Data models, schemas, and instances;

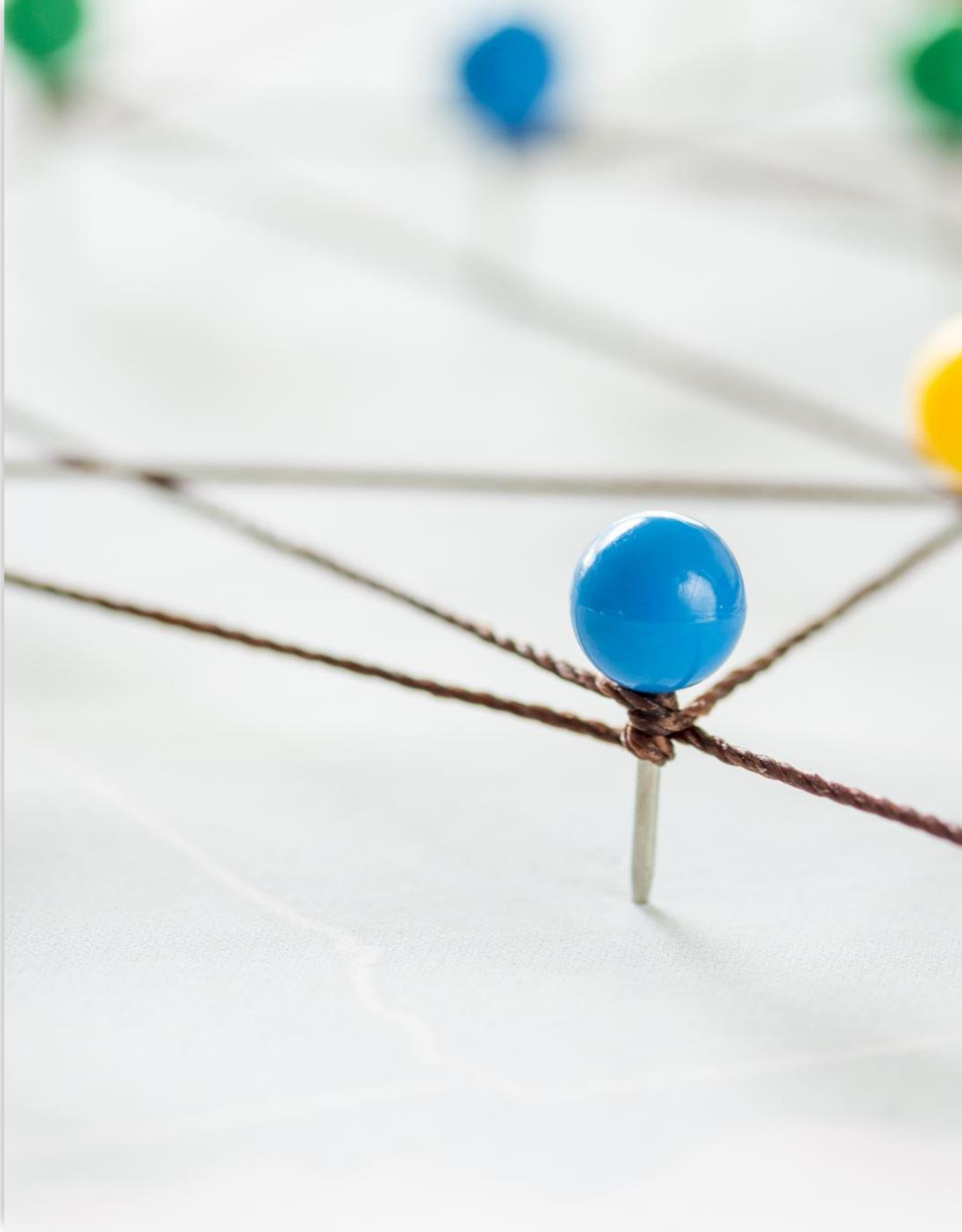
Data model

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.2 By *structure of a database* we mean the data types, relationships, and constraints that apply to the data. Most data models also include a set of **basic operations** for specifying retrievals and updates on the database.

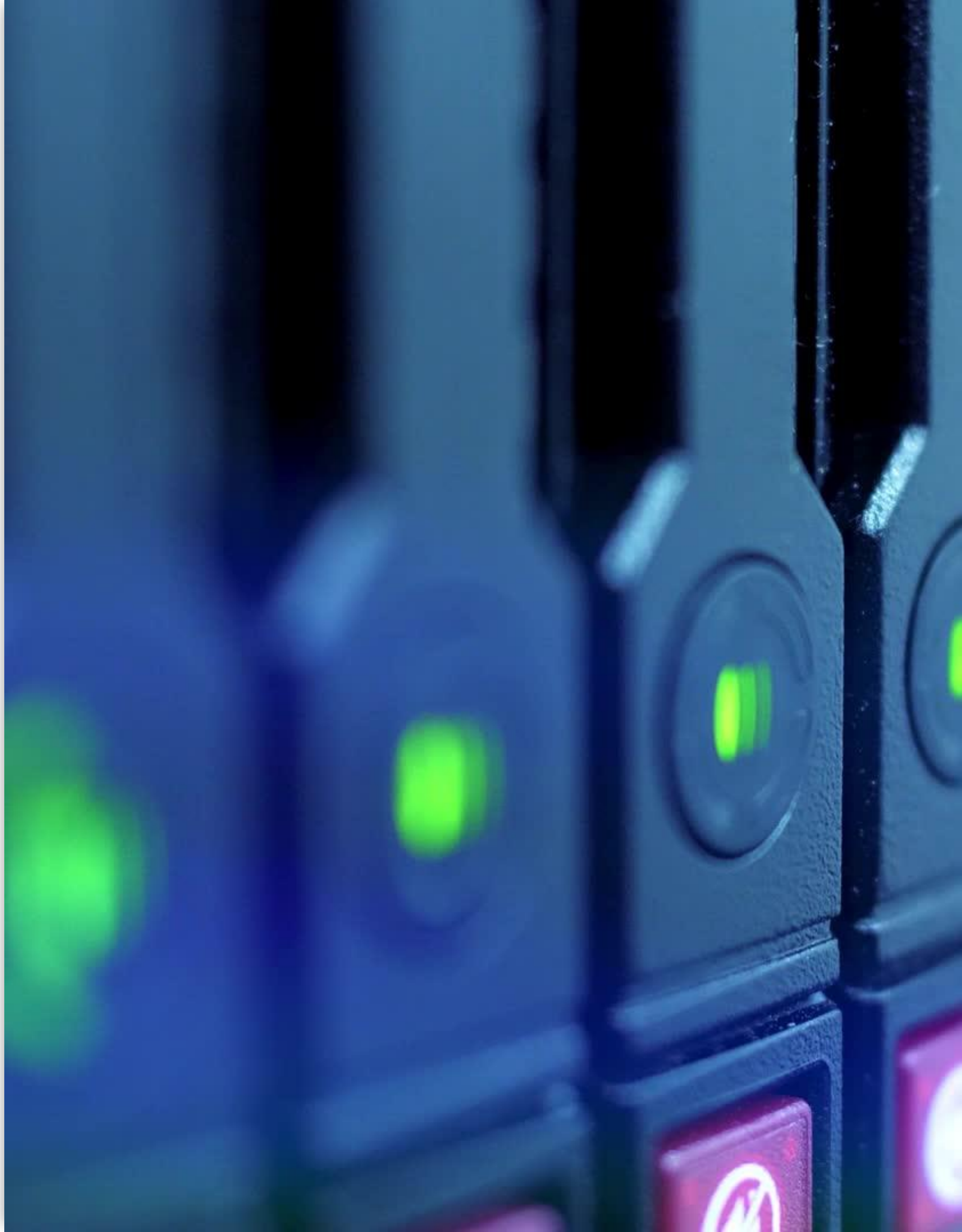
Categories of Data Models-

- **High-level** or **conceptual data models** provide concepts that are close to the way many users perceive data, Conceptual data models use concepts such as entities, attributes, and relationships.
- An **entity** represents a real-world object or concept, such as an employee or a project from the miniworld that is described in the database.
- An **attribute** represents some property of interest that further describes an entity, such as the employee's name or salary.
- A **relationship** among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project.



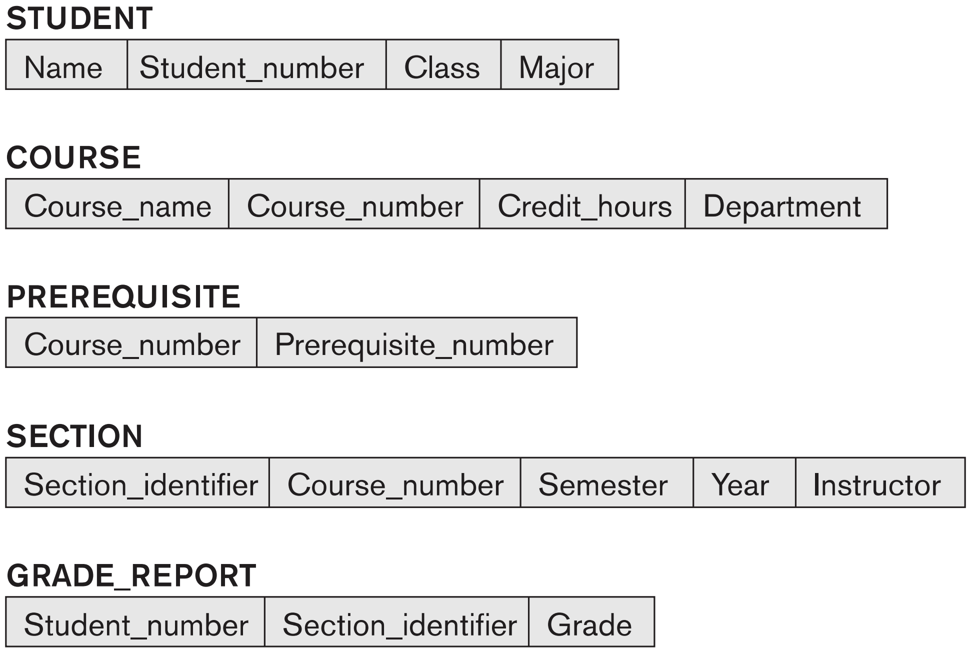
Categories of Data Models-

- **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.
- **Representational data models** hide many details of data storage on disk but can be implemented on a computer system directly.



Schemas

- it is important to distinguish between the *description* of the database and the *database itself*.
- 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**



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

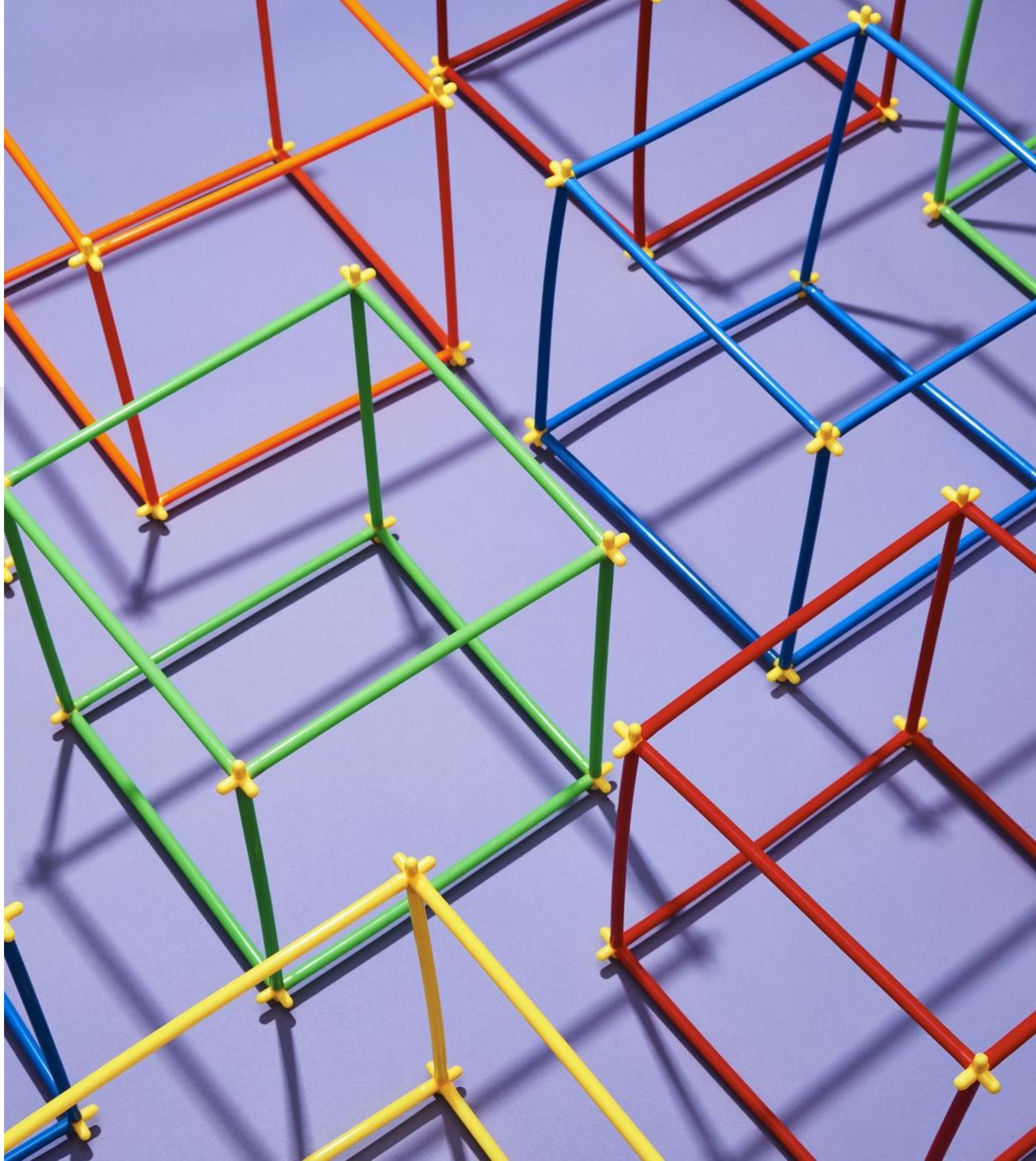
GRADE_REPORT

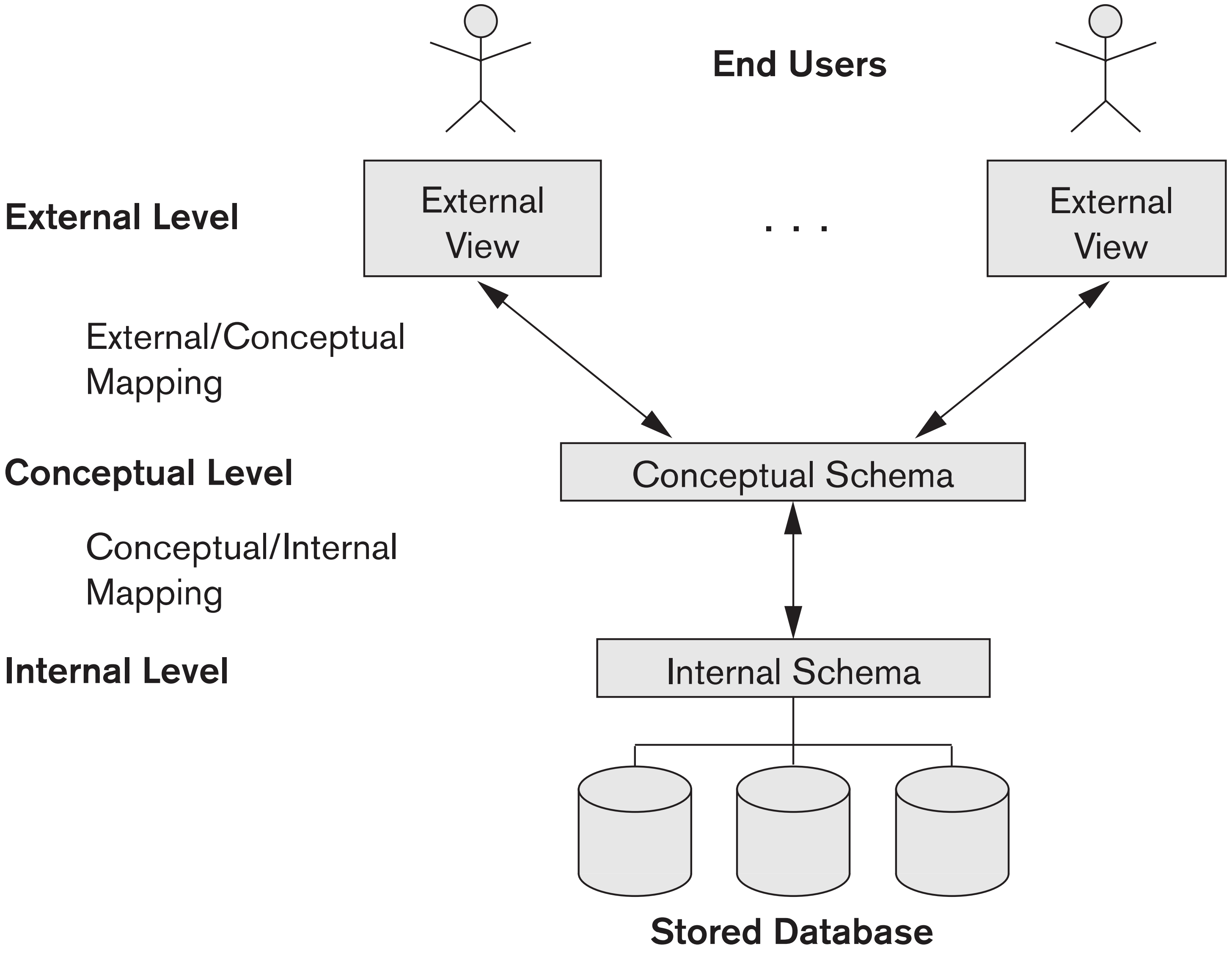
Student_number	Section_identifier	Grade

The Three-Schema Architecture

- The goal of the three-schema architecture is to separate the user applications from the physical database.
- In this architecture, schemas can be defined at the following three levels:

1. **The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.**
2. **The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints.**
3. **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.**





End Users

External Level

External View

. . .

External View

External/Conceptual Mapping

Conceptual Level

Conceptual Schema

Conceptual/Internal Mapping

Internal Level

Internal Schema

Stored Database

Data Independence

The three-schema architecture can be used to further explain the concept of **data independence**, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at the next higher level. We can define two types of data independence:

Logical data independence is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).

Physical data independence is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized—for example, by creating additional access structures—to improve the performance of retrieval or update. If the same data as before remains in the database, we should not have to change the conceptual schema.