Chapter 2: Concepts and Architecture

CS-6360 Database Systems

Chris Irwin Davis, Ph.D.

Email: cid021000@utdallas.edu
Phone: (972) 883-3574
Office: ECSS 4.705

Chapter 2 Outline

- 2.1 Data Models, Schemas, and Instances
- 2.2 Three-Schema Architecture and Data Independence
- 2.3 Database Languages and Interfaces
- 2.4 The Database System Environment
- 2.5 Centralized and Client/Server Architectures for DBMSs
- 2.6 Classification of Database Management Systems

2.1 Data Models, Schemas, and Instances

Data Models and Schemas

- **Data model**
  - One fundamental characteristic of the database approach is that it provides some level of **data abstraction**.
  - Collection of concepts that describe the structure of a database
  - **Basic operations**
    - Specify retrievals and updates on the database
  - **Dynamic aspect** or **behavior** of a database application
    - Allows the database designer to specify a set of valid operations allowed on database objects

§2.1.1 Categories of Data Models

- **High-level** or **conceptual data models**
  - Close to the way many users perceive data
- **Low-level** or **physical data models**
  - Describe the details of how data is stored on computer storage media
- **Representational data models**
  - Easily understood by end users
  - Also similar to how data organized in computer storage
  - Hides many details of data storage on disk, but can be implemented on a computer system directly

Categories of Data Models

- **Conceptual Types**
  - **Relational data model**
    - Used most frequently in traditional commercial DBMSs
  - **Object data model**
    - New family of higher-level implementation data models
    - Closer to conceptual data models
  - **Semantic data model**
    - Semantic Web

Categories of Data Models

- **Conceptual Features**
  - **Entity**
    - Represents a real-world object or concept
  - **Attribute**
    - Represents some property of interest
    - Further describes an entity
  - **Relationship** among two or more entities
    - Represents an association among the entities
    - **Entity-Relationship model**

Categories of Data Models

- **Physical**
  - Describe how data is stored as files in the computer
  - **Access path**
    - Structure that makes the search for particular database records efficient
  - **Index**
    - Example of an access path
    - Allows direct access to data using an index term or a keyword

Schemas, Instances, and Database State

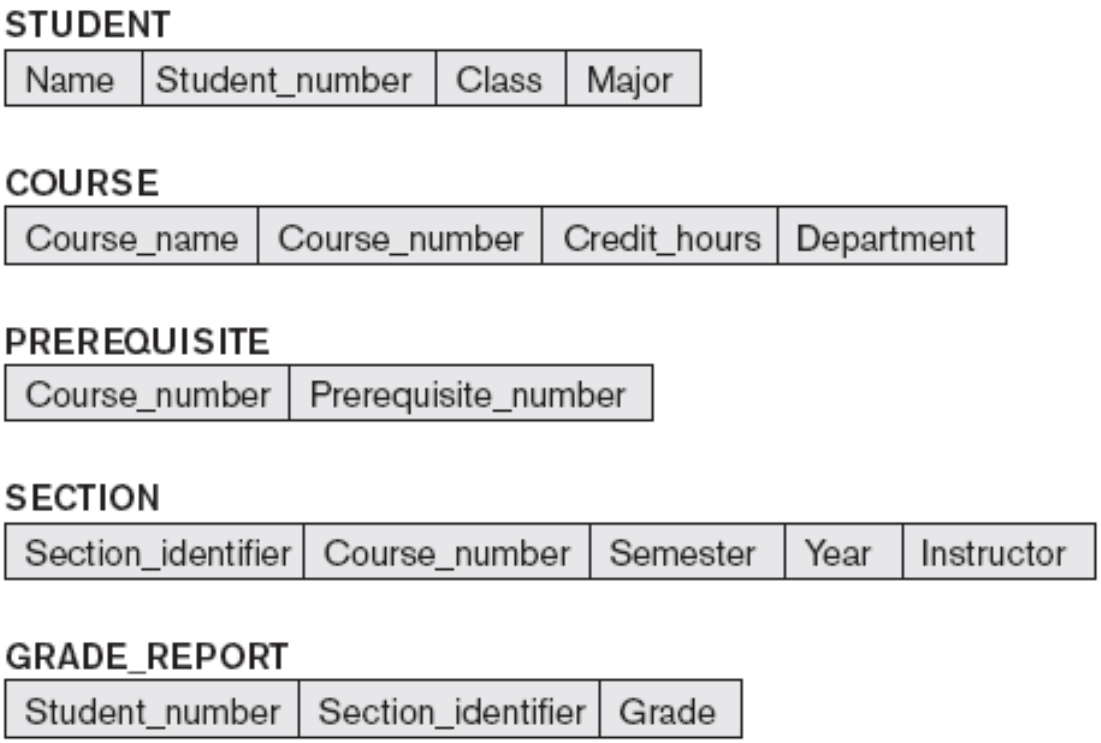
- In any data model, it is important to distinguish between the *description* of the database and the *database itself*

Schemas, Instances, and Database State

- **Database schema**
  - Description of a database
- **Schema diagram**
  - Displays selected aspects of schema
- **Schema construct**
  - Each object/instance in the schema (e.g. STUDENT or COURSE)
- **Database state or snapshot**
  - Data in database at a particular moment in time

Schemas, Instances, and Database State

Figure 2.1
Schema diagram for the database in Figure 1.2.



[6]Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

[7]It is customary in database parlance to use *schemas* as the plural for *schema*, even though *schemata* is the proper plural form. The word *scheme* is also sometimes used to refer to a schema.

Schemas, Instances, and Database State

Figure 2.1
Schema diagram for the database in Figure 1.2.

Table Name
Relation Name

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

[6]Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

[7]It is customary in database parlance to use *schemas* as the plural for *schema*, even though *schemata* is the proper plural form. The word *scheme* is also sometimes used to refer to a schema.

Schemas, Instances, and Database State

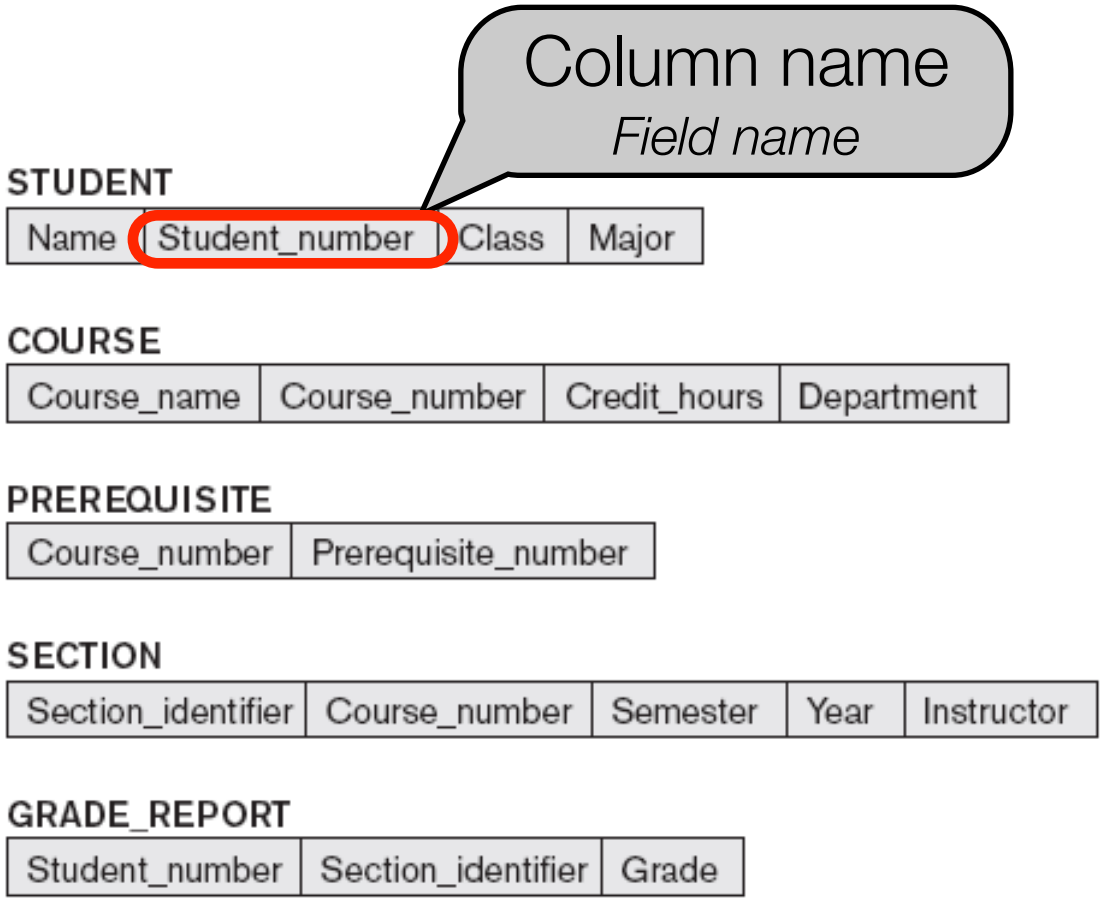


Figure 2.1
Schema diagram for the database in Figure 1.2.

[6]Schema changes are usually needed as the requirements of the database applications change. Newer database systems include operations for allowing schema changes, although the schema change process is more involved than simple database updates.

[7]It is customary in database parlance to use *schemas* as the plural for *schema*, even though *schemata* is the proper plural form. The word *scheme* is also sometimes used to refer to a schema.

Schemas, Instances, and Database State

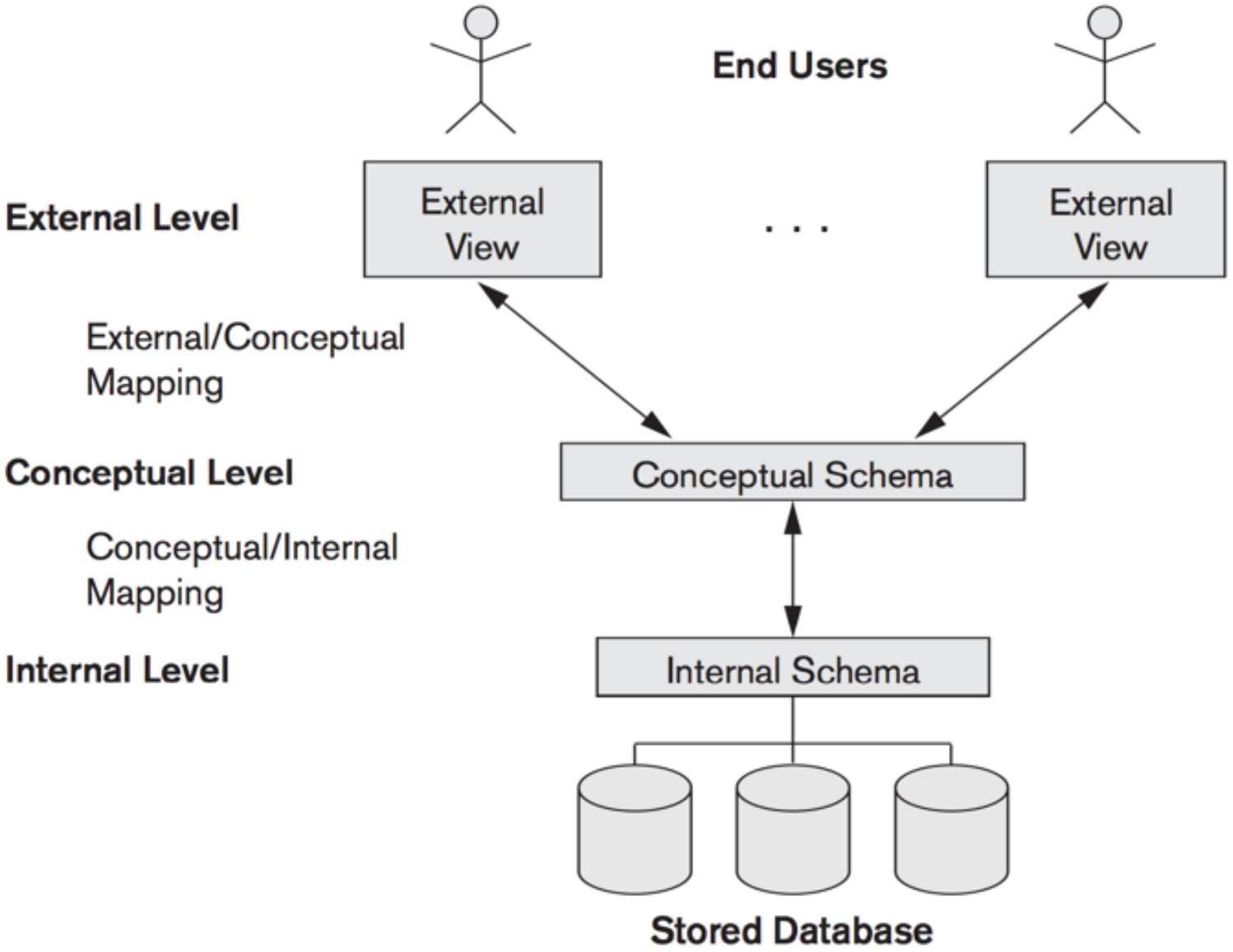
- **Define** a new database
  - Specify database schema to the DBMS
- **Initial state**
  - **Populated** or **loaded** with the initial data
- **Valid state**
  - Satisfies the structure and constraints specified in the schema
- **Schema evolution**
  - Changes applied to schema as application requirements change

2.2 Three-Schema Architecture and Data Independence

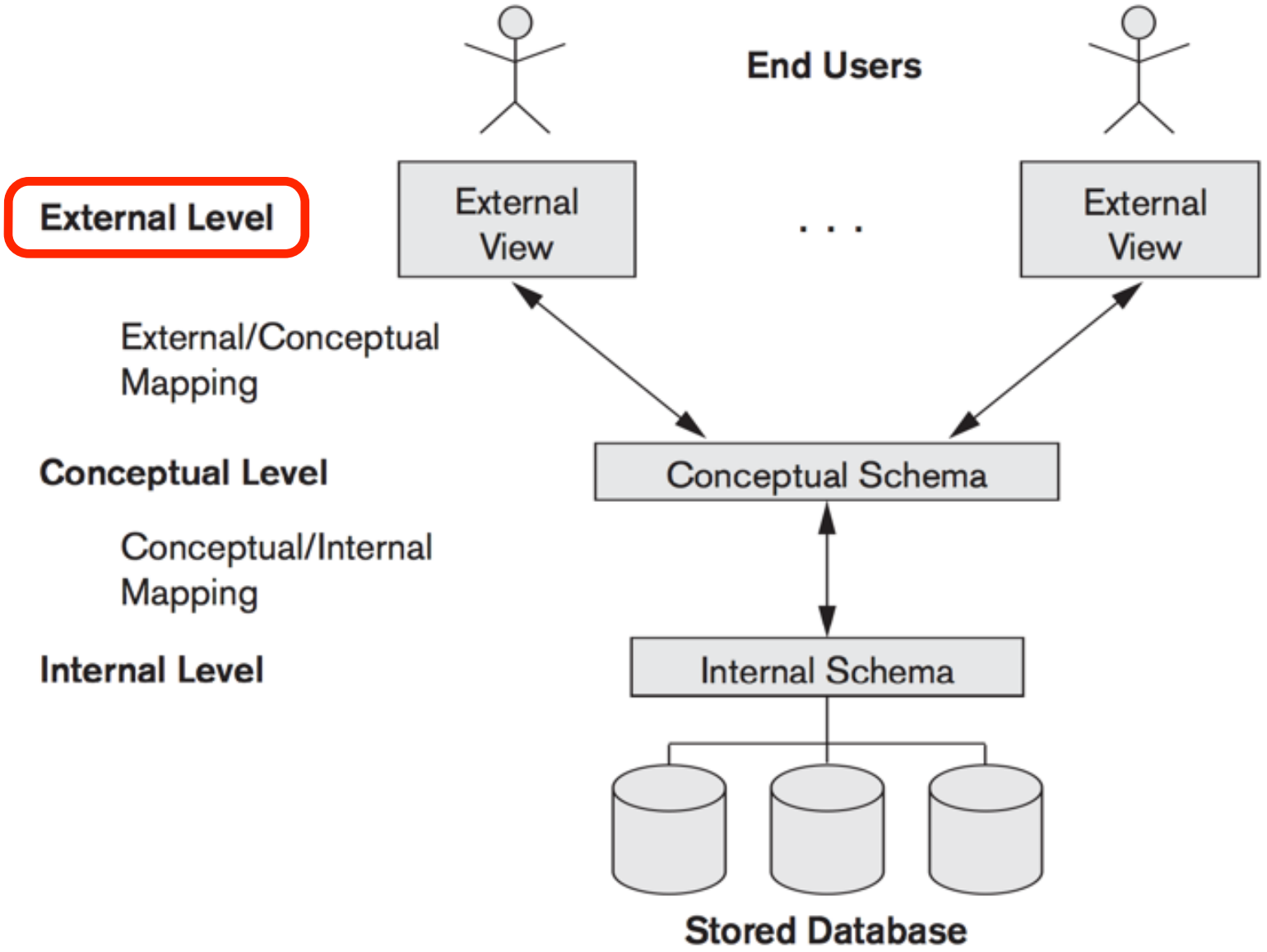
Three-Schema Architecture and Data Independence

- **Internal level**
  - Describes physical storage structure of the database
- **Conceptual level**
  - Describes structure of the whole database for a community of users
- **External** or **view level**
  - Describes part of the database that a particular user group is interested in

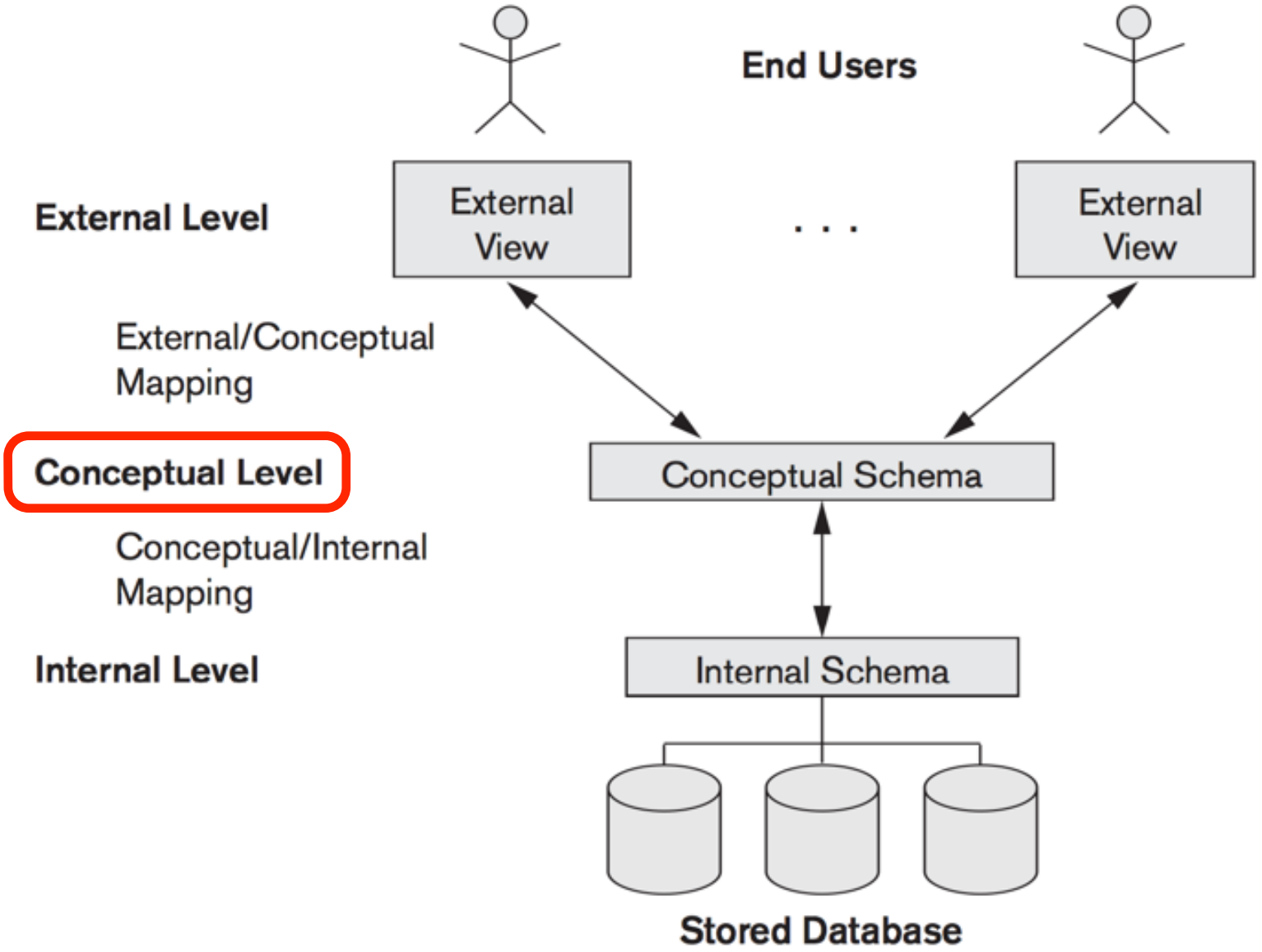
Three-Schema Architecture and Data Independence



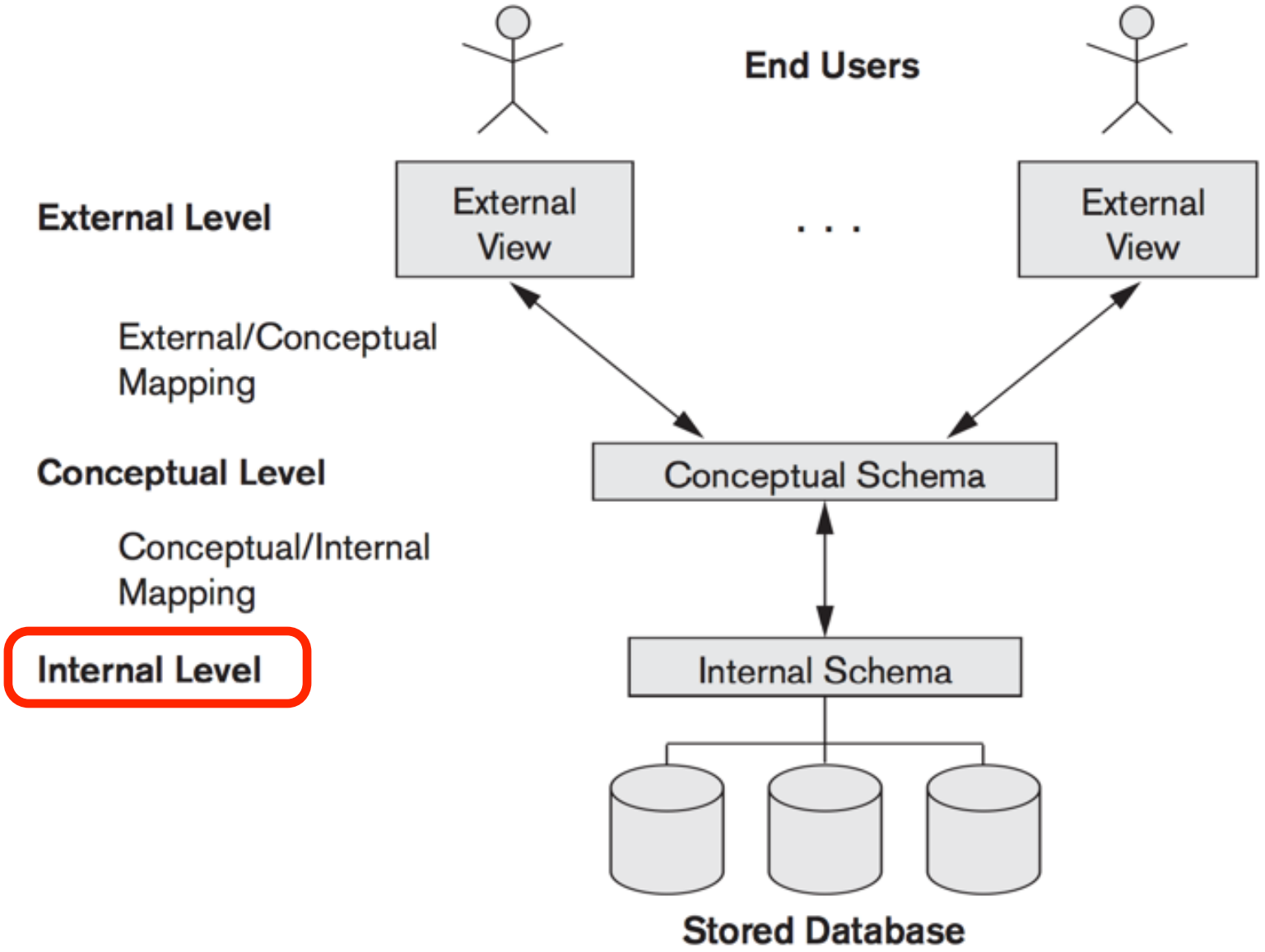
Three-Schema Architecture and Data Independence



Three-Schema Architecture and Data Independence



Three-Schema Architecture and Data Independence



Data Independence

- Capacity to change the schema at one level of a database system
  - Without having to change the schema at the next higher level
- Types:
  - **Logical**
  - **Physical**
- For example, changing to a different DBMS or disk drive without having to change the schema.

2.3 Database Languages and Interfaces

DBMS Languages

- **Data definition language (DDL)**
  - Defines both schemas
- **Storage definition language (SDL)**
  - Specifies the internal schema
- **View definition language (VDL)**
  - Specifies user views/mappings to conceptual schema
- **Data manipulation language (DML)**
  - Allows retrieval, insertion, deletion, modification

DBMS User Interfaces (UIs)

- Menu-based interfaces for Web clients or browsing
- Forms-based interfaces
- Graphical user interfaces (Like SSMS)
- Natural language interfaces
- Speech input and output
- Interfaces for parametric users
- Interfaces for the DBA

2.5 Centralized and Client/Server Architectures for DBMSs

Centralized and Client/Server Architectures for DBMSs

- **Centralized DBMSs Architecture**
  - All DBMS functionality, application program execution, and user interface processing carried out on one machine
  - This was the mainframe model, with dumb terminals. It is still widely used.

Basic Client/Server Architectures

- **Servers** with specific functionalities
  - **File server**
    - Maintains the files of the client machines.
  - **Printer server**
    - Connected to various printers; all print requests by the clients are forwarded to this machine
  - **Web servers** or **e-mail servers**

Basic Client/Server Architectures

- **Client machines**
  - Provide user with:
    - Appropriate interfaces to utilize these servers
    - Local processing power to run local applications
- **Server**
  - System containing both hardware and software
  - Provides services to the client machines
    - Such as file access, printing, archiving, or database access

Two-Tier Client/Server Architectures for DBMSs

- Server handles
  - Query and transaction functionality related to SQL processing
- Client handles
  - User interface programs and application programs

Two-Tier Client/Server Architectures for DBMSs

- Open Database Connectivity (**ODBC**)
  - Provides application programming interface (API)
  - Allows client-side programs to call the DBMS
    - Both client and server machines must have the necessary software installed
- Java Database Connectivity (**JDBC**)
  - Allows Java client programs to access one or more DBMSs through a standard interface

Three-Tier and n-Tier Architectures for Web Applications

- **Application server** or **Web server**
  - Adds intermediate layer between client and the database server
  - Runs application programs and stores business rules
- N-tier
  - Divide the layers between the user and the stored data further into finer components

Service Oriented Architecture

- Many different components of a system are services
- These may reside in the same machine or different machines.
- Database service, various business process services, calendar service, etc.

Three-Tier Architecture

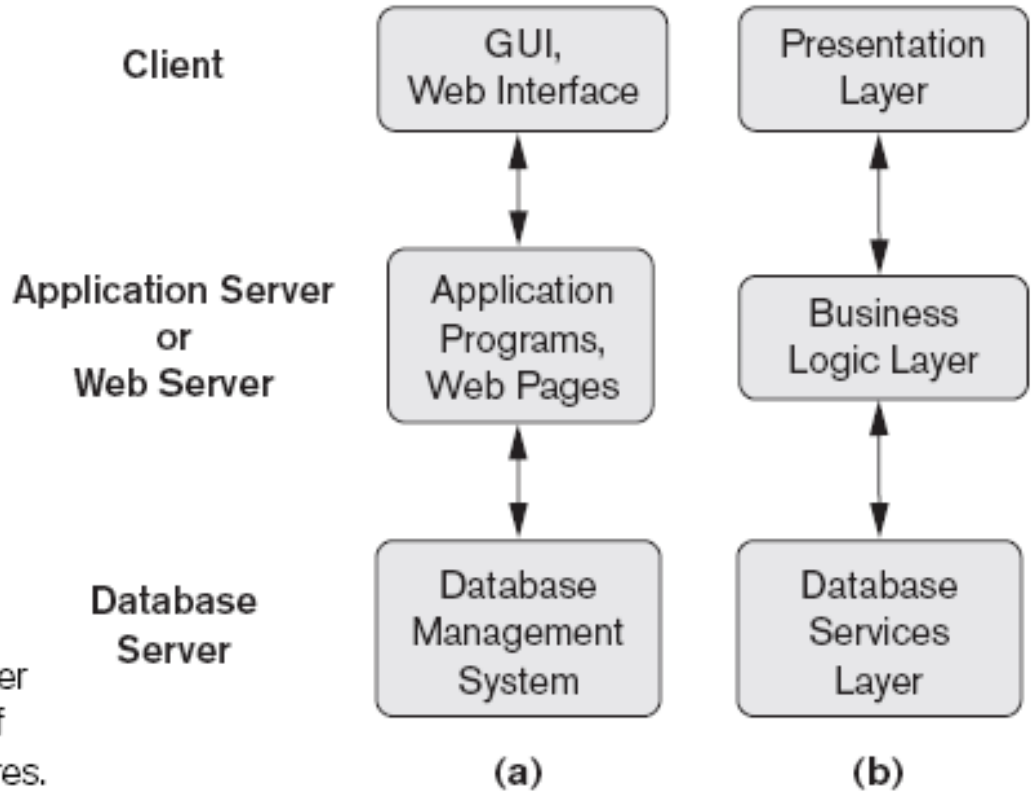


Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.

2.6 Classification of Database Management Systems

Classification of Database Management Systems

- Data model
  - Relational
  - Object
  - Hierarchical and network (legacy)
  - Native XML DBMS
- Number of users
  - Single-user
  - Multiuser

Classification of Database Management Systems

- **Number of sites**
  - **Centralized**
  - **Distributed**
    - **Homogeneous**
    - **Heterogeneous**
- **Cost**
  - Open source
  - Different types of licensing

Classification of Database Management Systems

- Types of access path options
- General or special-purpose

Summary

- Concepts used in database systems
- Main categories of data models
- Types of languages supported by DMBSs
- Interfaces provided by the DBMS
- DBMS classification criteria:
  - Data model, number of users, number of sties, access paths, cost