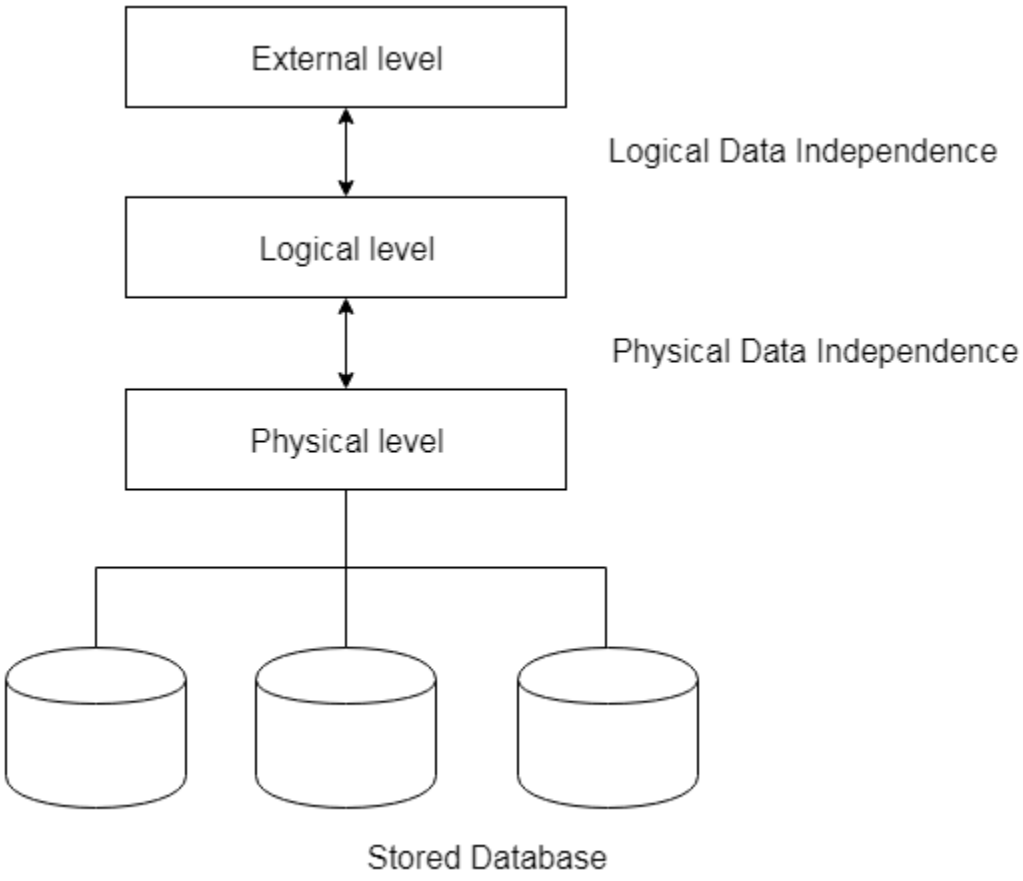
Data Independence

- Data independence refers characteristic of being able to modify the schema at one level of the database system without altering the schema at the next higher level.



There are two types of data independence:

1. Logical Data Independence

- Logical data independence refers characteristic of being able to change the conceptual schema without having to change the external schema.
- Logical data independence is used to separate the external level from the conceptual view.

- If we do any changes in the conceptual view of the data, then the user view of the data would not be affected.
- Logical data independence occurs at the user interface level.

2. Physical Data Independence

- Physical data independence can be defined as the capacity to change the internal schema without having to change the conceptual schema.
- If we do any changes in the storage size of the database system server, then the Conceptual structure of the database will not be affected.
- Physical data independence is used to separate conceptual levels from the internal levels.
- Physical data independence occurs at the logical interface level.

Database Users

Actors on the Scene: ----------

-People whose jobs involve the day-to-day use of a large database

- ♣ **Database administrator (DBA)** responsible for:
  - Authorizing access to the database
  - Coordinating and monitoring its use
  - Acquiring software and hardware resources
  - Tuning the DBMS for best performance
- ♣ **Database designer** responsible for:
  - Identifying the data to be stored
  - Choosing appropriate structures to represent and store this data
- ♣ **End users**
  - Those whose jobs require access to the database
    - **Naive or parametric end users**
      - Canned queries and updates
    - **Casual end users**
      - Occasional, special-purpose access
    - **Sophisticated end users**
      - Deep knowledge of database design and DBMS facilities
    - **Standalone users**
      - Users of personal databases
- ♣ **System analysts**
  - Determine requirements of end users
- ♣ **Application programmers**
  - Implement complex specifications (business logic) as programs

****

WORKERS BEHIND THE SCENE: ----------

♣ DBMS system designers and implementers

- Design and implement the DBMS modules and interfaces as a software package

♣ Tool developers

- Design and implement tools

♣ Operators and maintenance personnel

- Responsible for running and maintenance of hardware and software environment for database system

Relational Model concept

Relational model can represent as a table with columns and rows. Each row is known as a tuple. Each table of the column has a name or attribute.

Domain: It contains a set of atomic values that an attribute can take.

Attribute: It contains the name of a column in a particular table. Each attribute must have a domain

Relational instance: In the relational database system, the relational instance is represented by a finite set of tuples. Relation instances do not have duplicate tuples.

Relational schema: A relational schema contains the name of the relation and name of all columns or attributes.

Relational key: In the relational key, each row has one or more attributes. It can identify the row in the relation uniquely.

Example: STUDENT Relation

NAME	ROLL_NO	PHONE_NO	ADDRESS	AGE
Ram	14795	7305758992	Noida	24
Shyam	12839	9026288936	Delhi	35
Laxman	33289	8583287182	Gurugram	20
Mahesh	27857	7086819134	Ghaziabad	27
Ganesh	17282	9028 9i3988	Delhi	40

- In the given table, NAME, ROLL_NO, PHONE_NO, ADDRESS, and AGE are the attributes.
- The instance of schema STUDENT has 5 tuples.
- t3 = <Laxman, 33289, 8583287182, Gurugram, 20>

Properties of Relations

- Name of the relation is distinct from all other relations.
- Each relation cell contains exactly one atomic (single) value
- Each attribute contains a distinct name
- Attribute domain has no significance
- tuple has no duplicate value
- Order of tuple can have a different sequence

(Explain Integrity constraints and different types of keys on Relations)

Characteristics of the Database Approach

****

Self-Describing Nature of a Database System

Isolation between Programs and Data, and Data Abstraction

Support for Multiple Views of the Data

Sharing of knowledge and Multi-user Transaction Processing