# DBMS Tutorial



DBMS Tutorial provides basic and advanced concepts of Database. Our DBMS Tutorial is designed for beginners and professionals both.

Database management system is software that is used to manage the database.

Our DBMS Tutorial includes all topics of DBMS such as introduction, ER model, keys, relational model, join operation, SQL, functional dependency, transaction, concurrency control, etc.

## What is Database

The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views, and reports, etc.

For example: The college Database organizes the data about the admin, staff, students and faculty etc.

Using the database, you can easily retrieve, insert, and delete the information.

## Database Management System

* Database management system is a software which is used to manage the database. For example: MySQL, Oracle, etc are a very popular commercial database which is used in different applications.
* DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.
* It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

DBMS allows users the following tasks:

* Data Definition: It is used for creation, modification, and removal of definition that defines the organization of data in the database.
* Data Updation: It is used for the insertion, modification, and deletion of the actual data in the database.
* Data Retrieval: It is used to retrieve the data from the database which can be used by applications for various purposes.
* User Administration: It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by unexpected failure.

## Characteristics of DBMS

* It uses a digital repository established on a server to store and manage the information.
* It can provide a clear and logical view of the process that manipulates data.
* DBMS contains automatic backup and recovery procedures.
* It contains ACID properties which maintain data in a healthy state in case of failure.
* It can reduce the complex relationship between data.
* It is used to support manipulation and processing of data.
* It is used to provide security of data.
* It can view the database from different viewpoints according to the requirements of the user.

## Advantages of DBMS

* Controls database redundancy: It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
* Data sharing: In DBMS, the authorized users of an organization can share the data among multiple users.
* Easily Maintenance: It can be easily maintainable due to the centralized nature of the database system.
* Reduce time: It reduces development time and maintenance need.
* Backup: It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
* multiple user interface: It provides different types of user interfaces like graphical user interfaces, application program interfaces

## Disadvantages of DBMS

* Cost of Hardware and Software: It requires a high speed of data processor and large memory size to run DBMS software.
* Size: It occupies a large space of disks and large memory to run them efficiently.
* Complexity: Database system creates additional complexity and requirements.
* Higher impact of failure: Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

# Advantages of DBMS over file system

BY CHAITANYA SINGH | FILED UNDER: [**DBMS**](https://beginnersbook.com/category/dbms/)

In this guide, we will discuss what is a file processing system and how Database management systems are better than file processing systems.

## Drawbacks of File system

* Data redundancy: Data redundancy refers to the duplication of data, lets say we are managing the data of a college where a student is enrolled for two courses, the same student details in such case will be stored twice, which will take more storage than needed. Data redundancy often leads to higher storage costs and poor access time.
* Data inconsistency: Data redundancy leads to data inconsistency, lets take the same example that we have taken above, a student is enrolled for two courses and we have student address stored twice, now lets say student requests to change his address, if the address is changed at one place and not on all the records then this can lead to data inconsistency.
* Data Isolation: Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.
* Dependency on application programs: Changing files would lead to change in application programs.
* Atomicity issues: Atomicity of a transaction refers to “All or nothing”, which means either all the operations in a transaction executes or none.

For example: Lets say Steve transfers 100$ to Negan’s account. This transaction consists multiple operations such as debit 100$ from Steve’s account, credit 100$ to Negan’s account. Like any other device, a computer system can fail lets say it fails after first operation then in that case Steve’s account would have been debited by 100$ but the amount was not credited to Negan’s account, in such case the rollback of operation should occur to maintain the atomicity of transaction. It is difficult to achieve atomicity in file processing systems.

* Data Security: Data should be secured from unauthorised access, for example a student in a college should not be able to see the payroll details of the teachers, such kind of security constraints are difficult to apply in file processing systems.

## Advantage of DBMS over file system

There are several advantages of Database management system over file system. Few of them are as follows:

* No redundant data: Redundancy removed by data [**normalization**](https://beginnersbook.com/2015/05/normalization-in-dbms/). No data duplication saves storage and improves access time.
* Data Consistency and Integrity: As we discussed earlier the root cause of data inconsistency is data redundancy, since data normalization takes care of the data redundancy, data inconsistency also been taken care of as part of it
* Data Security: It is easier to apply access constraints in database systems so that only authorized user is able to access the data. Each user has a different set of access thus data is secured from the issues such as identity theft, data leaks and misuse of data.
* Privacy: Limited access means privacy of data.
* Easy access to data – Database systems manages data in such a way so that the data is easily accessible with fast response times.
* Easy recovery: Since database systems keeps the backup of data, it is easier to do a full recovery of data in case of a failure.
* Flexible: Database systems are more flexible than file processing systems.

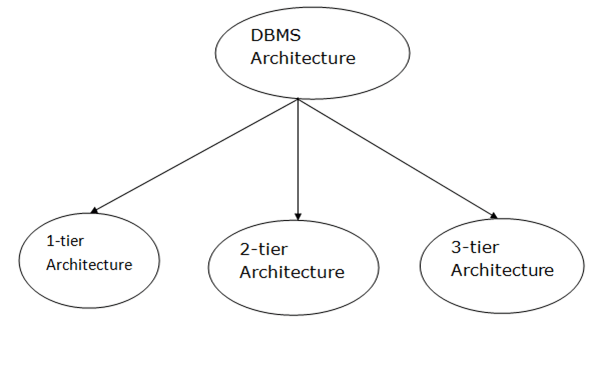
Disadvantages of DBMS:

* DBMS implementation cost is high compared to the file system
* Complexity: Database systems are complex to understand
* Performance: Database systems are generic, making them suitable for various applications. However this feature affect their performance for some applications

# DBMS Architecture

* The DBMS design depends upon its architecture. The basic client/server architecture is used to deal with a large number of PCs, web servers, database servers and other components that are connected with networks.
* The client/server architecture consists of many PCs and a workstation which are connected via the network.
* DBMS architecture depends upon how users are connected to the database to get their request done.

## Types of DBMS Architecture



Database architecture can be seen as a single tier or multi-tier. But logically, database architecture is of two types like: 2-tier architecture and 3-tier architecture.

### 1-Tier Architecture

* In this architecture, the database is directly available to the user. It means the user can directly sit on the DBMS and uses it.
* Any changes done here will directly be done on the database itself. It doesn't provide a handy tool for end users.
* The 1-Tier architecture is used for development of the local application, where programmers can directly communicate with the database for the quick response.

### 2-Tier Architecture

* The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: ODBC, JDBC are used.
* The user interfaces and application programs are run on the client-side.
* The server side is responsible to provide the functionalities like: query processing and transaction management.
* To communicate with the DBMS, client-side application establishes a connection with the server side.

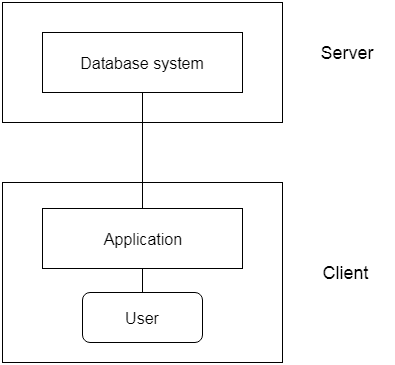


Fig: 2-tier Architecture

### 3-Tier Architecture

* The 3-Tier architecture contains another layer between the client and server. In this architecture, client can't directly communicate with the server.
* The application on the client-end interacts with an application server which further communicates with the database system.
* End user has no idea about the existence of the database beyond the application server. The database also has no idea about any other user beyond the application.
* The 3-Tier architecture is used in case of large web application.

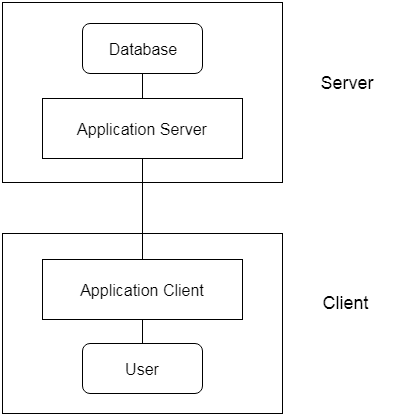
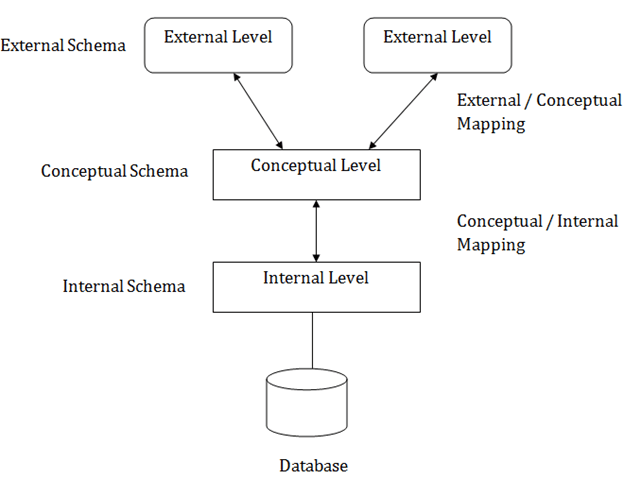


Fig: 3-tier Architecture

# Three schema Architecture

* The three schema architecture is also called ANSI/SPARC architecture or three-level architecture.
* This framework is used to describe the structure of a specific database system.
* The three schema architecture is also used to separate the user applications and physical database.
* The three schema architecture contains three-levels. It breaks the database down into three different categories.

The three-schema architecture is as follows:



In the above diagram:

* It shows the DBMS architecture.
* Mapping is used to transform the request and response between various database levels of architecture.
* Mapping is not good for small DBMS because it takes more time.
* In External / Conceptual mapping, it is necessary to transform the request from external level to conceptual schema.
* In Conceptual / Internal mapping, DBMS transform the request from the conceptual to internal level.

### 1. Internal Level

* The internal level has an internal schema which describes the physical storage structure of the database.
* The internal schema is also known as a physical schema.
* It uses the physical data model. It is used to define that how the data will be stored in a block.
* The physical level is used to describe complex low-level data structures in detail.

### 2. Conceptual Level

* The conceptual schema describes the design of a database at the conceptual level. Conceptual level is also known as logical level.
* The conceptual schema describes the structure of the whole database.
* The conceptual level describes what data are to be stored in the database and also describes what relationship exists among those data.
* In the conceptual level, internal details such as an implementation of the data structure are hidden.
* Programmers and database administrators work at this level.

### 3. External Level

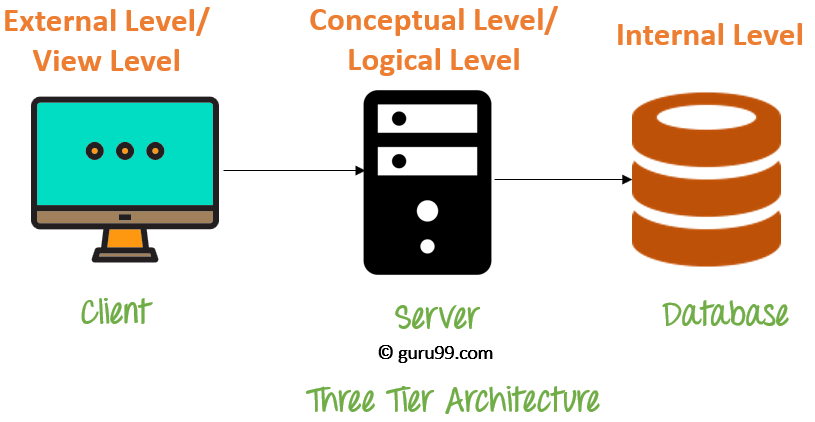
* At the external level, a database contains several schemas that sometimes called as subschema. The subschema is used to describe the different view of the database.
* An external schema is also known as view schema.
* Each view schema describes the database part that a particular user group is interested and hides the remaining database from that user group.
* The view schema describes the end user interaction with database systems.

# DBMS Schemas: Internal, Conceptual, External

# Database systems comprise of complex data structures. Thus, to make the system efficient for retrieval of data and reduce the complexity of the users, developers use the method of Data Abstraction.

There are mainly three levels of data abstraction:

1. Internal Level: Actual PHYSICAL storage structure and access paths.
2. Conceptual or Logical Level: Structure and constraints for the entire database
3. External or View level: Describes various user views



Let's study them in detail

## Internal Level/Schema

The internal schema defines the physical storage structure of the database. The internal schema is a very low-level representation of the entire database. It contains multiple occurrences of multiple types of internal record. In the ANSI term, it is also called "stored record'.

Facts about Internal schema:

* The internal schema is the lowest level of data abstraction
* It helps you to keeps information about the actual representation of the entire database. Like the actual storage of the data on the disk in the form of records
* The internal view tells us what data is stored in the database and how
* It never deals with the physical devices. Instead, internal schema views a physical device as a collection of physical pages

## Conceptual Schema/Level

The conceptual schema describes the Database structure of the whole database for the community of users. This schema hides information about the physical storage structures and focuses on describing data types, entities, relationships, etc.

This logical level comes between the user level and physical storage view. However, there is only single conceptual view of a single database.

Facts about Conceptual schema:

* Defines all database entities, their attributes, and their relationships
* Security and integrity information
* In the conceptual level, the data available to a user must be contained in or derivable from the physical level

## External Schema/Level

An external schema describes the part of the database which specific user is interested in. It hides the unrelated details of the database from the user. There may be "n" number of external views for each database.

Each external view is defined using an external schema, which consists of definitions of various types of external record of that specific view.

An external view is just the content of the database as it is seen by some specific particular user. For example, a user from the sales department will see only sales related data.

Facts about external schema:

* An external level is only related to the data which is viewed by specific end users.
* This level includes some external schemas.
* External schema level is nearest to the user
* The external schema describes the segment of the database which is needed for a certain user group and hides the remaining details from the database from the specific user group

## Goal of 3 level/schema of Database

Here, are some Objectives of using Three schema Architecture:

* Every user should be able to access the same data but able to see a customized view of the data.
* The user need not to deal directly with physical database storage detail.
* The DBA should be able to change the database storage structure without disturbing the user's views
* The internal structure of the database should remain unaffected when changes made to the physical aspects of storage.

## Advantages Database Schema

* You can manage data independent of the physical storage
* Faster Migration to new graphical environments
* DBMS Architecture allows you to make changes on the presentation level without affecting the other two layers
* As each tier is separate, it is possible to use different sets of developers
* It is more secure as the client doesn't have direct access to the database business logic
* In case of the failure of the one-tier no data loss as you are always secure by accessing the other tier

## Disadvantages Database Schema

* Complete DB Schema is a complex structure which is difficult to understand for every one
* Difficult to set up and maintain
* The physical separation of the tiers can affect the performance of the Database

### Summary

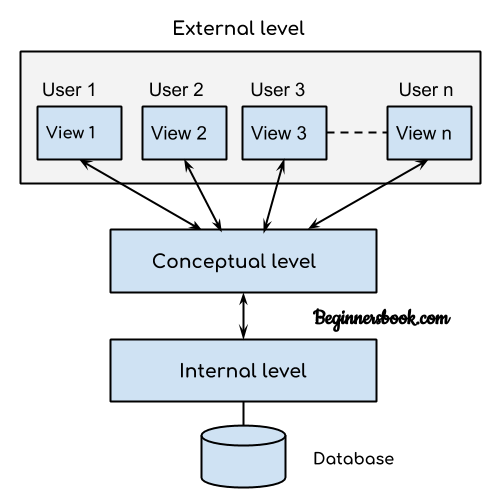
* There are mainly three levels of data abstraction: Internal Level, Conceptual or Logical Level or External or View level
* The internal schema defines the physical storage structure of the database
* The conceptual schema describes the Database structure of the whole database for the community of users
* An external schema describe the part of the database which specific user is interested in
* DBMS Architecture allows you to make changes on the presentation level without affecting the other two layers

# DBMS – Three Level Architecture

BY CHAITANYA SINGH | FILED UNDER: [**DBMS**](https://beginnersbook.com/category/dbms/)

In the previous tutorial we have seen the [**DBMS architecture**](https://beginnersbook.com/2018/11/dbms-architecture/) – one-tier, two-tier and three-tier. In this guide, we will discuss the three level DBMS architecture in detail.

## DBMS Three Level Architecture Diagram



This architecture has three levels:  
1. External level  
2. Conceptual level  
3. Internal level

## 1. External level

It is also called view level. The reason this level is called “view” is because several users can view their desired data from this level which is internally fetched from database with the help of conceptual and internal level mapping.

The user doesn’t need to know the database schema details such as data structure, table definition etc. user is only concerned about data which is what returned back to the view level after it has been fetched from database (present at the internal level).

External level is the “top level” of the Three Level DBMS Architecture.

## 2. Conceptual level

It is also called logical level. The whole design of the database such as relationship among data, schema of data etc. are described in this level.

Database constraints and security are also implemented in this level of architecture. This level is maintained by DBA (database administrator).

## 3. Internal level

This level is also known as physical level. This level describes how the data is actually stored in the storage devices. This level is also responsible for allocating space to the data. This is the lowest level of the architecture.

# Data Independence

* Data independence can be explained using the three-schema architecture.
* 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.

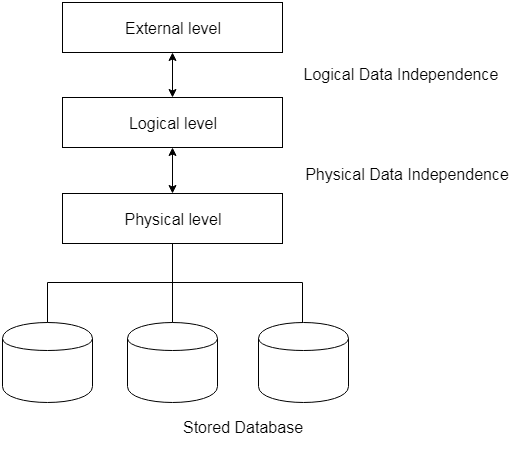


Fig: Data Independence

# View of Data in DBMS

BY CHAITANYA SINGH | FILED UNDER: [**DBMS**](https://beginnersbook.com/category/dbms/)

Abstraction is one of the main features of database systems. Hiding irrelevant details from user and providing abstract view of data to users, helps in easy and efficient user-database interaction. In the previous tutorial, we discussed the [**three level of DBMS architecture**](https://beginnersbook.com/2018/11/dbms-three-level-architecture/), The top level of that architecture is “view level”. The view level provides the “view of data” to the users and hides the irrelevant details such as data relationship, database schema, [**constraints**](https://beginnersbook.com/2015/04/constraints-in-dbms/), security etc from the user.

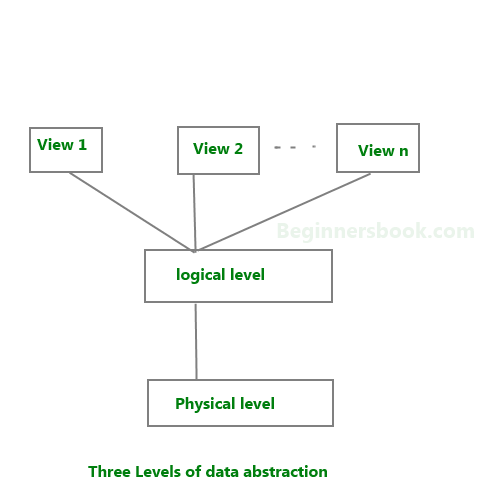
To fully understand the view of data, you must have a basic knowledge of data abstraction and instance & schema. Refer these two tutorials to learn them in detail.

1. [**Data abstraction**](https://beginnersbook.com/2015/04/levels-of-abstraction-in-dbms/)
2. [**Instance and schema**](https://beginnersbook.com/2015/04/instance-and-schema-in-dbms/)

# Data Abstraction in DBMS

BY CHAITANYA SINGH | FILED UNDER: [**DBMS**](https://beginnersbook.com/category/dbms/)

Database systems are made-up of complex data structures. To ease the user interaction with database, the developers hide internal irrelevant details from users. This process of hiding irrelevant details from user is called data abstraction.



We have three levels of abstraction:  
Physical level: This is the lowest level of data abstraction. It describes how data is actually stored in database. You can get the complex data structure details at this level.

Logical level: This is the middle level of 3-level data abstraction architecture. It describes what data is stored in database.

View level: Highest level of data abstraction. This level describes the user interaction with database system.

Example: Let’s say we are storing customer information in a customer table. At physical level these records can be described as blocks of storage (bytes, gigabytes, terabytes etc.) in memory. These details are often hidden from the programmers.

At the logical level these records can be described as fields and attributes along with their data types, their relationship among each other can be logically implemented. The programmers generally work at this level because they are aware of such things about database systems.

At view level, user just interact with system with the help of GUI and enter the details at the screen, they are not aware of how the data is stored and what data is stored; such details are hidden from them.

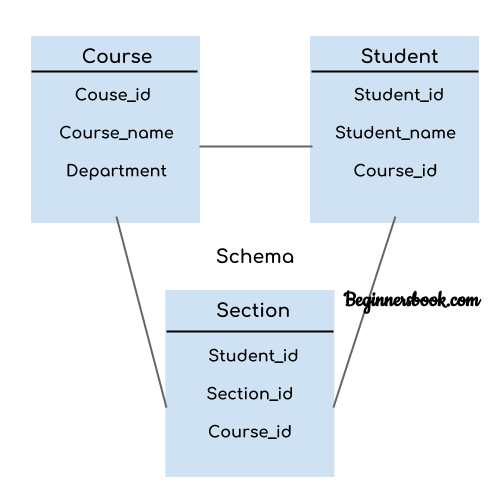
# Instance and schema in DBMS

BY CHAITANYA SINGH | FILED UNDER: [**DBMS**](https://beginnersbook.com/category/dbms/)

In this guide, we will learn what is an instance and schema in DBMS.

## DBMS Schema

Definition of schema: Design of a database is called the schema. Schema is of three types: Physical schema, logical schema and view schema.

For example: In the following diagram, we have a schema that shows the relationship between three tables: Course, Student and Section. The diagram only shows the design of the database, it doesn’t show the data present in those tables. Schema is only a structural view(design) of a database as shown in the diagram below.  


The design of a database at physical level is called physical schema, how the data stored in blocks of storage is described at this level.

Design of database at logical level is called logical schema, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is hidden at this level (available at physical level).

Design of database at view level is called view schema. This generally describes end user interaction with database systems.

To learn more about these schemas, refer [**3 level data abstraction architecture**](https://beginnersbook.com/2015/04/levels-of-abstraction-in-dbms/).

## DBMS Instance

Definition of instance: The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

For example, lets say we have a single table student in the database, today the table has 100 records, so today the instance of the database has 100 records. Lets say we are going to add another 100 records in this table by tomorrow so the instance of database tomorrow will have 200 records in table. In short, at a particular moment the data stored in database is called the instance, that changes over time when we add or delete data from the database.

# Data models in DBMS

A Database Model defines the logical design and structure of a database and defines how data will be stored,accessed and updated in a database management system.While The Relational model is the most widely used database model.

## Types of Data Models

There are several types of data models in DBMS-

Object based logical Models – Describe data at the conceptual and view levels.

1. [**E-R Model**](https://beginnersbook.com/2015/04/e-r-model-in-dbms/)
2. **Object oriented Model**

Record based logical Models – Like Object based model, they also describe data at the conceptual and view levels. These models specify logical structure of database with records, fields and attributes.

1. [**Relational Model**](https://beginnersbook.com/2015/04/relational-model-in-dbms/)
2. [**Hierarchical Model**](https://beginnersbook.com/2015/04/hierarchical-model-in-dbms/)
3. Network Model – Network Model is same as hierarchical model except that it has graph-like structure rather than a tree-based structure. Unlike hierarchical model, this model allows each record to have more than one parent record.

Physical Data Models – These models describe data at the lowest level of abstraction.

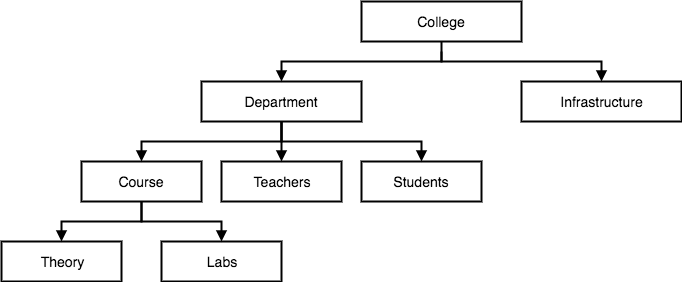
## Hierarchical Model

This database model organises data into a tree-like-structure, with a single root, to which all the other data is linked. The heirarchy starts from the **Root** data, and expands like a tree, adding child nodes to the parent nodes.

In this model, a child node will only have a single parent node.

This model efficiently describes many real-world relationships like index of a book, recipes etc.

**In hierarchical model, data is organised into tree-like structure with one one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.**

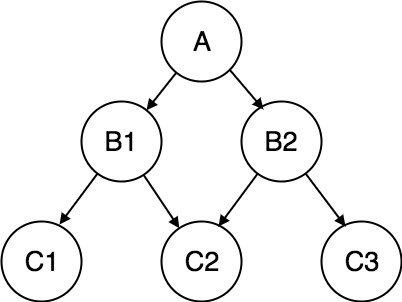


## Network Model

This is an extension of the Hierarchical model. In this model data is organised more like a graph, and are allowed to have more than one parent node.

In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.

This was the most widely used database model, before Relational Model was introduced.



## Entity-relationship Model

In this database model, relationships are created by dividing object of interest into entity and its characteristics into attributes.

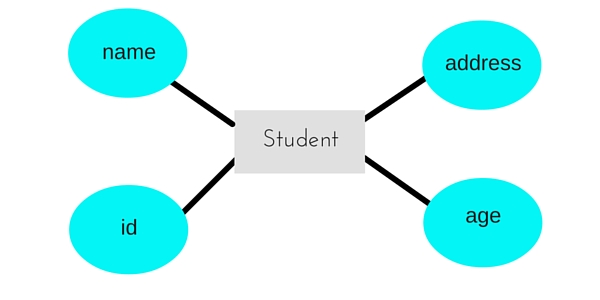
Different entities are related using relationships.

**E-R Models are defined to represent the relationships into pictorial form to make it easier for different stakeholders to understand.**

This model is good to design a database, which can then be turned into tables in relational model(explained below).

Let's take an example, If we have to design a School Database, then **Student** will be an **entity** with **attributes** name, age, address etc. As **Address** is generally complex, it can be another **entity** with **attributes** street name, pincode, city etc, and there will be a relationship between them.

Relationships can also be of different types. T



## Relational Model

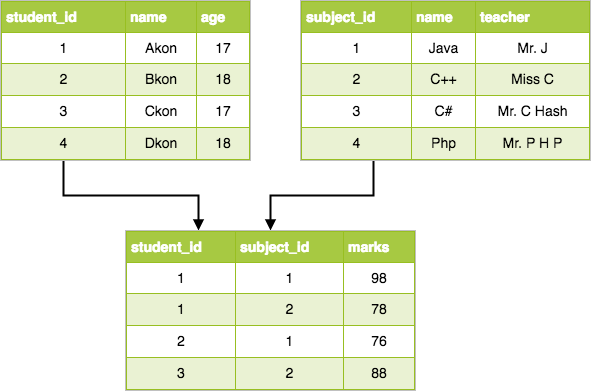
In this model, data is organised in two-dimensional **tables** and the relationship is maintained by storing a common field.

This model was introduced by E.F Codd in 1970, and since then it has been the most widely used database model, infact, we can say the only database model used around the world.

The basic structure of data in the relational model is tables. All the information related to a particular type is stored in rows of that table.

Hence, tables are also known as **relations** in relational model.

In the coming tutorials we will learn how to design tables, normalize them to reduce data redundancy and how to use Structured Query language to access data from tables.



# The Object-Oriented Data Model

1. A data model is a logic organization of the real world objects (entities), constraints on them, and the relationships among objects. A DB language is a concrete syntax for a data model. A DB system implements a data model.
2. A core object-oriented data model consists of the following basic object-oriented concepts:

(1) **object and object identifier**: Any real world entity is uniformly modeled as an object (associated with a unique id: used to pinpoint an object to retrieve).

(2) **attributes and methods**: every object has a state (the set of values for the attributes of the object) and a behavior (the set of methods - program code - which operate on the state of the object). The state and behavior encapsulated in an object are accessed or invoked from outside the object only through explicit message passing.

[ An attribute is an instance variable, whose domain may be any class: user-defined or primitive. A class composition hierarchy (aggregation relationship) is orthogonal to the concept of a class hierarchy. The link in a class composition hierarchy may form cycles. ]

(3) **class**: a means of grouping all the objects which share the same set of attributes and methods. An object must belong to only one class as an instance of that class (instance-of relationship). A class is similar to an abstract data type. A class may also be primitive (no attributes), e.g., integer, string, Boolean.

(4) **Class hierarchy and inheritance**: derive a new class (subclass) from an existing class (superclass). The subclass inherits all the attributes and methods of the existing class and may have additional attributes and methods. single inheritance (class hierarchy) vs. multiple inheritance (class lattice).

# Relational Data Model in DBMS: Concepts, Constraints, Example

## What is Relational Model

The relational model represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.

The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations. In the relational model, data are stored as tables. However, the physical storage of the data is independent of the way the data are logically organized.

Some popular Relational Database management systems are:

* DB2 and Informix Dynamic Server - IBM
* Oracle and RDB – Oracle
* SQL Server and Access - Microsoft

In this tutorial, you will learn

* [What is Relational Model](https://www.guru99.com/relational-data-model-dbms.html" \l "1)
* [Relational Model Concepts](https://www.guru99.com/relational-data-model-dbms.html" \l "2)
* [Relational Integrity constraints](https://www.guru99.com/relational-data-model-dbms.html" \l "3)
* [Operations in Relational Model](https://www.guru99.com/relational-data-model-dbms.html" \l "4)
* [Best Practices for creating a Relational Model](https://www.guru99.com/relational-data-model-dbms.html" \l "5)
* [Advantages of using Relational model](https://www.guru99.com/relational-data-model-dbms.html" \l "6)
* [Disadvantages of using Relational model](https://www.guru99.com/relational-data-model-dbms.html" \l "7)

## Relational Model Concepts

1. Attribute: Each column in a Table. Attributes are the properties which define a relation. e.g., Student\_Rollno, NAME,etc.
2. Tables – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.
3. Tuple – It is nothing but a single row of a table, which contains a single record.
4. Relation Schema: A relation schema represents the name of the relation with its attributes.
5. Degree: The total number of attributes which in the relation is called the degree of the relation.
6. Cardinality: Total number of rows present in the Table.
7. Column: The column represents the set of values for a specific attribute.
8. Relation instance – Relation instance is a finite set of tuples in the RDBMS system. Relation instances never have duplicate tuples.
9. Relation key - Every row has one, two or multiple attributes, which is called relation key.
10. Attribute domain – Every attribute has some pre-defined value and scope which is known as attribute domain

## Relational Integrity constraints

Relational Integrity constraints is referred to conditions which must be present for a valid relation. These integrity constraints are derived from the rules in the mini-world that the database represents.

There are many types of integrity constraints. Constraints on the Relational database management system is mostly divided into three main categories are:

1. Domain constraints
2. Key constraints
3. Referential integrity constraints

### Domain Constraints

Domain constraints can be violated if an attribute value is not appearing in the corresponding domain or it is not of the appropriate data type.

Domain constraints specify that within each tuple, and the value of each attribute must be unique. This is specified as data types which include standard data types integers, real numbers, characters, Booleans, variable length strings, etc.

Example:

Create DOMAIN CustomerName

CHECK (value not NULL)

The example shown demonstrates creating a domain constraint such that CustomerName is not NULL

### Key constraints

An attribute that can uniquely identify a tuple in a relation is called the key of the table. The value of the attribute for different tuples in the relation has to be unique.

Example:

In the given table, CustomerID is a key attribute of Customer Table. It is most likely to have a single key for one customer, CustomerID =1 is only for the CustomerName =" Google".

|  |  |  |
| --- | --- | --- |
| CustomerID | CustomerName | Status |
| 1 | Google | Active |
| 2 | Amazon | Active |
| 3 | Apple | Inactive |
|  |  |  |

### Referential integrity constraints

Referential integrity constraints is base on the concept of Foreign Keys. A foreign key is an important attribute of a relation which should be referred to in other relationships. Referential integrity constraint state happens where relation refers to a key attribute of a different or same relation. However, that key element must exist in the table.

Example:

In the above example, we have 2 relations, Customer and Billing.

Tuple for CustomerID =1 is referenced twice in the relation Billing. So we know CustomerName=Google has billing amount $300

## Operations in Relational Model

Four basic update operations performed on relational database model are

Insert, update, delete and select.

* Insert is used to insert data into the relation
* Delete is used to delete tuples from the table.
* Modify allows you to change the values of some attributes in existing tuples.
* Select allows you to choose a specific range of data.

Whenever one of these operations are applied, integrity constraints specified on the relational database schema must never be violated.

### Inset Operation

The insert operation gives values of the attribute for a new tuple which should be inserted into a relation.

### Update Operation

You can see that in the below-given relation table CustomerName= 'Apple' is updated from Inactive to Active.

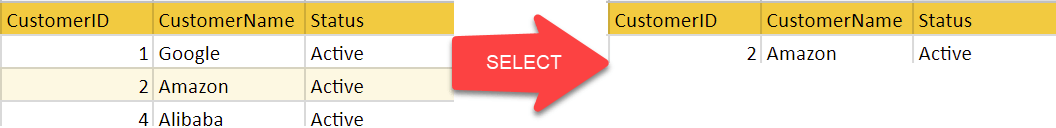
### Delete Operation

To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.

In the above-given example, CustomerName= "Apple" is deleted from the table.

The Delete operation could violate referential integrity if the tuple which is deleted is referenced by foreign keys from other tuples in the same database.

### Select Operation



In the above-given example, CustomerName="Amazon" is selected

## Best Practices for creating a Relational Model

* Data need to be represented as a collection of relations
* Each relation should be depicted clearly in the table
* Rows should contain data about instances of an entity
* Columns must contain data about attributes of the entity
* Cells of the table should hold a single value
* Each column should be given a unique name
* No two rows can be identical
* The values of an attribute should be from the same domain

## Advantages of using Relational model

* Simplicity: A relational data model is simpler than the hierarchical and network model.
* Structural Independence: The relational database is only concerned with data and not with a structure. This can improve the performance of the model.
* Easy to use: The relational model is easy as tables consisting of rows and columns is quite natural and simple to understand
* Query capability: It makes possible for a high-level query language like SQL to avoid complex database navigation.
* Data independence: The structure of a database can be changed without having to change any application.
* Scalable: Regarding a number of records, or rows, and the number of fields, a database should be enlarged to enhance its usability.

## Disadvantages of using Relational model

* Few relational databases have limits on field lengths which can't be exceeded.
* Relational databases can sometimes become complex as the amount of data grows, and the relations between pieces of data become more complicated.
* Complex relational database systems may lead to isolated databases where the information cannot be shared from one system to another.

### Summary

* The Relational database model represents the database as a collection of relations (tables)
* Attribute, Tables, Tuple, Relation Schema, Degree, Cardinality, Column, Relation instance, are some important components of Relational Model
* Relational Integrity constraints are referred to conditions which must be present for a valid relation
* Domain constraints can be violated if an attribute value is not appearing in the corresponding domain or it is not of the appropriate data type
* Insert, Select, Modify and Delete are operations performed in Relational Model
* The relational database is only concerned with data and not with a structure which can improve the performance of the model
* Advantages of relational model is simplicity, structural independence, ease of use, query capability, data independence, scalability.
* Few relational databases have limits on field lengths which can't be exceeded.