DATABASE MANAGEMENT SYSTEM

The success of an organization depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use it to analyze and guide its activities. The phrase information superhighway refers information processing which is rapidly growing domain industry. Data Management System simplify the tasks of managing the data and extracting useful information in a timely fashion.

A database is a collection of data, typically describing the activities of one or more related organizations.

A database management system, or DBMS, is software designed to assist in maintaining and utilizing large collections of data

A HISTORICAL PERSPECTIVE

The first general-purpose DBMS was designed by Charles Bachman at General Electric in the early 1960s and was called the Integrated Data Store.

In the late 1960s, IBM developed the Information Management System, which is an alternative data representation framework called the hierarchical data model

In 1970, Edgar Codd, at IBM's San Jose Research Laboratory, proposed a new data representation framework called the relational data model

In the 1980s, the relational model consolidated its position as the dominant DBMS paradigm,

In the late 1980s and the 1990s, advances have been made in many areas of database systems. Several vendors (e.g., IBM's DB2, Oracle 8, Informix UDS) have extended their systems with the ability to store new data types such as images and text, and with the ability to ask more complex queries.

Specialized systems have been developed by numerous vendors for creating data warehouses, consolidating data from several databases, and for carrying out specialized analysis

DBMSs have entered the Internet Age and use of a DBMS to store data that is accessed through a Web browser is becoming widespread.

FILE SYSTEMS VERSUS A DBMS

Traditional file system called flat files, store the data in plain text format and columns are delimited by comma/ space/any special symbols. It store the data of an organization in group of file.

It has few limitations:

- Each file independent of other files
- Less flexible and many limitations
- Difficult to maintain file process system
- Duplication of data
- Data inconsistency
- Poor data modeling of world
- Transactional and concurrency problem
- Any change in one file affects all the files

The comparison between file and database system is listed below

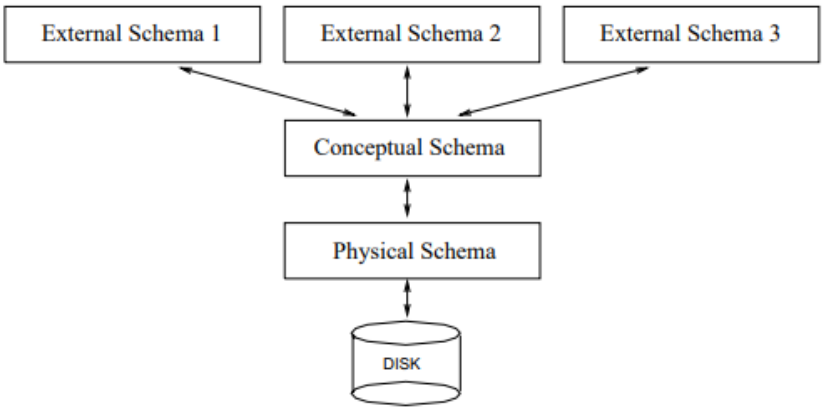
FILE SYSTEM	DBMS
Files stored in the storage unit of the computer	A software to store and retrieve the user's data
Redundant data may be present	No presence of redundant data
Query processing is not so efficient	Query processing is efficient
Data consistency is low	Data consistency is high due to normalization
Less complex, does not support complicated transactions	More complexity in managing the data, easier to implement complicated transactions
Less security	Supports more security mechanisms
Less expensive	Higher cost than the File system
Does not support crash recovery	Crash recovery mechanism is highly supported

Advantages of DBMS

- Data independence – Application is independent from data representation
- Efficient data access – uses sophisticated techniques to store and retrieve data efficiently
- Data integrity and security - can enforce access controls that govern what data is visible to different classes of users
- Data administration - When several users share the data, centralizing the administration of data can offer significant improvement.
- Concurrent access and crash recovery: Data is being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.
- Reduced application development time - DBMS supports many important functions that are common to many applications accessing data stored in the DBMS

LEVELS OF ABSTRACTION IN A DBMS

The data in a DBMS is described at three levels of abstraction, as illustrated in Figure: conceptual, physical, and external schemas.



The **conceptual schema (**sometimes called the logical schema) describes the stored data in terms of the data model of the DBMS. In a relational DBMS, the conceptual schema describes all relations that are stored in the database. In our sample university database, these relations contain information about entities, such as students and faculty, and about relationships, such as students' enrollment in courses.

Students(sid: string, name: string, login: string, age: integer, gpa: real)

Faculty(fid: string, fname: string, sal: real)

Courses(cid: string, cname: string, credits: integer)

The process of arriving at a good conceptual schema is called **conceptual database design**

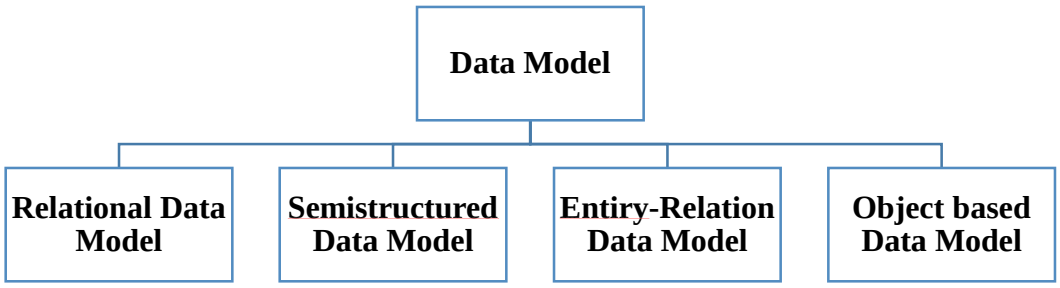
The **physical schema** summarizes how the relations described in the conceptual schema are actually stored on secondary storage devices. It includes file organizations to use to store the relations, and create auxiliary data structures called indexes to speed up data retrieval operations.

The process of arriving at a good physical schema is called **physical database design**

External schemas, which usually are also in terms of the data model of the DBMS, allow data access to be customized (and authorized) at the level of individual users or groups of users. Each external schema consists of a collection of one or more views and relations from the conceptual schema. A view is conceptually a relation, but the records in a view are not stored in the DBMS. The external schema design is guided by end user requirements.

DATA MODELS

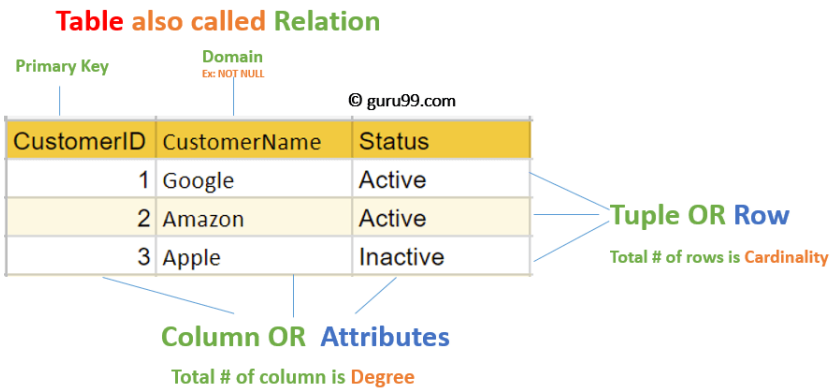
A data model is a collection of high-level data description constructs that hide many low-level storage details. It contains data description, data semantics, and consistency constraints of the data.



Relational Data Model

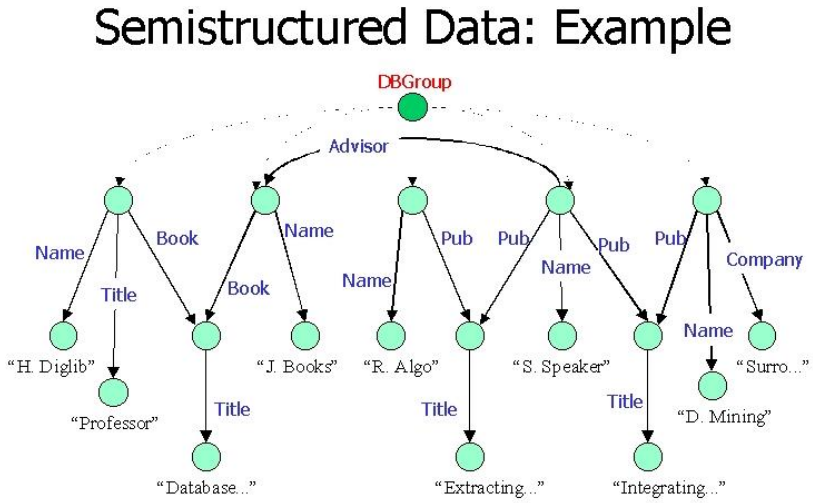
It designs the data in the form of rows and columns within a table. A Relational model uses tables for representing data and in-between relationships, described by Edgar F. Codd

A description of data in terms of a data model is called a **schema** and Tables are also called **Relations.** Integrity Constraints, are conditions that the records in a relation must satisfy.



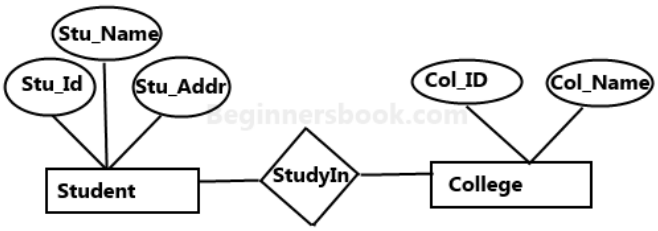
Semi Structured Data Model

It allows the data specifications at places where the individual data items of the same type may have different attributes sets. The Extensible Markup Language (XML) is widely used for representing the semistructured data.



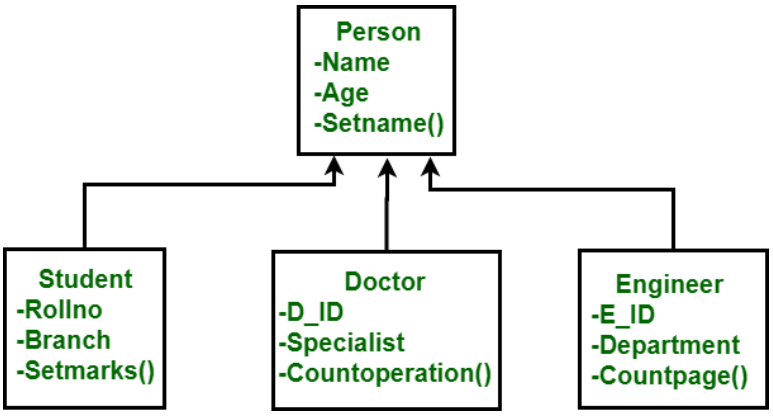
Entity Relationship Model

This model is the logical representation of data as objects and relationships among them. These objects are known as entities, and relationship is an association among these entities.



Object Based Data Model

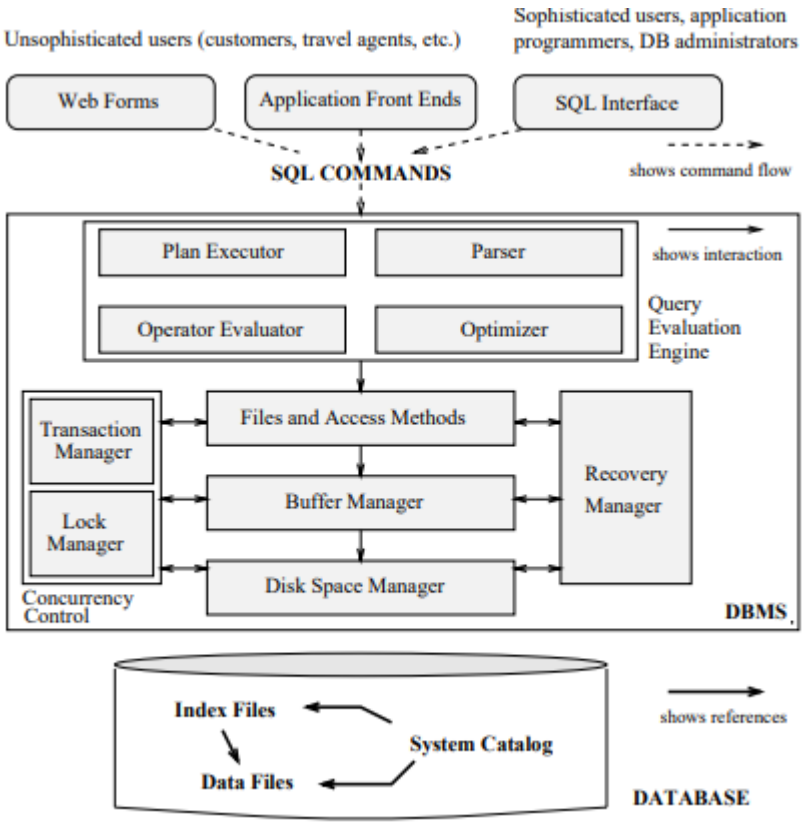
An extension of the ER model with notions of functions, encapsulation, and object identity, as well. This model supports a rich type system that includes structured and collection types. Here, the objects are nothing but the data carrying its properties.



DBMS ARCHITECTURE

A typical DBMS is based on the relational data model.

DBMS accepts SQL commands generated from a variety of user interfaces, produces query evaluation plans, executes these plans against the database, and returns the answers.



When a user issues a query, the parsed query is presented to a query optimizer, which uses information about how the data is stored to produce an efficient execution plan for evaluating the query. An execution plan is a blueprint for evaluating a query, and is usually represented as a tree of relational operators.

The code that implements relational operators sits on top of the file and access methods layer. This layer includes a variety of software for supporting the concept of a file, which, in a DBMS, is a collection of pages or a collection of records. Buffer manager, brings pages in from disk to main memory as needed in response to read requests.

The lowest layer of the DBMS software deals with management of space on disk, where the data is stored. Higher layers allocate, deallocate, read, and write pages through (routines provided by) this layer, called the disk space manager.

Transaction manager, ensures that transactions request and release locks according to a suitable locking protocol and schedules the execution transactions. Lock manager, which keeps track of requests for locks and grants locks on database objects when they become available; and the recovery manager, which is responsible for maintaining a log, and restoring the system to a consistent state after a crash.

PEOPLES DEAL WITH DBMS

Application programmers develop packages that facilitate data access for end users.

Database administrator (DBA) is responsible for designing and maintaining the database. DBA is responsible for many critical tasks:

- Design of the conceptual and physical schemas
- Security and authorization
- Data availability and recovery from failures

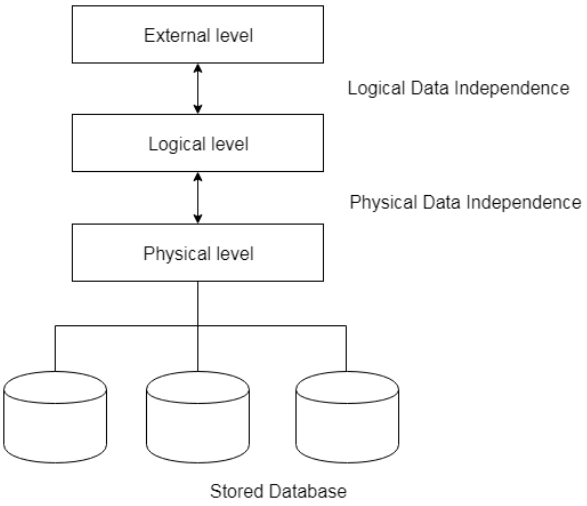
- Database tuning

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

2. Physical Data Independence

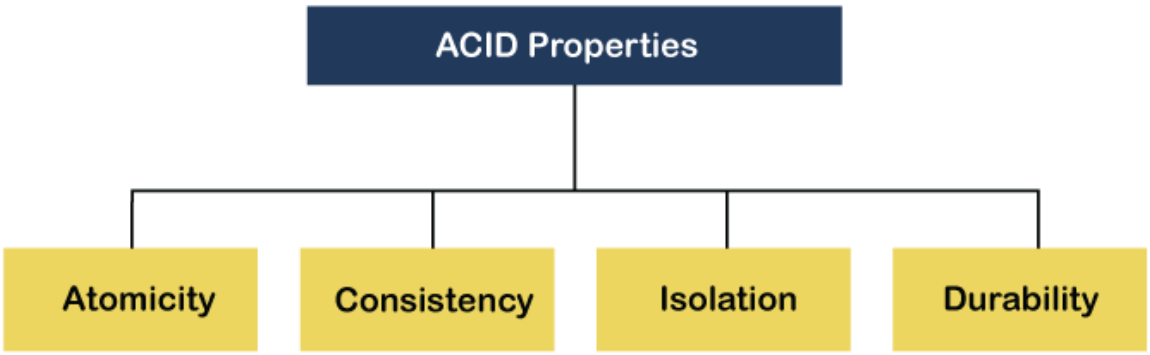


Logical data independence refers characteristic of being able to change the conceptual schema without having to change the external schema. It is used to separate the external level from the conceptual view. If we do any changes in the conceptual view of the data, then the user view of the data would not be affected. Logical data independence occurs at the user interface level

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. It occurs at the logical interface level

ACID Properties

To maintain the integrity of the data, there are four properties described in the database management system, which are known as the ACID properties.



Atomicity: The term atomicity defines that the data remains atomic. It means if any operation is performed on the data, either it should be performed or executed completely or should not be executed at all. It further means that the operation should not break in between or execute partially.

Consistency**:** In DBMS, the integrity of the data should be maintained, which means if a change in the database is made, it should remain preserved always. In the case of transactions, the integrity of the data is very essential so that the database remains consistent before and after the transaction. The data should always be correct.

Isolation**:** In DBMS, Isolation is the property of a database where no data should affect the other one and may occur concurrently. In short, the operation on one database should begin when the operation on the first database gets complete. It means if two operations are being performed on two different databases, they may not affect the value of one another.

Durability**:** Durability ensures that the data after the successful execution of the operation becomes permanent in the database. The durability of the data should be so perfect that even if the system fails or leads to a crash, the database still survives.

RELATIONAL DATA MODEL (INTRO)

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 Ai must have a domain, dom(Ai)

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

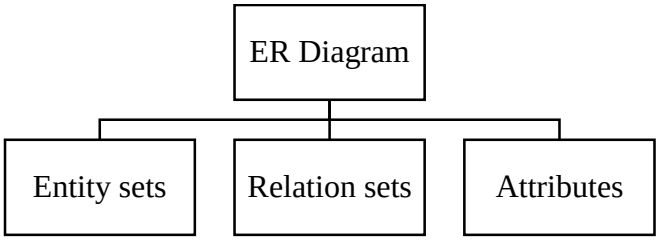
E-R MODELING (ER DIAGRAM)

A graphical representation of entities and their relationships which helps in understanding data independent of the actual database implementation.

Entity is a “thing” or “object” in the enterprise that is distinguishable from other objects

Relationships are the association of one entity with another entity

The components of E-R Diagram is depicted in the figure.



An entity can be place, person, object, event or a concept, which stores data in the database. The characteristics of entities are must have an attribute, and a unique key. Every entity is made up of some 'attributes' which represent that entity.

An entity set is a group of similar kind of entities. It may contain entities with attribute sharing similar values. Entities are represented by their properties, which also called attributes. All attributes have their separate values.

An entity that depends on another entity called a ***weak entity***. The weak entity doesn't contain any key attribute of its own.

The attribute is used to describe the property of an entity.

- ***Key attribute*** is used to represent the main characteristics of an entity. It represents a primary key
- An attribute that composed of many other attributes is known as a ***composite attribute***.
- An attribute can have more than one value. These attributes are known as a **multivalued attribute**
- An attribute that can be derived from other attribute is known as a ***derived attribute***

Key

- Primary Key: It is the first key which is used to identify one and only one instance of an entity uniquely
- A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.
- Super key is a set of an attribute which can uniquely identify a tuple
- Foreign keys are the column of the table which is used to point to the primary key of another table

Relationship

A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship

- When only one instance of an entity is associated with the relationship, then it is known as ***one to one*** relationship
- When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a ***one-to-many*** relationship
- When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a ***many-to-one*** relationship
- When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a ***many-to-many*** relationship

In ER diagram, many notations are used to express the cardinality. These notations are as follows