DATABASE SYSTEM

Unit 01

T.Y. B. Tech Computer

SVKM'S Institute of Technology, Dhule

Mr. Bhushan R. Nandwalkar

DATABASE CONCEPTS

What is Data?

Database contains data and this data is **raw facts**.

The word raw indicates that the facts have not yet been processed to revel their meaning.

Information is the result of processing row data to revel it's meaning.

Raw data must be properly formatted for storage, processing and representation.

DATABASE CONCEPTS

What is DBMS?

A database management system (DBMS) is a software package designed to **define, manipulate, retrieve and manage data in a database.** A DBMS generally manipulates the data itself, the data format, field names, record structure and file structure. It also defines rules to validate and manipulate this data.

OR

Database is a collection of inter-related data which helps in efficient retrieval, insertion and deletion of data from database and organizes the data in the form of tables, views, schemas, reports etc.

DATABASE CONCEPTS

What is DBMS?(Conti..)

So Database is shared, integrated computer structure that stores the collection of -

A) End Users Data

B) Meta Data- i.e. Data about data through which end user data is integrated and manage.

DATABASE CONCEPTS

What is DBMS?(Conti..)

So, DBMS is an arrangement of data, hardware and software that helps to manage users operational data. The main function of DBMS is to provide efficient and reliable methods of data retrieval to many users.

1. Database System Hardware:-

DBMS are in most cases installed on general purpose computers. In examining the memory needs of DBMS, We need to consider the following issues.

A) Data must remain available long after program completed it's works. Also data must available even if system break down.

DATABASE CONCEPTS

What is DBMS?(Conti..)

B) A DBMS must access data at relatively high rate.

C) Storage medium must be low rate.

2. Database System Software:-

User interact with database system through query language. The Query languages of DBMS has two broad tasks:

- **To define data structure** that serves the data of the database and to **allow the speedy retrieval and modification of data.**

DATABASE CONCEPTS

What is DBMS?(Conti..)

A Central task of DBMS is transaction management.

Atomicity:

A transaction is a sequence of database operations that must be executed in it's entirely or not at all. This property of transaction is known as **"Atomicity"**

For example, in an application that transfers funds from one account to another, the atomicity property ensures that, if a debit is made successfully from one account, the corresponding credit is made to the other account.

DATABASE CONCEPTS

What is DBMS?(Conti..)

Consistency:

Data is in a consistent state when a transaction starts and when it ends.

For example, in an application that transfers funds from one account to another, the consistency property ensures that the total value of funds in both the accounts is the same at the start and end of each transaction.

DATABASE CONCEPTS

What is DBMS?(Conti..)

Isolation:

The transaction management component ensures that the execution of one transaction is not influence by execution of any other transaction.

Durability:

After a transaction successfully completes, changes to data persist and are not undone, even in the event of a system failure.

For example, in an application that transfers funds from one account to another, the durability property ensures that the changes made to each account will not be reversed.

DATABASE CONCEPTS

Need of Database

1) Processing Queries and Object Management:

In traditional file systems, we cannot store data in the form of objects. In practical-world applications, data is stored in objects and not files.

2) Controlling Redundancy and Inconsistency:

Redundancy refers to repeated instances of the same data. A database system provides redundancy control whereas in a file system, same data may be stored multiple times. If data is repeated may occurs inconsistency. Data inconsistency occur between files when similar data is kept in different formats in two different files

DATABASE CONCEPTS

Need of Database(Conti..)

3) Efficient Memory Management and Indexing:

DBMS makes complex memory management easy to handle. In file systems, query operations require entire file scans whereas in a DBMS , using indexing we can retrieve data efficiently.

4)Data Integrity:

Data integrity ensures that only required data is stored in the database. Data is validated before entered into database using integrity constraints such as primary key foreign key.

DATABASE CONCEPTS

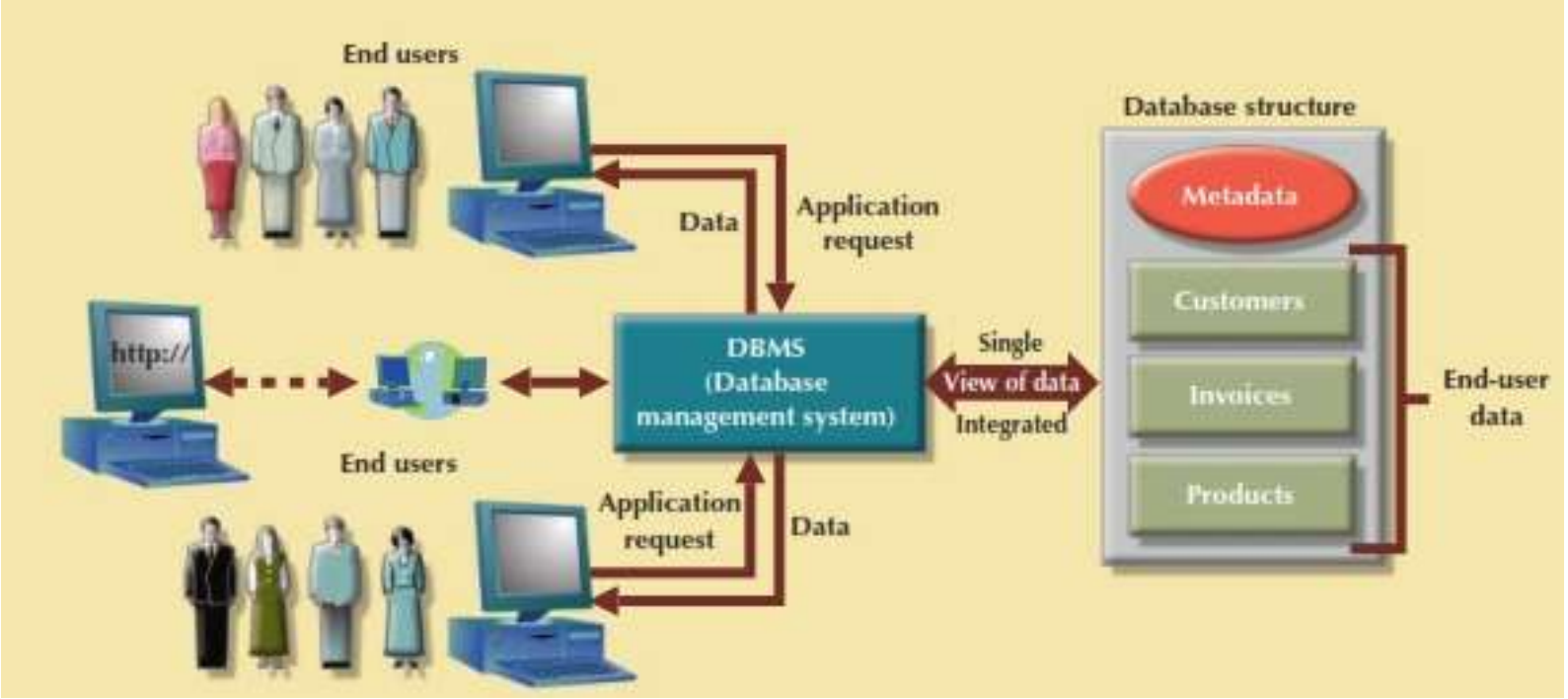
Need of Database(Conti..)

5) Data Security:

In traditional file management there is no authentication system where as DBMS provides level of security authentication.

DATABASE CONCEPTS

The DBMS manages the interaction between the end user and the database



DATABASE CONCEPTS

Types of Databases

- A **single-user database** supports only one user at a time. In other words, if user A is using the database, users B and C must wait until user A is done. A single-user database that runs on a personal computer is called a **desktop database.**

-**Multiuser database** supports multiple users at the same time. When the multiuser database supports a relatively small number of users (usually fewer than 50) or a specific department within an organization, it is called a **workgroup database.**

DATABASE CONCEPTS

Types of Databases(Conti..)

- When the database is used by the entire organization and supports many users (more than 50, usually hundreds) across many departments, the database is known as an **enterprise database.**

-Location might also be used to classify the database. For example, a database that supports data located at a single site is called a **centralized database.** A database that supports data distributed across several different sites is called a **distributed database.**

File System Vs DBMS

File System	Database Management System (DBMS)
1. It is a software system that manages and controls the data files in a computer system.	1. It is a software system used for creating and managing the databases. DBMS provides a systematic way to access, update, and delete data.
2. File system does not support multi-user access.	2. Database Management System supports multi-user access.
3. Data consistency is less in the file system.	3. Data consistency is more due to the use of normalization.
4. File system is not secured.	4. Database Management System is highly secured.
5. File system is used for storing the unstructured data.	5. Database management system is used for storing the structured data.
6. In the file system, data redundancy is high.	6. In DBMS, Data redundancy is low.

FILE SYSTEM VS DBMS

7. No data backup and recovery process is present in a file system.	7. There is a backup recovery for data in DBMS.
8. Handling of a file system is easy.	8. Handling a DBMS is complex.
9. Cost of a file system is less than the DBMS.	9. Cost of database management system is more than the file system.
10. If one application fails, it does not affect other application in a system.	10. If the database fails, it affects all application which depends on it.
11. In the file system, data cannot be shared because it is distributed in different files.	11. In DBMS, data can be shared as it is stored at one place in a database.
12. These system does not provide concurrency facility.	12. This system provides concurrency facility.
13. **Example:** NTFS (New technology file system), EXT (Extended file system), etc.	13. **Example:** Oracle, MySQL, MS SQL Server, DB2, Microsoft Access, etc.

DATABASE SYSTEM ARCHITECTURE

The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical.

The architecture of a DBMS can be seen as either single tier or multi-tier. The tiers are classified as follows :

- **1-tier architecture**
- **2-tier architecture**
- **3-tier architecture**

DATABASE SYSTEM ARCHITECTURE

1 Tier Architecture

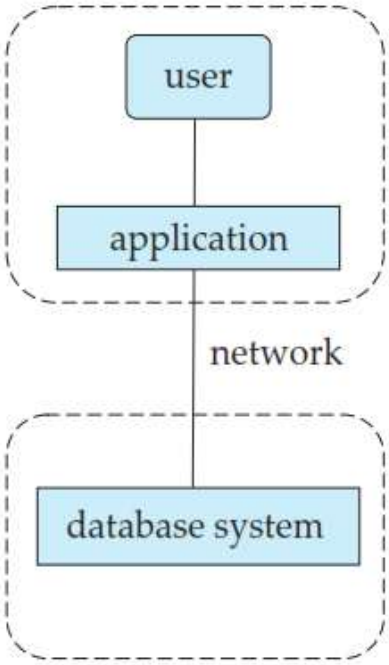
One-tier architecture involves putting all of the required components for a software application or technology on a single server or platform.

a one-tier architecture keeps all of the elements of an application, including the interface, Middleware and back-end data, in one place.

DATABASE SYSTEM ARCHITECTURE

2 Tier Architecture

The two-tier is based on Client Server architecture. The two-tier architecture is like client server application. The direct communication takes place between client and server. There is no intermediate between client and server.



DATABASE SYSTEM ARCHITECTURE

3 Tier Architecture

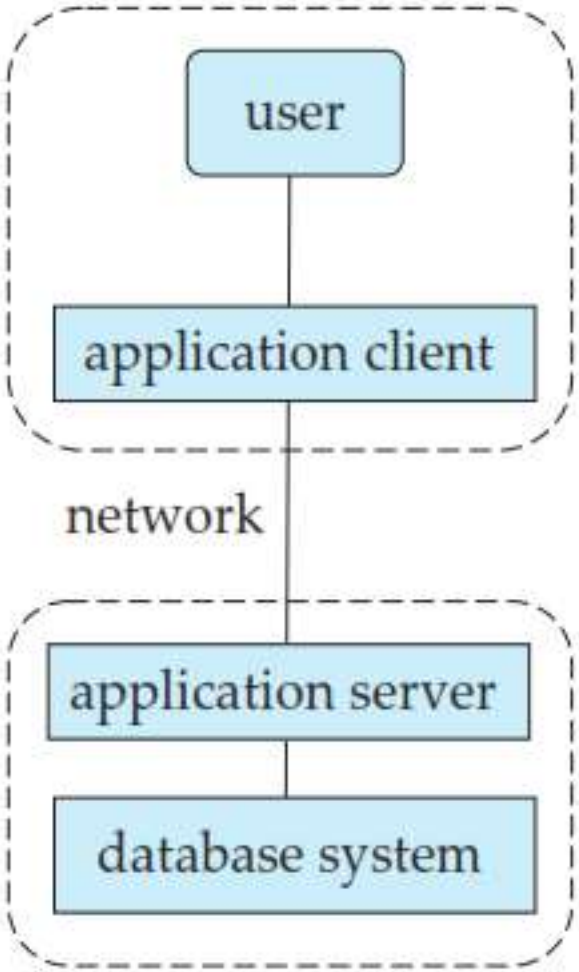
In a **three-tier architecture**, the client machine acts as front end and does not contain any direct database calls. Instead, the client end communicates with an application server, usually through interface.

The **application server** in turn communicates with a database system to access data. The **business logic** of the application, which says what actions to carry out under what conditions, is embedded in the application server.

Three-tier applications are more appropriate for large applications, and for applications that run on the World Wide Web.

DATABASE SYSTEM ARCHITECTURE

3 Tier Architecture



DATA ABSTRACTION

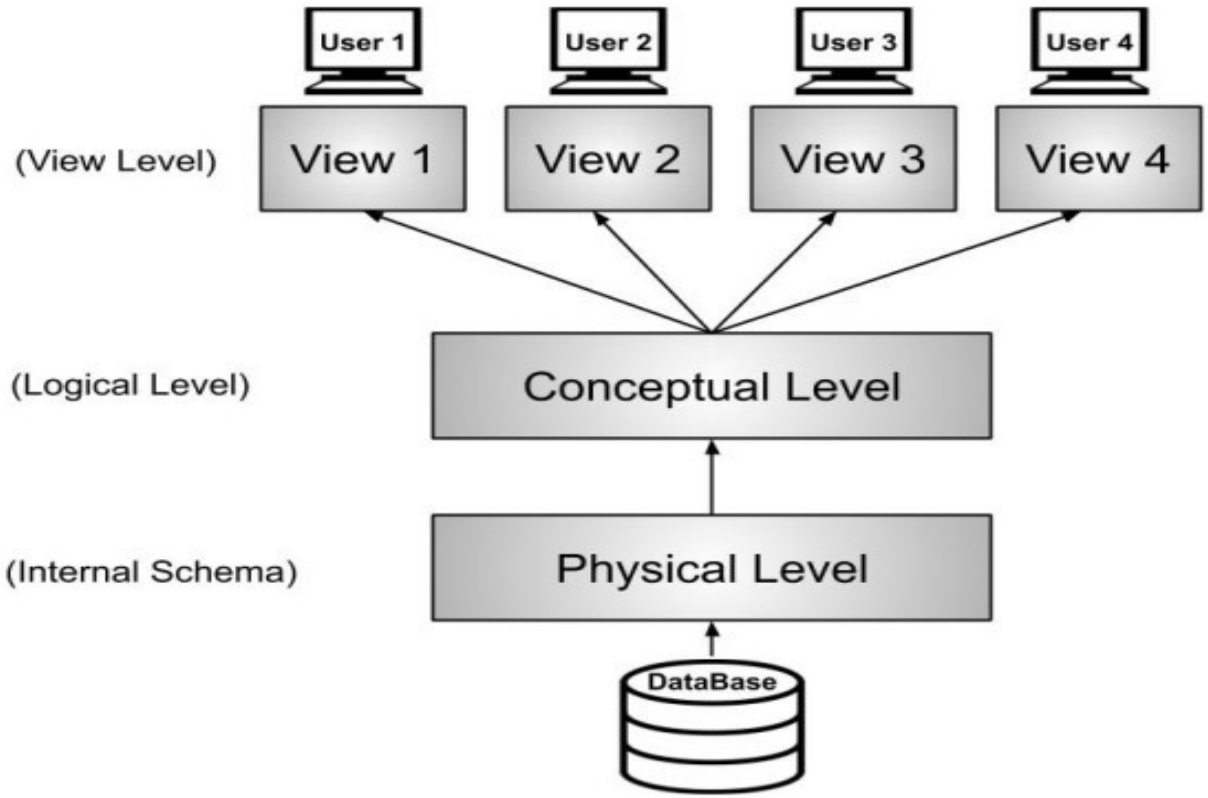
View of Data :

A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data. A major purpose of a database system is to provide users with an ***abstract view*** *of* the data. That is, the system **hides certain details** of how the data are stored and maintained.

There are mainly **3** levels of data abstraction:

1. Physical Level Abstraction
2. Logical Level Abstraction
3. View Level Abstraction

DATA ABSTRACTION



Levels of Data Abstraction

DATA ABSTRACTION

1. Physical Level Abstraction

This is the lowest level of data abstraction. It tells us how the data is actually stored in memory. i.e. it tells the actual location of the data that is being stored by the user.

e.g. The Database Administrators(DBA) decide that which data should be kept at which particular disk drive, how the data has to be fragmented, where it has to be stored etc.

DATA ABSTRACTION

2. Logical Level Abstraction/ (Conceptual Level)

Logical level of abstraction describes ***what data are*** stored in the database, and what relationships exist among those data.

We have different data models by which we can store the data.(e.g. Relational Models, Object Models etc.)

3. View Level Abstraction/

The highest level of abstraction describes only part of the entire database. i.e. The system may provide many views for the same database.

e.g. student and faculty have different view on university database.

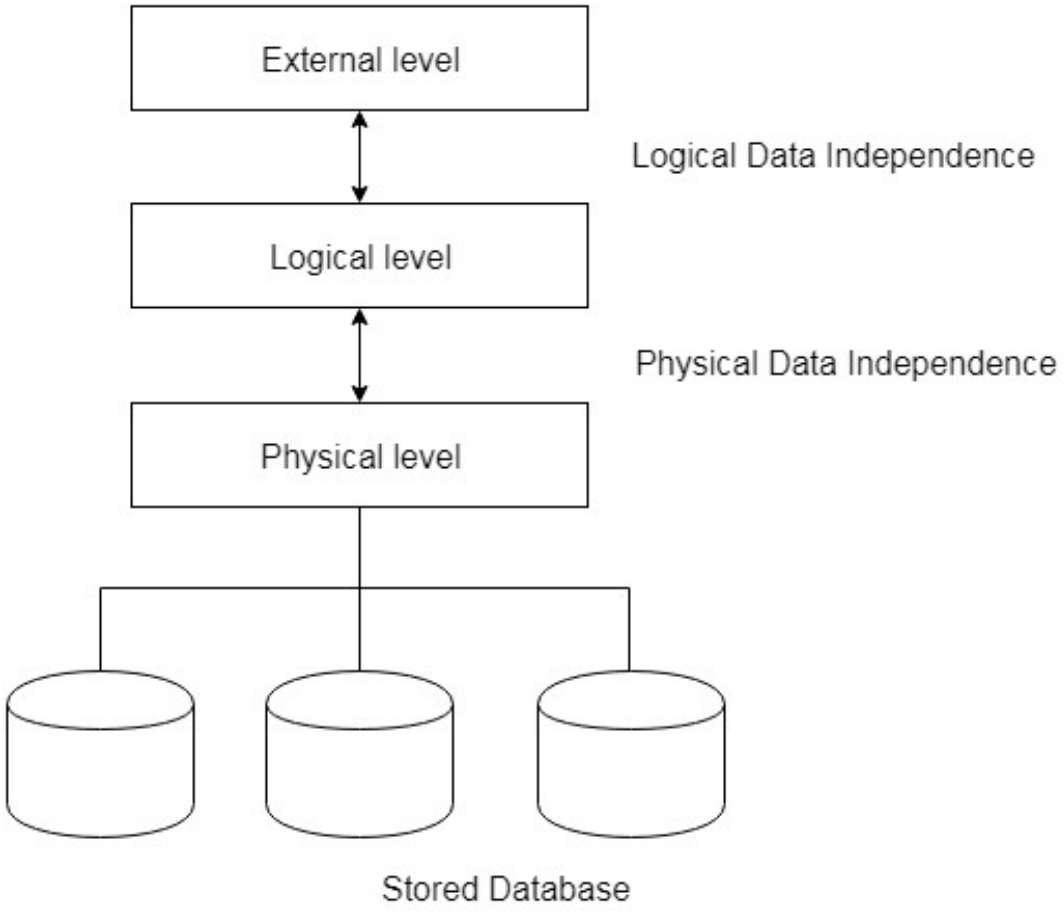
DATA INDEPENDENCE

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. Physical Data Independence
2. Logical Data Independence

DATA INDEPENDENCE



DATA INDEPENDENCE

1. Physical Data Independence

Physical data independence can be defined as the capacity to change the physical schema without having to change the conceptual(Logical) schema.

e.g. Suppose there was a change in the memory size of the database servers. This will not affect the logical structure of any of the objects in the database. They are completely independent of the physical structure. This is called physical data independence.

DATA INDEPENDENCE

2. Logical Data Independence

Logical data independence refers characteristic of being able to change the conceptual(Logical level) schema without having to change the external(view level) schema.

e.g. Any changes to the database objects like changes to the table structure, size, or addition/removal of columns from the table will not affect user views. They will see the data like before. This is called logical data independence.

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.

Data model is to give an idea that how final system or software will look like after development is completed.

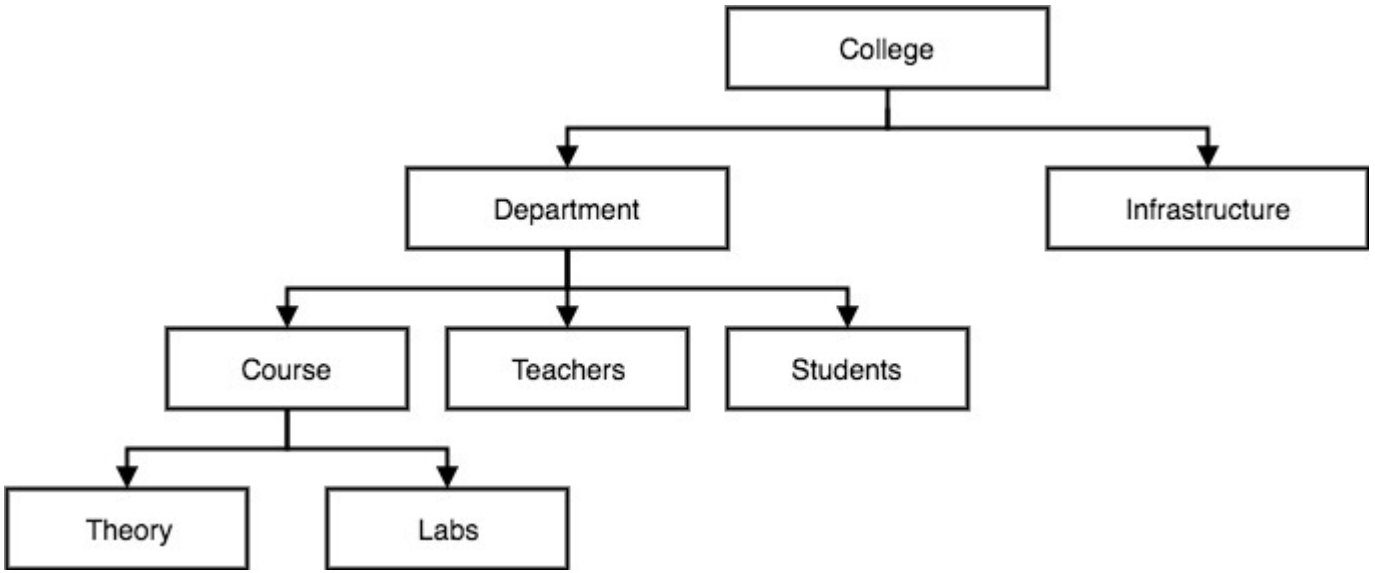
Types of Data Models:

1. Hierarchical Model
2. Network Model
3. Entity-relationship Model
4. Relational Model
5. Object Data Models

DATA MODELS IN DBMS

1. Hierarchical Model

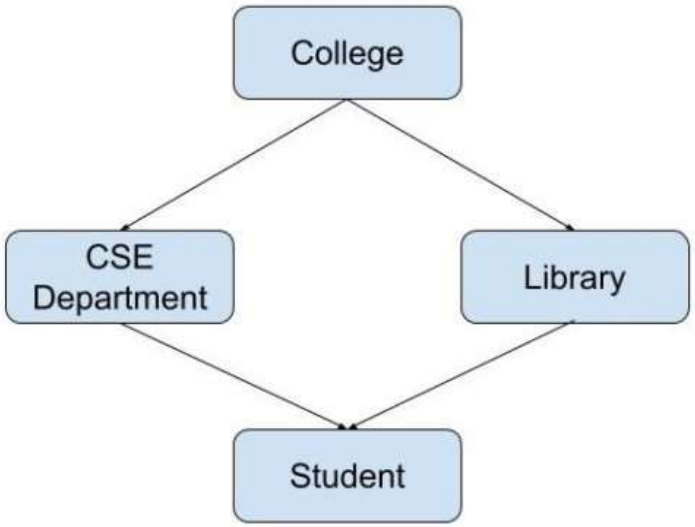
Hierarchical Model was the first DBMS model. This model organizes the data in the hierarchical tree structure. The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node.



DATA MODELS IN DBMS

2. Network Model

This model is an extension of the hierarchical model. It was the most popular model before the relational model. This model is the same as the hierarchical model, the only difference is that a **record can have more than one parent.** It replaces the hierarchical tree with a graph.

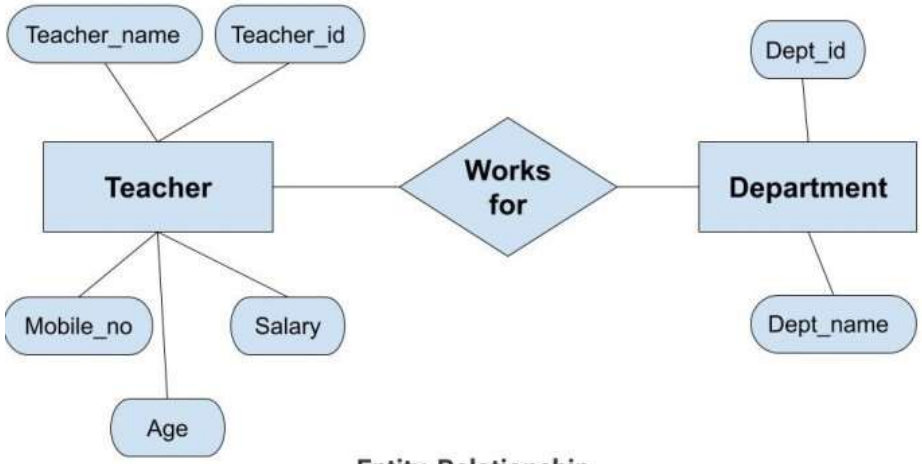


Network Model

DATA MODELS IN DBMS

3. Entity-relationship Model

Entity-Relationship Model or simply ER Model is a high-level data model diagram. In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand. It is also very easy for the developers to understand the system by just looking at the ER diagram.



Entity-Relationship Model

DATA MODELS IN DBMS

4. Relational Model

Relational Model is the most widely used model. In this model, the data is maintained in the form of a two-dimensional table. All the information is stored in the form of **row and columns.** The basic structure of a relational model is tables. So, the tables are also called *relations* in the relational model.

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
3	SUJIT	ROHTAK	9156253131	20
4	SURESH	DELHI		18

DATA MODELS IN DBMS

5. Object Data Models

The real-world problems are more closely represented through the object-oriented data model. In this model, both the data and relationship are present in a single structure known as an object.

In this model, two are more objects are connected through links. We use this link to relate one object to other objects.

DATA DEFINITION LANGUAGE (DDL)

- A data definition language (DDL) is a computer language used to create and modify the structure of database objects in a database. These database objects include views, schemas, tables, indexes, etc.

- This term is also known as data description language in some contexts, as it describes the fields and records in a database table.

DATA MANIPULATION LANGUAGE (DML)

A data manipulation language (DML) is a family of computer languages including commands permitting users to manipulate data in a database. This manipulation involves inserting data into database tables, retrieving existing data, deleting data from existing tables and modifying existing data.

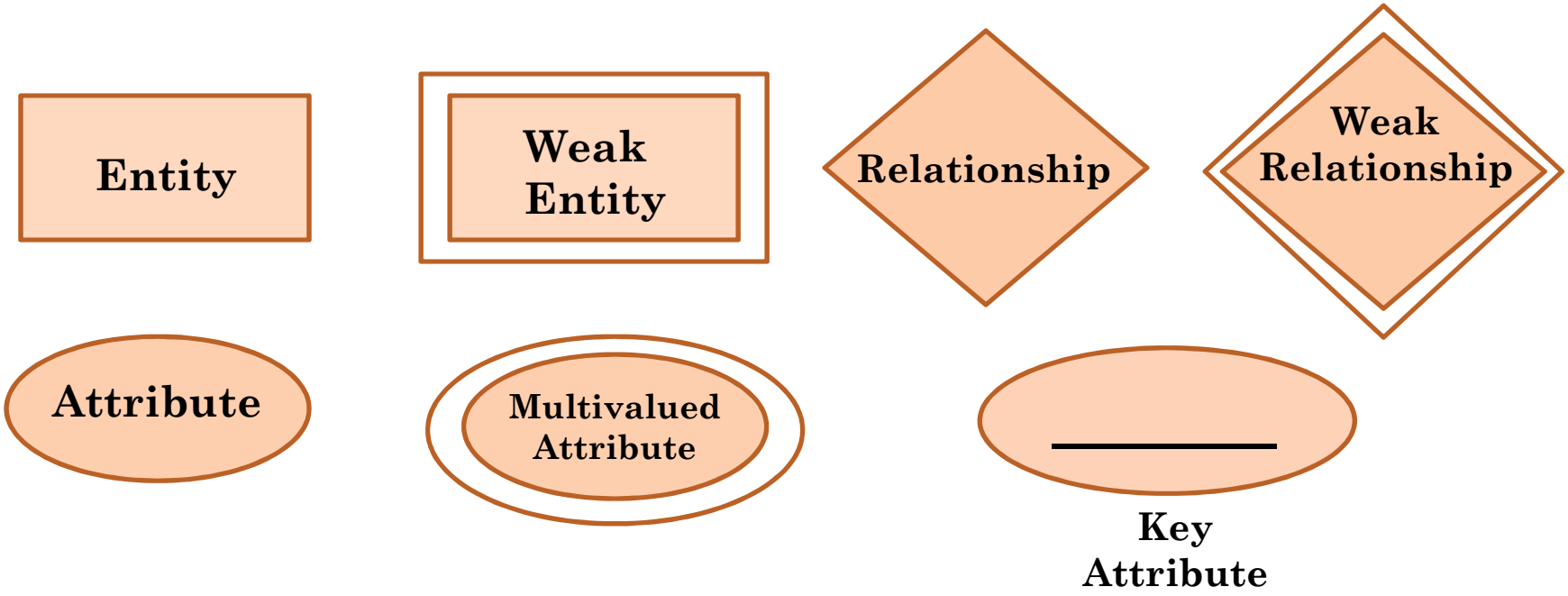
A popular data manipulation language is that of Structured Query Language (SQL), which is used to retrieve and manipulate data in a relational database

ENTITY-RELATIONSHIP MODEL

An **Entity–relationship model (ER model)** describes the structure of a database with the help of a diagram, which is known as **Entity Relationship Diagram (ER Diagram)**. An ER model is a design or blueprint of a database that can later be implemented as a database.

ENTITY-RELATIONSHIP MODEL

ERD entity symbols

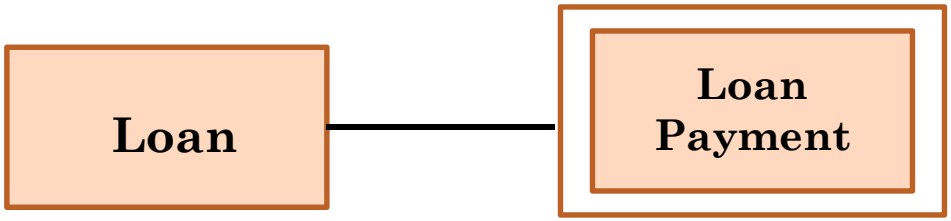


Weak entities or relations can also be represented by Dotted Line

ENTITY-RELATIONSHIP MODEL

ERD entity symbols

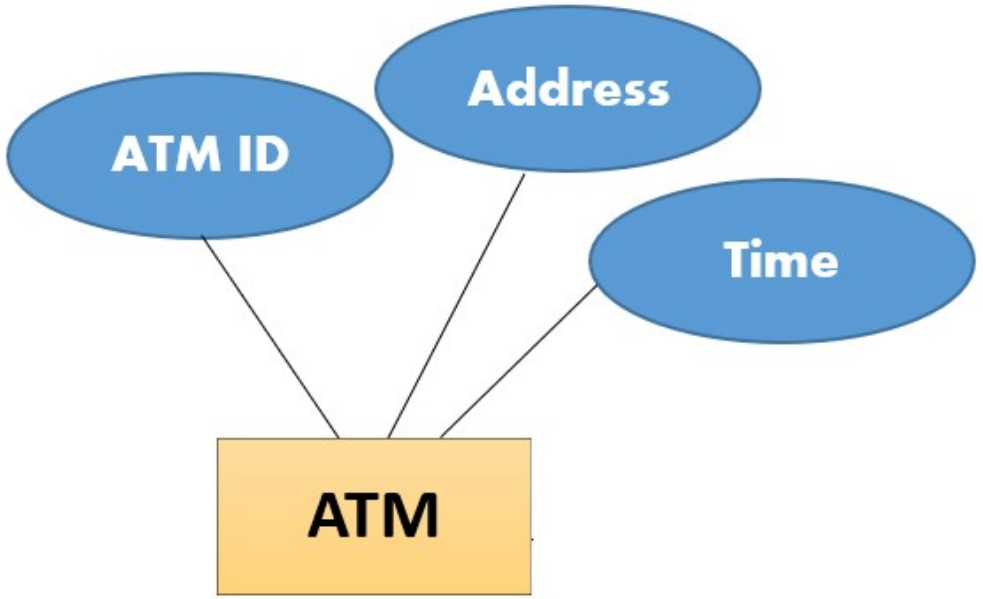
Weak Entity: A weak entity is an entity that depends on the existence of another entity.



ENTITY-RELATIONSHIP MODEL

ERD entity symbols

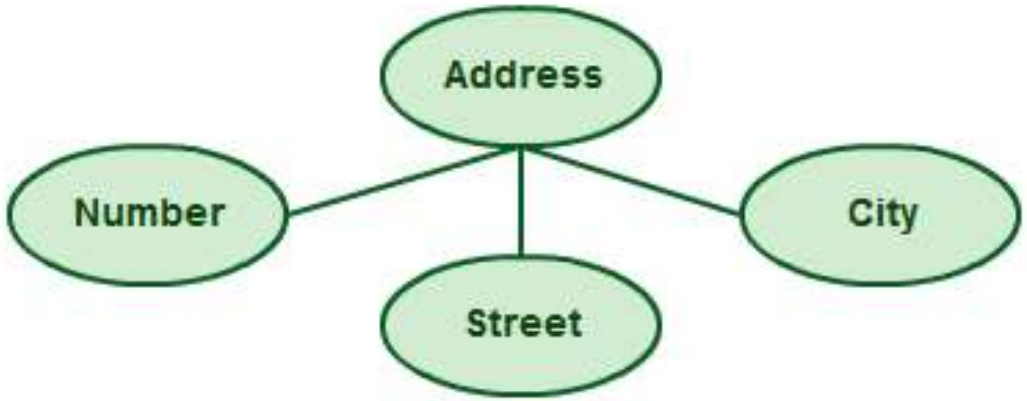
Attribute: Attributes describe property or character of an entity.



ENTITY-RELATIONSHIP MODEL

ERD entity symbols

Composite Attribute : An attribute can also have their own attribute. These attribute are known as composite attribute.



ENTITY-RELATIONSHIP MODEL

ERD entity symbols

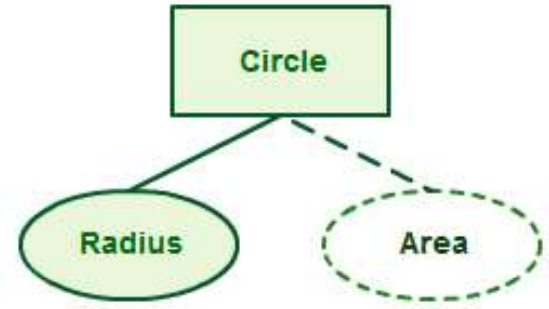
Relationship : Relationship describe relations between entities. Or A relationship describes how entities interact. For example, the entity "Carpenter" may be related to the entity "table" by the relationship "builds" or "makes". Relationships are represented by diamond shapes and are labeled using verbs.

ENTITY-RELATIONSHIP MODEL

ERD entity symbols

Multivalued Attribute: If an attribute can have more than one value it is called a multi-valued attribute. For example, a teacher entity can have multiple subject values.

Derived Attribute: An attribute based on another attribute. This is found rarely in ER diagrams. For example, for a circle, the area can be derived from the radius.



ENTITY-RELATIONSHIP MODEL

Cardinality: The number of times an entity of an entity set participates in a relationship set is known as cardinality.

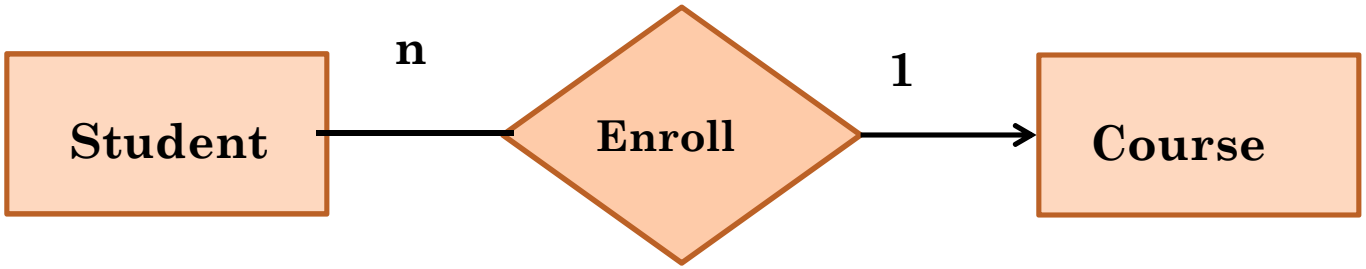
Types of Relations:

1. Binary Relationship : Means Relation between two entities.

1.1. One to One: When each entity in each entity set can take part **only once in the relationship**, the cardinality is one to one.

ENTITY-RELATIONSHIP MODEL

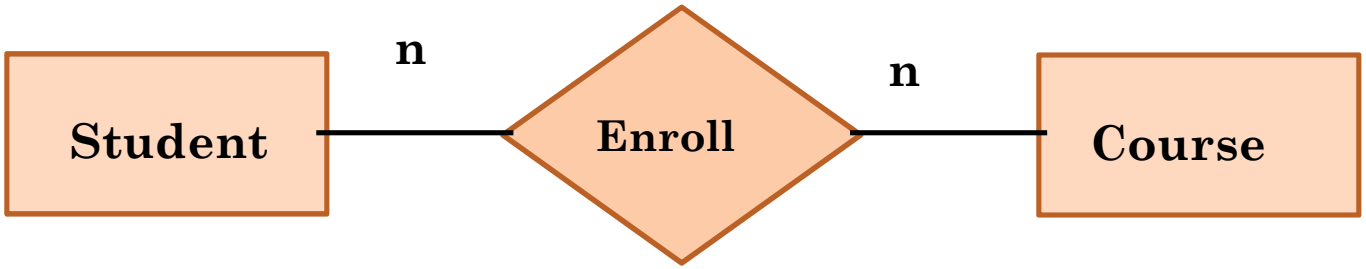
1.2 Many- To- One : When entities in one entity set can take part only once in the relationship set and entities in other entity set can take part more than once in the relationship set, cardinality is many to one.



Many students enrolls only one course but course can have many students.

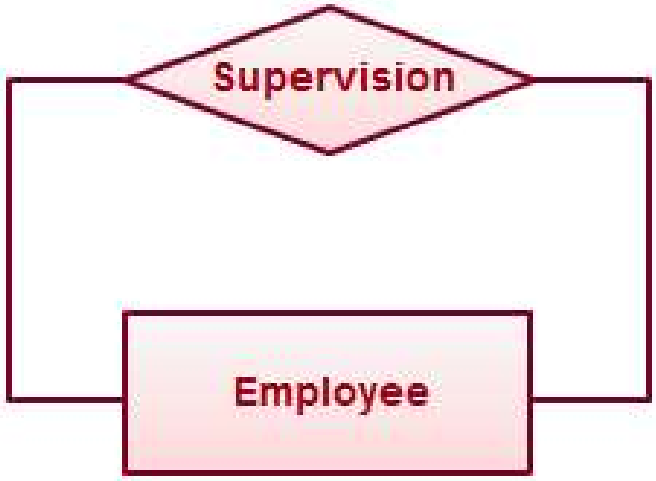
ENTITY-RELATIONSHIP MODEL

1.3 Many- To- Many : When entities in all entity sets can take part more than once in the relationship cardinality is many to many.



ENTITY-RELATIONSHIP MODEL

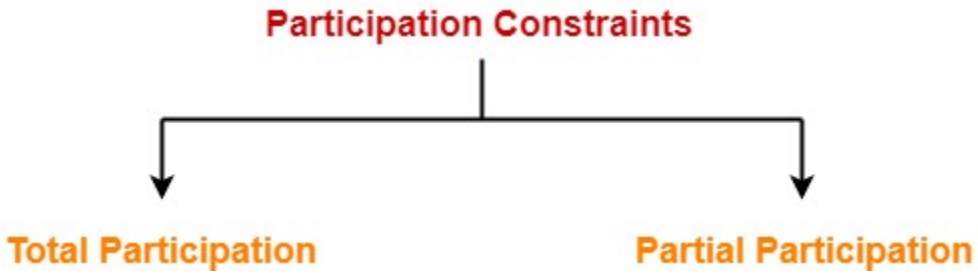
2. Recursive Relationship: If the same entity participates more than once in a relationship it is known as a recursive relationship. In the below example an employee can be a supervisor and be supervised, so there is a recursive relationship.



ENTITY-RELATIONSHIP MODEL

Participation Constraints:

There are two types of participation constraints-



1. Total Participation : Each entity is involved in the relationship. Total participation is represented by double lines.

ENTITY-RELATIONSHIP MODEL

Participation Constraints:

2. Partial Participation : Not all entities are involved in the relationship. Partial participation is represented by single lines.

ENTITY-RELATIONSHIP MODEL

As the complexity of data increased in the late 1980s, it became more and more difficult to use the traditional ER Model for database modelling. Hence some improvements or enhancements were made to the existing ER Model to make it able to handle the complex applications better.

Hence, as part of the **Enhanced ER Model**, along with other improvements, three new concepts were added to the existing ER Model, they were:

1.Generalization

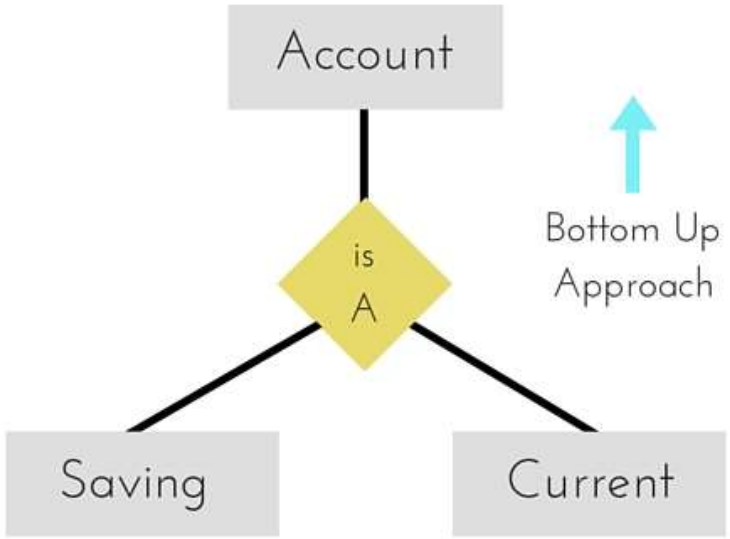
2.Specialization

3.Aggregration

ENTITY-RELATIONSHIP MODEL

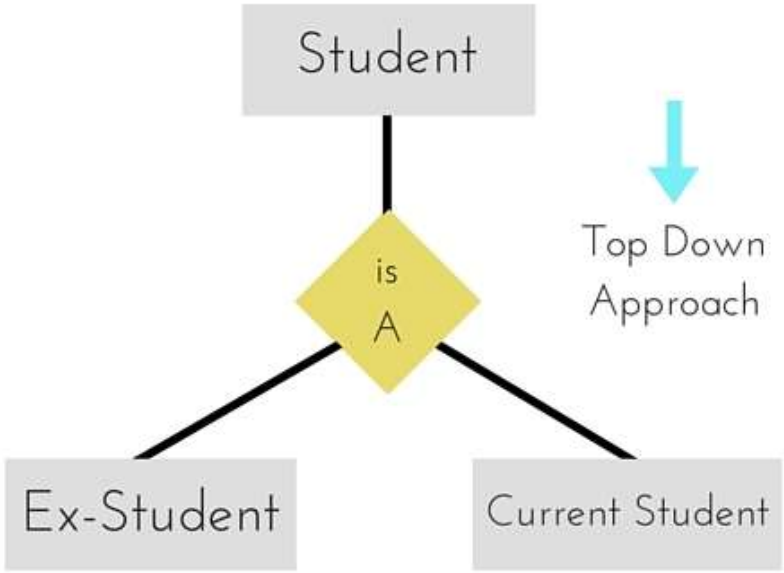
1.Generalization : Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entities to make further higher level entity.

For example, **Saving** and **Current** account types entities can be generalized and an entity with name **Account** can be created, which covers both.



ENTITY-RELATIONSHIP MODEL

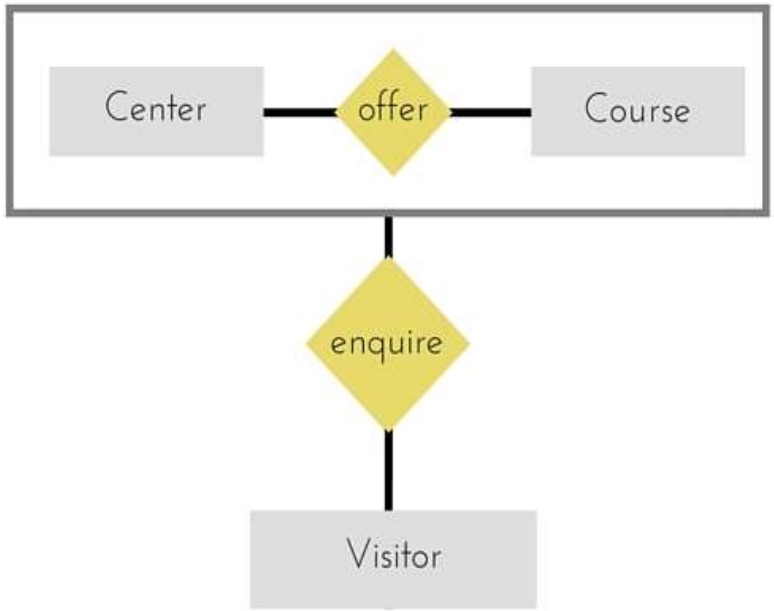
2. Specialization: Specialization is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, a higher level entity may not have any lower-level entity sets, it's possible.

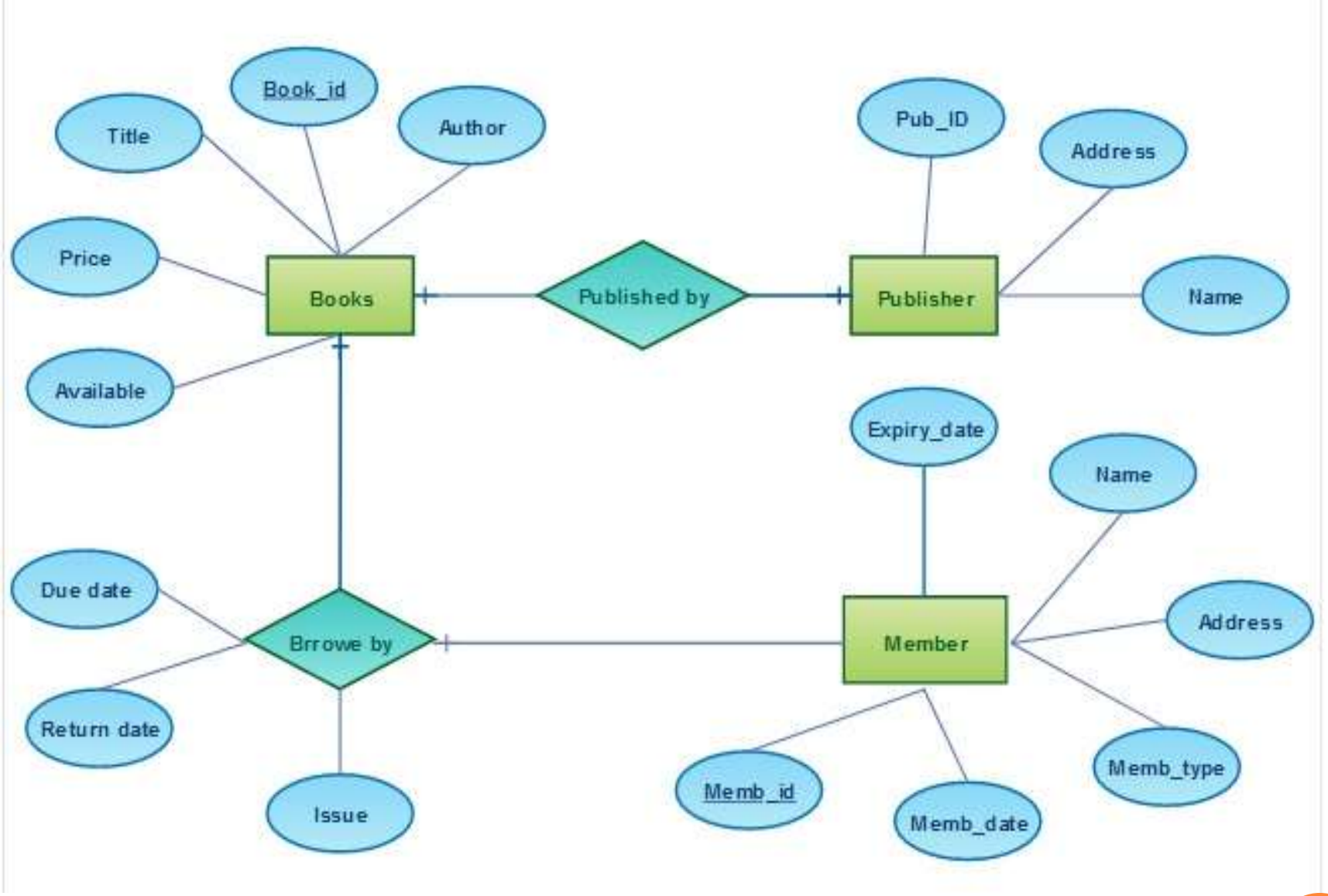


ENTITY-RELATIONSHIP MODEL

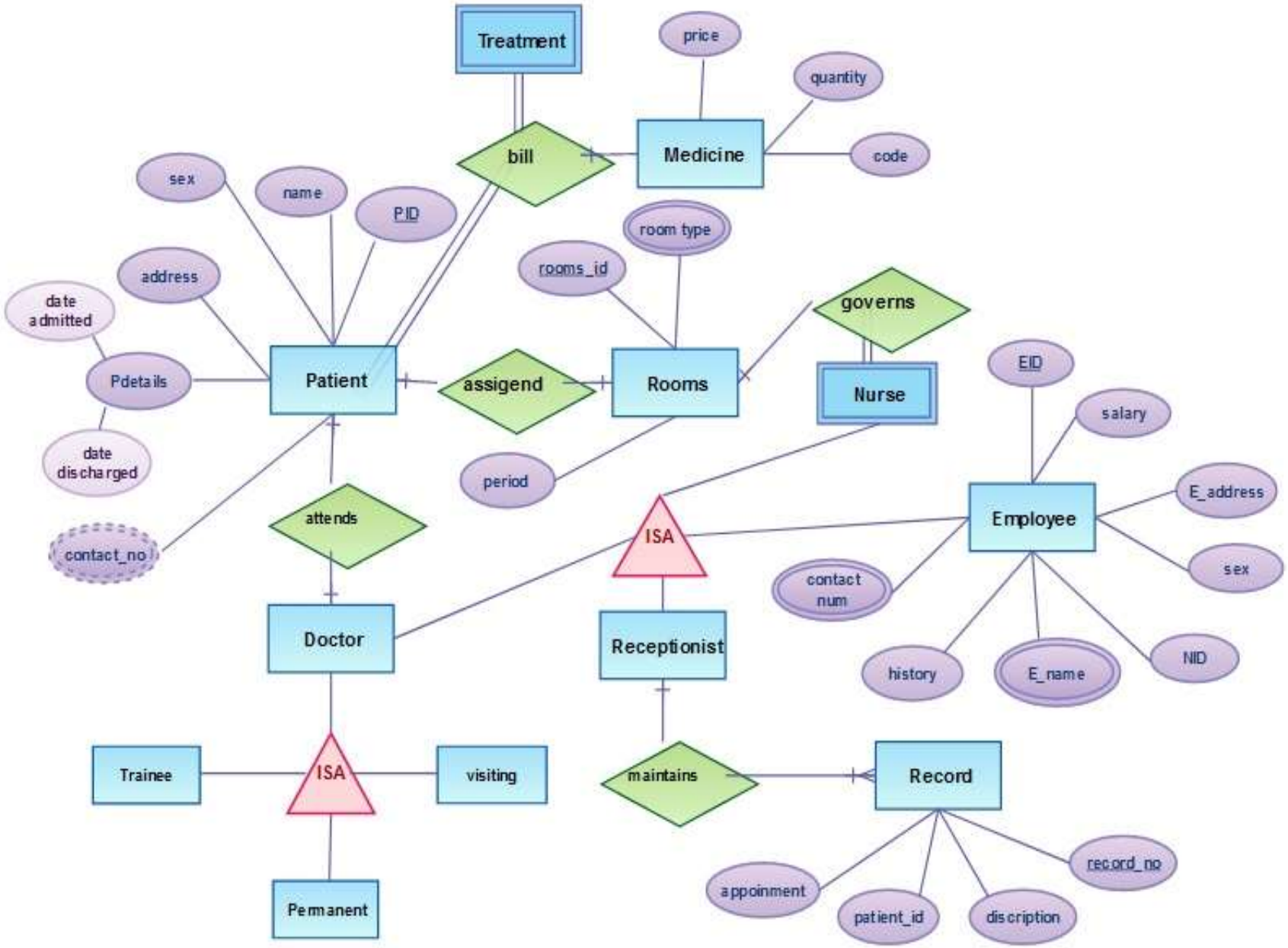
3. Aggregation : Aggregation is a process when relation between two entities is treated as a **single entity**.

In the diagram above, the relationship between **Center** and **Course** together, is acting as an Entity, which is in relationship with another entity **Visitor**. Now in real world, if a Visitor or a Student visits a Coaching Center, he/she will never enquire about the center only or just about the course, rather he/she will ask enquire about both.





ER Diagram Library Management System



ER Diagram For Hospital Management

RELATIONAL INTEGRITY CONSTRAINT

Integrity constraints ensure that changes made to the database by authorized users do not result in a loss of data consistency. Thus, integrity constraints guard against accidental damage to the database.

Examples of integrity constraints are:

-An instructor name cannot be ***null.***

-No two instructors can have the same instructor ID.

-Every department name in the ***course*** relation must have a matching department name in the *department* relation.

- The budget of a department must be greater than Rs 0.0.

RELATIONAL INTEGRITY CONSTRAINT

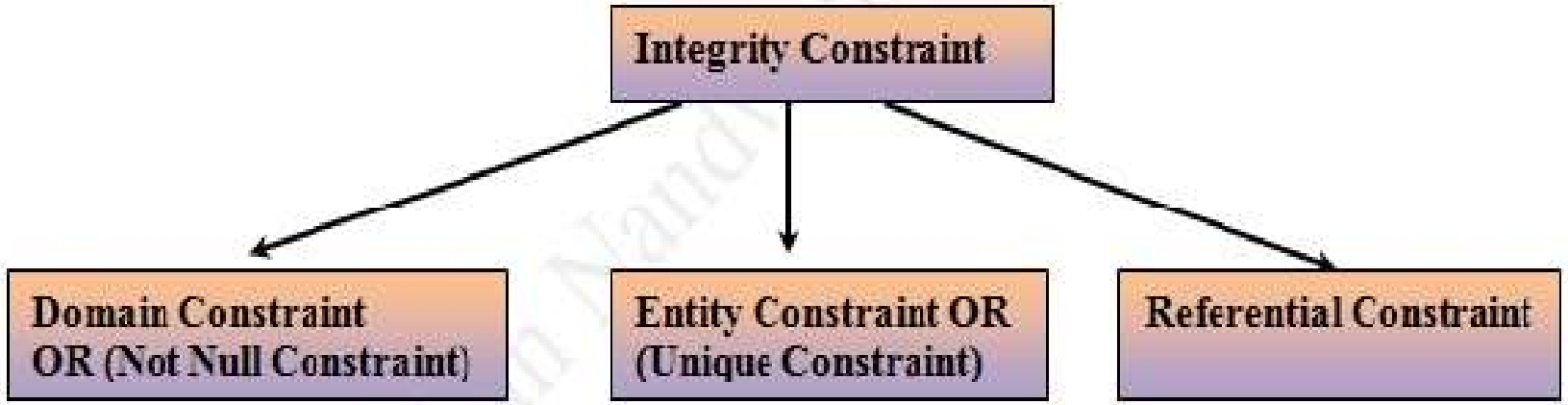
-Integrity constraints are usually identified as part of the database schema design process.

-So, Integrity constraints are a set of rules. It is used to maintain the quality of information.

- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.

RELATIONAL INTEGRITY CONSTRAINT

Types of Integrity Constraint



RELATIONAL INTEGRITY CONSTRAINT

1. Domain Constraint or Not Null Constraint

- Domain constraints can be defined as the definition of a valid set of values for an attribute.

- The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

- Consider a tuple (row) in the ***student*** table where ***name*** is ***null***. Such a row gives student information for an unknown student; thus, it does not contain useful information. Similarly, we would not want the department budget to be ***null.***

RELATIONAL INTEGRITY CONSTRAINT

- The **not null** constraint prohibits the insertion of a null value for the attribute. So by restricting the domain of the attributes *name* and *budget* to exclude null values, by declaring it as follows:

name **varchar**(20) **not null**

budget **numeric**(12,2) **not null**

RELATIONAL INTEGRITY CONSTRAINT

2. Entity Constraint

- The entity integrity constraint states that primary key value can't be null or contains unique value. This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.

- A table can contain a null value other than the primary key field.

RELATIONAL INTEGRITY CONSTRAINT

Student ID	Student Name	Dept_No
1	N. Bhushan	101
	Lobhas	102

Primary Key Can't Contains Null Value

RELATIONAL INTEGRITY CONSTRAINT

3. Referential Integrity

A referential integrity constraint is specified between two tables. Means Primary key of table B is used as foreign key in table A. This condition is called **referential integrity**. Then every value of the Foreign Key in Table A must be null or be available in Table B.

RELATIONAL INTEGRITY CONSTRAINT

Foreign Key

Student ID	Student Name	Dept_No
1	N. Bhushan	101
2	Lobhas	103
3	Raghavi	102

Not Allowed
Because 103 is not
in Table Dept.

Table A

Primary Key

Dept_No	Dept Name
101	Computer
102	IT

Table B

DATA MANIPULATION LANGUAGE

A **data manipulation language** (**DML**) is a computer programming language used for adding (inserting), deleting, and modifying (updating) data in a database.

Main SQL Statements are

1. Select - For Retrieval
2. Insert into - For Insertion
3. Delete - For Deletion
4. Update - For Updating


SVKM


SVKM


SVKM


SVKM


SVKM