Database Systems Concepts and Architecture

Dr. Jyotismita Chaki

History of database systems

- Early 1960s: Charles Bachman at GE creates the first general purpose DBMS Integrated Data Store. It creates the basis for the network model (standardized by CODASYL)
- Late 1960s: IBM develops the Information Management System (IMS). It uses an alternate model, called the hierarchical data model. SABRE is created around IMS.
- 1970: Edgar Codd, from IBM creates the relational data model. In 1981 Codd receives the Turing Award for his contributions to database theory.
- 1980s SQL, developed by IBM, becomes the standard query language for databases. SQL is standardized by ISO.
- 1980s and 1990s, IBM, Oracle, Informix and others develop powerful DBMS.
- In the Internet Age, DBMS are showing how useful they can be.

Motivation for database systems

- Assume you work for Amazon (and database management systems have not been invented) and you are asked to write a collection of programs that can store and retrieve every single sell in every store of the chain (could be a Tbyte of info)
- How do you do it?
  - How do you find and retrieve data?
  - How many files do you need?
  - How many disks do you need?
- And if we add complexity:
  - How do you operate on the data?
  - How do you allow concurrent access and modifications to the data?

Introduction: Database

- A database is a collection of related data
- By data, we mean known facts that can be recorded and that have implicit meaning.
- A database has the following implicit properties:
  - Represents some aspect of the real world, sometimes called the miniworld or the universe of discourse (UoD).
  - A logically coherent collection of data with some inherent meaning.
  - Designed, built, and populated with data for a specific purpose.
- Database has some source from which data is derived, some degree of interaction with events in the real world, and an audience that is actively interested in its contents.

Introduction: Database

- A database can be of any size and complexity.
  - The computerized catalog of a large library may contain half a million entries organized under different categories
  - A database of even greater size and complexity would be maintained by a social media company such as Facebook, which has more than a billion users: constantly changing information required by the social media Web site
  - An example of a large commercial database is Amazon.com. It contains data for over 60 million active users, and millions of books, CDs, videos, DVDs, games, electronics, apparel, and other items: continually updated as new books and other items are added to the inventory, and stock quantities are updated as purchases are transacted.

Introduction: DBMS

- A **database management system (DBMS)** is a computerized system that enables users to create and maintain a database.
- a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications.
- The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called **meta-data**

Introduction: DBMS

- **Constructing** the database is the process of storing the data on some storage medium that is controlled by the DBMS.
- **Manipulating** a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
- **Sharing** a database allows multiple users and programs to access the database simultaneously.
- An **application program** accesses the database by sending queries or requests for data to the DBMS.
- A **query** typically causes some data to be retrieved;
- A **transaction** may cause some data to be read and some data to be written into the database.

DBMS: Example

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

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

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	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

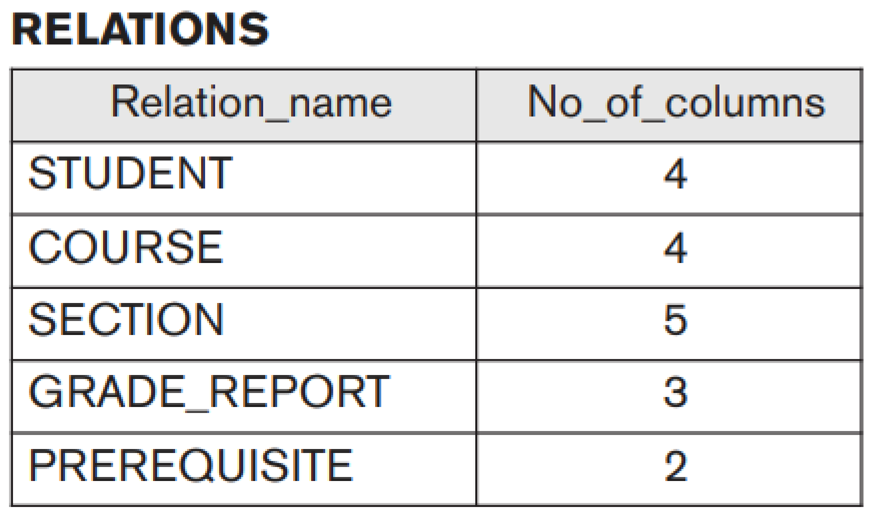
Characteristics of database approach

- In traditional file processing, each user defines and implements the files needed for a specific software application as part of programming the application.
- In the database approach, a single repository maintains data that is defined once and then accessed by various users repeatedly through queries, transactions, and application programs.
- The main characteristics of the database approach:
  - Self-describing nature of a database system
  - Insulation between programs and data, and data abstraction
  - Support of multiple views of the data
  - Sharing of data and multiuser transaction processing

Characteristics of database approach: Self-describing nature

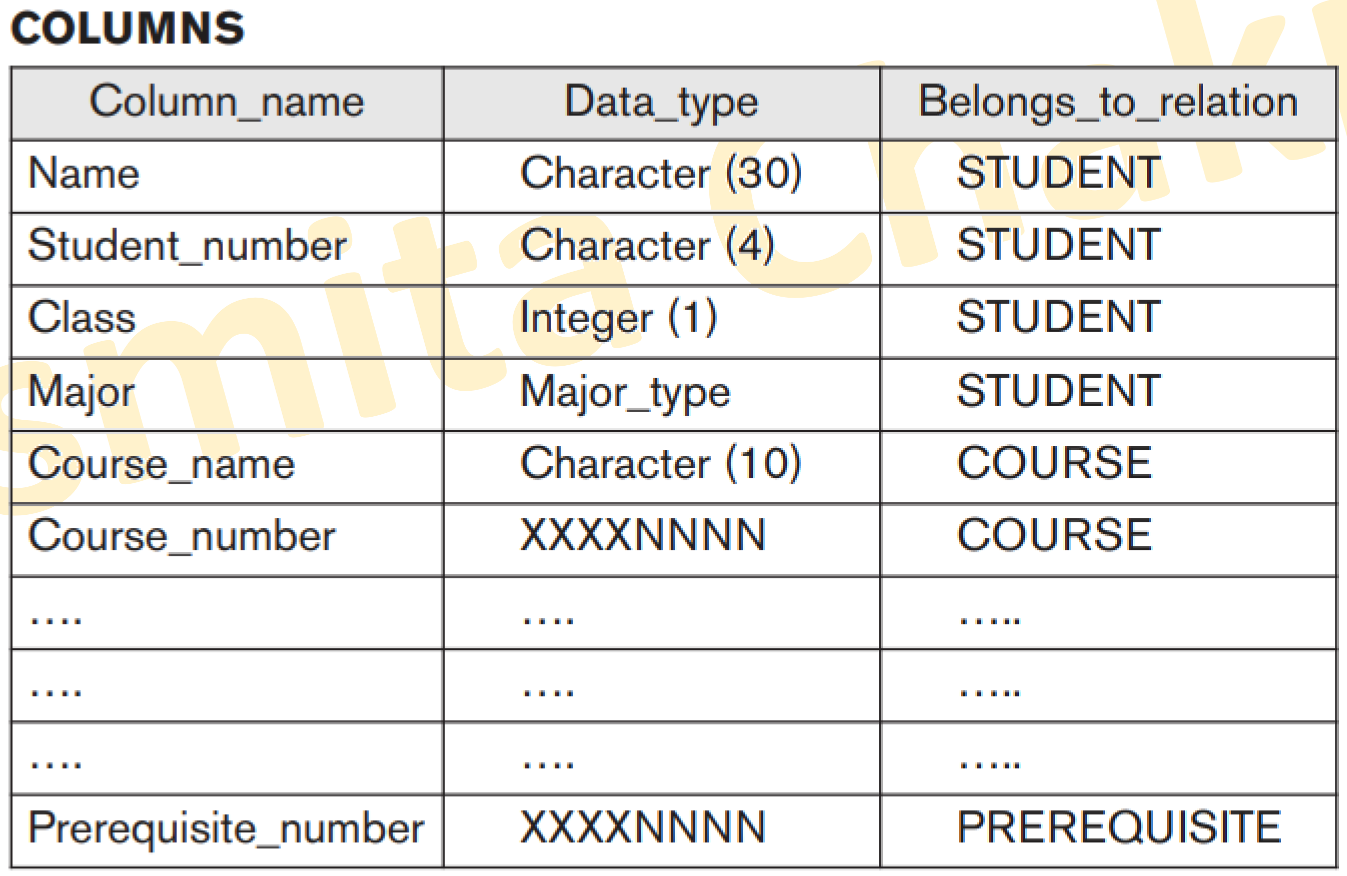
- Meta-data contains the catalogue.
- A complete definition or description of the database structure and constraints is stored in the DBMS catalog, which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data.
- NOSQL systems, do not require meta-data: the data is stored as self-describing data

Characteristics of database approach: Self-describing nature: Example



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

- XXXXNNNN is used to define a type with four alphabetic characters followed by four numeric digits.

Characteristics of database approach: Insulation between programs and data, and data abstraction

- In traditional file processing, the structure of data files is embedded in the application programs
- The structure of data files is stored in the DBMS catalog separately from the access programs: **program-data independence**.
- In object-oriented and object-relational systems, users can define operations on data as part of the database definitions.
- An operation (also called a function or method) is specified in two parts:
  - The interface (or signature): the operation name and the data types of its arguments
  - The implementation (or method) of the operation
- Application programs can use operations, regardless of knowing how the operations are implemented: **program-operation independence**.

Characteristics of database approach: Insulation between programs and data, and data abstraction

- **Data abstraction** allows program-data independence and program-operation independence
- **Conceptual representation** of data that does not include many of the details of how the data is stored or how the operations are implemented: **Data Model**
- The data model hides storage and implementation details that are not of interest to most database users.
- The data model uses logical concepts, their properties, and their interrelationships, that may be easier for most users to understand.

Characteristics of database approach: Insulation between programs and data, and data abstraction

Data Item Name	Starting Position in Record	Length in Characters (bytes)
Name	1	30
Student_number	31	4
Class	35	1
Major	36	4

Internal storage format for a STUDENT record, based on the database catalog

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

A conceptual representation of the STUDENT records

Characteristics of database approach: Support of multiple views of the data

- A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

TRANSCRIPT

Student_name	Student_transcript				
	Course_number	Grade	Semester	Year	Section_id
Smith	CS1310	C	Fall	08	119
	MATH2410	B	Fall	08	112
Brown	MATH2410	A	Fall	07	85
	CS1310	A	Fall	07	92
	CS3320	B	Spring	08	102
	CS3380	A	Fall	08	135

The TRANSCRIPT view.

COURSE_PREREQUISITES

Course_name	Course_number	Prerequisites
Database	CS3380	CS3320
		MATH2410
Data Structures	CS3320	CS1310

The COURSE_PREREQUISITES view

Characteristics of database approach: Sharing of data and multiuser transaction processing

- A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time.
- The DBMS must include concurrency control software to ensure that when several users trying to update the same data in a controlled manner, the result of the updates are correct.
- A **transaction** is an executing program or process that includes one or more database accesses, such as reading or updating of database records.
- The DBMS must enforce several transaction properties:
    - The isolation property ensures that each transaction appears to execute in isolation from other transactions
    - The atomicity property ensures that either all the database operations in a transaction are executed or none are

Actors on the scene: Database Administrator

- In a database environment:
  - primary resource→ database itself,
  - Secondary resource → DBMS and related software
- Administering these resources is the responsibility of the **database administrator (DBA)**.
- The DBA is responsible for
  - authorizing access to the database,
  - coordinating and monitoring its use, and
  - acquiring software and hardware resources as needed

Actors on the scene: Database Designer

- Responsible for
  - Identifying the data to be stored in the database
  - Choosing appropriate structures to represent and store this data
  - Communicating with all prospective database users in order to understand their requirements
  - Creating a design that meets user requirements.
  - Developing views of the database: Each view is then analyzed and integrated with the views of other user groups. The final database design must be capable of supporting the requirements of all user groups

Actors on the scene: End Users

- End users are the people whose jobs require access to the database.
- There are several categories of end users:
  - **Casual end users** occasionally access the database, but they may need different information each time.
  - **Naive** or **parametric end users** constantly querying and updating the database, using standard types of queries and updates— called **canned transactions**—that have been carefully programmed and tested. Examples are:
    - Bank customers and tellers check account balances and post withdrawals and deposits.
    - Social media users post and read items on social media Web sites.
  - **Sophisticated end users** include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.
  - **Standalone users** (user of a financial software package) maintain personal databases by using ready-made program packages that provide easy-to-use menu-based or graphics-based interfaces.

Actors on the scene: System Analysts and Application Programmers (Software Engineers)

- **System analysts** determine
  - The requirements of end users, especially naive and parametric end users
  - Develop specifications for standard canned transactions that meet these requirements.
- **Application programmers** (**software developers** or **software engineers**) implement these specifications as programs; then they
  - Test,
  - Debug,
  - Document, and
  - Maintain these canned transactions.

Workers behind the scene

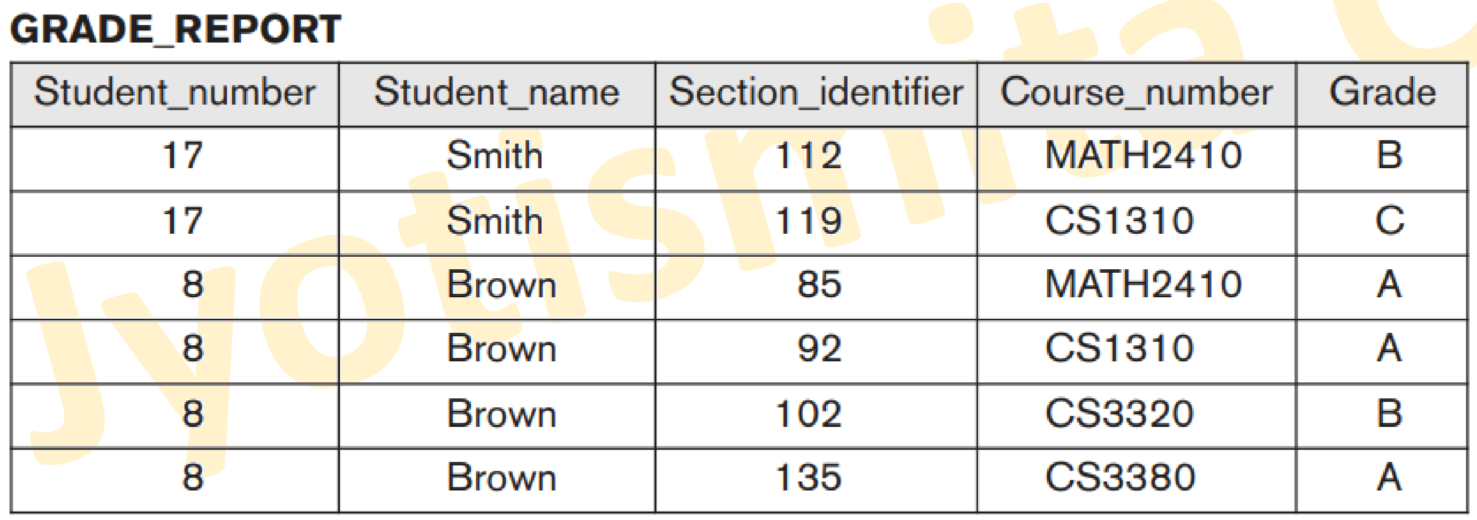
- Associated with the design, development, and operation of the DBMS software and system environment.
- they include the following categories:
  - **DBMS system designers and implementers** design and implement the DBMS modules and interfaces as a software package.
  - **Tool developers** design and implement tools—the software packages that facilitate database modeling and design, database system design, and improved performance.
  - **Operators and maintenance personnel** (system administration personnel) are responsible for the actual running and maintenance of the hardware and software environment for the database system.

Advantages of using DBMS approach: Controlling Redundancy

- **Redundancy** means storing the same data multiple times which leads to several problems
  - Duplication of effort
  - Storage space is wasted
  - Files that represent the same data may become inconsistent
- **Data normalization** ensures consistency and saves storage space
  - We should have a database design that stores each logical data item—such as a student's name or birth date—in only one place in the database.

Advantages of using DBMS approach: Controlling Redundancy

- It is sometimes necessary to use controlled redundancy to improve the performance of queries



GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Smith	112	MATH2410	B
17	Smith	119	CS1310	C
8	Brown	85	MATH2410	A
8	Brown	92	CS1310	A
8	Brown	102	CS3320	B
8	Brown	135	CS3380	A

- By placing all the data together, we do not have to search multiple files to collect this data: **Denormalization**

Advantages of using DBMS approach: Controlling Redundancy

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Smith	112	MATH2410	B
17	Smith	119	CS1310	C
8	Brown	85	MATH2410	A
8	Brown	92	CS1310	A
8	Brown	102	CS3320	B
8	Brown	135	CS3380	A

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Brown	112	MATH2410	B

Advantages of using DBMS approach: Restricting Unauthorized Access

- Users or user groups are given account numbers protected by passwords, which they can use to gain access to the database.
- A DBMS should provide a **security and authorization subsystem**, which the DBA uses to create accounts and to specify account restrictions.
- Then, the DBMS should enforce these restrictions automatically.
- We can apply similar controls to the DBMS software.
  - Only the DBA's staff may be allowed to use certain privileged software
  - Parametric users may be allowed to access the database only through the predefined apps

Advantages of using DBMS approach: Providing Persistent Storage for Program Objects

- Databases can be used to provide **persistent storage** for program objects and data structures.
- This is one of the main reasons for **object-oriented database systems**
- Object-oriented database systems are compatible with programming languages such as C++ and Java, and the DBMS software automatically performs any necessary program variable conversions.
- Hence, a complex object in C++ can be stored permanently in an object-oriented DBMS.
- Such an object is said to be **persistent**, since it survives the termination of program execution and can later be directly retrieved by another program.

Advantages of using DBMS approach: Providing Storage Structures and Search Techniques for Efficient Query Processing

- Database systems must provide capabilities for efficiently executing queries and updates.
- Because the database is typically stored on disk, the DBMS must provide specialized data structures and search techniques to speed up disk search for the desired records → Done by **Indexes**
- DBMS often has a buffering or caching module that maintains parts of the database in main memory buffers.
- The **query processing and optimization** module of the DBMS is responsible for choosing an efficient query execution plan for each query based on the existing storage structures.

Advantages of using DBMS approach: Providing Backup and Recovery

- The **backup and recovery subsystem** of the DBMS is responsible for recovery.
- For example, if the computer system fails in the middle of a complex update transaction, the recovery subsystem is responsible for making sure that the database is restored to the state it was in before the transaction started executing.
- Disk backup is also necessary in case of a **catastrophic disk failure** (complete, sudden, often unexpected breakdown in a machine, electronic system, computer or network. Such a breakdown may occur as a result of a hardware event such as a disk drive crash, memory chip failure or surge on the power line).

Advantages of using DBMS approach: Providing Multiple User Interfaces

- Because many types of users with varying levels of technical knowledge use a database, a DBMS should provide a variety of user interfaces.
- These include:
  - apps for mobile users,
  - query languages for casual users,
  - programming language interfaces for application programmers,
  - forms and command codes for parametric users, and menu-driven interfaces
  - natural language interfaces for standalone users.
- Forms-style interfaces and menu-driven interfaces: **graphical user interfaces (GUIs)**

Advantages of using DBMS approach: Representing Complex Relationships among Data

- A database may include numerous varieties of data that are interrelated in many ways.
- A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to retrieve and update related data easily and efficiently.

GRADE_REPORT

Student_number	Student_name	Section_identifier	Course_number	Grade
17	Smith	112	MATH2410	B
17	Smith	119	CS1310	C
8	Brown	85	MATH2410	A
8	Brown	92	CS1310	A
8	Brown	102	CS3320	B
8	Brown	135	CS3380	A

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

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

Data models

- **Data abstraction** generally refers to the suppression of details of data organization and storage, and the highlighting of the essential features for an improved understanding of data.
- **A data model**—a collection of concepts that can be used to describe the structure of a database—provides the necessary means to achieve this abstraction.
- **Structure of a database** → the data types, relationships, and constraints that apply to the data

Data models: Categories

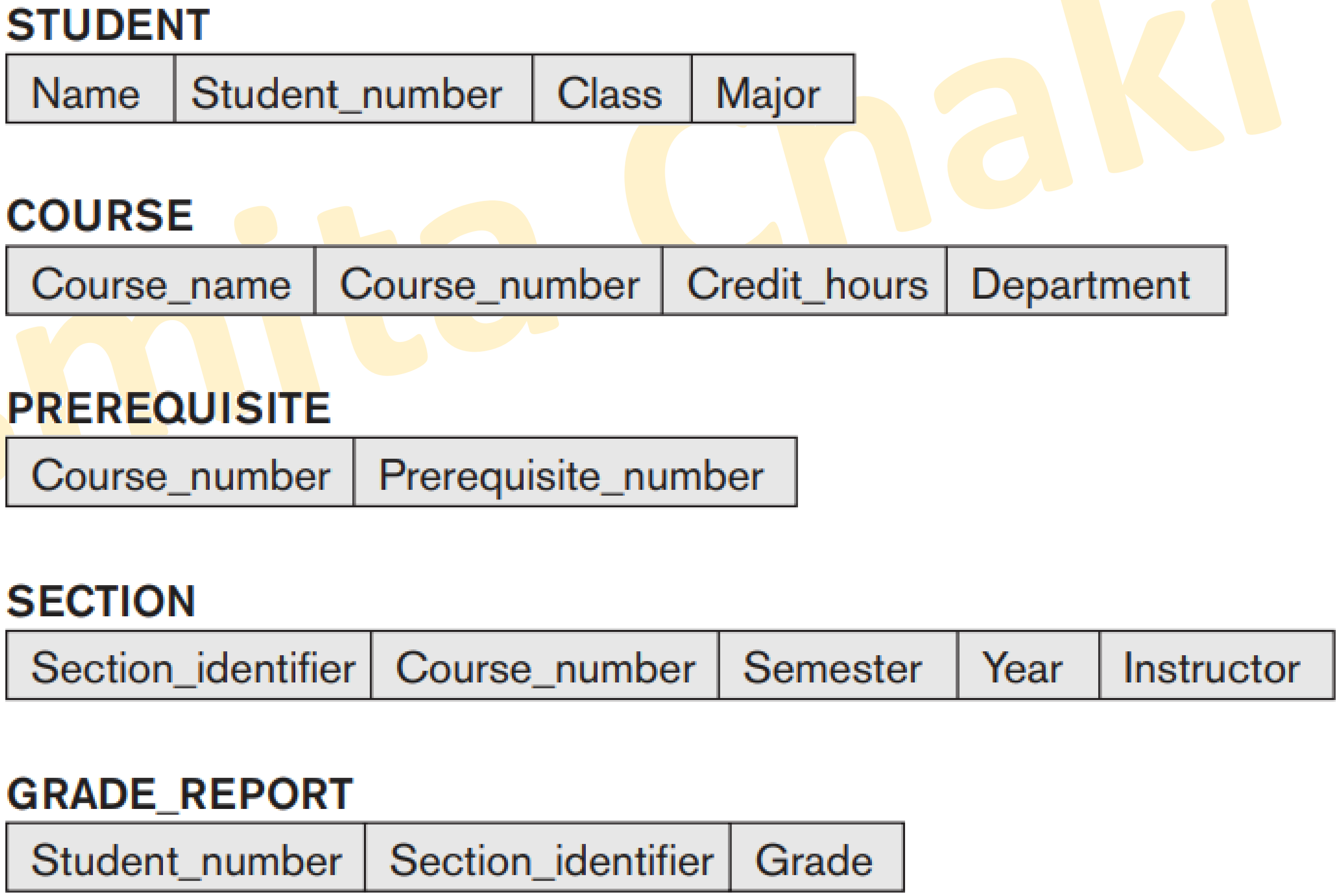
- **High-level** or **conceptual data models** provide concepts that are close to the way many users perceive data. Use concept such as
  - **Entities**: a real-world object or concept, such as an employee or a project from the miniworld that is described in the database
  - **Attributes**: some property of interest that further describes an entity, such as the employee’s name or salary.
  - **Relationships**: association among the entities, for example, a works-on relationship between an employee and a project.
- **Low-level** or **physical data models** provide concepts that describe the details of how data is stored on the computer storage media, typically magnetic disks.

Data models: Categories

- **Representational** (or **implementation**) data models provide concepts that may be easily understood by end users but that are not too far removed from the way data is organized in computer storage.
  - include the widely used **relational data model**, as well as the so-called legacy data models—the **network** and **hierarchical models**
  - represent data by using record structures → **record-based data models**
- **Self-describing data models**: the data storage in systems based on these models combines the description of the data with the data values themselves. Eg., XML, NOSQL etc

Schemas, Instances

- The description of a database is called the database **schema**, which is specified during database design and is not expected to change frequently
- A displayed schema is called a **schema diagram.**
- The diagram displays the structure of each record type but not the actual instances of records.
- Each object in the schema—such as STUDENT or COURSE—a **schema construct**.



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

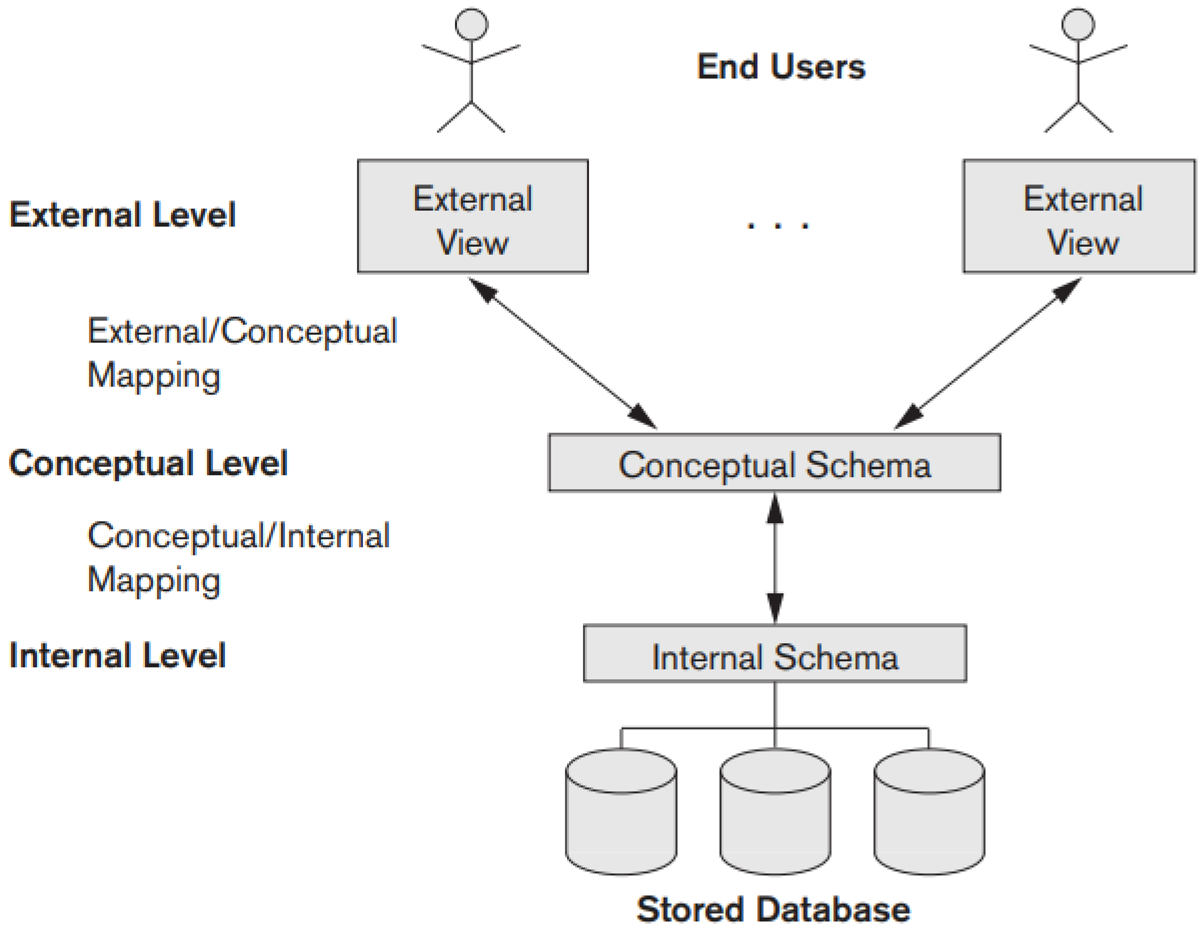
Student_number	Section_identifier	Grade

Schemas, Instances

- The data in the database at a particular moment in time is called a **database state** or **snapshot**.
- It is also called the current set of **occurrences** or **instances** in the database.
- In a given database state, each schema construct has its own current set of instances

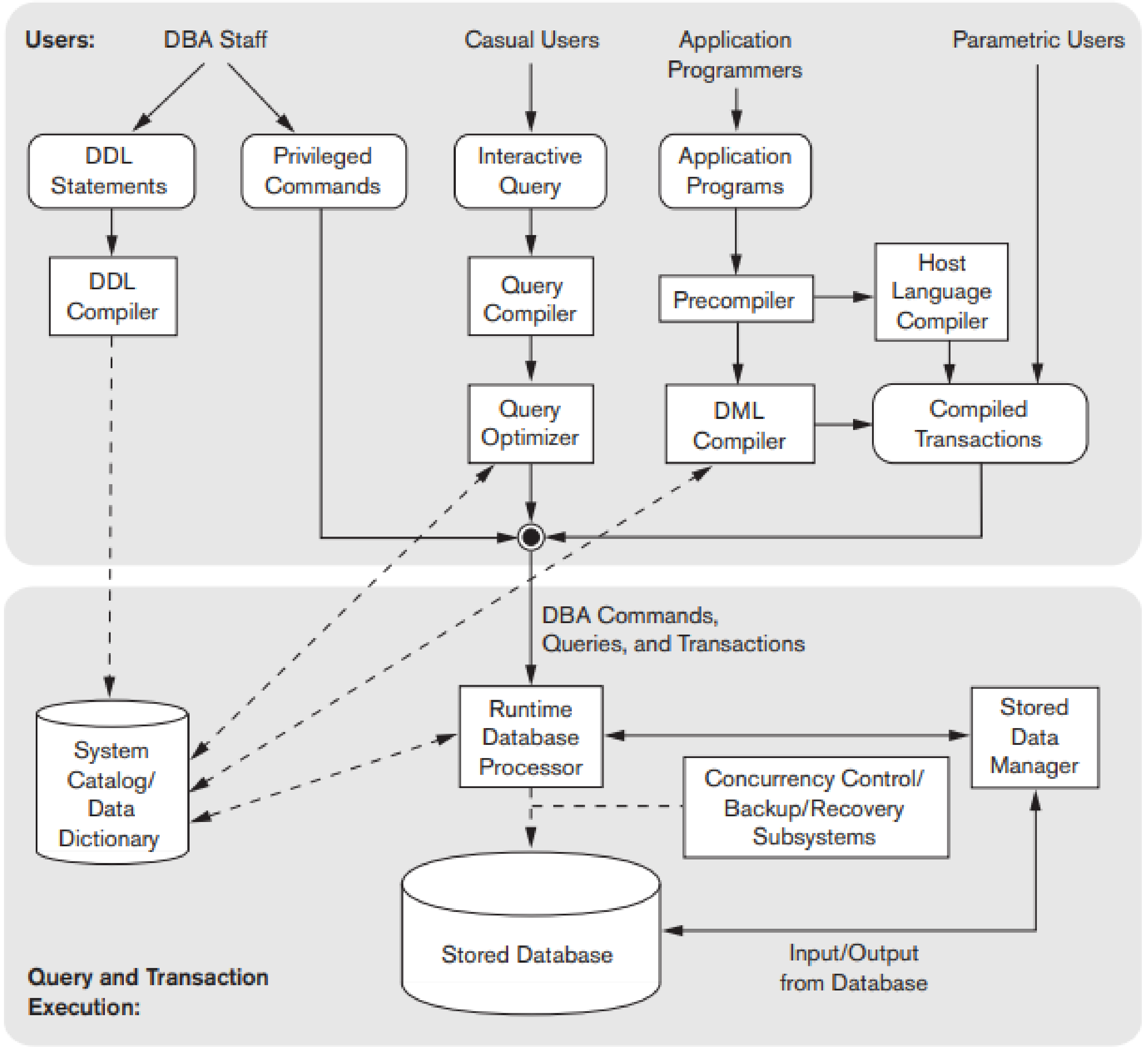
Three-Schema Architecture and Data Independence

- The goal of the three-schema architecture is to separate the user applications from the physical database.
- The **internal level** has an **internal schema**, which describes the physical storage structure of the database.
- The **conceptual level** has a **conceptual schema**, which describes the structure of the whole database for a community of users.
- The **external** or **view level** includes 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 from that user group.



The Database System Environment: DBMS Component Modules

- The **database** and the **DBMS catalog** are usually stored on disk.
- Access to the disk is controlled primarily by the **operating system (OS)**, which schedules disk read/write.
- Many DBMSs have their own buffer management module to schedule disk read/write.
- A higher-level **stored data manager module** of the DBMS controls access to DBMS information that is stored on disk, whether it is part of the database or the catalog.



The Database System Environment: Database System Utilities

- **Database utilities** help the DBA manage the database system.
- Common utilities have the following types of functions:
    - **Loading**: A loading utility is used to load existing data files—such as text files or sequential files—into the database.
    - **Backup**: A backup utility creates a backup copy of the database, usually by dumping the entire database onto tape or other mass storage medium. Incremental backups are also often used, where only changes since the previous backup are recorded.
    - **Database storage reorganization**: This utility can be used to reorganize a set of database files into different file organizations and create new access paths to improve performance.
    - **Performance monitoring**: Such a utility monitors database usage and provides statistics to the DBA.

Centralized and Client/Server Architectures for DBMS: Basic Client/Server Architectures

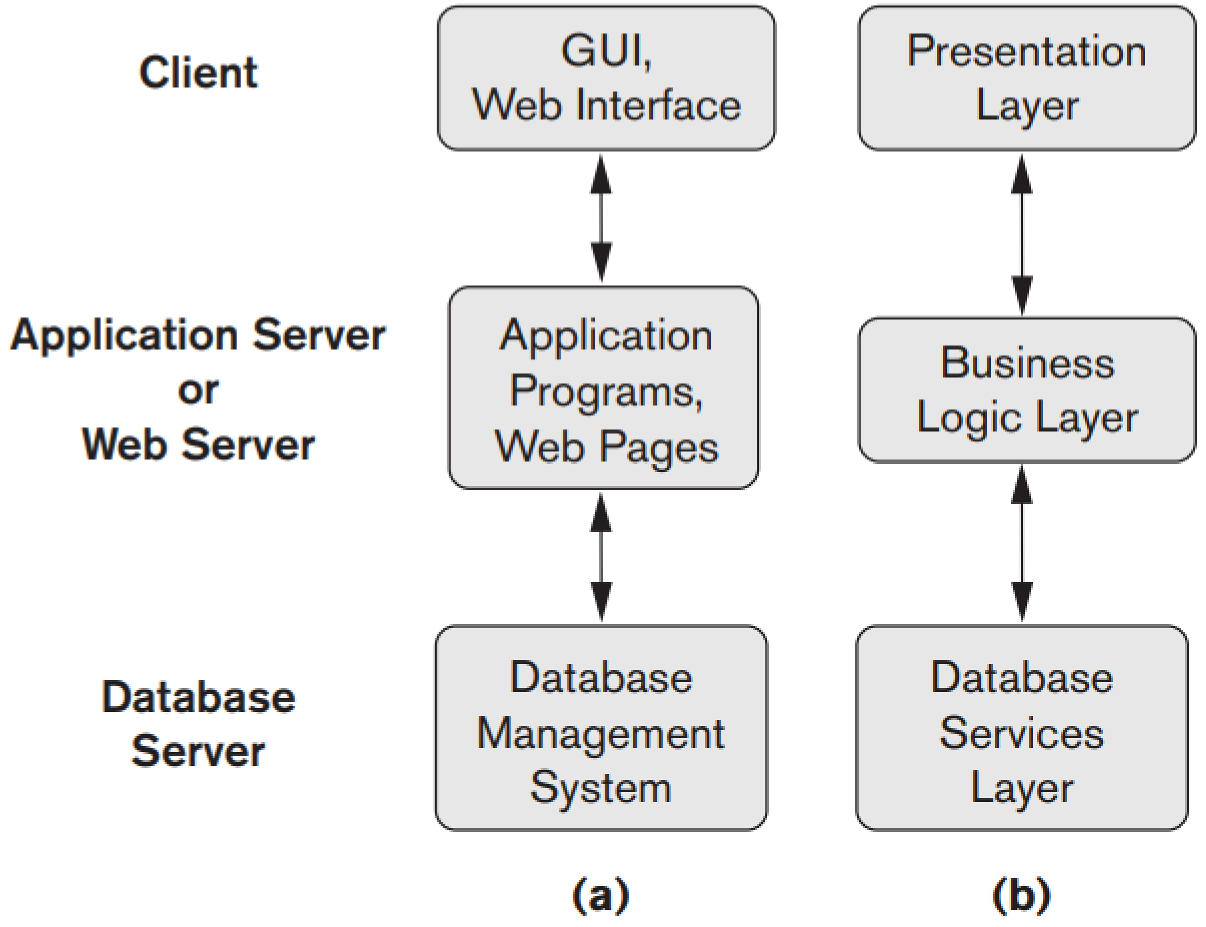
- The concept of client/server architecture assumes an underlying framework that consists of many PCs/workstations and mobile devices as well as a smaller number of server machines, connected via wireless networks or LANs and other types of computer networks.
- A client in this framework is typically a user machine that provides user interface capabilities and local processing.
- A server is a system containing both hardware and software that can provide services to the client machines, such as file access, printing, archiving, or database access.
- Two main types of basic DBMS architectures were created on this underlying client/server framework:
  - Two-tier and
  - Three-tier.

Centralized and Client/Server Architectures for DBMS: Two-Tier Client/Server Architectures

- Two-tier architecture: the software components are distributed over two systems: client and server
- the query and transaction functionality related to SQL processing remained on the server side: **query server** or **transaction server**
- In an RDBMS, the server is also often called an SQL server.
- The user interface programs and application programs can run on the client side.
- A standard called **Open Database Connectivity (ODBC)** provides an **application programming interface (API)**, which allows client-side programs to call the DBMS, as long as both client and server machines have the necessary software installed.
- The advantages of this architecture are its simplicity and seamless compatibility with existing systems.

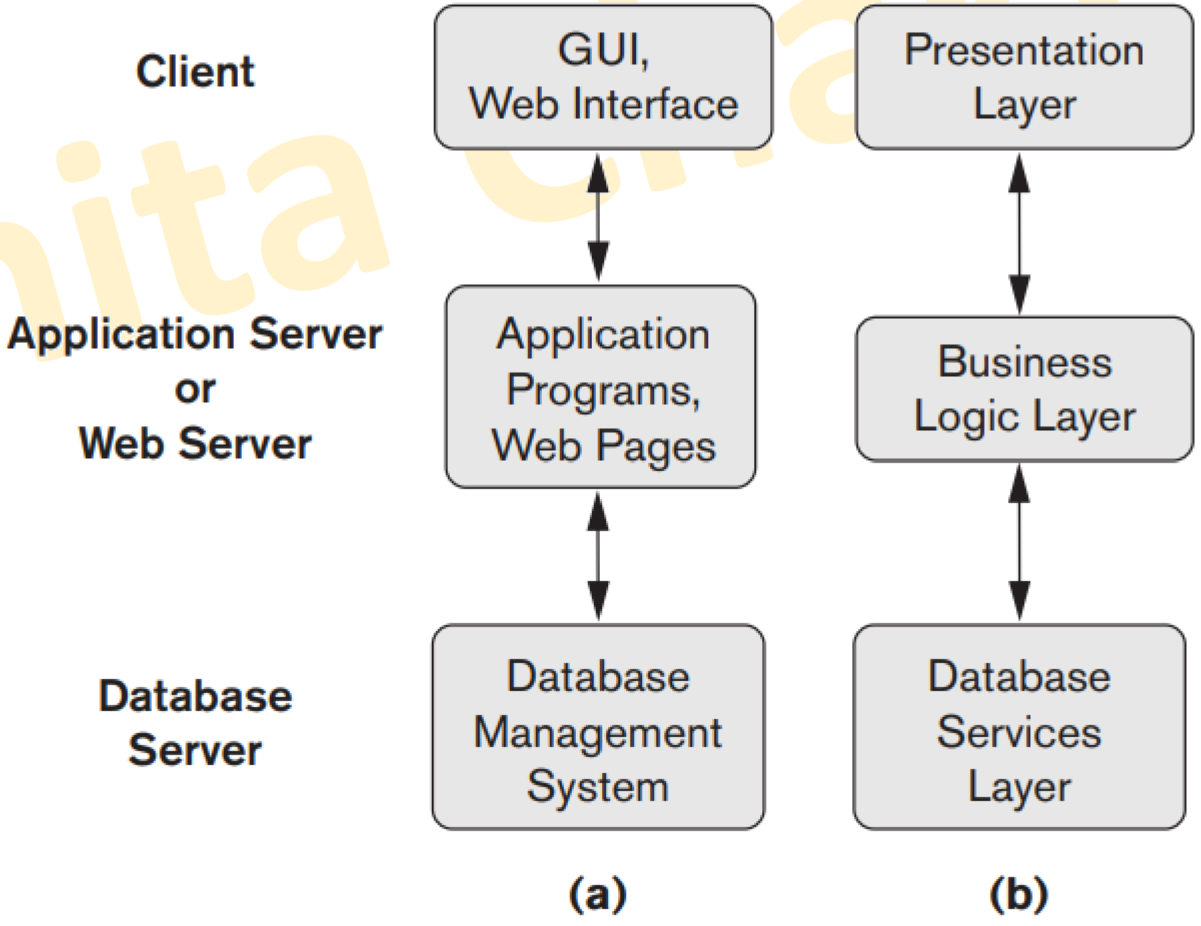
Centralized and Client/Server Architectures for DBMS: Three-Tier Client/Server Architectures

- Many Web applications use an architecture called the three-tier architecture, which adds an intermediate layer between the client and the database server, as illustrated in Figure (a).
- This intermediate layer or middle tier is called the application server or the Web server, depending on the application.



Centralized and Client/Server Architectures for DBMS: Three-Tier Client/Server Architectures

- Figure (b) shows another view of the three-tier architecture used by database and other application package vendors.
- The presentation layer displays information to the user and allows data entry. The business logic layer handles intermediate rules and constraints before data is passed up to the user or down to the DBMS.
- The bottom layer includes all data management services.
- The middle layer can also act as a Web server.



Classification of database management systems

- Several criteria can be used to classify DBMSs. The first is the data model on which the DBMS is based.
- We can categorize DBMSs based on the data model: relational, object, object-relational, NOSQL, key-value, hierarchical, network
- The relational DBMSs are evolving continuously, and, in particular, have been incorporating many of the concepts that were developed in object databases: **object-relational DBMS**
- Some experimental DBMSs are based on the XML (eXtended Markup Language) model, which is a tree-structured data model**: native XML DBMS**

Classification of database management systems

- The second criterion used to classify DBMSs is the number of users supported by the system.
  - **Single-user systems** support only one user at a time and are mostly used with PCs.
  - **Multiuser systems**, which include the majority of DBMSs, support concurrent multiple users.
- The third criterion is the number of sites over which the database is distributed.
  - A DBMS is **centralized** if the data is stored at a single computer site.
  - A **distributed DBMS** (DDBMS) can have the actual database and DBMS software distributed over many sites connected by a computer network.