

COMP 3311 DATABASE MANAGEMENT SYSTEMS

LECTURE 1 DATABASE MANAGEMENT SYSTEMS

DATABASE MANAGEMENT SYSTEMS: OUTLINE

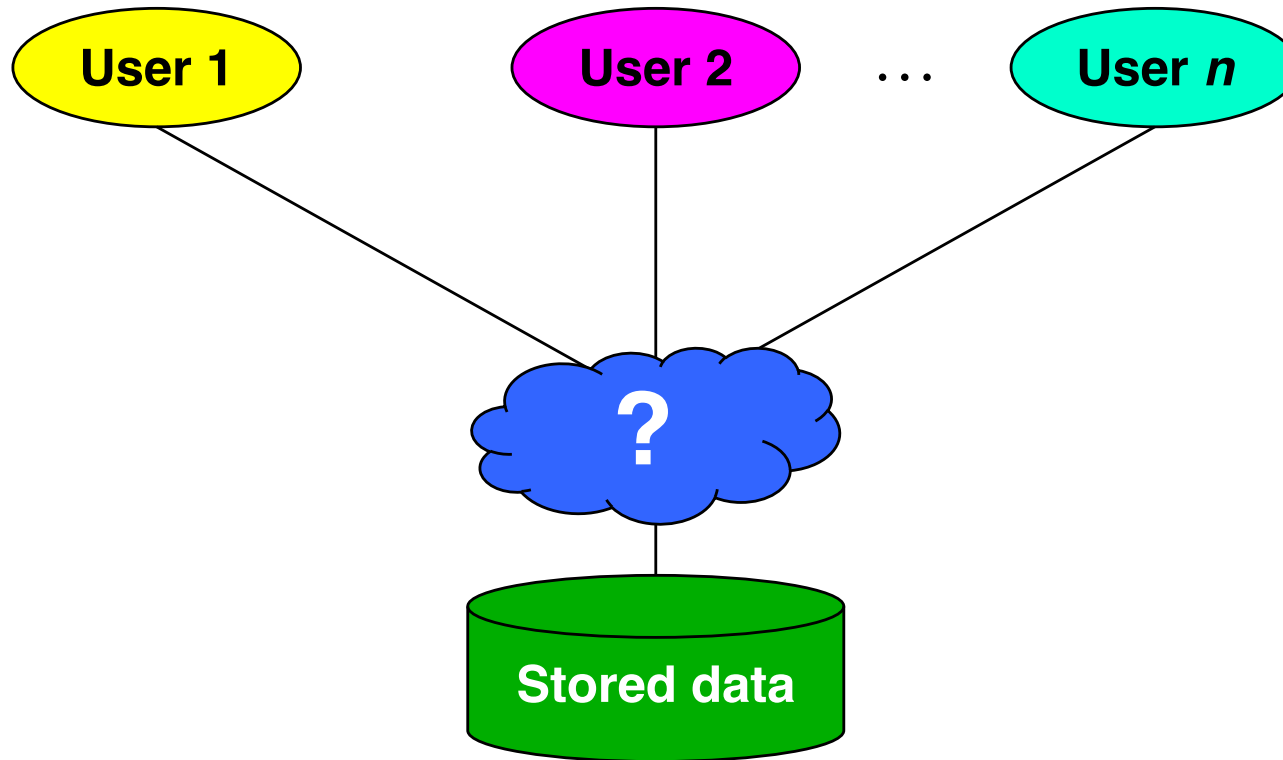
What Is A Database Management System (DBMS)?

Why Do We Need Database Management Systems?

How Does A DBMS Manage Data?

What Are The Major Components Of A DBMS?

THE PROBLEM WE WANT TO ADDRESS



How best to manage stored data?

👉 **organize, access, share, protect, ...**

DATABASE

A *database* is a collection of related data within a specific business process or problem setting.

Data are facts such as age, salary, name, address, etc.

- A **database** has the following properties.
 - It is designed, built and populated with data for a specific purpose.
Applications: sales, human resources, manufacturing, banking, real estate, stock trading, inventory management, social media, ride sharing, ...
 - It usually represents some aspect of the real world.
 - The data have some inherent meaning.

 **Databases touch all aspects of our lives!**

DATABASE MANAGEMENT SYSTEM

A database management system (DBMS) is a general-purpose software system used to manage databases.

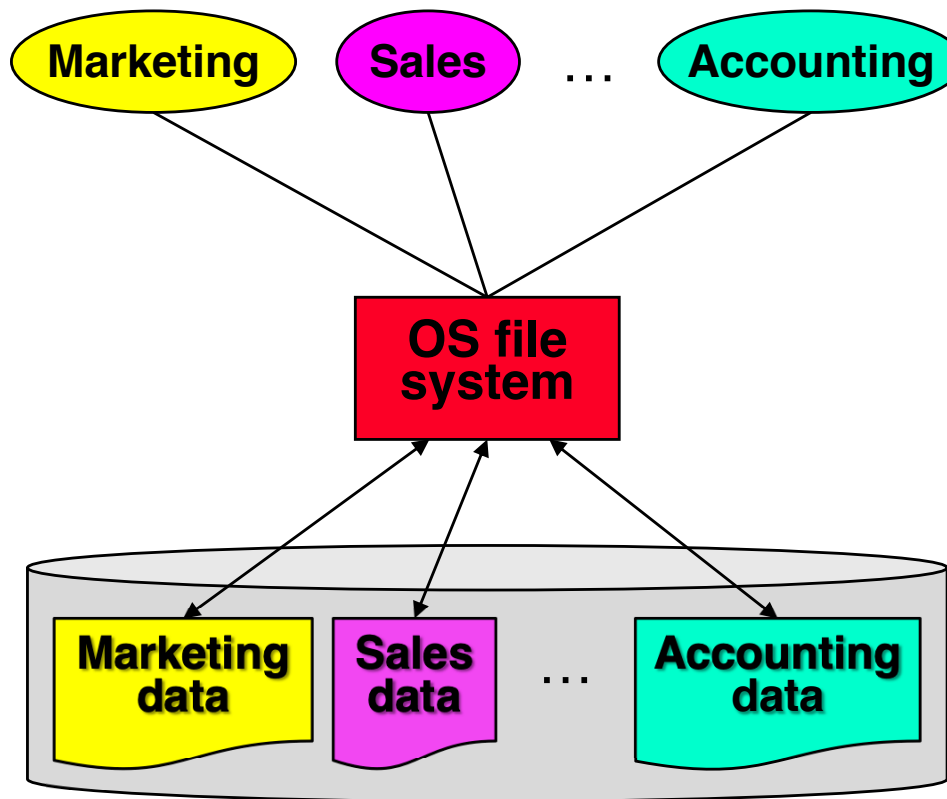
- A DBMS provides support/facilities for:
 - **defining** what data to store (types, structures, constraints)
 - **storing** and **managing** data on a storage device
 - **manipulating** data (querying, updating)
 - **sharing** data among many users
 - **protecting** data from loss, corruption, unauthorized access

| <i>Company</i> | <i>Product</i> |
|-----------------------|-----------------------|
| Oracle | Oracle Database |
| IBM | DB2 |
| Microsoft | Access, SQL Server |
| Sybase | Adaptive Server |
| Informix | Dynamic Server |

A DBMS provides an environment for managing data that is both *convenient* and *efficient* to use.

FILE-BASED APPROACH TO MANAGING DATA

Applications access stored data using the facilities provided by an **operating system file system**.



Drawbacks

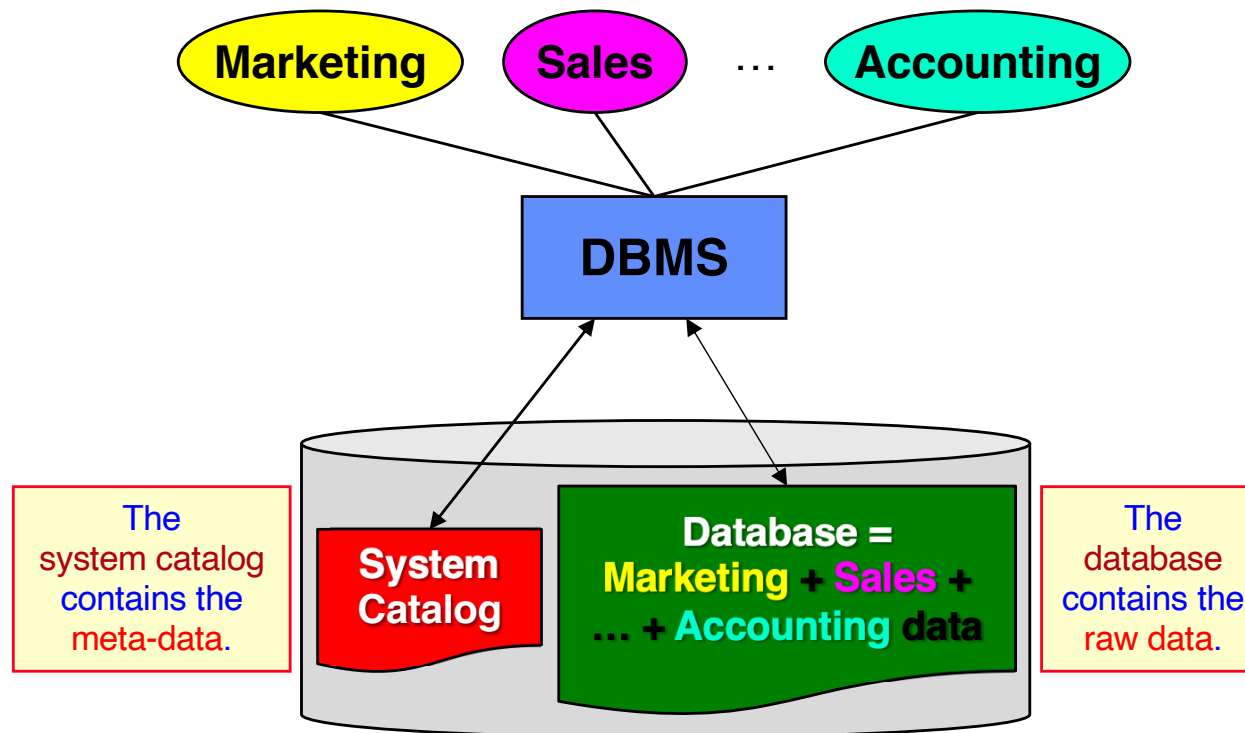
- Data duplication and inconsistency
- Difficulty meeting unanticipated needs
- Data isolation
- Data integrity problems
- Atomicity of updates
- Concurrent access
- Security problems

FILE-BASED APPROACH: DRAWBACKS

- **Data duplication and inconsistency**
 - Data may be duplicated in several files leading to data inconsistency.
- **Difficulty meeting unanticipated needs**
 - Unanticipated data needs often require writing new application programs.
- **Data isolation**
 - Data are often scattered in different files and in different formats making writing new application programs challenging.
- **Data integrity problems**
 - Constraints on data become part of the program code making them hard to change and making it hard to know what constraints apply.
 - New constraints are often hard to enforce especially across several files.
- **Atomicity of updates**
 - Since changes to data may involve several updates (e.g., a bank account transfer), hardware/software failures can cause data to become inconsistent.
- **Concurrent access**
 - Data may become inconsistent due to simultaneous access by many users (e.g. lost update).
- **Security problems**
 - Enforcing security can be challenging due to *ad hoc* data management.

DATABASE APPROACH TO MANAGING DATA

Applications access stored data using the facilities provided by a **DBMS**.



Major Principles

- **integrates** an organization's data.
- **separates** meta-data (description of data) and data.
- **supports** multiple views of data.
- **controls** definition and access of data centrally.

A DBMS provides automated solutions for the data management problems encountered when using file systems.

DATA MODELS

A data model is a set of concepts for describing data that defines
– properties – relationships – semantics – constraints

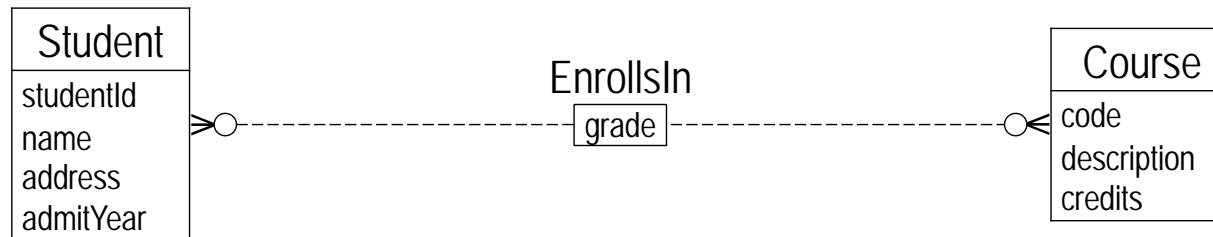
A data model is the fundamental mechanism used by a DBMS to logically describe and organize data and consists of:

1. **data structure types** \Rightarrow specify logical organization (properties, relationships and semantics)
2. **integrity constraints** \Rightarrow specify constraints (restrictions on properties and relationships)
3. **operations** \Rightarrow specify how data is accessed (e.g., R, I, U, D—Read, Insert, Upsert, Delete)

A data model is used to describe and organize data as well as to state any restrictions on the data.

DATA MODELS: EXAMPLE

Entity-Relationship (E-R) model



👉 Users view data as **entities** and explicit **relationships** among entities.

Relational model



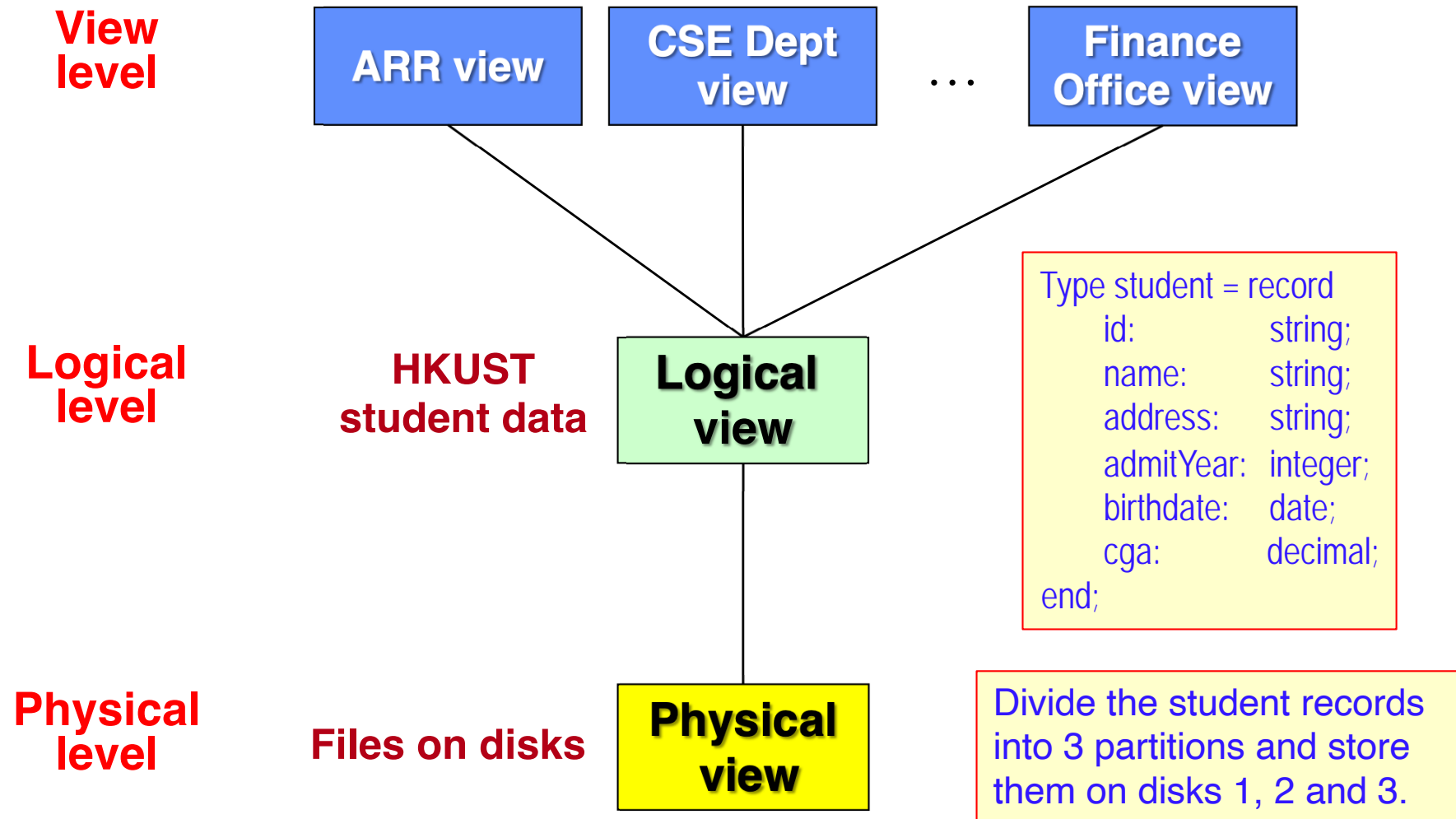
👉 Users view data as **tables** and implicit **relationships** among tables.

LEVELS OF ABSTRACTION

- One big problem in application development is the *separation* of **application programs** (i.e., code) from the **data** that they access.
- **Do I have to change my application program when I ...**
 - replace my hard drive?
 - partition the data into two physical files (or merge two physical files into one)?
 - store salary as a floating-point number instead of an integer?
 - develop other applications that use the same data?
 - add more data fields to support other applications?
 - index the data using a B⁺-tree instead of a hash index?

A DBMS provides separation of application programs and data via several levels of abstraction.

LEVELS OF ABSTRACTION (CONT'D)



LEVELS OF ABSTRACTION (CONT'D)

Physical level: Describes how the data are stored on disk.

- Divide the student records into 3 partitions and store them on disks 1, 2 and 3.

Logical level: Describes what data are stored in the database and the relationships among the data.

- E.g., students are related to courses and to fee information.
- Hides the details of how the data are physically stored from users.

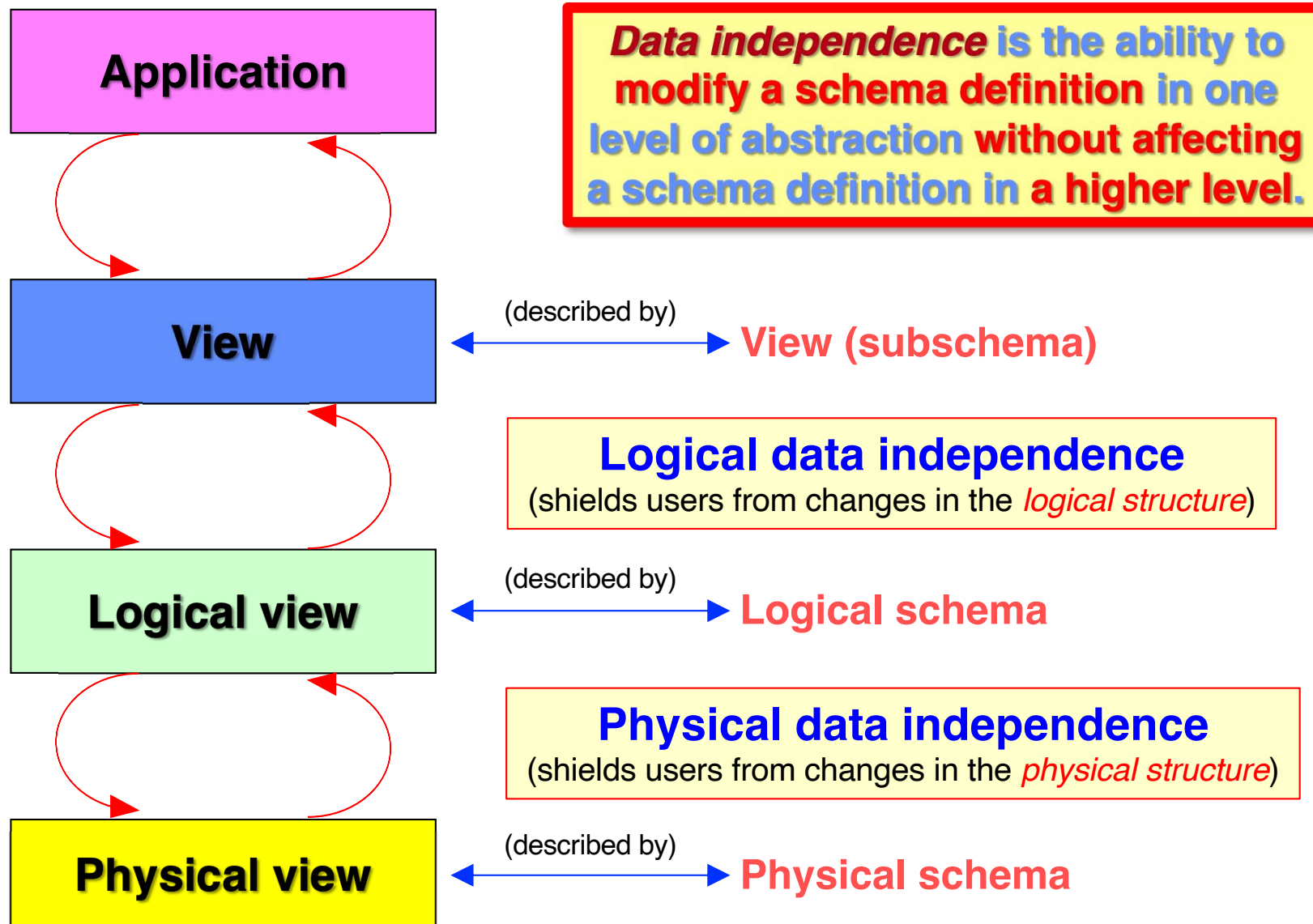
View level: Defines a subset (part) of the database for a particular application.

- Views can hide information for security/privacy purposes (e.g., cga) or add information not actually stored (e.g., age).

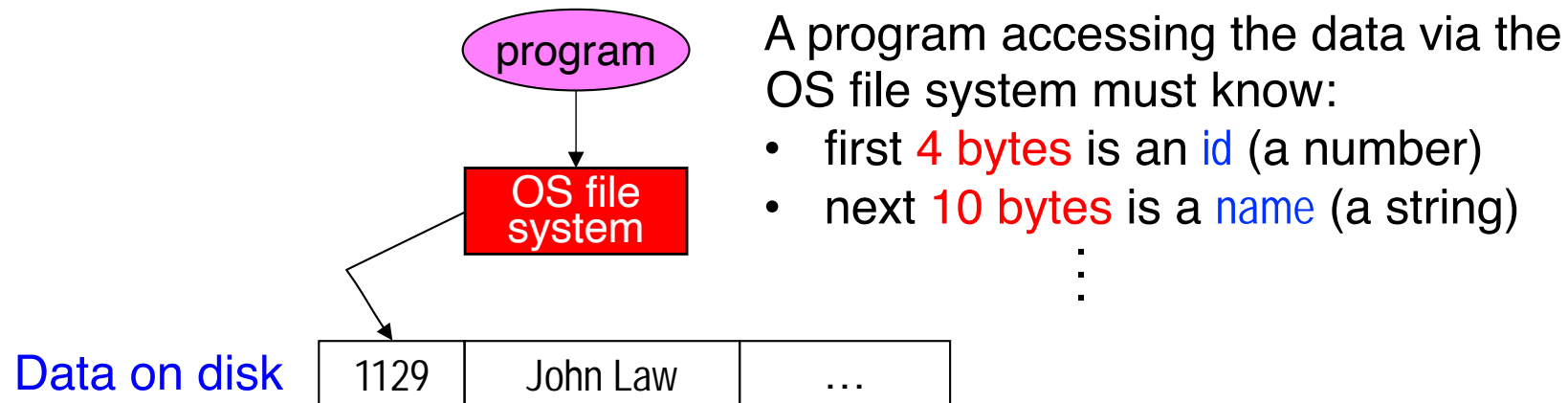
SCHEMAS AND INSTANCES

- A **database schema** describes the **overall design of a database according to a data model**.
 - ✎ **Stored in the system catalog; changes infrequently, if at all.**
- A DBMS uses several schemas, one for each level of abstraction, which **describes the data** at the corresponding level.
 - A **view** (subschema) describes the **data that a user can access**.
 - A **logical schema** describes the **logical structure of the database** (e.g., the set of students, courses and the relationship between them).
 - A **physical schema** describes the **file formats and locations where the data are stored on disk**.
- A **database instance** refers to the **actual content** of the database at a particular **point in time**.
 - ✎ **Conforms to its corresponding schema; changes frequently as data are changed.**

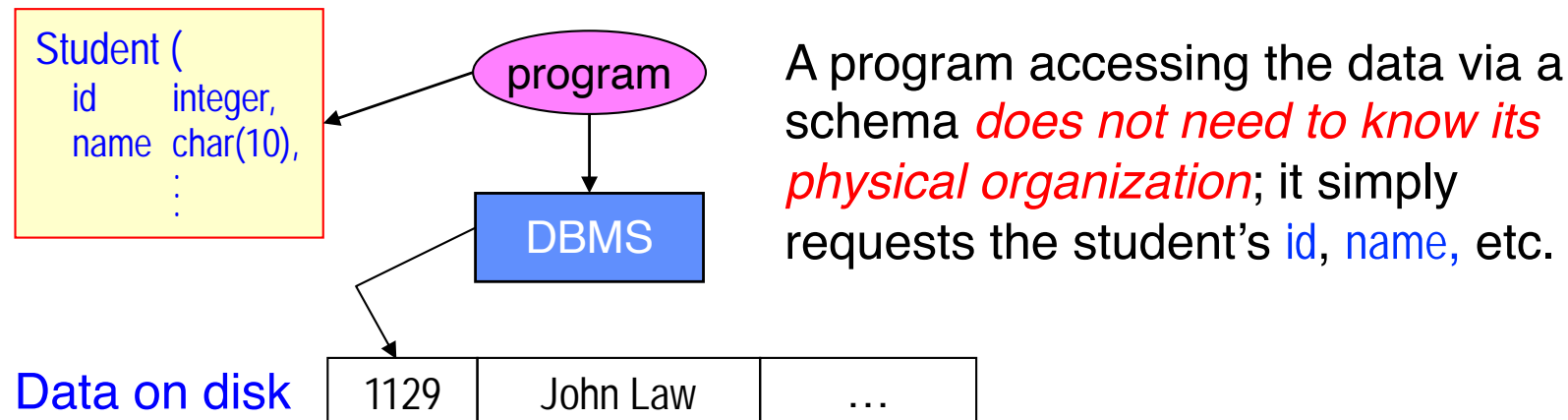
DATA INDEPENDENCE



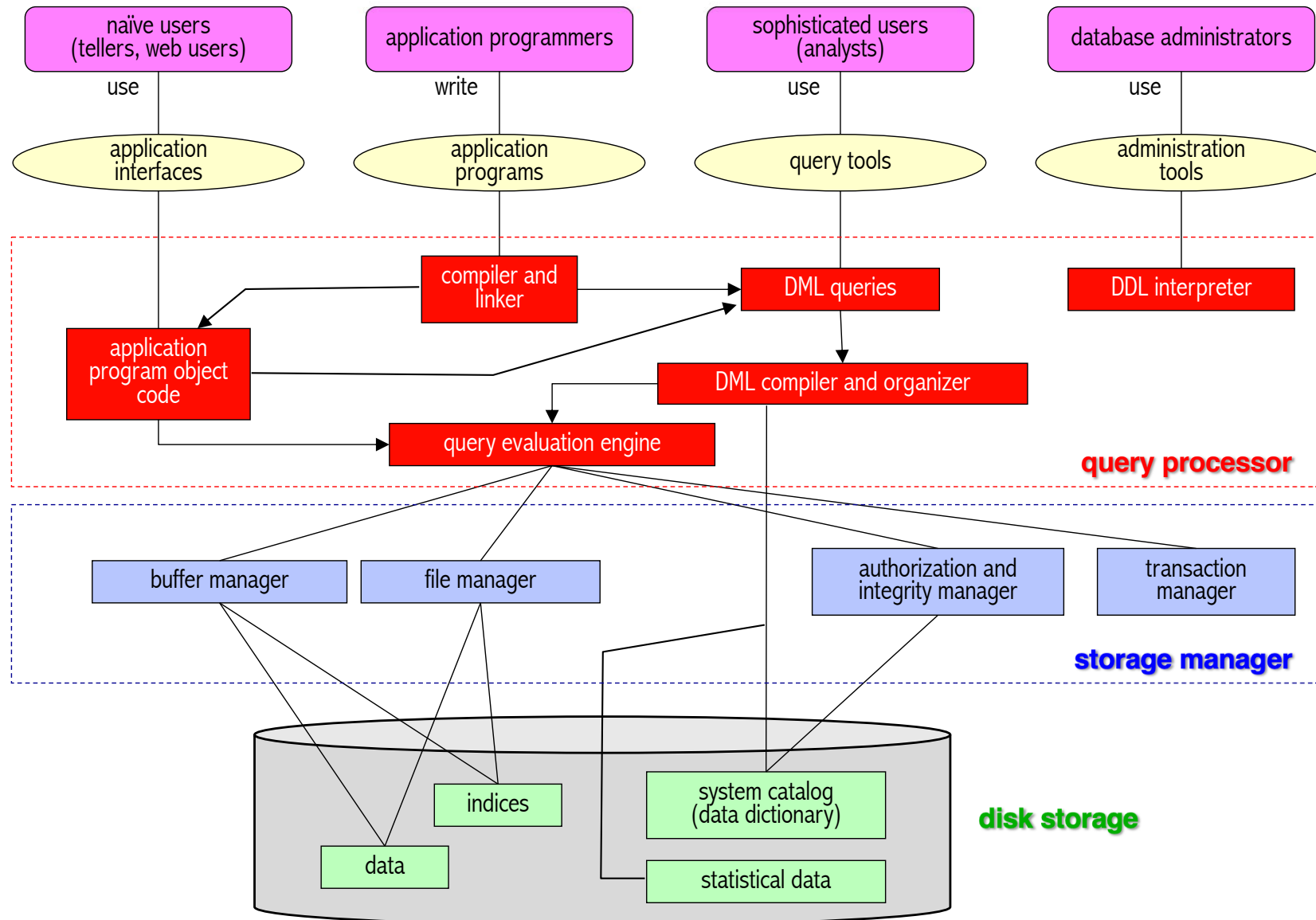
DATA INDEPENDENCE: **EXAMPLE**



Schema



DBMS ARCHITECTURE



DBMS ARCHITECTURE: STORAGE MANAGER

- The **storage manager** provides the **interface** between the **low-level data** stored in the database and the **application programs and queries** submitted to the system.

 **It is responsible for storing, retrieving and updating data in the database.**

- The storage manager components include:
 - **buffer manager** (*fetches data from disk and caches it in main memory*)
 - **file manager** (*manages allocation of disk space*)
 - **authorization and integrity manager** (*controls database access; enforces constraints*)
 - **transaction manager** (*ensures database consistency despite system failures and concurrent access by multiple users*)

DBMS ARCHITECTURE: QUERY PROCESSOR

- The **query processor** translates nonprocedural queries and updates into efficient physical disk-level operations.

 **It simplifies and facilitates data access while providing good performance.**

- The query processor components include two languages:
 - **data definition language (DDL)** used to **define the database schema** (i.e., meta-data) in a formal language according to a data model.
 - **data manipulation language (DML)** used to **access and manipulate data** as organized by a data model.
 - **Procedural DML**: user specifies **both what** data is required **and how** to get the data.
 - **Nonprocedural (declarative) DML**: user specifies **only what** data is required **not** how to get the data
 - **Structured Query Language (SQL)** is a nonprocedural DML used in all commercial relational DBMSs.

DBMS USERS

End Users

Naïve users

- Use existing application programs (e.g., print monthly sales report).
- Interact with applications through a graphical user interface (GUI).

Application programmers

- Develop applications that interact with DBMS through DML calls.

Sophisticated users

- Issue queries either directly using a database query language (e.g., SQL) or via tools such as data analysis software.

Database Administrator (DBA)

- Coordinates all activities of the database system.
 - Defines and maintains the schemas.
 - Defines and maintains the physical organization.
 - Monitors and optimizes the database performance.
 - Monitors access and grants access rights.

A DBA must have a good understanding of an enterprise's information resources and needs.

DATABASE MANAGEMENT SYSTEMS: SUMMARY

- **Database management systems (DBMSs)** address the limitations of OS file systems for managing an enterprise's data.
- **Data models** are the foundation for developing a database—the entity-relationship (E-R) model and relational model are commonly used in practice.
- **Data independence** is fundamental to understanding how a DBMS manages data at different abstraction levels.
- **A DBMS provides many facilities** for query processing and storage management to efficiently handle the data management and data access needs of various users.