Introduction to Databases

Aims:

At the end of this group of two lectures you should be able to understand basic data management operations and ways of storing data on computers. You should be able to understand in general how databases work

Reading:

Elmasri & Navathe, 7th edition, 2016

Chapters 1 & 2

Overview

1. Data management
   - Data requirements
   - Data management operations
2. Computer data organization
3. Database systems
4. Data models
5. DBMS
6. Advantages and disadvantages of databases

Data management

- Entity
- Attribute
- Data
- Record

Data requirements

Data management operations

Overview

1. Data management
2. Computer data organization
   - Files
   - Databases
3. Database systems
4. Data models
5. DBMS
6. Advantages and disadvantages of databases

Files

- Unstructured
- Structured
  - Record
  - Field
    - Alphanumeric
    - Numeric
    - Currency
    - Date
    - Memo
    - …
  - Key

Field name	Start position	Length
Name	1	30
StudNo	31	4
Major	35	4

Conventional data processing

- Several programs for each application
- One or more files for each application
- Programmers must write their own programs for accessing, inserting and updating data in files

Problems with processing data stored in files

- Separation and isolation of data
- Data duplication (redundancy)
- Data integrity
- Data availability
- Maintenance
- Programs are dependent on data

Database

- A collection of related data
- **Mini-world**: A part of the real world about which data is stored in a database
- A typical database
  - Represents some aspects of the mini-world
  - Is used for specific purposes
  - By one or more groups of users

Overview

1. Data management
2. Computer data organization
3. Database systems
   - Organization of databases
   - Database features
   - People involved with database systems
4. Data models
5. DBMS
6. Advantages and disadvantages of databases

Databases

- **Database Management System (DBMS)**: A software system to facilitate the creation and maintenance of a computerized database
- **Database Software = DBMS + Applications**
- **Database System = Database + Software**

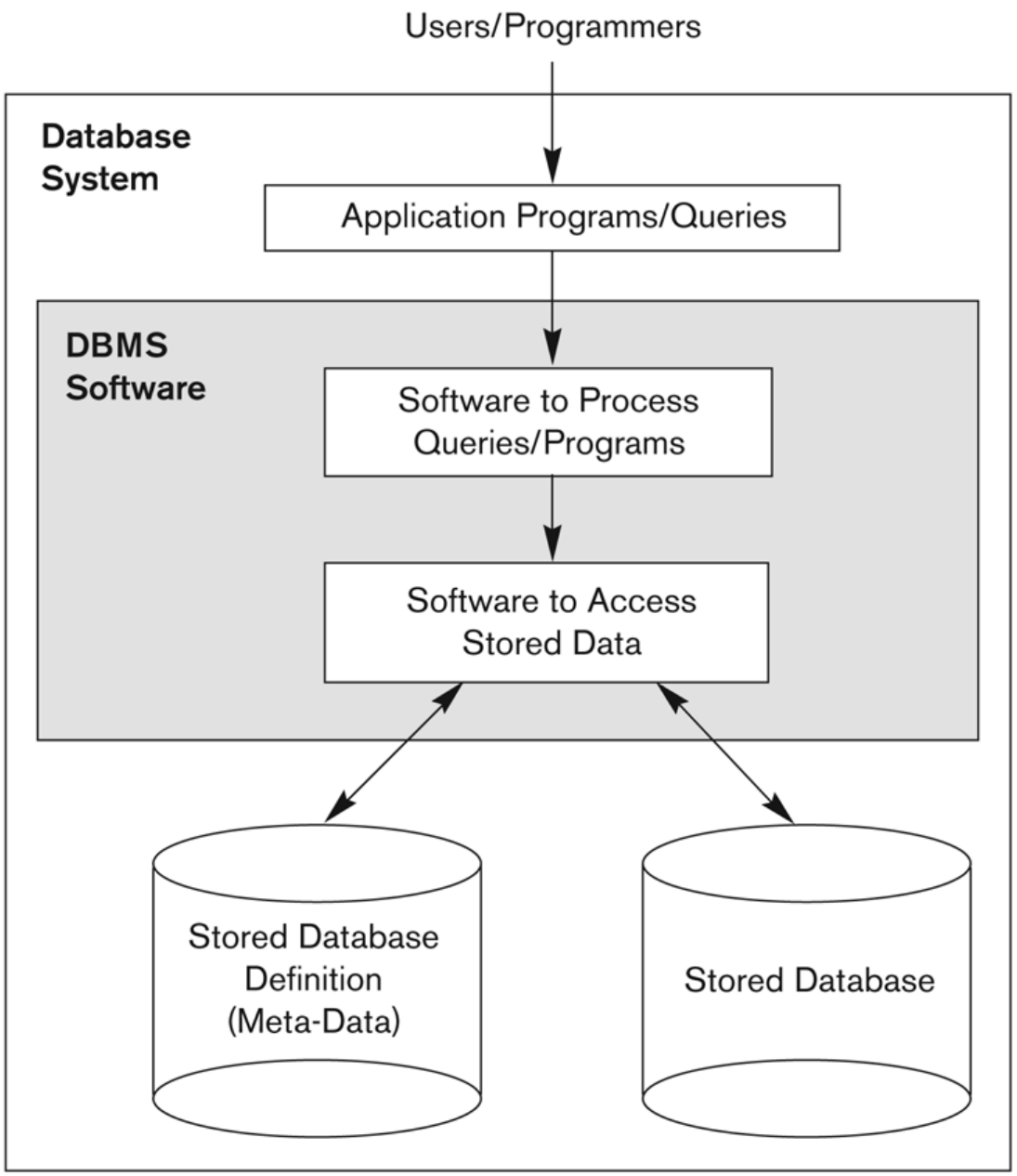


Figure 1.1
A simplified database system environment.

Example Database

- **Mini-world for the example**: Part of a UNIVERSITY environment
- **Some mini-world *entities***:
  - Students
  - Courses
  - Sections (of courses)
  - Departments
  - Instructors

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.

Example Database (cont.)

- **Some mini-world *relationships*:**
  - Sections *are of* specific Courses
  - Students *take* Courses
  - Courses *have* prerequisite Courses
  - Instructors *teach* Sections
  - Courses *are offered by* Departments
  - Students *major in* Departments

Typical DBMS Functionality

- Define a database: in terms of data types, structures and constraints
- Construct or load the database on a secondary storage medium
- Manipulating the database: querying, generating reports, insertions, deletions and modifications to its content
- Concurrent processing and sharing by a set of users and programs – yet, keeping all data valid and consistent
- Protection or security measures to prevent unauthorized access

Types of Databases and Database Applications

- Numeric and textual databases
- Multimedia databases
- Geographic Information Systems (GIS)
- Data Warehouses
- Real-time and active databases

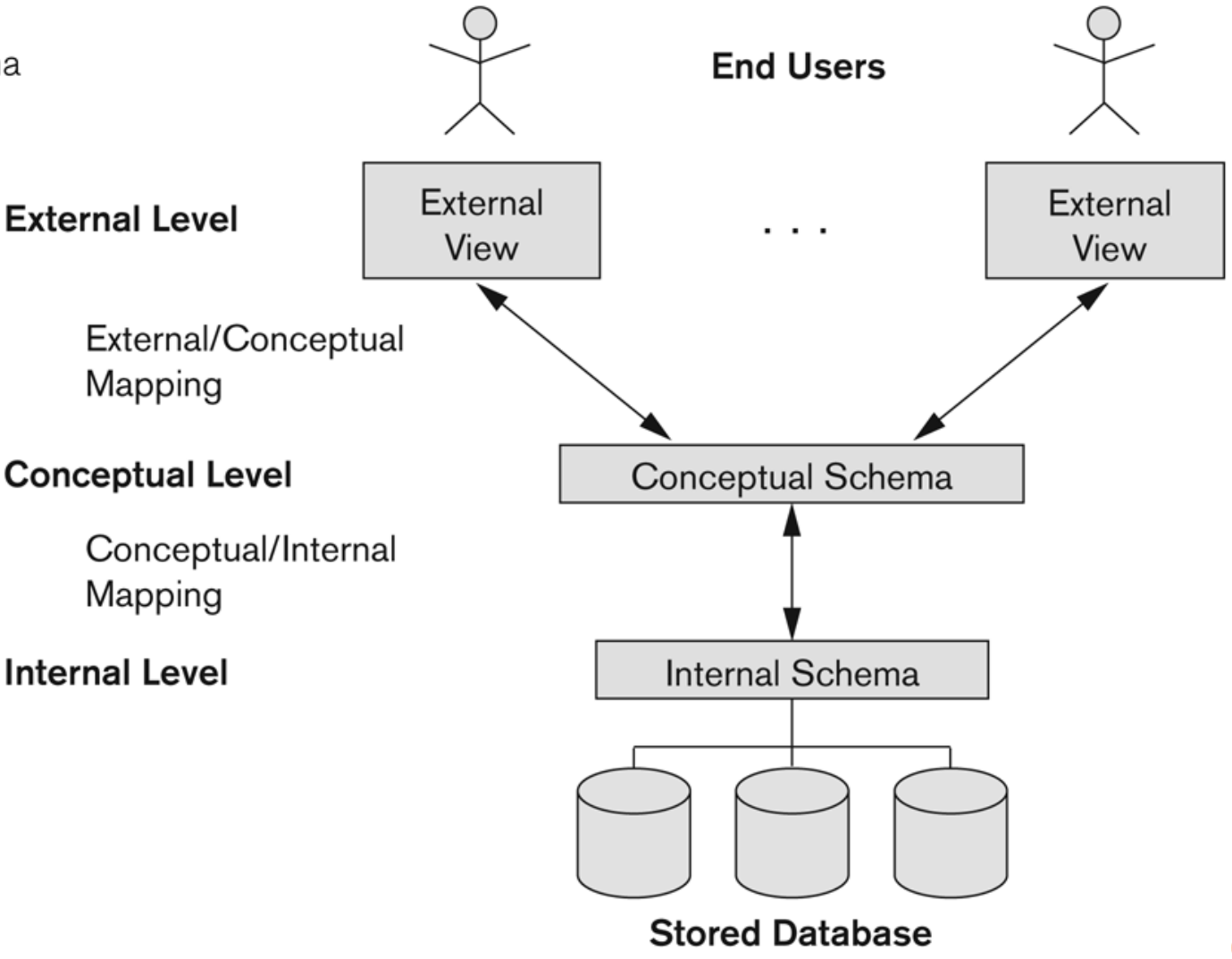
A number of these databases and applications are described in the textbook

Organization of databases

Three schema architecture

- External level
- Conceptual level
- Internal level

Figure 2.2
The three-schema architecture.



Database features

- **Data catalog (dictionary)**

  Description of the database (**meta-data**)

  Allows DBMS to work with different DBs
- **Data abstraction**
- **Data integrity**
- **Data independence**
  - Logical data independence
  - Physical data independence

Database features (cont.)

- Support of multiple views of the data
- Sharing of data (concurrency control)
- Controlled data redundancy
- Query languages
- Interfaces
- Backup and recovery
- Utilities (creating files, monitoring, …)

People involved with DBs

- Actors on the scene
  - Database administrators
  - Database designers
  - End-users
- Workers behind the scene
  - DBMS designers and implementers
  - Tool developers
  - Operators and maintenance personnel

Actors on the scene: DBA

- Creates the database
- Creates reports
- Grants permit for access
- Maintains storage structures and the catalog
- Monitors database activities
- Performs backup and recovery
- Tunes database performance

End-users

- Casual
- Naïve/Parametric ("canned transactions")
- Sophisticated
- Stand-alone

Overview

1. Data management
2. Computer data organization
3. Database systems
4. Data models
   - Components of data models
   - Categories of data models
5. DBMS
6. Advantages and disadvantages of databases

Data model

- A set of concepts (formalism) used to describe the structure of a database
- We need a tool that permits us to model the real world in a way that the achieved representation is abstract enough to suppress unnecessary details, however, at the same time is detailed enough to express all necessary facts and relationships (from the application's point of view).

Schemas versus Instances

- **Database schema**: The *description* of a database (database intension)
- **Schema diagram**: A diagrammatic display of (some aspects of) a database schema
- **Database instance**: The actual data stored in a database at a *particular moment in time*
- **Database state (extension, occurrence)**

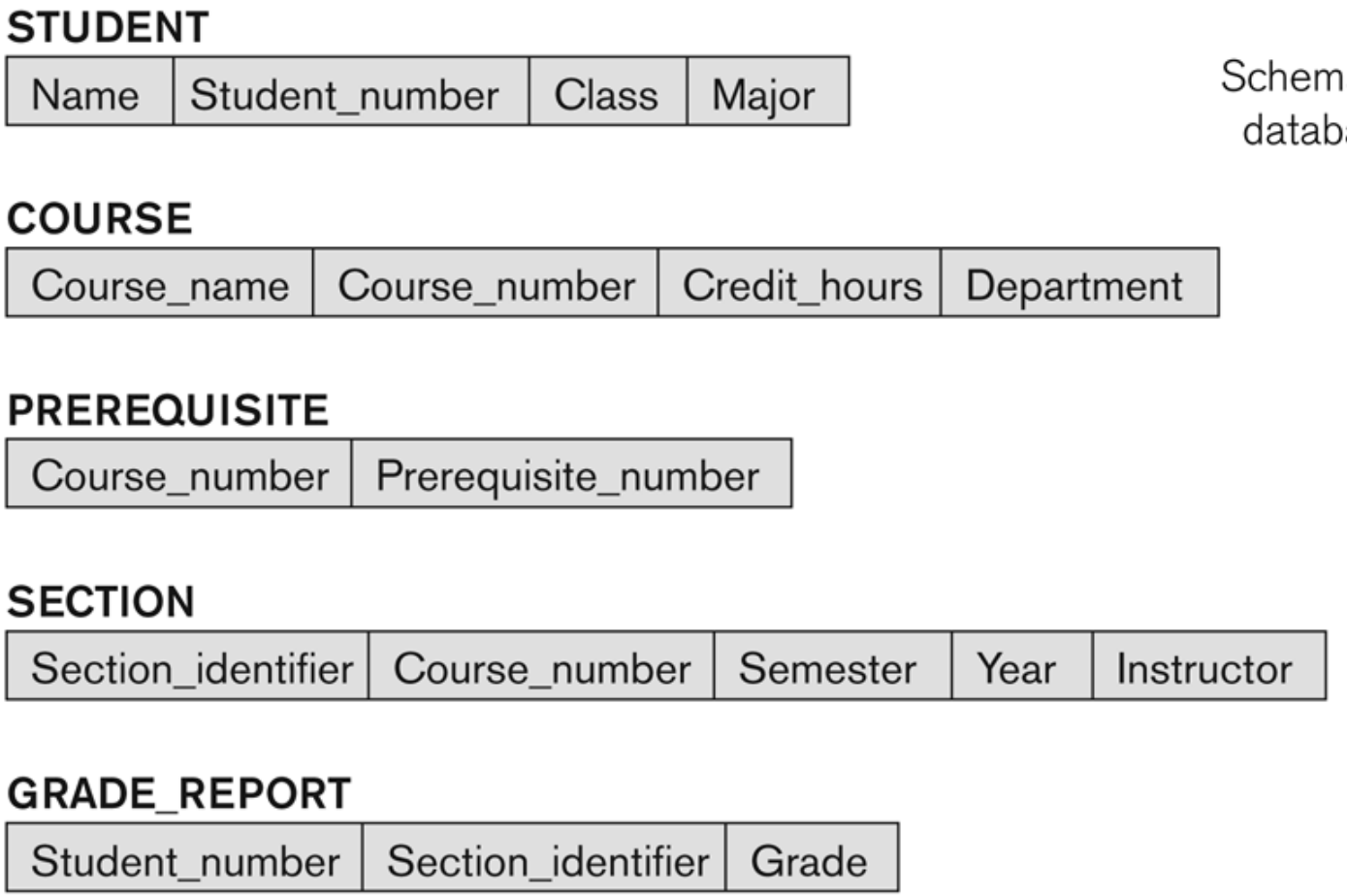


Figure 2.1
Schema diagram for the database in Figure 1.2.

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.

Schema vs. Database State

- **Database state:** Refers to the content of a database at a moment in time
- **Initial database state:** Refers to the database when it is loaded
- **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*.

Components of data models

- Structural component
- Integrity component
- Operational component

Categories of data models

- **Conceptual (high-level, semantic)**
  Provide concepts that are close to the way many users *perceive* data
- **Physical (low-level, internal)**
  Provide concepts that describe details of how data is stored in the computer.
- **Implementation (representational)**
  Provide concepts that fall between the above two, balancing user views with some computer storage details.

History of Data Models

- **Network Model**: Honeywell 1964-65 (IDS System). IDMS, VAX -DBMS
- **Hierarchical Data Model**: IBM 1965. Resulted in the IMS family of systems. The most popular model.
- **Relational Model**: proposed in 1970 by E.F. Codd (IBM), first commercial system in 1981-82. Now in several commercial products (DB2, ORACLE, SQL Server, SYBASE, INFORMIX).
- **Object-oriented Data Models**: OO languages such as C++ (OBJECTSTORE, VERSANT), and Smalltalk (GEMSTONE). Also O_2, ORION, IRIS.
- **Object-Relational Models**: Most recent trend. Informix Universal Server, Oracle-10i, DB2, SQL Server

Overview

1. Data management
2. Computer data organization
3. Database systems
4. Data models
5. DBMS
   - DBMS languages
   - DBMS Interfaces
   - DBMS Components
   - Utilities
   - History
   - Architectures
   - Classification
6. Advantages and disadvantages of databases

DBMS LANGUAGES

- Data Definition Language (DDL)
- View Definition Language (VDL)
- Storage Definition Language (SDL)
- Data Manipulation Language (DML)
  - High-level (non-procedural) languages (set-at-a-time)
  - Low-level (procedural) languages (record-at-a-time)

DBMS Interfaces

- Stand-alone query language interfaces
- Programmer interfaces
- User-friendly interfaces:
  - Menu-based
  - Forms-based
  - Graphics-based
  - Natural language
  - Combinations of the above

Other DBMS Interfaces

- Speech as Input and Output
- Web Browser as an interface
- Parametric interfaces (e.g., bank tellers) using function keys.
- Interfaces for the DBA:
  - Creating accounts, granting authorizations
  - Setting system parameters
  - Changing schemas or access path

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.
  - *Report generation* utilities.
  - *Performance monitoring* utilities.
  - Other functions, such as *sorting*, *user monitoring*, *data compression*, etc.

Other Tools

- **Data dictionary / repository**
  - Used to store schema descriptions and other information such as design decisions, application program descriptions, user information, usage standards, etc.
  - *Active* data dictionary is accessed by DBMS software and users/DBA.
  - *Passive* data dictionary is accessed by users/DBA only.

- **Application Development Environments and CASE tools**

  Examples – Power builder (Sybase), Builder (Borland)

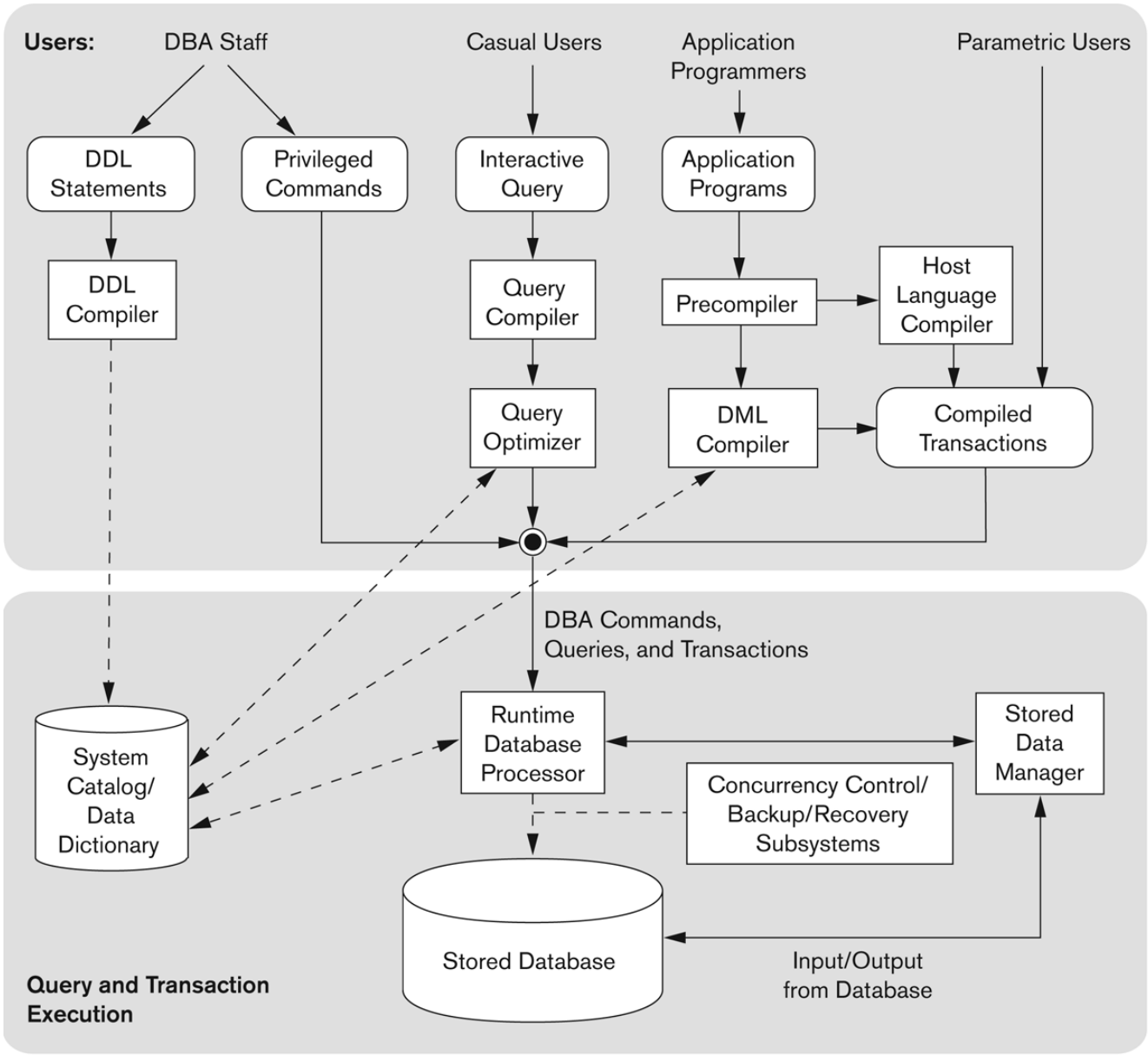


Figure 2.3
Component modules of a DBMS and their interactions.

History

- **1st generation**: File systems (1950s)
- **2nd generation**: Hierarchical and Network Models (1960s)

  Bulk of the worldwide DB processing still occurs using these models.
- **3rd generation**: Relational Model (1980s)
- **4th generation**: Post-relational (1990s-)

  OO and object-relational

  Data on the Web and E-commerce

Architectures

- Centralized DBMS
- Client-server architecture
  - Specialized Servers
    - File servers
    - Printer Servers
    - Web Servers
    - E-mail Servers
  - Clients
  - DBMS Server

Clients

- Provide appropriate interfaces and a client-version of the system to access and utilize the server resources.
- Clients maybe diskless machines or PCs or workstations with disks with only the client software installed.
- Connected to the servers via a network.

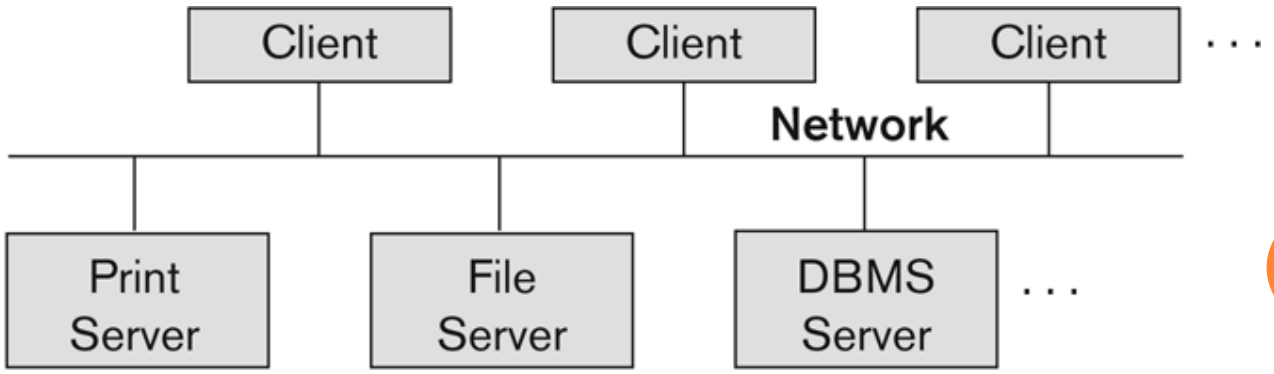


Figure 2.5
Logical two-tier client/server architecture.

Two Tier Client-Server Architecture

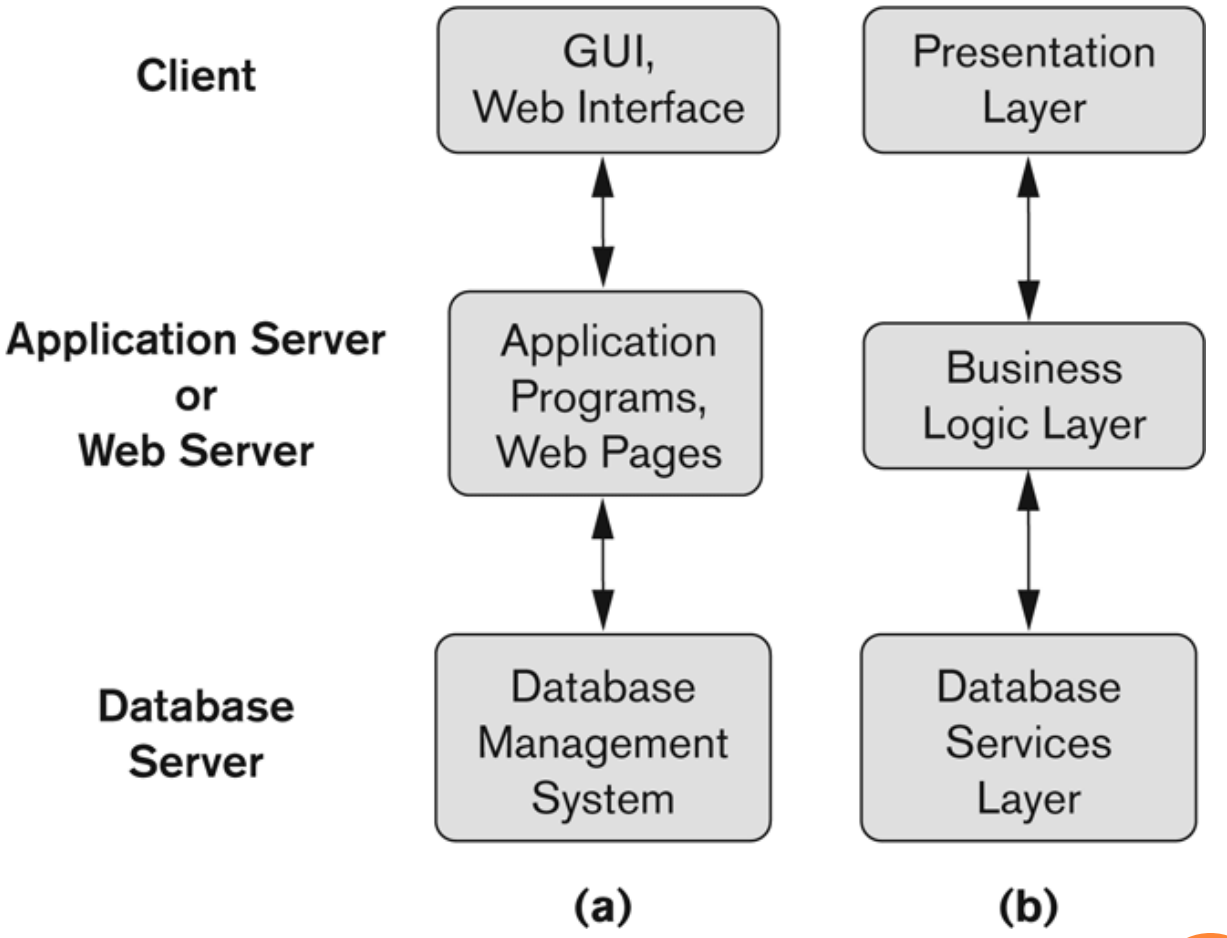
- Client side: User interface and application programs
- ODBC provides an API which allows the client side programs to call the DBMS
- Client may connect to several DBMSs
- Other variations of clients: data servers

  More functionality transferred to clients (data dictionary functions, optimization and recovery across multiple servers, etc)

Three Tier Client-Server Architecture

- Common for Web applications
- Intermediate Layer: **Application (Web) Server**
  - stores the web connectivity software and **the rules and business logic** of the application used to access the right amount of data from the database server
  - acts like a conduit for sending partially processed data between the database server and the client.
- Additional Features - Security:
  - encrypt the data at the server before transmission
  - decrypt data at the client

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



Classification of DBMSs

- Based on the data model used:
  - Traditional: Relational, Network, Hierarchical.
  - Emerging: Object-oriented, Object-relational.
- Other classifications:
  - Single-user (typically used with micro- computers) vs. multi-user (most DBMSs).
  - Centralized (uses a single computer with one database) vs. distributed (uses multiple computers, multiple databases)

Distributed DBMSs

- Homogeneous DDBMS
- Heterogeneous DDBMS
- Federated or Multidatabase Systems

Overview

1. Data management
2. Computer data organization
3. Database systems
4. Data models
5. DBMS
6. Advantages and disadvantages of databases

Advantages of DB Approach

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

Extending Database Capabilities

- New functionality is being added to DBMSs in the following areas:
  - Scientific Applications
  - Image Storage and Management
  - Audio and Video data management
  - Data Mining
  - Spatial data management
  - Time Series and Historical Data Management

The above gives rise to new research and development in incorporating new data types, complex data structures, new operations and storage and indexing schemes in database systems.

Disadvantages of DB approach

- Complexity
- Size
- Cost
- Additional hardware cost
- Performance
- Higher impact of failure

When not to use a DBMS

- Small data set, well defined, not expected to change
- Strict real-time requirements
- Access by multiple users is not required
- When no DBMS may suffice
  - Complex data
  - Special operations required