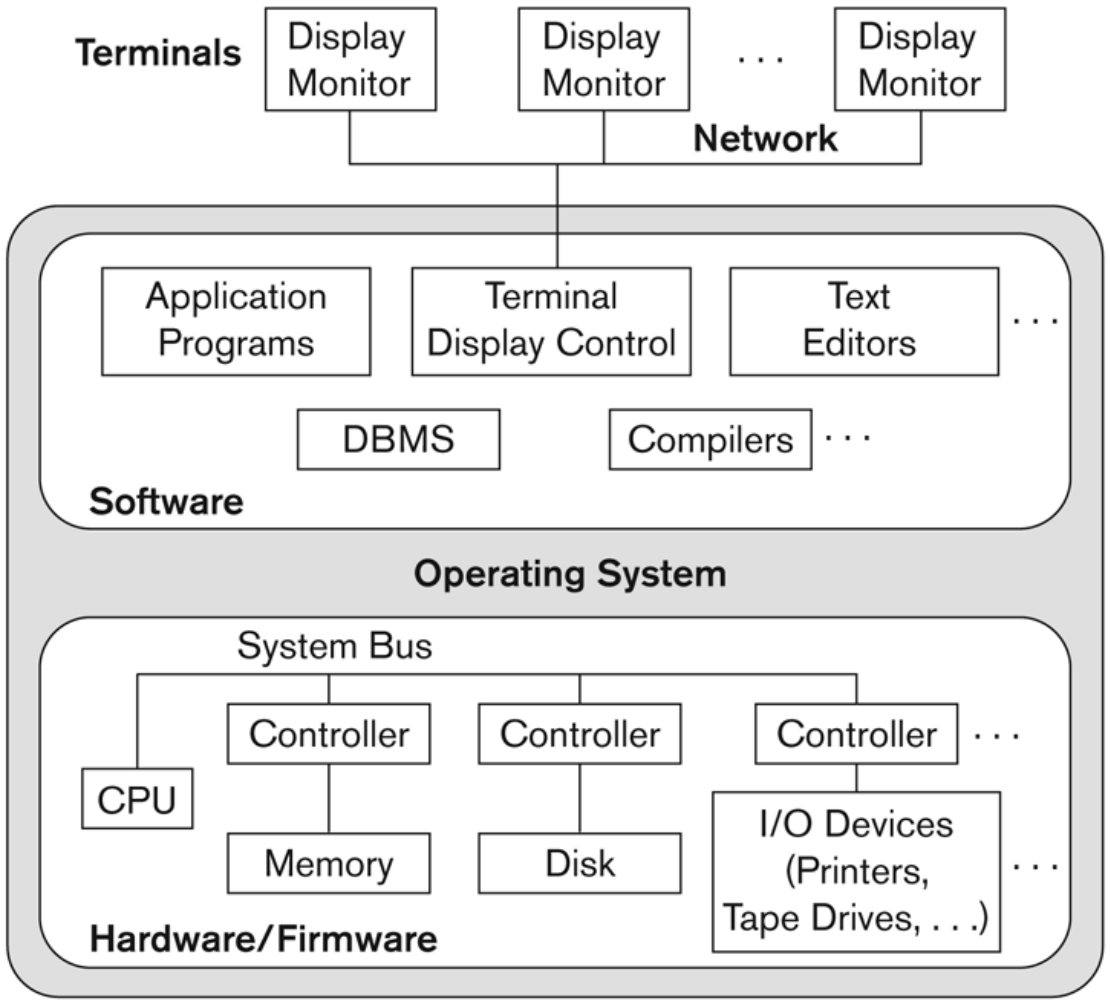
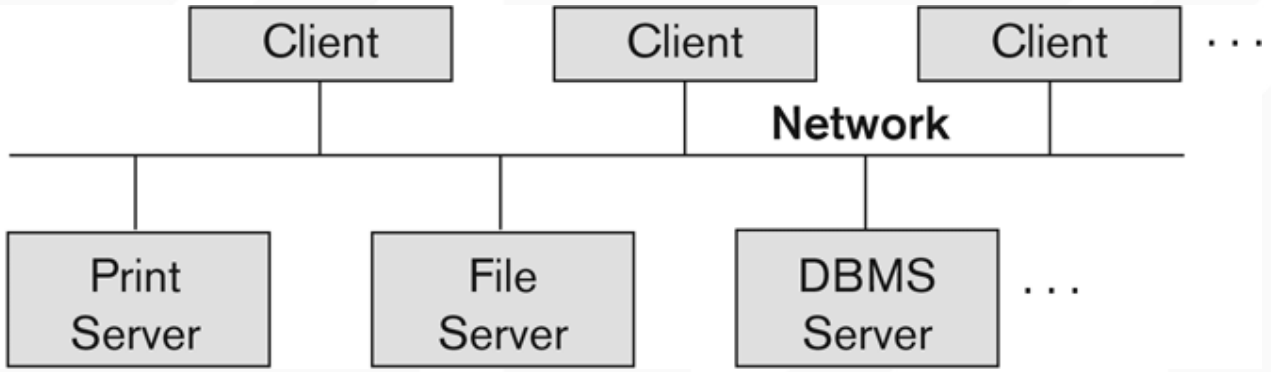


Figure 2.4
A physical centralized architecture.

- Centralized DBMS:
  - Combines everything into single system including- DBMS software, hardware, application programs, and user interface processing software.
  - User can still connect through a remote terminal – however, all processing is done at centralized site

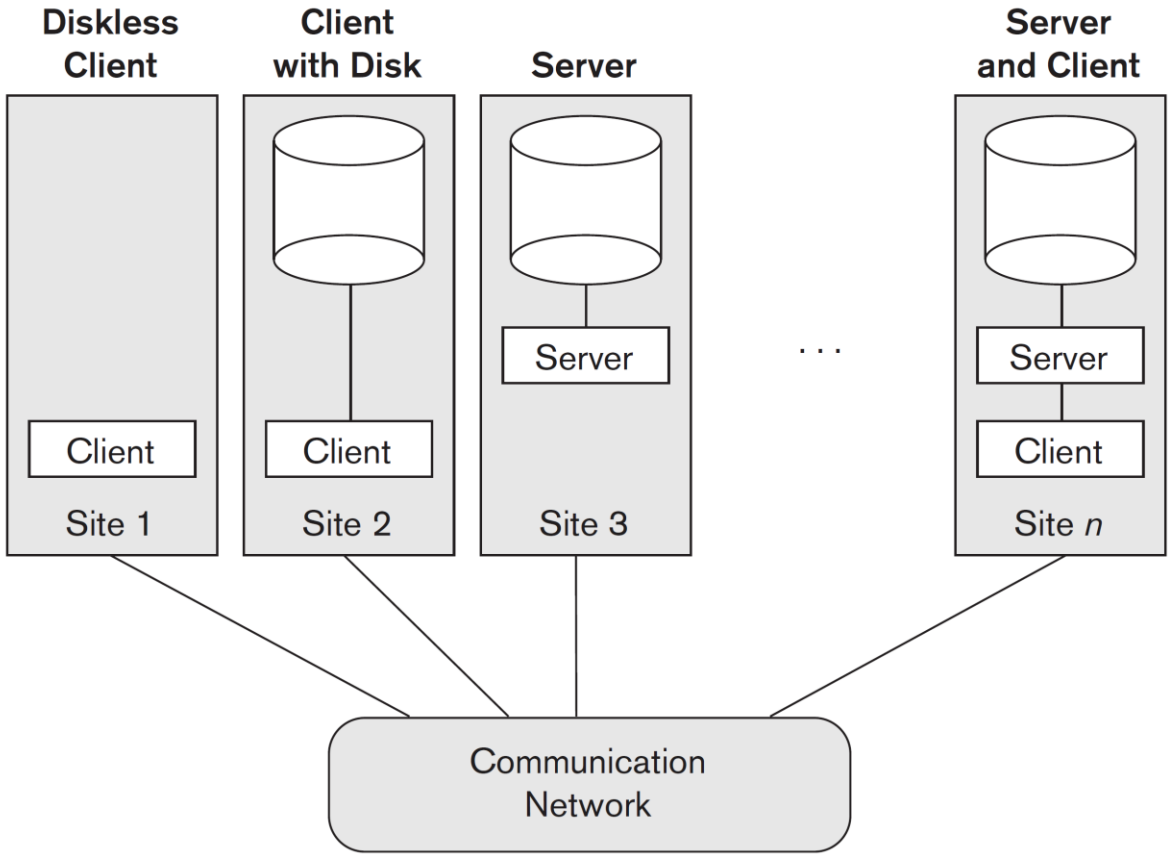
Two-tier Client/Server

Figure 2.5
Logical two-tier client/server architecture.



- Specialized Servers with Specialized functions
  - Print server
  - File server
  - DBMS server
  - Web server
  - Email server
- Clients can access the specialized servers as needed

Figure 2.6
Physical two-tier client/server architecture.



- Clients
  - Provide appropriate interfaces through a client software module to access and utilize the various server resources.
  - Clients may be **diskless machines** or **PCs** or **Workstations with disks** with only the client software installed.
  - Connected to the servers via some form of a network.
    - (LAN: local area network, wireless network, etc.)

- DBMS Server
  - Provides database query and transaction services to the clients
  - Relational DBMS servers are often called **SQL servers**, **query servers**, or **transaction servers**
  - Applications running on clients utilize an Application Program Interface (API) to access server databases via standard interface such as:
    - ODBC: Open Database Connectivity standard
    - JDBC: for Java programming access

Three-tier Client/Server

Database Server **Application Server** **Thin Client**

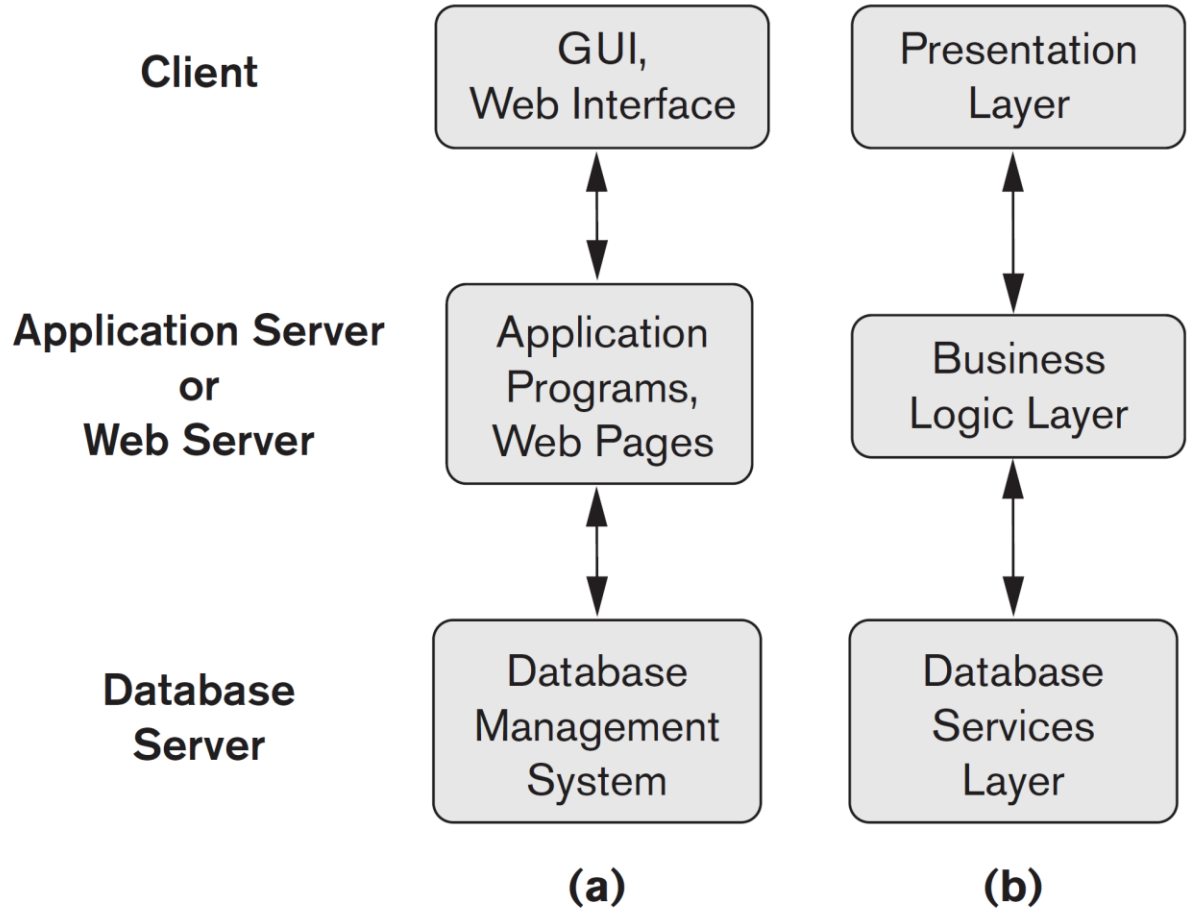


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

- Common for Web applications
- Intermediate Layer called **Application Server** or **Web Server**:
  - Stores the web connectivity software and the business logic part of the application used to access the corresponding data from the database server
  - Acts like a conduit for sending partially processed data between the database server and the client.
- Three-tier Architecture Can Enhance Security:
  - Database server only accessible via middle tier
  - Clients cannot directly access database server
  - Clients contain user interfaces and Web browsers
  - The client is typically a PC or a mobile device connected to the Web

Classification of DBMSs

- Based on the data model used
  - Legacy:
    - Network, Hierarchical.
  - Currently Used:
    - Relational, Object-oriented, Object-relational
  - Recent Technologies:
    - Key-value storage systems, NOSQL systems: document based, column-based, graph-based and key-value based. Native XML DBMSs.
- Other classifications
  - Single-user (typically used with personal computers) vs. multi-user (most DBMSs).
  - Centralized (uses a single computer with one database) vs. distributed (multiple computers, multiple DBs)

Cost considerations for DBMSs

- Cost Range:
  - from free open-source systems to configurations costing millions of dollars
- Examples of free relational DBMSs:
  - MySQL, PostgreSQL, others
- Commercial DBMS offer additional specialized modules,
  - e.g. time-series module, spatial data module, document module, XML module
  - These offer additional specialized functionality when purchased separately
  - Sometimes called cartridges (e.g., in Oracle) or blades
- Different licensing options:
  - site license, maximum number of concurrent users (seat license), single user, etc.


?
Q
&
A
!