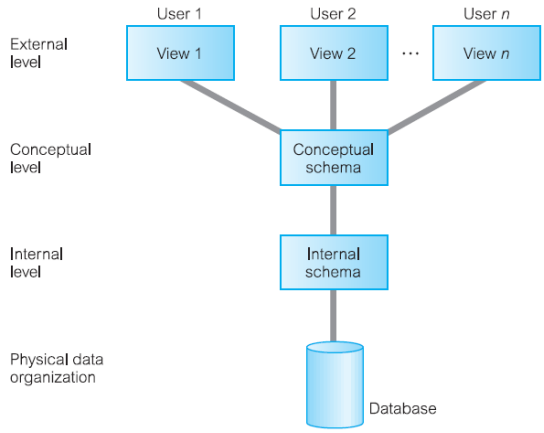
Topic Two: The Three- Level Database Architecture

- A major aim of a database system is to provide users with an abstract view of data, hiding certain details of how data is stored and manipulated.
- Therefore, the starting point for the design of a database must be an abstract and general description of the information requirements of the organization that is to be represented in the database.
- A database is a shared resource, each user may require a different view of the data held in the database.
- To satisfy these needs, the architecture of most commercial DBMSs available today is based to some extent on the so-called ANSI-SPARC architecture.
- Fundamental point of these architecture is the identification of three levels of abstraction, that is, three distinct levels at which data items can be described.
- The levels form a **three-level architecture** comprising an **external**, a **conceptual**, and an **internal** level.
- The way users perceive the data is called the **external level**.
- The way the DBMS and the operating system perceive the data is the **internal level**, where the data is actually stored using the data structures and file.



- The **conceptual level** provides both the **mapping** and the desired **independence** between the external and internal levels.
- The objective of the three-level architecture is to separate each user's view of the database from the way the database is physically represented.
- There are several reasons why this separation is desirable:
  - ❖ Each user should be able to access the same data, but have a different customized view of the data. Each user should be able to change the way he or she views the data, and this change should not affect other users.

- ❖ Users should not have to deal directly with physical database storage details, such as indexing or hashing. In other words, a user's interaction with the database should be independent of storage considerations.
- ❖ The Database Administrator (DBA) should be able to change the database storage structures without affecting the users' views.
- ❖ The internal structure of the database should be unaffected by changes to the physical aspects of storage, such as the changeover to a new storage device.
- ❖ The DBA should be able to change the conceptual structure of the database without affecting all users.

External Level

- The users' view of the database. This level describes that part of the database that is relevant to each user.
- The external level consists of a number of different external views of the database.
- Each user has a view of the 'real world' represented in a form that is familiar for that user.
- The external view includes only those entities, attributes, and relationships in the 'real world' that the user is interested in.
- Other entities, attributes, or relationships that are not of interest may be represented in the database, but the user will be unaware of them.
- In addition, different views may have different representations of the same data.
- For example, one user may view dates in the form (day, month, year), while another may view dates as (year, month, day).
- Some views might include derived or calculated data: data not actually stored in the database as such, but created when needed.

Conceptual Level

- The community view of the database. This level describes what data is stored in the database and the relationships among the data.
- The middle level in the three-level architecture is the conceptual level. This level contains the logical structure of the entire database as seen by the DBA.
- It is a complete view of the data requirements of the organization that is independent of any storage considerations.
- The conceptual level represents:
    - ❖ all entities, their attributes, and their relationships;
    - ❖ the constraints on the data;
    - ❖ semantic information about the data;
    - ❖ security and integrity information.

- The conceptual level supports each external view, in that any data available to a user must be contained in, or derivable from, the conceptual level.
- However, this level must not contain any storage-dependent details.
- For instance, the description of an entity should contain only data types of attributes (for example, integer, real, character) and their length (such as the maximum number of digits), but not any storage considerations, such as the number of bytes occupied.

Internal Level

- The physical representation of the database on the computer. This level describes how the data is stored in the database.
- The internal level covers the physical implementation of the database to achieve optimal runtime performance and storage space utilization.
- It covers the data structures and file organizations used to store data on storage devices.
- It interfaces with the operating system access methods (file management techniques for storing and retrieving data records) to place the data on the storage devices, build the indexes, retrieve the data, and so on.
- The internal level is concerned with such things as:
    - ❖ storage space allocation for data and indexes;
    - ❖ record descriptions for storage (with stored sizes for data items)
    - ❖ record placement;
    - ❖ data compression and data encryption techniques.
- Below the internal level there is a physical level that may be managed by the operating system under the direction of the DBMS.
- However, the functions of the DBMS and the operating system at the physical level are not clear-cut and vary from system to system.
- Some DBMSs take advantage of many of the operating system access methods, while others use only the most basic ones and create their own file organizations.
- The physical level below the DBMS consists of items only the operating system knows, such as exactly how the sequencing is implemented and whether the fields of internal records are stored as contiguous bytes on the disk.

Schemas, Mappings, and Instances

- The overall description of the database is called the **database schema**.
- There are three different types of schema in the database and these are defined according to the levels of abstraction of the three-level architecture.
- At the highest level, we have multiple **external schemas** (also called **subschemas**) that correspond to different views of the data.
- At the conceptual level, we have the **conceptual schema**, which describes all the entities, attributes, and relationships together with integrity constraints.

- At the lowest level of abstraction we have the **internal schema**, which is a complete description of the internal model, containing the definitions of stored records, the methods of representation, the data fields, and the indexes and storage structures used.
- There is only one conceptual schema and one internal schema per database.

Mapping

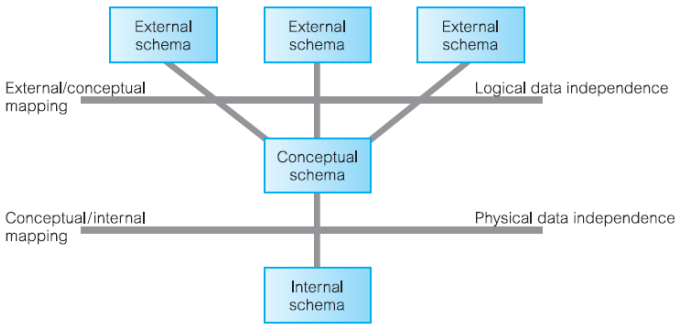
- The DBMS is responsible for mapping between these three types of schema.
- It must also check the schemas for consistency; in other words, the DBMS must check that each external schema is derivable from the conceptual schema, and it must use the information in the conceptual schema to map between each external schema and the internal schema.
- The conceptual schema is related to the internal schema through a **conceptual/internal mapping**.
- This enables the DBMS to find the actual record or combination of records in physical storage that constitute a **logical record** in the conceptual schema, together with any constraints to be enforced on the operations for that logical record.
- It also allows any differences in entity names, attribute names, attribute order, data types, and so on, to be resolved. Finally, each external schema is related to the conceptual schema by the **external/conceptual mapping**.
- This enables the DBMS to map names in the user's view on to the relevant part of the conceptual schema.
- The conceptual level is then mapped to the internal level, which contains a physical description of the structure for the conceptual record.
- Note that the order of fields at the internal level is different from that at the conceptual level. Again, the DBMS maintains the **conceptual/internal mapping**.

Schema and Instance

- It is important to distinguish between the description of the database and the database itself.
- The description of the database is the **database schema**.
- The schema is specified during the database design process and is not expected to change frequently.
- However, the actual data in the database may change frequently; for example, it changes every time we insert details of a new member of staff or a new property.
- The data in the database at any particular point in time is called a **database instance**.
- Therefore, many database instances can correspond to the same database schema.
- The schema is sometimes called the **intension** of the database, while an instance is called an **extension** (or **state**) of the database.

Data Independence

- A major objective for the three-level architecture is to provide **data independence**, which means that upper levels are unaffected by changes to lower levels.
- There are two kinds of data independence: **logical** and **physical**.



Logical data independence

- Logical data independence refers to the immunity of the external schemas to changes in the conceptual schema.
- Changes to the conceptual schema, such as the addition or removal of new entities, attributes, or relationships, should be possible without having to change existing external schemas or having to rewrite application programs.
- Clearly, the users for whom then changes have been made need to be aware of them, but what is important is that other users should not be.

Physical data independence

- Physical data independence refers to the immunity of the conceptual
- schema to changes in the internal schema.
- Changes to the internal schema, such as using different file organizations or storage structures, using different storage devices, modifying indexes, or hashing algorithms, should be possible without having to change the conceptual or external schemas.
- From the users' point of view, the only effect that may be noticed is a change in performance. In fact, deterioration in performance is the most common reason for internal schema changes.