Database Management Systems (CSE 220)

Vikas Bajpai

Acknowledgements

Many slides of this presentation are from "Fundamentals of Database Systems, by Elmasri, Navathe".

Text Book(s):

- R. Elmasri and S. Navathe, *Fundamentals of Database Systems*, Addison-Wesley, 6th ed., 2011
- Silberschatz, H. Korth, and S. Sudarshan, *Database System Concepts*, McGraw-Hill.

Reference Book(s):

- R. Ramakrishnan, *Database Management Systems*, WCB/McGraw-Hill.
- C.J. Date, An Introduction to *Database Systems*, Pearson, 8th ed.

Data:

- information in raw or unorganized form.
- facts and statistics collected together for reference or analysis.

Database:

- Is an organized collection of data.
- Collection of related data.
- Collection of schemas, tables, queries, reports, views etc. etc.

Database:

DATABASE
BIG DATA
SETS
VOLUME
SOCIAL
VIRTUAL
TECHNOLOGY
INTERNET
NAS
SAN
SEARCH
LARGE
STORAGE
INFORMATION
PETABYTES
ANALYSIS
SOFTWARE
RESEARCH
TECHNOLOGIES
MANAGEMENT
MILLION
FUTURE
NEW
WORLD
DEVELOPMENT
GOVERNMENT
PROCESS
SCIENCE
EXABYTES
SIZE

Databases these days:

- Used daily: ☺
  - Facebook
    - Posts
    - Likes
  - Twitter
    - Tweets
  - Online shopping
    - amazon.com
    - flipcart.com

Database Management System (DBMS):

- A collection of programs that enables users to create and maintain a database.
- A general purpose software system that facilitates the process of:
  - Defining
  - Constructing
  - Manipulating
  - Sharing

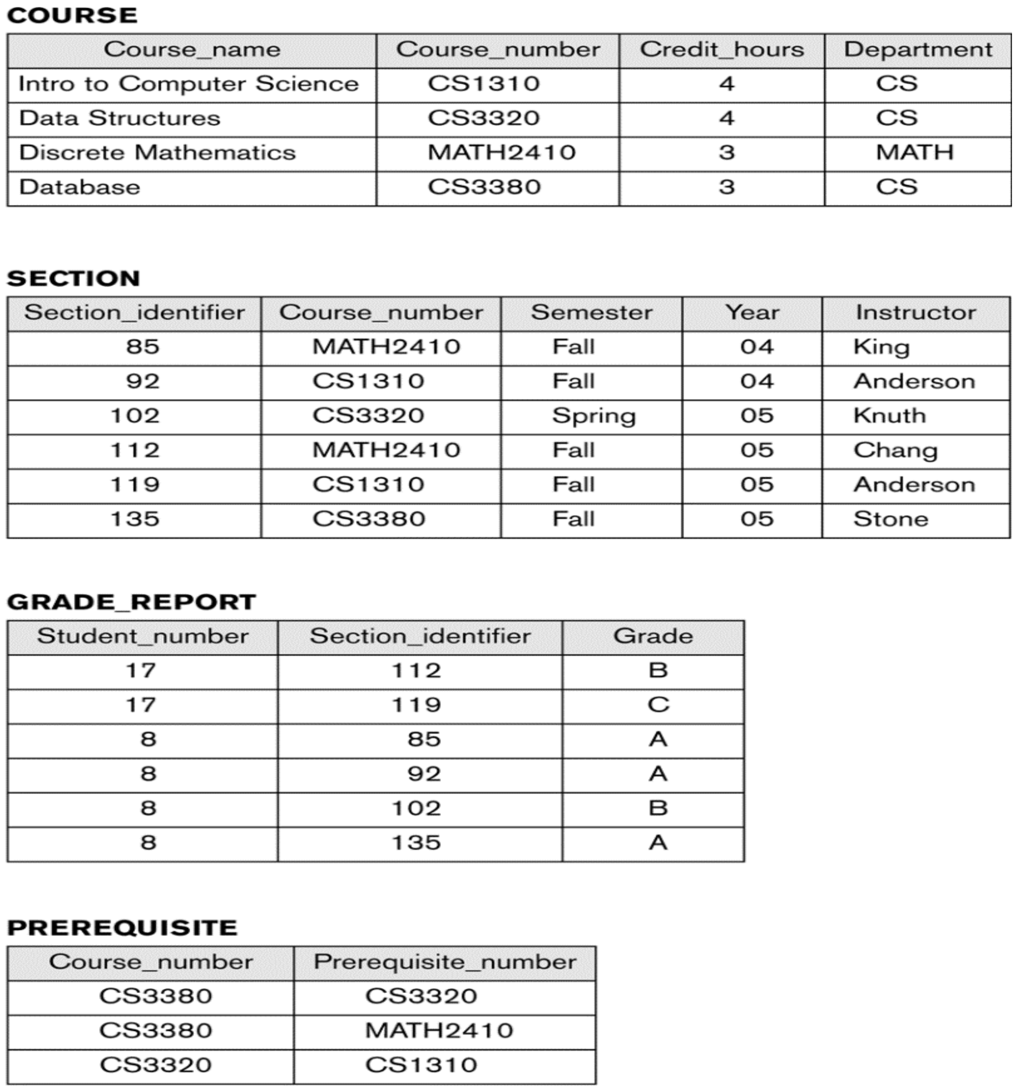
DBMS Functionalities

databases among various users and applications.

DBMS Functionalities:

- 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

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

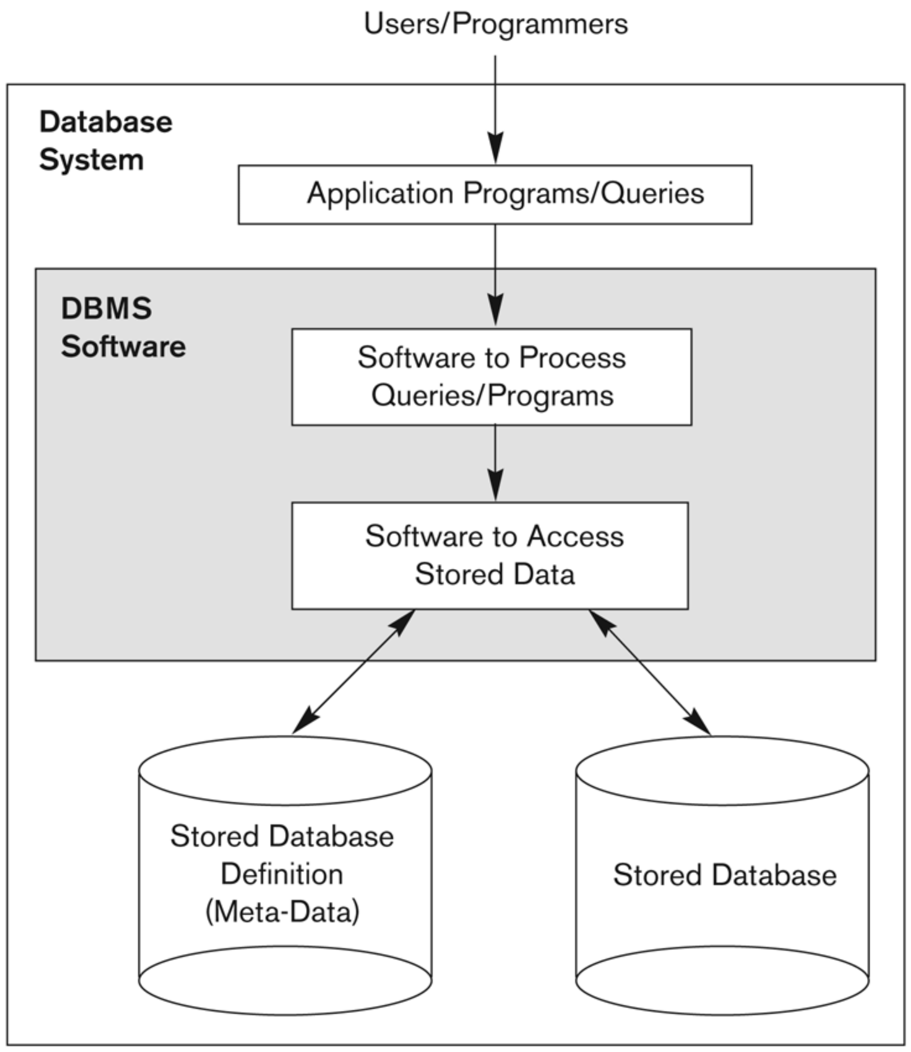
Source: Rameez Elmasri and Shamkant B. Navathe

DBMS Functionalities:

Other features:

- 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

Simplified database system environment:

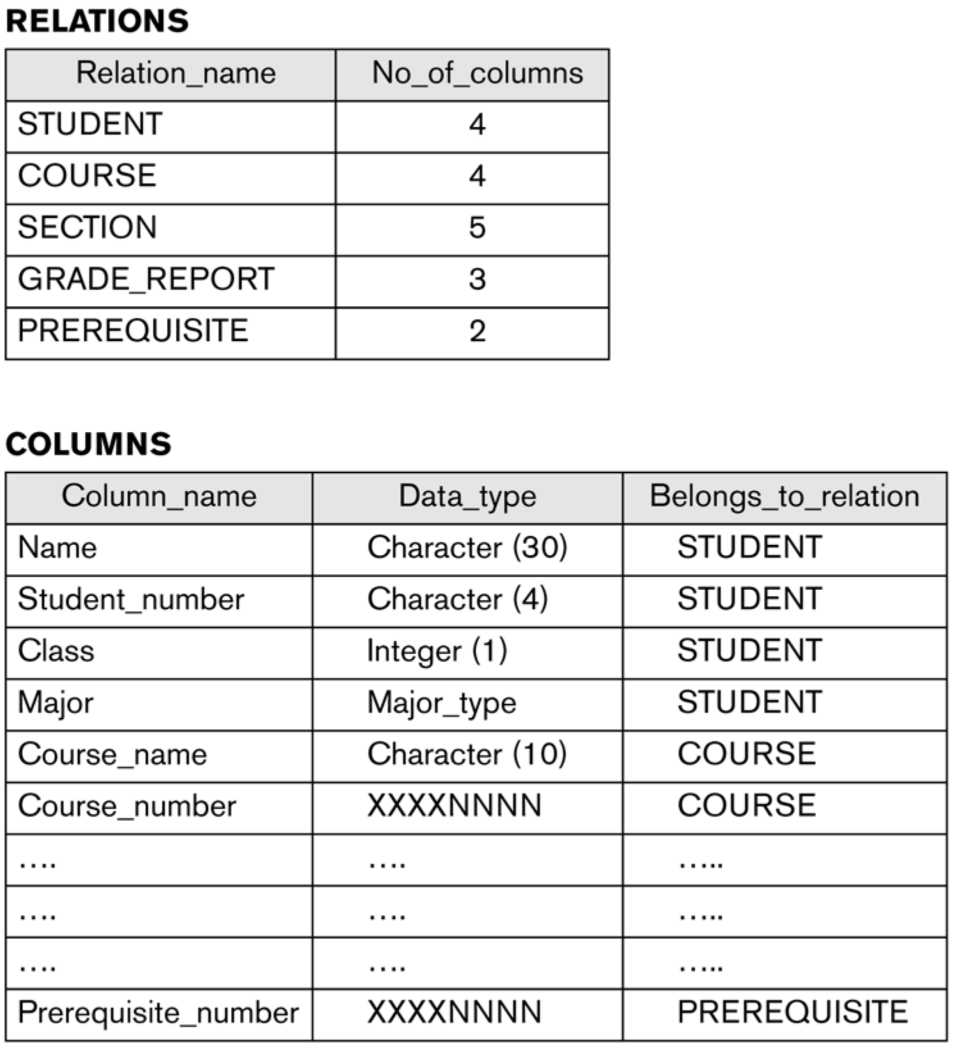

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Source: Rameez Elmasri and Shamkant B. Navathe

Database approach vs file processing approach:

- Self describing nature of a database system.
- Insulation between programs and data, and data abstraction.
- Support of multiple views of data.
- Sharing of data and multiuser transaction processing.
  - Allowing concurrent users for retrieval & updating
  - Concurrency control
  - Ensuring complete transaction
  - OLTP (online transaction processing)

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

Source: Rameez Elmasri and Shamkant B. Navathe

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.

Database Users:

- Users may be divided into:
  - Those who actually use and control the database content, (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").

1. Actors on the scene:

1. 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.

2. 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.

Contd…

1. Actors on the scene:

3. End Users : They use the data for queries, reports and some of them update the database content. End-users can be categorized into:
   a. Casual end users: access database occasionally
   b. Naive or Parametric end users: 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

Contd…

1. Actors on the scene:

c. Sophisticated end users:

- 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.

d. Stand-alone users:

- 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.

Contd…

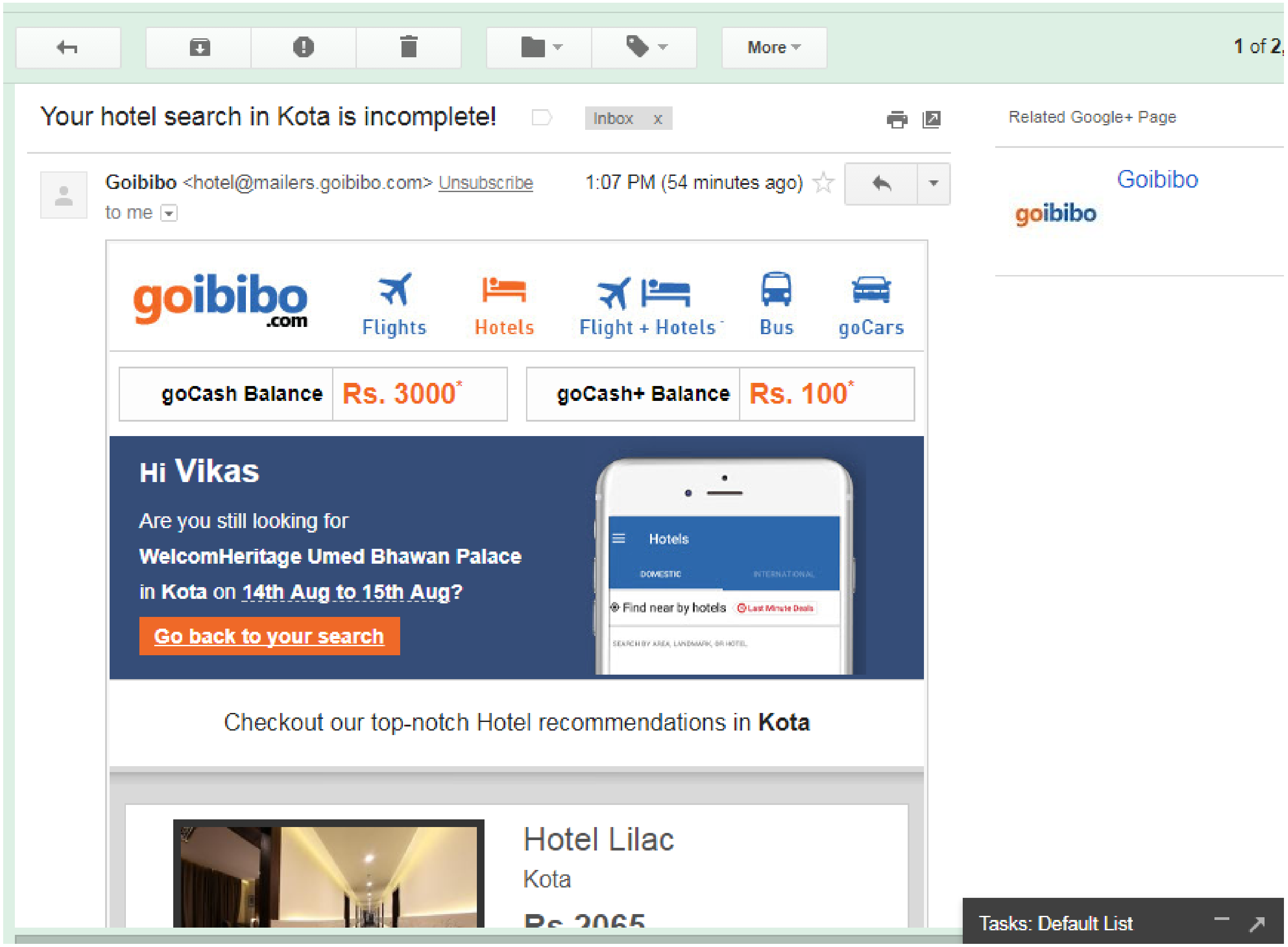
1. Actors on the scene:

4. System Analysts and Application Programmers (S/W Engineers): This category currently accounts for a very large proportion of the IT work force.

   a. **System Analysts**: They understand the user requirements of naïve and sophisticated users and design applications including canned transactions to meet those requirements.

   b. **Application Programmers:** Implement the specifications developed by analysts and test and debug them before deployment.

   c. **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.



Your hotel search in Kota is incomplete!

Goibibo <hotel@mailers.goibibo.com> Unsubscribe 1:07 PM (54 minutes ago)
to me

goCash Balance	Rs. 3000*	goCash+ Balance	Rs. 100*

Hi Vikas

Are you still looking for **WelcomHeritage Umed Bhawan Palace** in **Kota** on **14th Aug to 15th Aug?**

Checkout our top-notch Hotel recommendations in **Kota**

Hotel Lilac
Kota

2. Workers behind the scene:

1. DBMS system designers and implementers: 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.
2. 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.
3. Operators and maintenance personnel: They manage the actual running and maintenance of the database system hardware and software environment.

Advantages of using DBMS approach:

- Controlling redundancy
- Restricting unauthorized access
- Providing persistent storage for program objects
- Providing storage structures for efficient query processing
- Providing backup and recovery
- Providing multiple user interfaces
- Representing complex relationships among data

Contd…

Advantages of using DBMS approach:

- Enforcing integrity constraints
- Permitting inferencing and actions using rules
- Additional implications of using database approach:
  - Potential for enforcing standards
  - Reduced application development time
  - Flexibility
  - Availability of up-to-date information

When not to use a DBMS:

Overhead cost of using DBMS because of:

- High initial investment in hardware, software and training
- The generality that a DBMS provides for defining and processing data.
- Overhead for providing security, concurrency control, recovery and integrity functions.

When not to use a DBMS:

If the database designers and DBA do not properly design the database or if the database systems are not implemented properly:

- The database and applications are simple, well defined and not expected to change.
- There are stringent real time requirements for some programs that may not be met because of DBMS overhead.
- Multiple-user access to data is not required.

View of Data:

- View is a single table, derived from other tables. These other tables could be base tables or previously defined views.
- A view doesn't exist in physical form, it is considered as a virtual table.

In other words:

- View logically represents subset of data from one or more tables.

Why to use View:

- To restrict database access
- To make complex queries easy
- To allow data independence
- To present different views of same data

Data Abstraction:

Once Again??

- 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

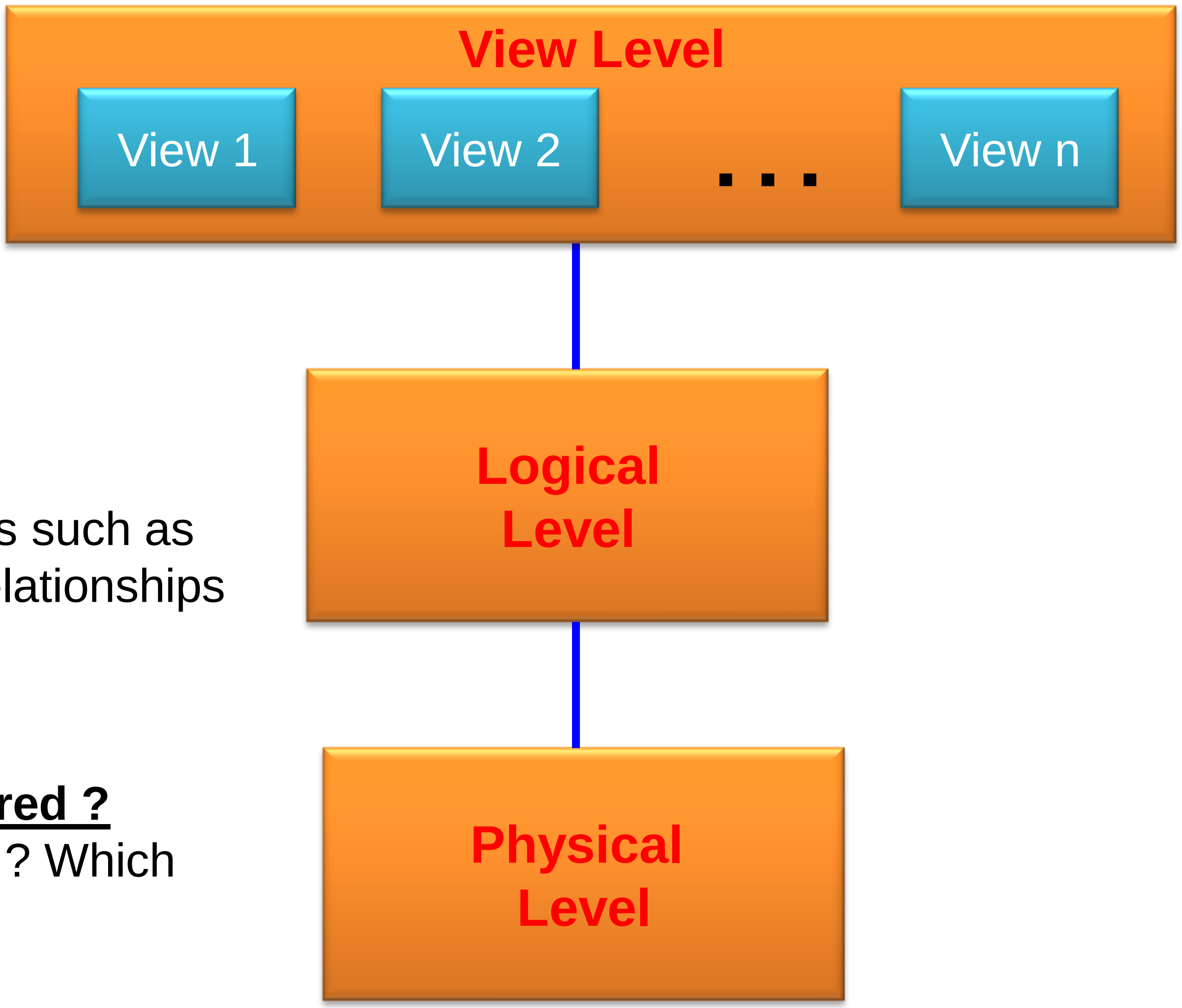
Data Abstraction:

- For a system to be usable, it must retrieve data efficiently.
- Developers hide the complexity from users through several levels of abstraction.

Levels of Data Abstraction:

1. Physical Level (Internal): The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low level data structures in detail.
2. Logical Level (Conceptual): It describes ‘what data are stored in database’ and what relationship exists among those data.
3. View Level (External): The highest level of abstraction which describes only part of entire database.

Data Abstraction:



<u>What data users and application programs see ?</u>

<u>What data is stored ?</u>
describe data properties such as data semantics, data relationships

<u>How data is actually stored ?</u>
e.g. are we using disks ? Which file system ?

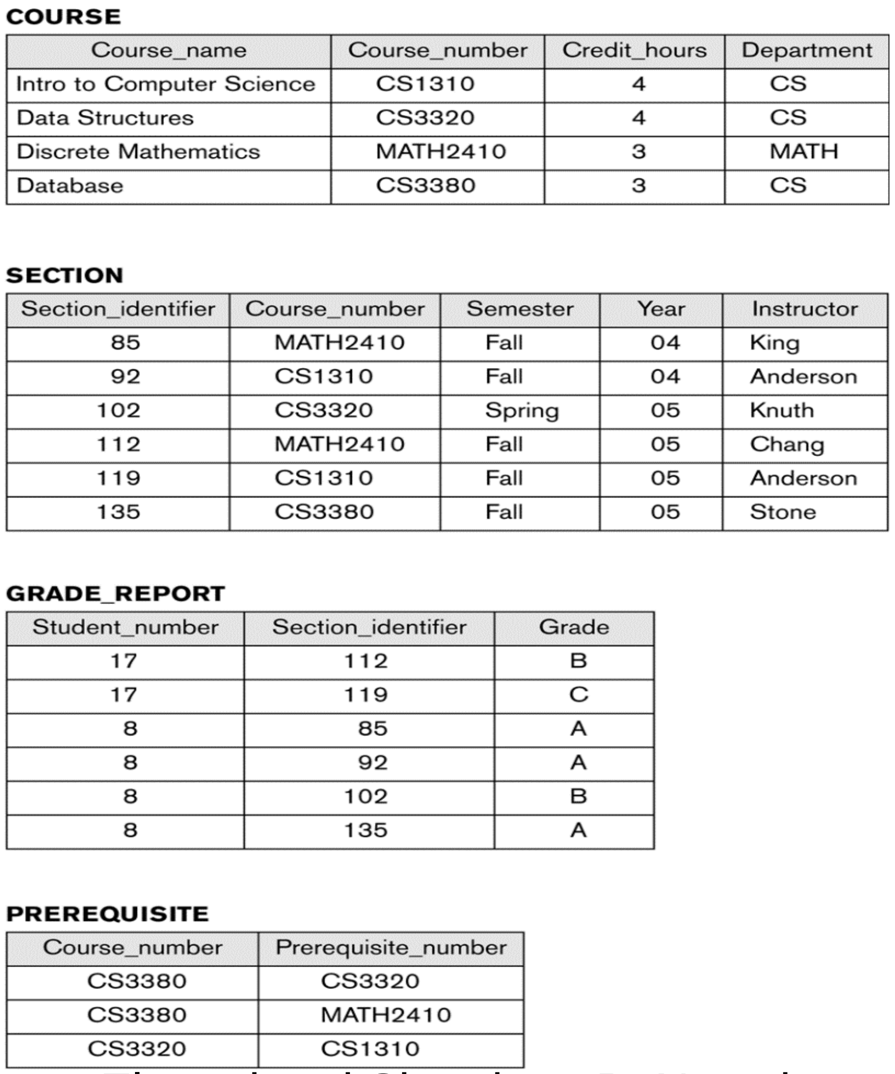
Database Instances/ Database state:

- The collection of information stored in the database at a particular moment is called an instance of a database.

Database Instances/ Database state:

- 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.

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

Source: Rameez Elmasri and Shamkant B. Navathe

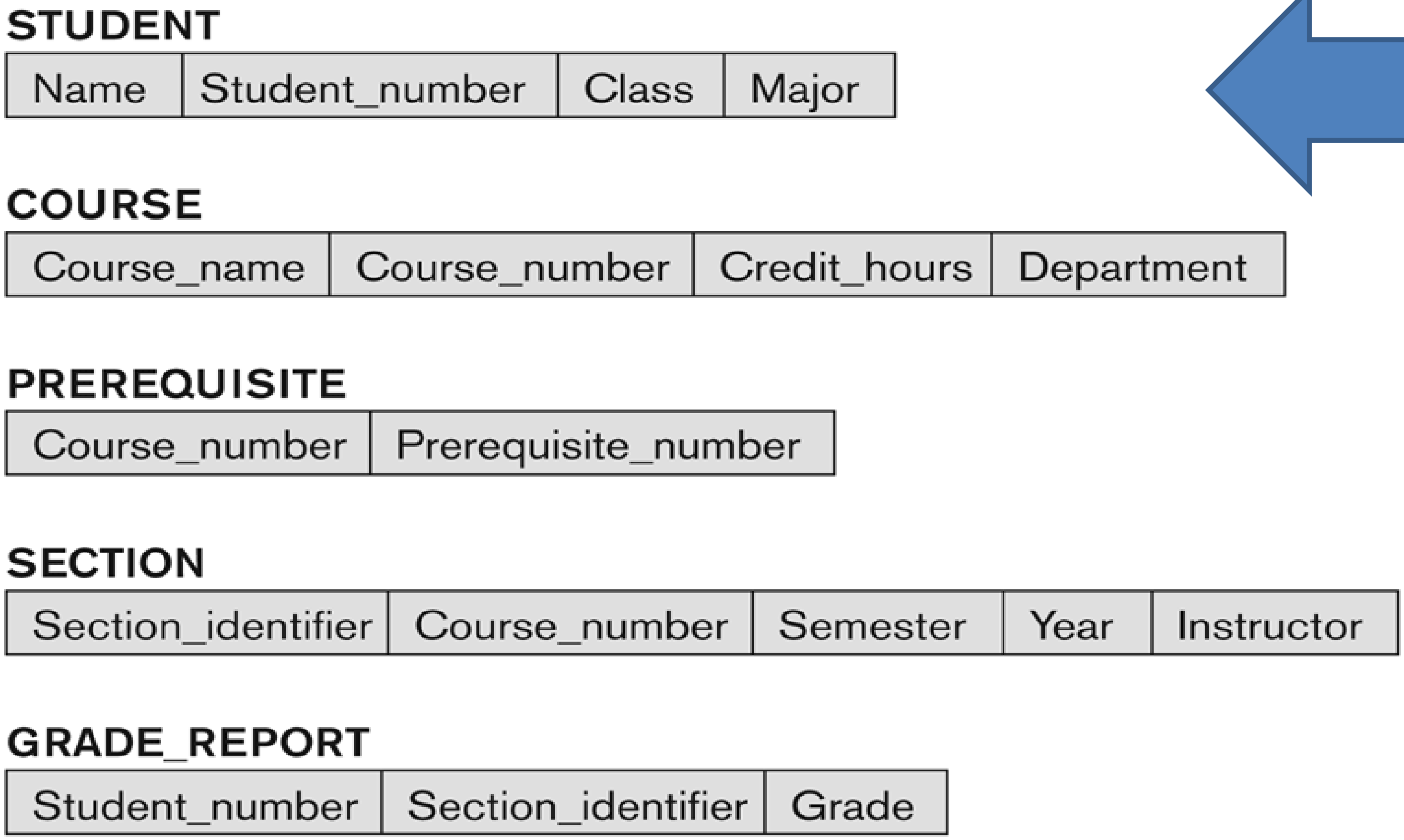
Database Schemas:

- The overall design of the database is called the database schema.

In other words:

- The description of a database is called the database schema. Includes descriptions of the database structure, data types, and the constraints on the database.

Example of a Database Schema:



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

Source: Rameez Elmasri and Shamkant B. Navathe

Data Model:

- A collection of concepts, used to describe the structure* of a database.
- It provides the necessary means to achieve abstraction.

* Structure of a database means data types, relationships and constraints that should hold for the data.

Categories of data models:

1. High level or conceptual data models
2. Low level or physical data models
3. Representation or implementation data models
4. Self-Describing Data Models

Categories of Data Models:

1. High level or conceptual data models:
   – Provide concepts that are close to the way many users perceive data.
     - (Also called ***entity-based*** or ***object-based*** data models.)
2. Low level or physical data models:
   – Provide concepts that describe details of how data is stored in the computer. These are usually specified in an ad-hoc manner through DBMS design and administration manuals
3. Representation or implementation 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).
4. Self-Describing Data Models:
   – Combine the description of data with the data values. Examples include XML, key-value stores and some NOSQL systems.

Three Schema Architecture:

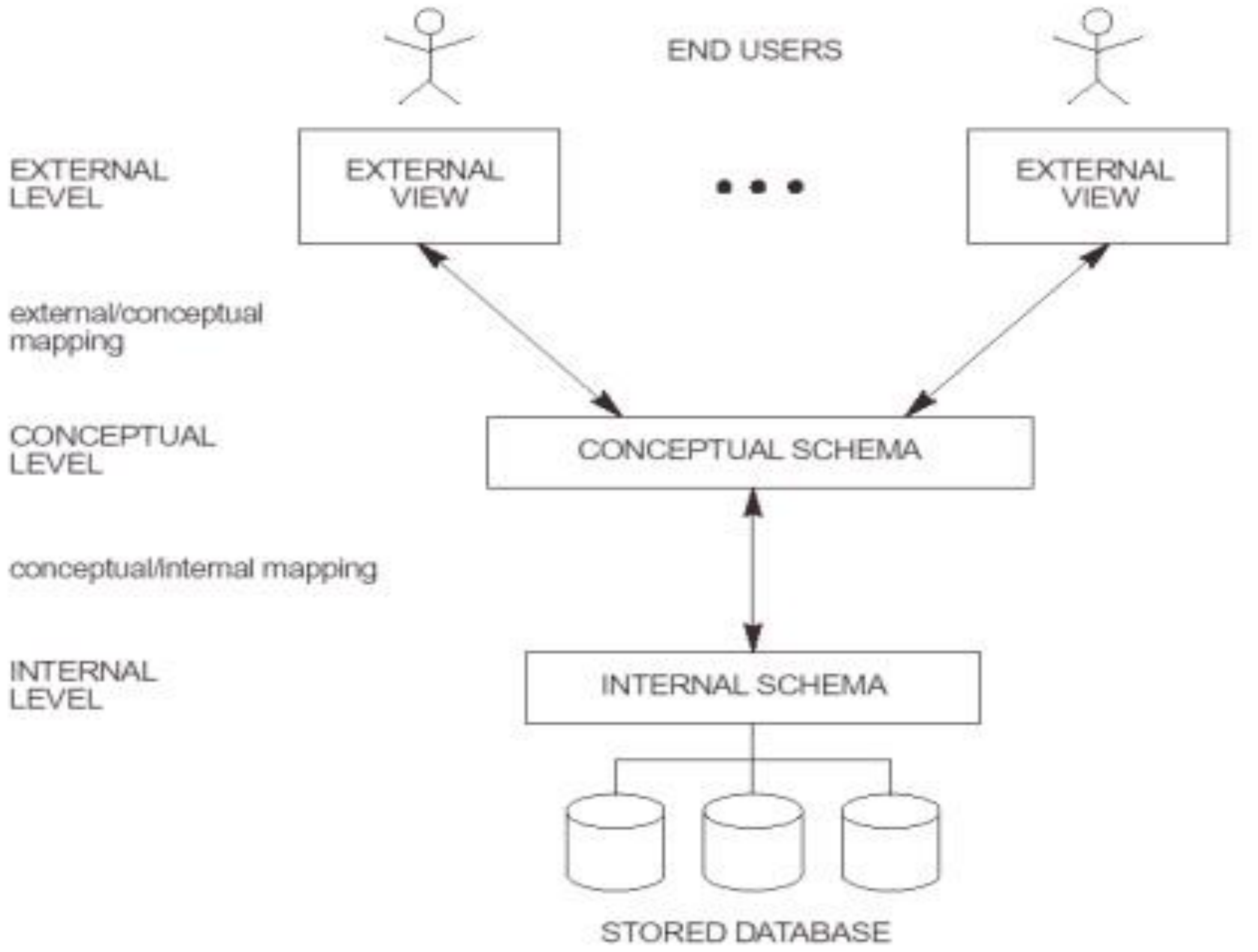
- Goal is to separate "user application" and "physical database".
- Not explicitly used in commercial DBMS products, but has been useful in explaining database system organization
- Three schema architecture defines DBMS schemas at three levels levels are:
    1. Internal schema
    2. Conceptual schema
    3. External schema

Three Schema Architecture:

DBMS schemas at three levels levels are:

1. Internal schema at the internal level to describe physical storage structures and access paths (e.g indexes).
    - Typically uses a physical data model.
2. 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.
3. External schemas at the external level to describe the various user views.
    - Usually uses the same data model as the conceptual schema.

Three Schema Architecture:



Three-Schema Architecture:

- Mappings among schema levels are needed to transform requests and data.
  - Programs refer to an external schema, and are mapped by the DBMS to the internal schema for execution.
  - Data extracted from the internal DBMS level is reformatted to match the user’s external view (e.g. formatting the results of an SQL query for display in a Web page)

Data Independence:

- Data independence is the capacity to change the schema at one level of a database system without any change in the schema at the next higher level. Only the mapping between the two levels is changed.
- Two types of Data Independence:
  1. Logical Data Independence
  2. Physical Data Independence

1. Logical Data Independence:

- It is the capacity to change conceptual schema without having any change in external schema.
- We may change conceptual schema to expand the database, to change the constraints, or to reduce the database.
- After conceptual schema undergoes a logical reorganization, application program that refer the external schema must work as before.

2. Physical Data Independence:

- It is the capacity to change the internal schema without having any change in the conceptual schema. Hence the external schema need not be changed as well .
- Change to the internal schema may be needed because some physical files had to be reorganized.

Database Languages:

1. Data-definition Language (DDL)
2. Data-manipulation Language (DML)
    a. Procedural DMLs
    b. Declarative DMLS (Non Procedural DML)

1. Data-definition Language (DDL):

- DDL is a set of definitions which specify a database schema.
- DDL is used by DBA and database designers to define all schemas.

Eg. -> create table ACCOUNT (name char(10), balance integer)

Execution of above DDL statement creates ACCOUNT table plus updates a special set of tables called Data Dictionary* or Data Directory.

2. Data-manipulation Language (DML):

- DML is a language that enables users to access or manipulate data as organized by appropriate data model.
- Two types of DML:
  a. Procedural DMLs-require a user to specify what data are needed and how to get those data.
  b. Declarative DMLs (Non-Procedural)- require a user to specify what data are needed without specifying how to get those data.

Data Manipulation Language is:

- The retrieval of information stored in database.
- The insertion of information into the database.
- The deletion of information from the database.
- The modification of information stored in database.

DBMS Programming Language Interfaces:

- Programmer interfaces for embedding DML in a programming languages:
  - Embedded Approach: e.g embedded SQL (for C, C++, etc.), SQLJ (for Java)
  - Procedure Call Approach: e.g. JDBC for Java, ODBC (Open Databse Connectivity) for other programming languages as API's (application programming interfaces)
  - Database Programming Language Approach: e.g. ORACLE has PL/SQL, a programming language based on SQL; language incorporates SQL and its data types as integral components
  - Scripting Languages: PHP (client-side scripting) and Python (server-side scripting) are used to write database programs.

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:

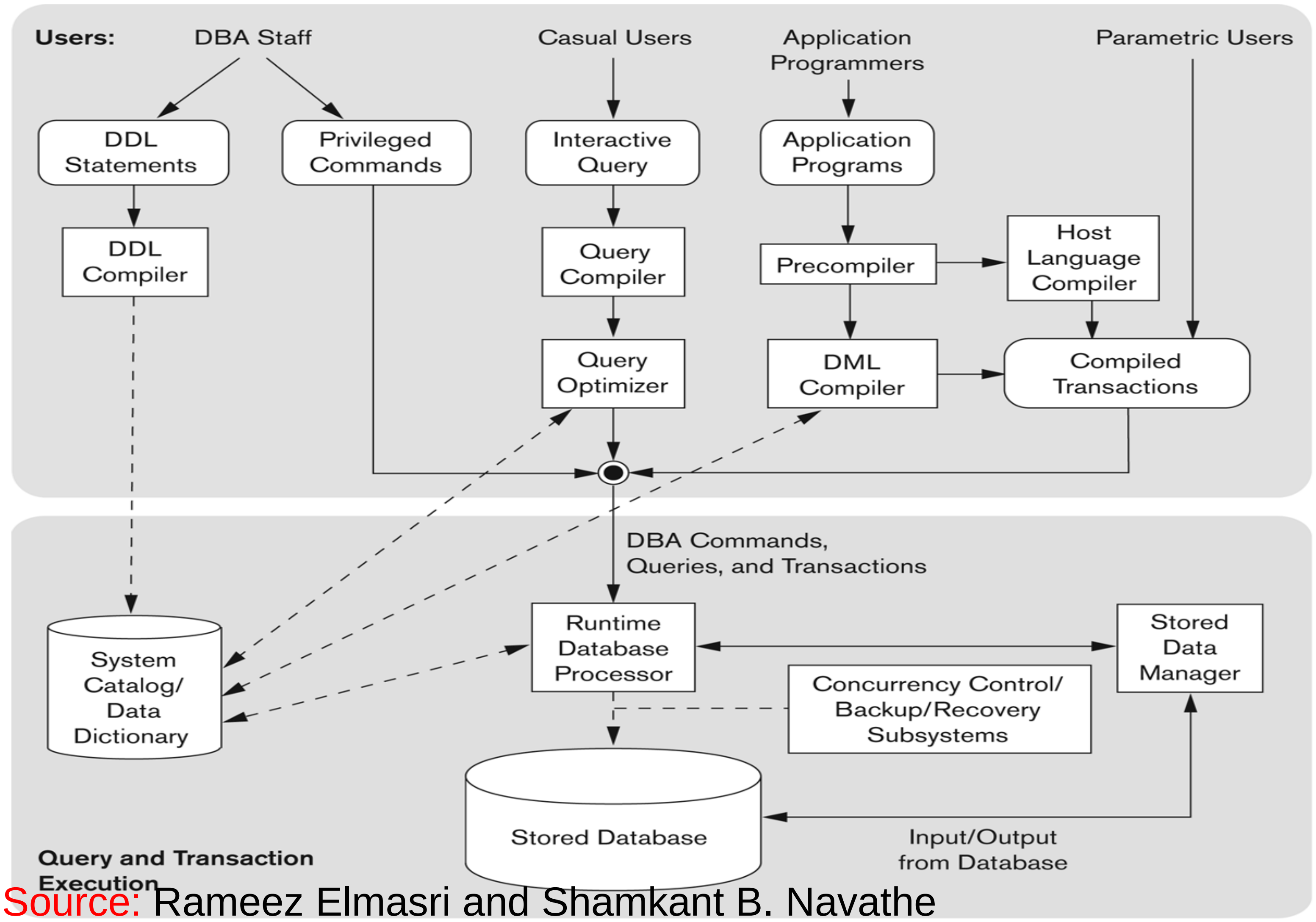
Data dictionary / repository:

- 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.

Other Tools:

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

DBMS component modules and their interactions:

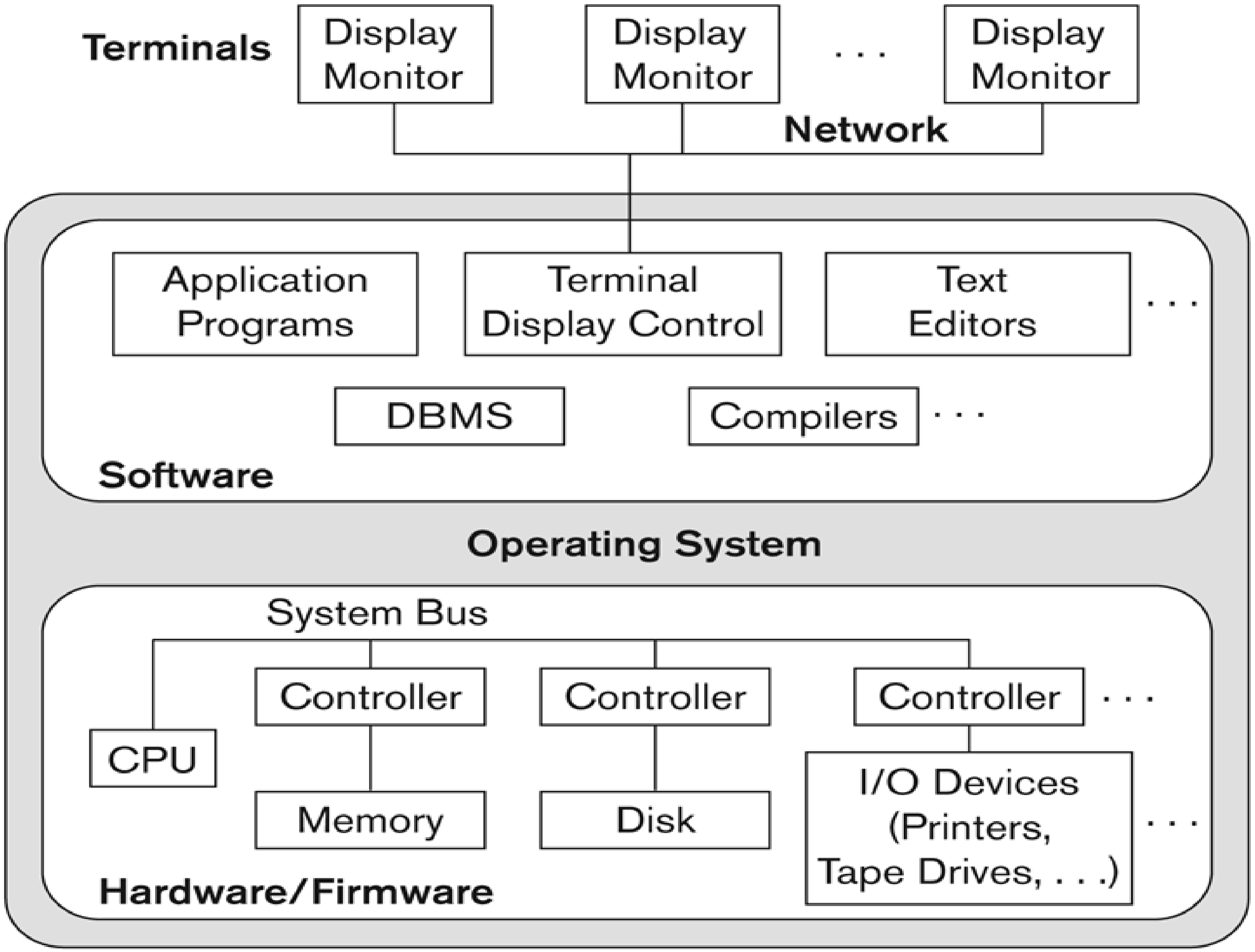


Source: Rameez Elmasri and Shamkant B. Navathe

Centralized DBMS Architectures:

- 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.

Centralized DBMS Architectures:



Source: Rameez Elmasri and Shamkant B. Navathe

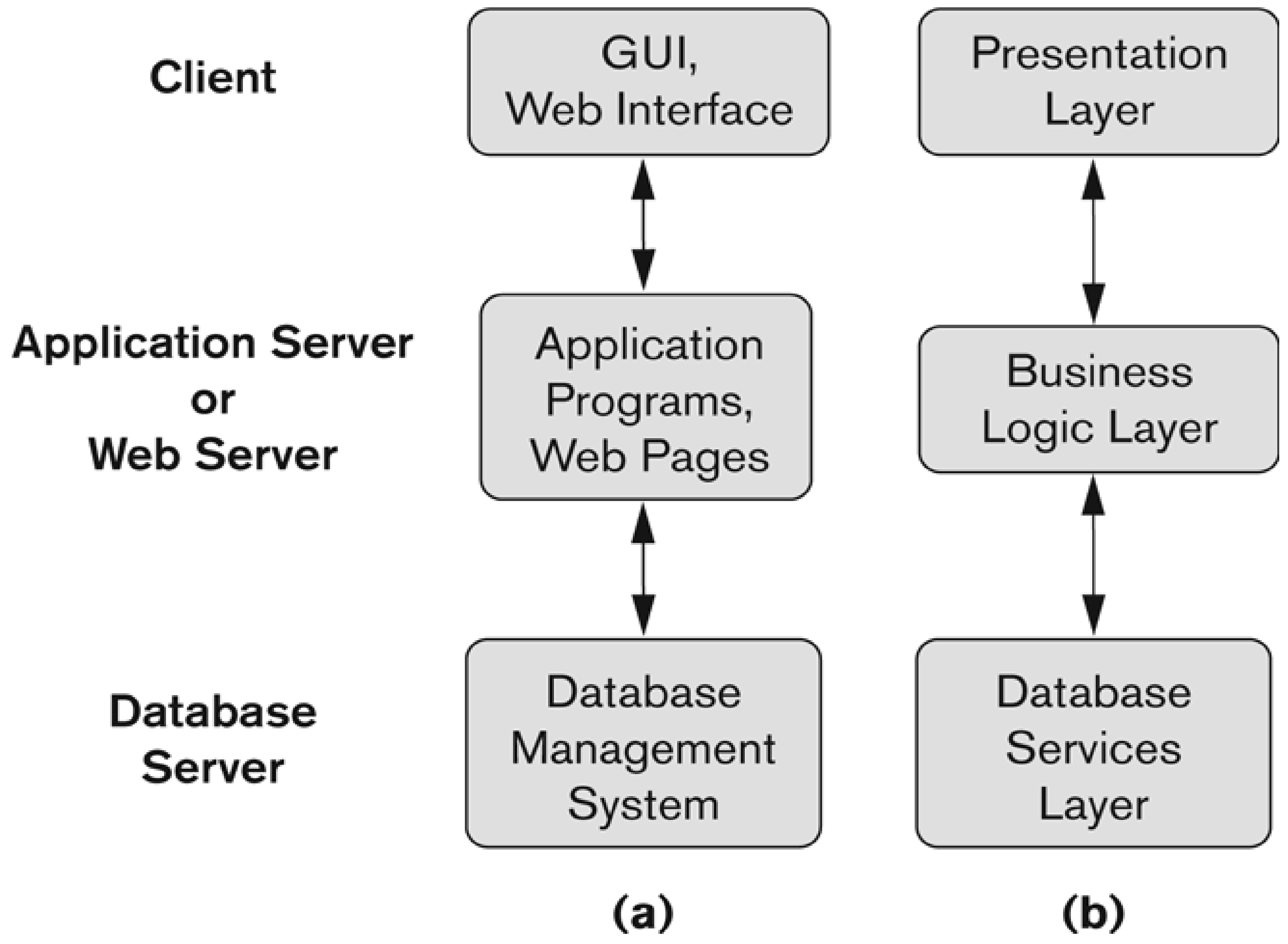
Client Server DBMS Architecture:

- Client Module: handles user interaction and provides the user friendly interfaces such as forms or menu based GUIs
- Server Module: handles data storage, access, search and other functions.

Three Tier Client-Server Architecture

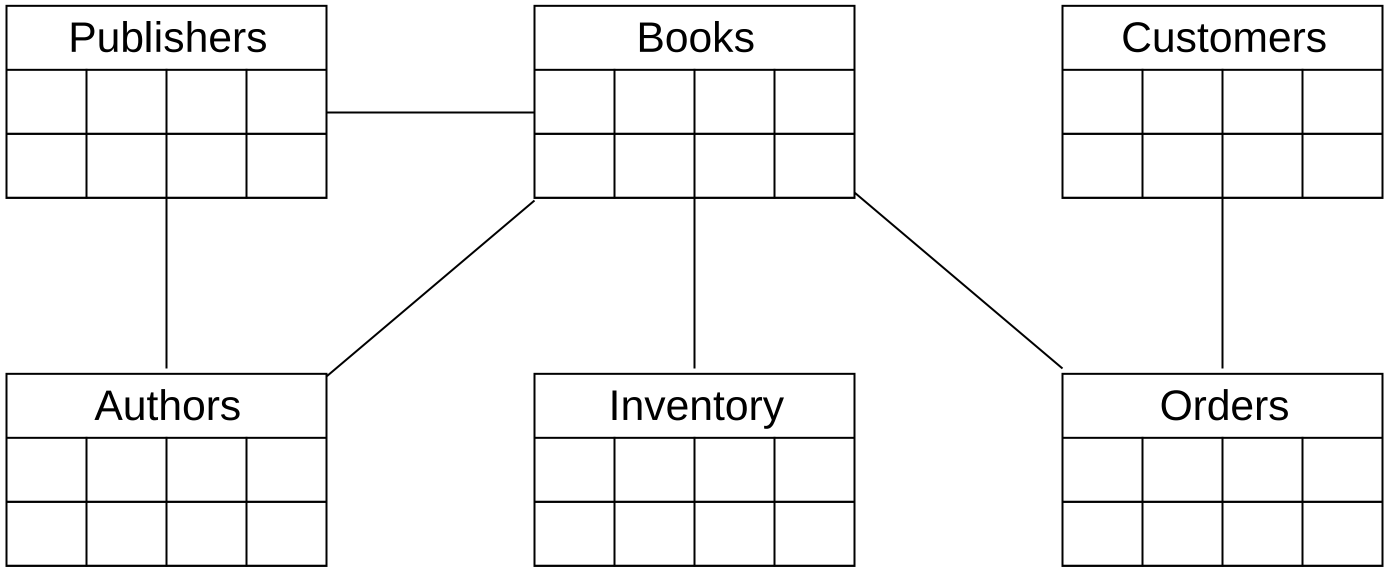
- Common for Web applications
- Intermediate Layer called Application Server or Web Server:
  - Stores the web connectivity software and the business logic part of the application used to access the corresponding data from the database server
  - Acts like a conduit for sending partially processed data 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

Three-tier client-server architecture



Source: Rameez Elmasri and Shamkant B. Navathe

A Database with Multiple Tables



DBMS at a Glance

1. Data Modeling
2. Data Retrieval
3. Data Storage
4. Data Integrity