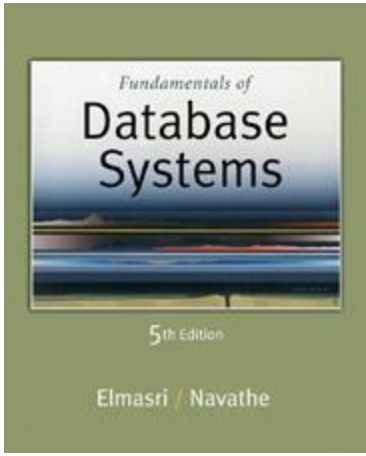
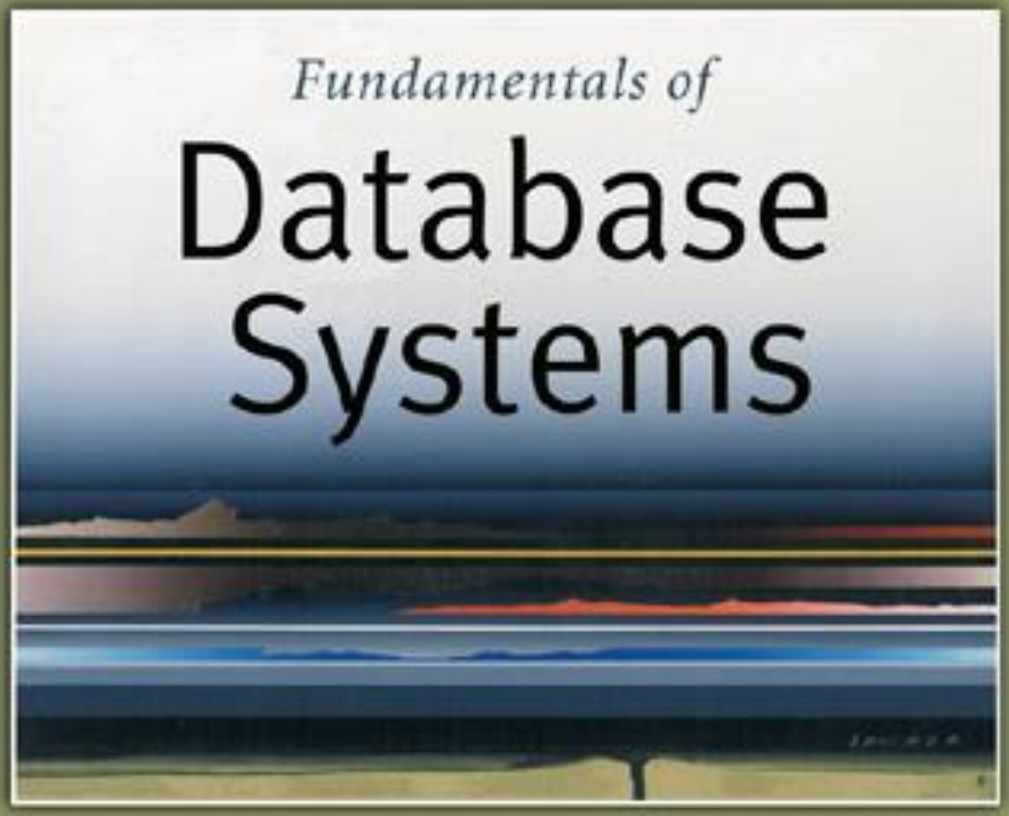
Database Management Systems

Dr. Alshaimaa abo-alian

a_alian@cis.asu.edu.eg



Lecture 2

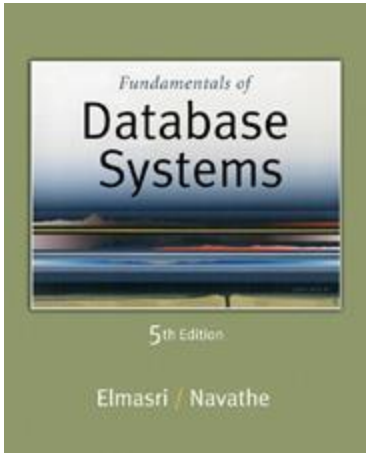


5th Edition

Elmasri / Navathe

Chapter 2

Database System Concepts and Architecture



Outline

- Data Models and Their Categories
- Schemas, Instances, and States
- Three-Schema Architecture
- Data Independence
- DBMS Languages and Interfaces
- Database System Utilities and Tools
- Centralized and Client-Server Architectures
- Classification of DBMSs

Data Models

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

Data Models (continued)

- **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) and ***user-defined operations*** (e.g. compute_student_gpa, balance_transfer)

Categories of Data Models

- **Conceptual (high-level, semantic) data models:**
  - Provide concepts that are close to the way many users perceive data.
    - (Also called ***entity-based*** or ***object-based*** data models.)
- **Physical (low-level, internal) data models:**
  - Provide concepts that describe details of how data is stored in the computer. by representing information such as record formats, record orderings, and access paths.
- **Implementation (representational) data models:**
  - Provide concepts that fall between the above two, used by many commercial DBMS implementations (e.g. relational data models used in many commercial systems).

Database Schema versus Database Instance

- Database Schema:
  - The ***description*** of a database.
  - Includes descriptions of the database structure, data types, and the constraints on the database.
- Schema Diagram:
  - An ***illustrative*** display of (most aspects of) a database schema.
  - Other aspects are not specified in the schema diagram??
- Schema Construct:
  - A ***component*** of the schema or an object within the schema, e.g., STUDENT, COURSE.

Example of a Database Schema

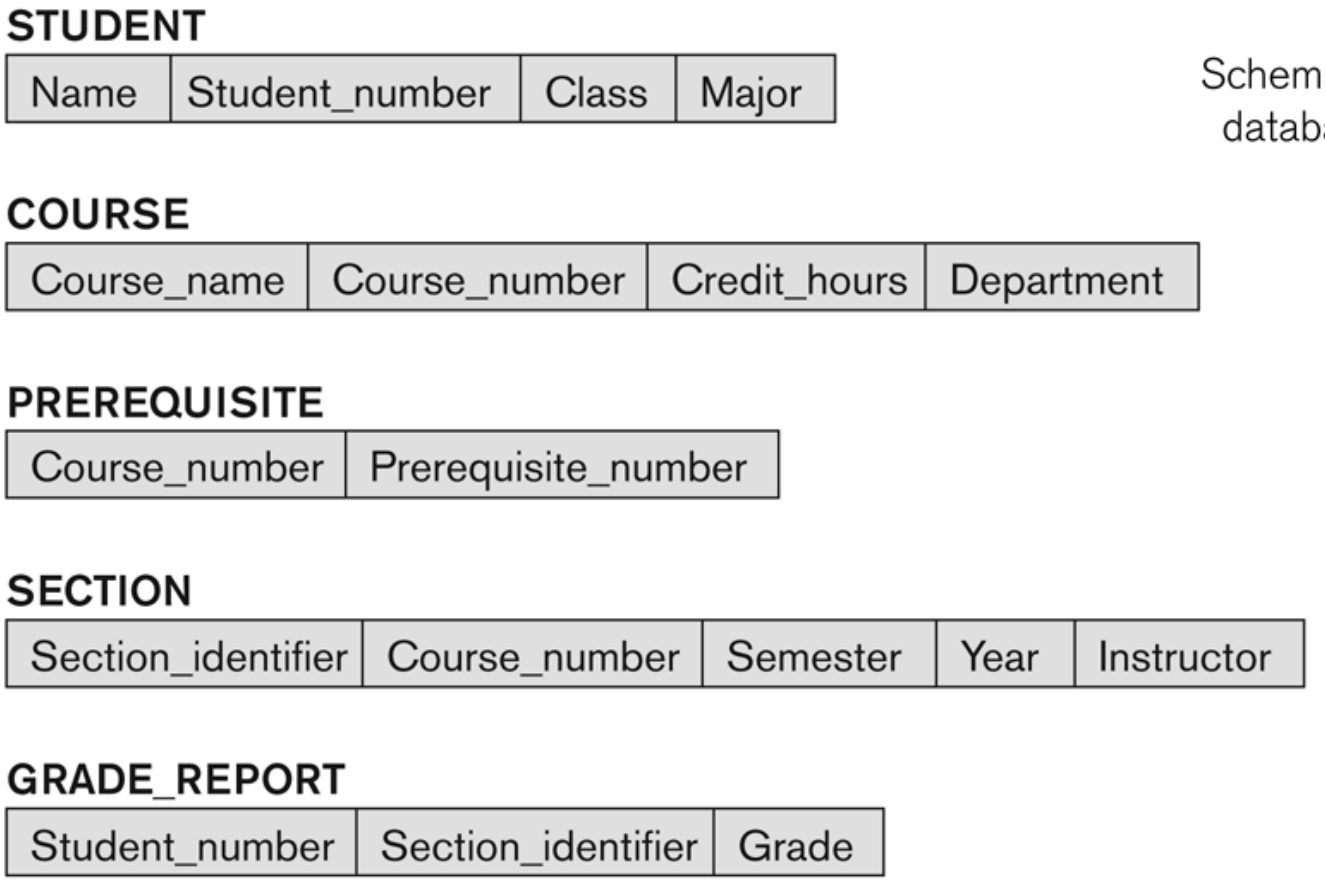


Figure 2.1
Schema diagram for the database in Figure 1.2.

Database Schema versus Database Instance

- 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, …*
- Database State:
  - Refers to the ***content*** of a database at a moment in time.

Example of a database state

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.

Database Schema vs. Database State

- Empty Database State **(***with no data)*:
  - When we define a new database, we specify its database schema only to the DBMS**.**
- Initial Database State:
  - Refers to the database state when it is initially loaded with the initial data into the system.
- Valid State:
  - A state that satisfies the structure and constraints of the database.
  - DBMS responsibility: stores the descriptions of the schema constructs and constraints (**meta-data)**—in the DBMS catalog so that it can refer to the schema whenever it needs to.

Database Schema vs. Database State

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

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

Three-Schema Architecture

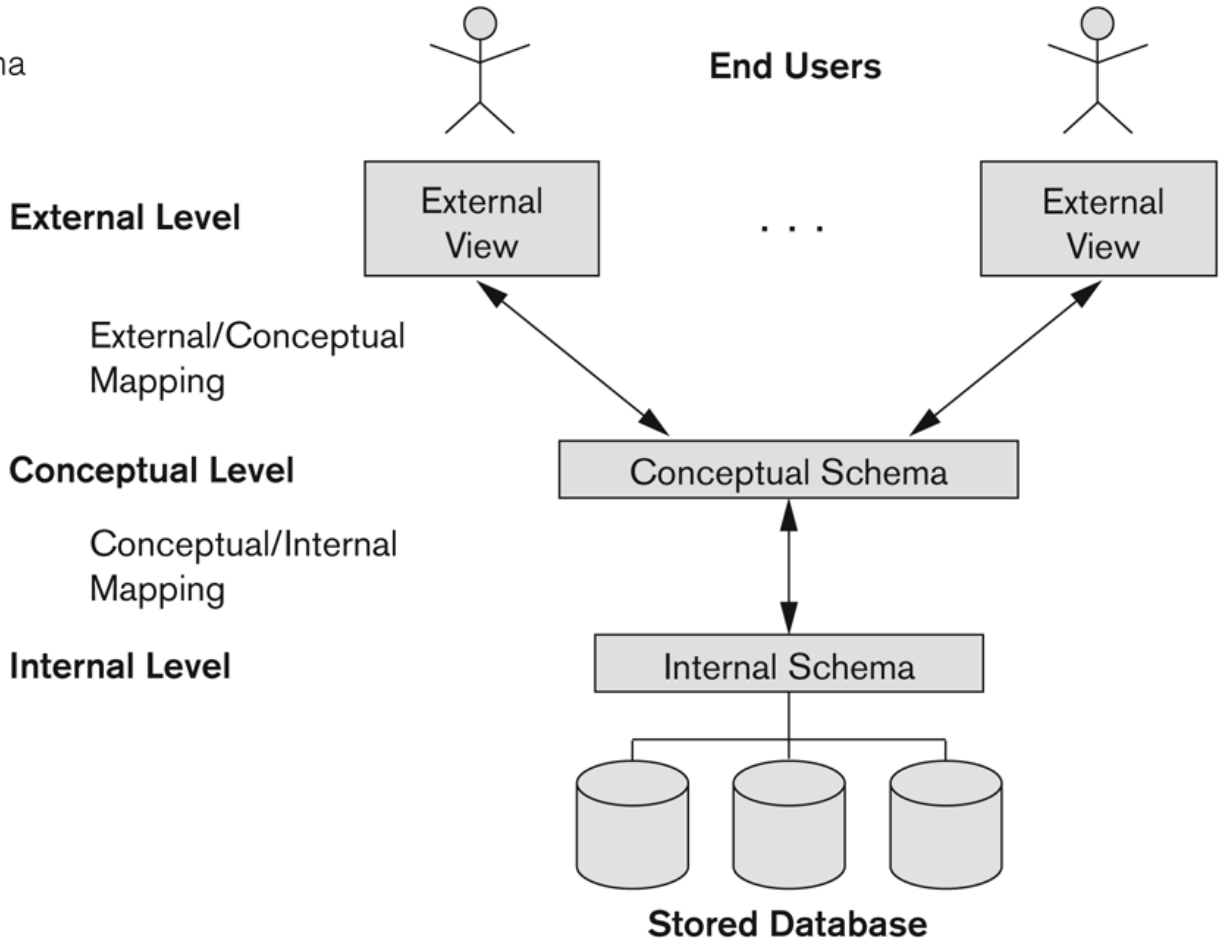
- Proposed to support DBMS characteristics of:
  - **Program-data independence.**
  - Support of **multiple views** of the data.
- Not explicitly used in commercial DBMS products, but has been useful in explaining database system organization
- The goal of the three-schema architecture is to separate the user applications from the physical database

Three-Schema Architecture

- Defines DBMS schemas at ***three*** levels:
  - **Internal schema** at the internal level to describe physical storage structures and access paths (e.g indexes).
    - Typically uses a **physical** data model.
  - **Conceptual schema** at the conceptual level to describe the structure and constraints for the whole database for a community of users.
    - Uses a **conceptual** or an **implementation** data model.
  - **External schemas** at the external level to describe the various user views.
    - Usually uses the same data model as the conceptual schema.
    - Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

The three-schema architecture

Figure 2.2
The three-schema architecture.



Data Independence

- The capacity to change the schema at one level of a database system without having to change the schema at the next higher level.
- **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 (ex. improve retrieval speed)

DBMS Languages

- The DBMS must provide appropriate languages and interfaces for each category of users
  - Data Definition Language (DDL)
  - Data Manipulation Language (DML)

DBMS Languages

- **Data Definition Language (DDL):**
  - Used by the DBA and database designers to specify the conceptual schema of a database.
  - In many DBMSs, the DDL is also used to define internal and external schemas (views).
  - In some DBMSs, separate **storage definition language (SDL)** and **view definition language (VDL)** are used to define internal and external schemas.
    - SDL is typically realized via DBMS commands provided to the DBA and database designers

DBMS Languages

- **Data Manipulation Language (DML):**
  - Used to specify database retrievals and updates (insertion, deletion, and modification of the data)
  - DML commands (**data sublanguage)** can be *embedded* in a general-purpose programming language (**host language**), such as C++, or Java.
    - A library of functions can also be provided to access the DBMS from a programming language
  - Alternatively, **stand-alone** DML commands can be applied directly (called a ***query language***).

Types of DML

- **High Level or Non-procedural Language:**
  - For example, the SQL relational language
  - Are “set”-oriented and specify what data to retrieve rather than how to retrieve it.
  - Also called **declarative** languages.
- **Low Level or Procedural Language:**
  - Retrieve data one record-at-a-time;
  - Constructs such as looping are needed to retrieve multiple records, along with positioning pointers.

DBMS Interfaces

- Stand-alone query language interfaces
  - Example: Entering SQL queries at the DBMS interactive SQL interface (e.g. SQL*Plus in ORACLE)
- Programmer interfaces for embedding DML in programming languages
- User-friendly interfaces
  - Menu-based, forms-based, graphics-based, etc.
- Interfaces for the DBA:
    - Creating user accounts, granting authorizations
    - Setting system parameters
    - Changing schemas or access paths

Database System Utilities

- Database utilities that help the DBA manage the database system to perform certain functions such as:
  - Loading data stored in files into a database. Includes data conversion tools.
  - **Backing** up the database periodically on another storage medium.
  - Report generation utilities.
  - Performance monitoring utilities.
  - Other functions, such as sorting, user monitoring, data compression, etc.
  - Data dictionary / repository:
    - Used to store schema descriptions, constraints and other information such as application program descriptions, user information, etc

Centralized and Client-Server DBMS Architectures

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

A Physical Centralized Architecture

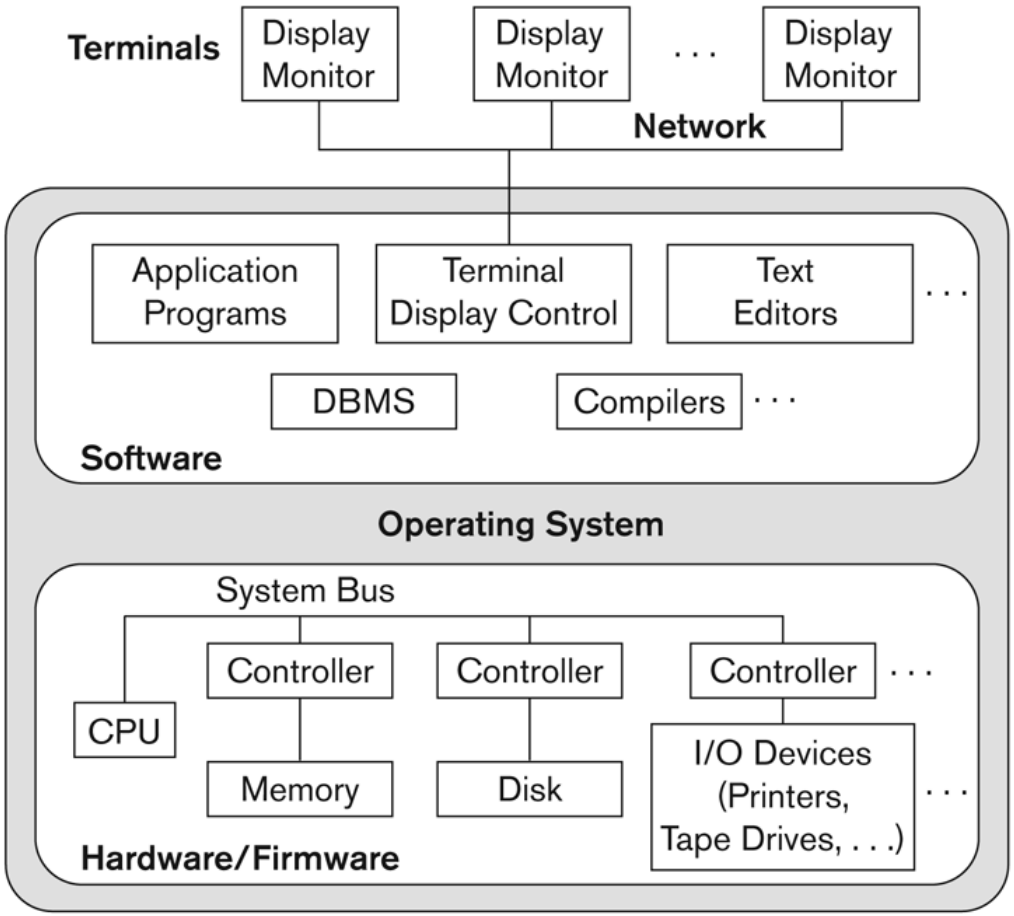


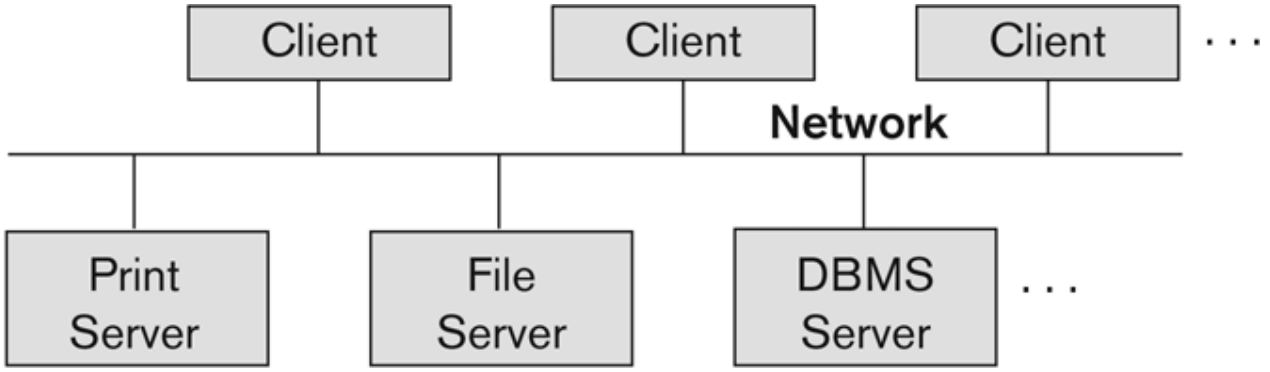
Figure 2.4
A physical centralized architecture.

Basic 2-tier Client-Server Architectures

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

Logical two-tier client server architecture

Figure 2.5
Logical 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, transaction servers, or data 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
- Client and server must install appropriate client module and server module software for ODBC or JDBC
- See Chapter 9

Physical two-tier client server architecture

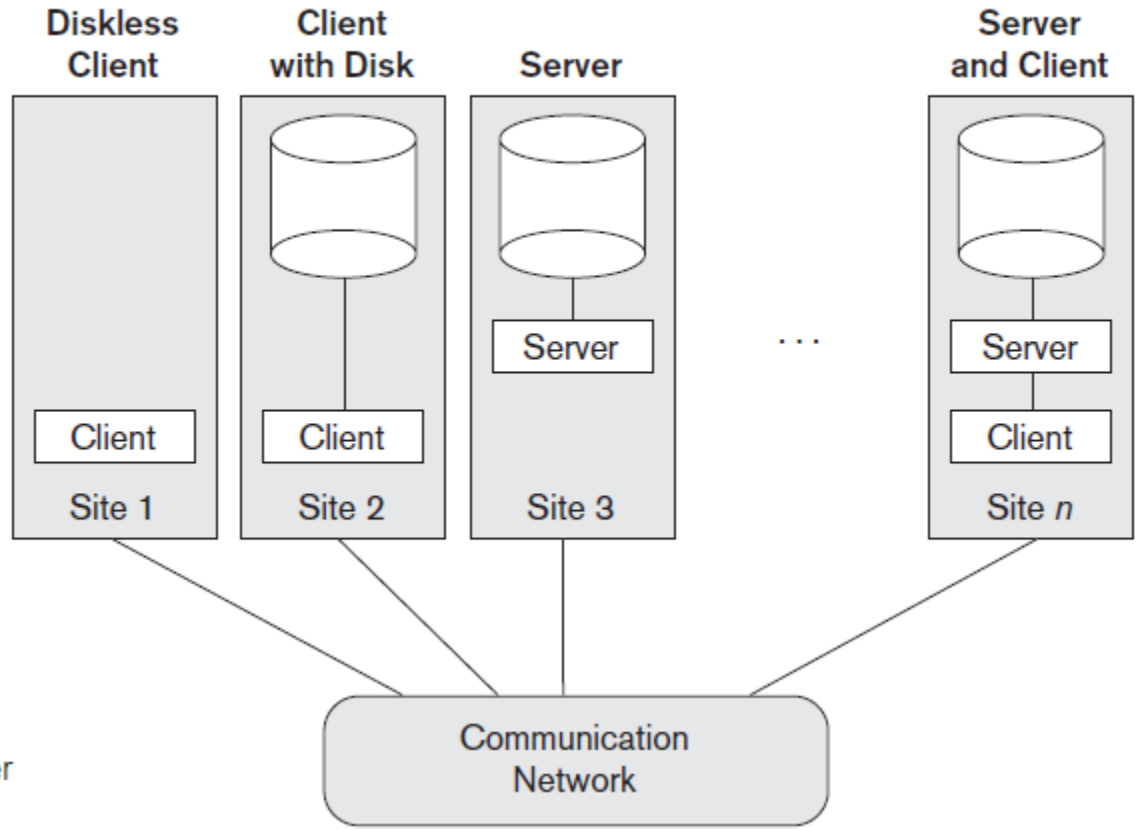


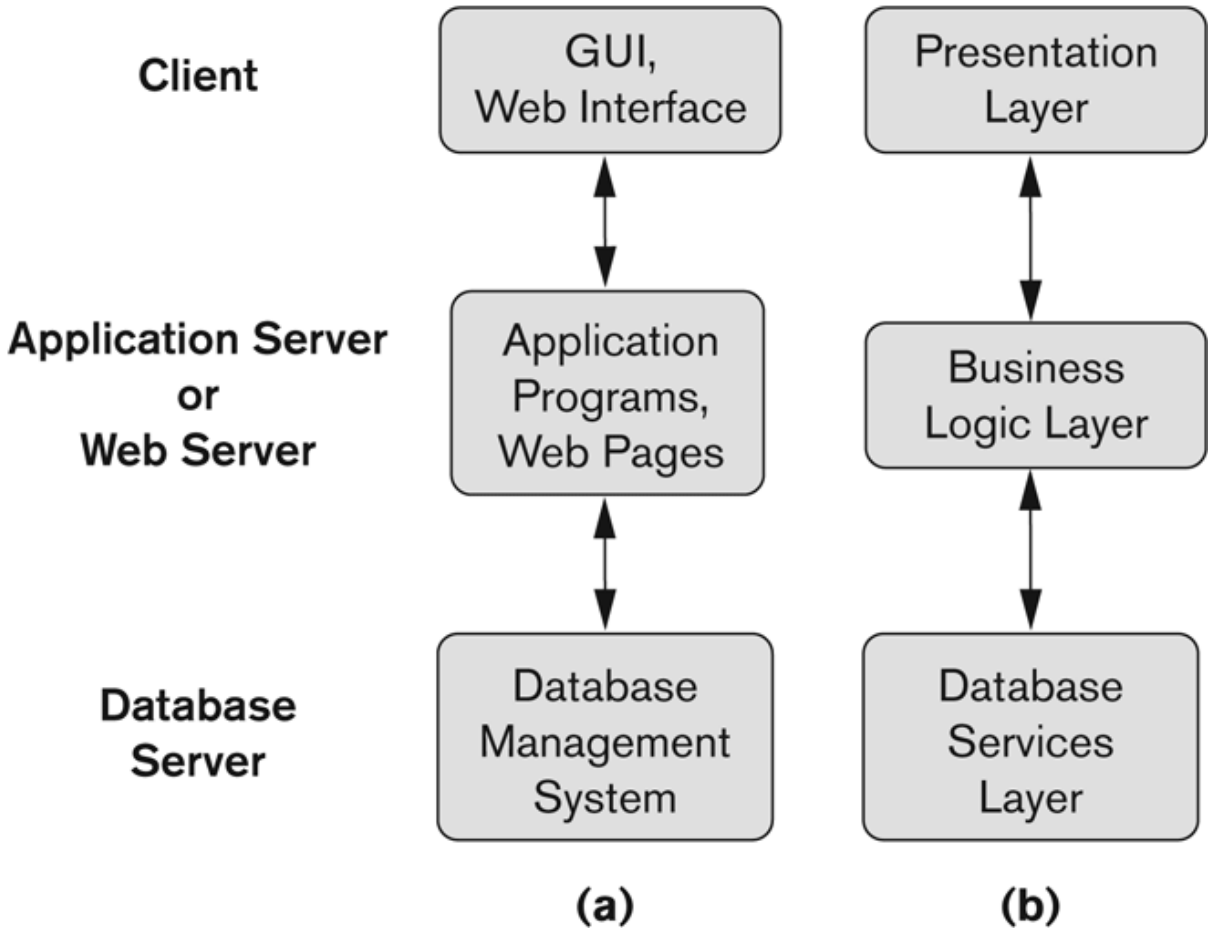
Figure 2.6
Physical two-tier client/server architecture.

Three Tier Client-Server Architecture

- 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 channel 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

Three-tier client-server architecture

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



Classification of DBMSs

We can classify DBMSs according to several criteria:

1. Number of users
   - Single-user (typically used with personal computers)
     vs. multi-user (most DBMSs).
2. Number of sites
   - Centralized (data is stored at a single computer site)
     vs. distributed (database & DBMS software are distributed over many sites)
     - Homogeneous DDBMS *(same DBMS software at all the sites) vs.* Heterogeneous DDBMS

Classification of DBMSs

We can classify DBMSs according to several criteria:

3. Data model
   - Relational, object, Network, Hierarchical
4. Cost.
   - open source (free) DBMS products like MySQL
   - License-based like Oracle

Summary

- Data Models and Their Categories
- Schemas, Instances, and States
- Three-Schema Architecture
- Data Independence
- DBMS Languages and Interfaces
- Database System Utilities and Tools
- Centralized and Client-Server Architectures
- Classification of DBMSs