R23 DBMS SYLLABUS

III SEMESTER CODE:23CS3T03	L	T	P	C
	3	-	-	3
DATABASE MANAGEMENT SYSTEMS				

Course Objectives:

The main objectives of the course is to

- Introduce database management systems and to give a good formal foundation on the relational model of data and usage of Relational Algebra
- Introduce the concepts of basic SQL as a universal Database language
- Demonstrate the principles behind systematic database design approaches by covering conceptual design, logical design through normalization
- Provide an overview of physical design of a database system, by discussing Database indexing techniques and storage techniques

Course Outcomes:

- Understand database systems, characteristics, architectures, and ER modeling.
- Learn the relational model, constraints, and basic SQL operations
- Perform advanced SQL queries and manage relational databases
- Apply normalization techniques and understand functional dependencies
- Grasp transaction properties, concurrency control, recovery, and indexing methods.

UNIT I:

Introduction: Database system, Characteristics (Database Vs File System), Database Users, Advantages of Database systems, Database applications, Database Languages. Brief introduction of different Data Models; Concepts of Schema, Instance and data independence; Three tier schema architecture for data independence; Database system structure, environment, Centralized and Client Server architecture for the database.

Entity Relationship Model: Introduction, Representation of entities, attributes, entity set, relationship, relationship set, constraints, sub classes, super class, inheritance, specialization, generalization using ER Diagrams.

UNIT II:

Relational Model: Introduction to relational model, concepts of domain, attribute, tuple, relation, importance of null values, constraints (Domain, Key constraints, integrity constraints) and their importance, Relational Algebra, Relational Calculus.

BASIC SQL: Simple Database schema, data types, table definitions (create, alter), different DML operations (insert, delete, update).

UNIT III:

SQL: Basic SQL querying (select and project) using where clause, arithmetic & logical operations, SQL functions(Date and Time, Numeric, String conversion).Creating tables with relationship, implementation of key and integrity constraints, nested queries, sub queries, grouping, aggregation, ordering, implementation of different types of joins, view(updatable and non-updatable), relational set operations.

UNIT IV:

Schema Refinement (Normalization): Purpose of Normalization or schema refinement, concept of functional dependency, normal forms based on functional dependency Lossless join and dependency preserving decomposition, (1NF, 2NF and 3 NF), concept of surrogate key, Boyce-Codd normal form(BCNF), MVD, Fourth normal form(4NF), Fifth Normal Form (5NF).

UNIT V:

Transaction Concept: Transaction State, ACID properties, Concurrent Executions, Serializability, Recoverability, Implementation of Isolation, Testing for Serializability, lockbased,time stamp based ,optimistic, concurrency protocols, Deadlocks, Failure Classification, Storage, Recovery and Atomicity, Recovery algorithm.
Introduction to Indexing Techniques: B+ Trees, operations on B+ Trees, Hash Based Indexing.

Text Books:

1) Database Management Systems, 3rd edition, Raghurama Krishnan, Johannes Gehrke, TMH (For Chapters 2, 3, 4)
2) Database System Concepts,5th edition, Silberschatz, Korth, Sudarsan,TMH (For Chapter 1 and Chapter 5)

Reference Books:

1) Introduction to Database Systems, 8thedition, C J Date, Pearson.
2) Database Management System, 6th edition, Ramez Elmasri, Shamkant B. Navathe, Pearson
3) Database Principles Fundamentals of Design Implementation and Management, Corlos Coronel, Steven Morris, Peter Robb, Cengage Learning.

Web-Resources:

1) https://nptel.ac.in/courses/106/105/106105175/
2) https://infyspringboard.onwingspan.com/web/en/app/toc/lex_auth_01275806667282022456_shared/overview

UNIT-I

INTRODUCTION

What is Data?

Data is a collection of a raw facts, stored and used for future analysis or reference. It can be used in a variety of forms like text, numbers, media, bytes, mobile no:9876545555 etc.

Information:

Information in DBMS is processed, organized, or summarized data. It may be defined as a collection of related data that, when put together, becomes a useful message to a recipient.

Example of information: The number given above belongs to Dr S.P.S becomes information.

What is Database?

A database is an organized collection of inter related data stored at one place and accesses by different users.

You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.

The **main purpose** of the database is to operate a large amount of information by storing, retrieving, and managing data.

There are many databases available like MySQL, Sybase, Oracle, MongoDB, Informix, PostgreSQL, SQL Server, etc.

SQL or Structured Query Language is used to operate on the data stored in a database. SQL dependson relational algebra and tuple relational calculus.

DBMS

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

The Database Management System is a collection of inter related data and set of programs used to store and access the data in an efficient and effective manner.

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

CHARACTERISTICS OF A DATABASE SYSTEM

There are so many characteristics of a database management system, which are as follows

- A database management system is able to store any kind of data in a database.
- The database management system has to support ACID (atomicity, consistency, isolation, durability) properties.
- The Database management system allows so many users to access databases atthe same time.
- Backup and recovery are the two main methods which allow users to protect the data from damage or loss.
- It also provides multiple views for different users in a single organization.
- It follows the concept of normalization which is helpful to minimize the redundancy of a

relation.

- It also provides users query language, helpful to insert, retrieve, update, anddelete the data in a data

Difference between File System and DBMS

File System :

The file system is basically a way of arranging the files in a storage medium like a hard disk.

DBMS(Database Management System) :

Database Management System is basically software that manages the collection of related data. It is used for storing data and retrieving the data effectively when it is needed.

Difference between File System and DBMS:

DATABASE Vs FILE SYSTEM

Basis	**FILE SYSTEM**	**DBMS**
Data Redundancy	Redundant data can be present in file system.	There is no redundant data.
Consistency	There is less data consistency in file system.	There is more data consistency because of the process of normalization.
Security	File System provide less Security.	DBMS has more security mechanisms as compared to the file system.
Complexity	It is less Complex as compared to DBMS.	It has more complexity in handling as compared to the file system.
User Access	Only one user can access data at a time.	Multiple users can access data at a time.
Cost	It is less Expensive than DBMS.	It has a comparatively higher cost than a file system.
Query Processing	There is no efficient query processing in the file system.	Efficient query processing is there in DBMS.
Backup and Recovery	It doesn't provide backup and recovery of data if it is lost.	It Provides backup and recovery of data even if it is lost.
Data Independence	There is no Data Independence.	In DBMS data independence exists.
Sharing	Data is distributed in many files.So not easy to share data.	Due to centralized nature,sharing is easy.
Structure	Simple	Complex
Suitable	Suitable for small organizations.	Suitable for both small & large

		organizations.
Integrity issues	Has Integrity issues.	No Integrity issues.

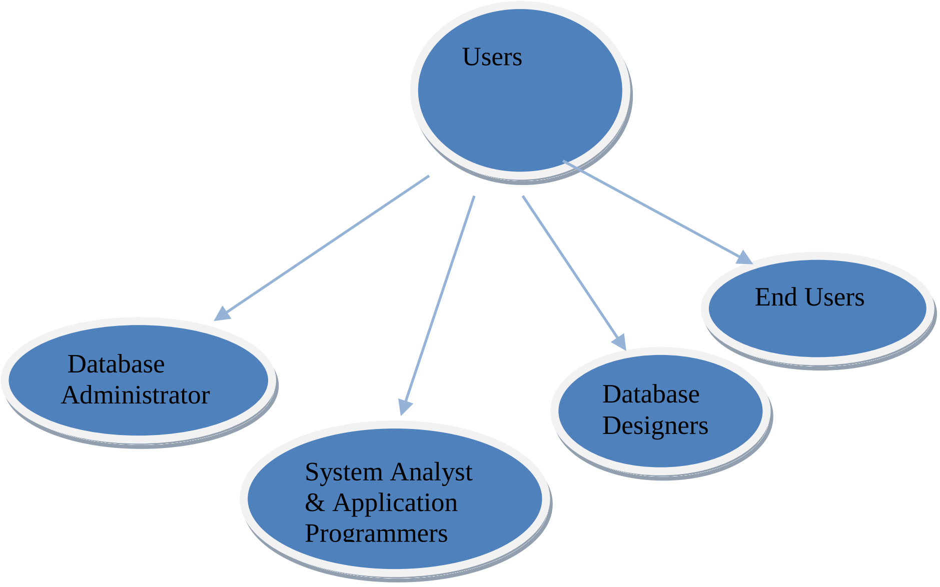
DATABASE USERS

Actors on the Scene:

Classification of Users

DBMS mainly classified into Four users −

- Database Administrator
- System Analyst & Application Programmers.
- Database Designers
- End Users.



1.Data base Administrator:
The functions of a DBA include the following

i) Schema Definition : The DBA creates the original database schema by executing a set of ddl statements.

ii) Schema and Physical Organization modification: The DBA carries out changes to the schema and physical organization to reflect the changes needs to the organization or to alter the physical organization to improve the performance.

iii) Granting of Authorization for data access: DBA grants different types of authorizations for different users.The authorization information is kept in special system structure that the database system consults whenever it requires.

iv) Routine Maintenance:
i) Backing up the database either in tapes or remote servers.
ii) Prevent loss of data in case of disasters, floods, System failures etc.
iii) Ensuring that enough free disk space is available for normal operations and upgrading disk space.

2. System Analysts and Application Programmers:

System Analysts : Determine needs of end users, especially naive and parametric users,and develop specifications for canned transactions that meet these needs. A System Analyst has also known as a business technology analyst. These professionals are responsible for the design, structure, and properties of databases.

Application Programmers: The application programmer uses the specifications provided by the system analyst to construct the software that is used by end users. Implement, test, document, and maintain programs that satisfy the specifications mentioned above.

3. Database Designers: These are responsible for identifying the data to be stored and for choosing an appropriate way to organize it. Database designers typically interact with each potential group of users and develop views of the database that meet the data and processing requirements of these groups. The final database design must be capable of supporting the requirements of all user groups.

4. End Users: These are persons who access the database for querying,updating,and report generation. There are several categories of end users

i) Casual end users : use database occasionally ,needing different information each time; use query language to specify their requests ; typically middle or high-level managers.

ii) Naive/parametric end users: Parametric End Users are the unsophisticated who don’t have any DBMS knowledge but they frequently use the database applications in their daily life to get the desired results. Biggest group of users;

For example, Railway’s ticket booking users are naive users. Clerks in any bank is a naive user because they don’t have any DBMS knowledge but they still use the database and perform their given task.

iii) Sophisticated end users: Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database. They can develop their own database applications according to their requirement. They don’t write the program code but they interact the database by writing SQL queries directly through the query processor.

iv) Stand-alone users: maintain personal database by using ready- made program packages that provide easy-to-use-menu-based or graphics-based interface. **Example:** user of a tax package that stores a variety of personal financial data for tax purposes.

v) Specialised Users: The special users are responsible for writing specialized database-related programs and also have the task of creating the actual database as well as implementing technical controls needed to enforce policies and decisions.

Workers behind the Scene:

DBMS system designers and implementers : Design and implement the DBMS modules and interfaces as a software package. A DBMS is a very complex software system that consists of many components, or **modules**, including modules for implementing the catalog, query language processing etc.The DBMS must interface with other system software such as the operating system and compilers for various programming languages.

Tool developers : Design and implement **tools**—the software packages that facilitate database modeling and design, database system design, and improved performance. Tools are optional packages that are often purchased separately. In many cases, independent software vendors develop and market these tools.

Operators and maintenance personnel :(system administration personnel) are responsible for the actual running and maintenance of the hardware and software environment for the database system.

ADVANTAGES OF DATABASE SYSTEM

Reducing Data Redundancy:

The file based data management systems contained multiple files that were stored in many different locations in a system or even across multiple systems. Because of this, there were sometimes multiple copies of the same file which lead to data redundancy.

This is prevented in a database as there is a single database and any change in it is reflected immediately. Because of this, there is no chance of encountering duplicate data.

Sharing of Data

In a database, the users of the database can share the data among themselves. There are various levels of authorization to access the data, and consequently the data can only be shared based on the correct authorization protocols being followed**.**

Data Integrity

Data integrity means that the data is accurate and consistent in the database. Data Integrity is very important as there are multiple databases in a DBMS. All of these databases contain data that is visible to multiple users. So it is necessary to ensure that the data is correct and consistent in all the databases and for all the users.

Data Security

Data Security is vital concept in a database. Only authorized users should be allowed to access the database and their identity should be authenticated using a username and password. Unauthorized users should not be allowed to access the database under any circumstances as it violates the integrity constraints.

Concurrent Access

Databases handle concurrent access by multiple users or applications, ensuring that data remains consistent even when accessed simultaneously.

Privacy

The privacy rule in a database means only the authorized users can access a database according to its privacy constraints. There are levels of database access and a user can only view the data he is allowed to. For example - In social networking sites, access constraints are different for different accounts a user may want to access.

Backup and Recovery

Database Management System automatically takes care of backup and recovery. The users don't need to backup data periodically because this is taken care of by the DBMS. Moreover, it also restores the database after a crash or system failure to its previous condition.

Data Consistency

Data consistency is ensured in a database because there is no data redundancy. All data appears consistently across the database and the data is same for all the users viewing the database. Moreover, any changes made to the database are immediately reflected to all the users and there is no data inconsistency.

DISADVANTAGES OF DATABASE SYSTEM

Cost: Setting up and maintaining a database system can be expensive, including hardware, software licenses, and personnel costs.

Complexity: Managing databases can be complex, and organizations may require skilled database administrators to ensure optimal performance, security, and reliability.

Data Migration: Moving data between different database systems or versions can be challenging and time-consuming.

Security Concerns: While databases offer security features, they are still vulnerable to security breaches if not configured and maintained properly.

Compatibility Issues: Different database management systems (DBMS) have variations in SQL syntax and features, which can lead to compatibility issues when migrating or using different DBMS.

DATABASE APPLICATIONS

Banking: For Customer information, accounts, loans, and banking transactions.

Airlines: For reservations and schedule information .Airlines were among the first to use databases in a geographically distributed manner.

Universities: For student information ,course registrations and grades.

Credit card transactions: For purchases on credit cards and generation of monthly statements.

Telecommunications: For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.

Finance: For storing information about holdings,sales,and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable on-line trading by customers and automated trading by the firm.

Sales: For customer,product,and purchase information.

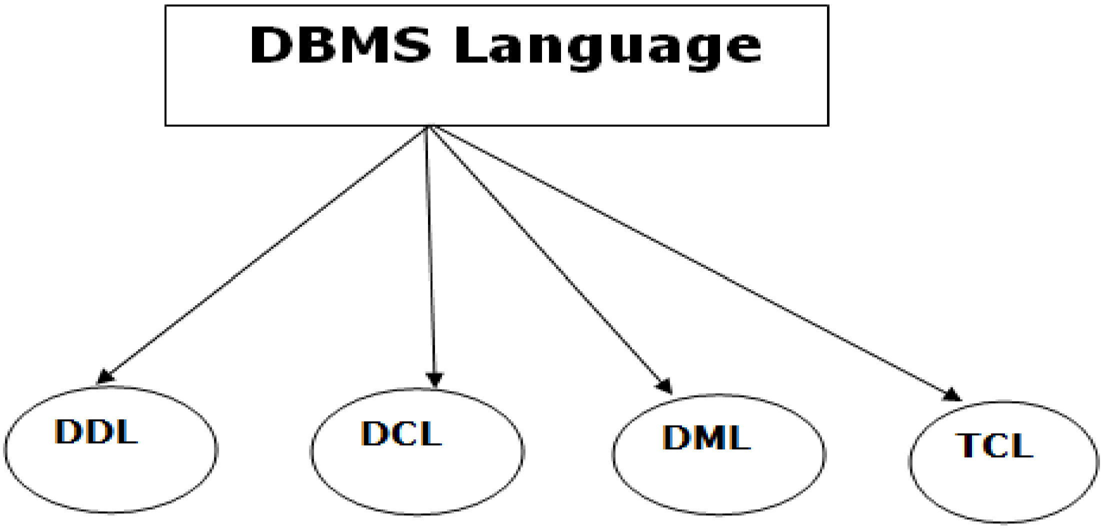
Online retailers: For sales data noted above plus online order tracking ,generation of recommendation lists, and maintenance of online product evaluation.

Manufacturing: For management of supply chain and for tracking production of items in factories ,inventories of items in warehouses and stores, and orders for items.

HumanResources: For information about employees,salaries,payroll taxes,benefits,and for generation of paychecks.

DATABASE LANGUAGES

- A DBMS has appropriate languages and interfaces to express database queries and updates.
- Database languages can be used to read, store and update the data in the database.



1. Data Definition Language (DDL)

- **DDL** stands for **D**ata **D**efinition **L**anguage. It is used to define database structure or pattern.
- It is used to create schema, tables, indexes, constraints, etc. in the database.
- Using the DDL statements, you can create the skeleton of the database.
- Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Here are some tasks that come under DDL:

- **Create:** It is used to create objects in the database.
- **Alter:** It is used to alter the structure of the database.
- **Drop:** It is used to delete objects from the database.
- **Truncate:** It is used to remove all records from a table.
- **Rename:** It is used to rename an object.

2. Data Manipulation Language (DML)

DML stands for **D**ata **M**anipulation **L**anguage. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.
- **Insert:** It is used to insert data into a table.
- **Update:** It is used to update existing data within a table.
- **Delete:** It is used to delete all records from a table.

3. Data Control Language (DCL)

- **DCL** stands for **D**ata **C**ontrol **L**anguage. It is used to retrieve the stored or saved data.
- The DCL execution is transactional. It also has rollback parameters.

(But in Oracle database, the execution of data control language does not have the feature of rolling back.)

Here are some tasks that come under DCL:

- **Grant:** It is used to give user access privileges to a database.
- **Revoke:** It is used to take back permissions from the user.

4. Transaction Control Language (TCL)

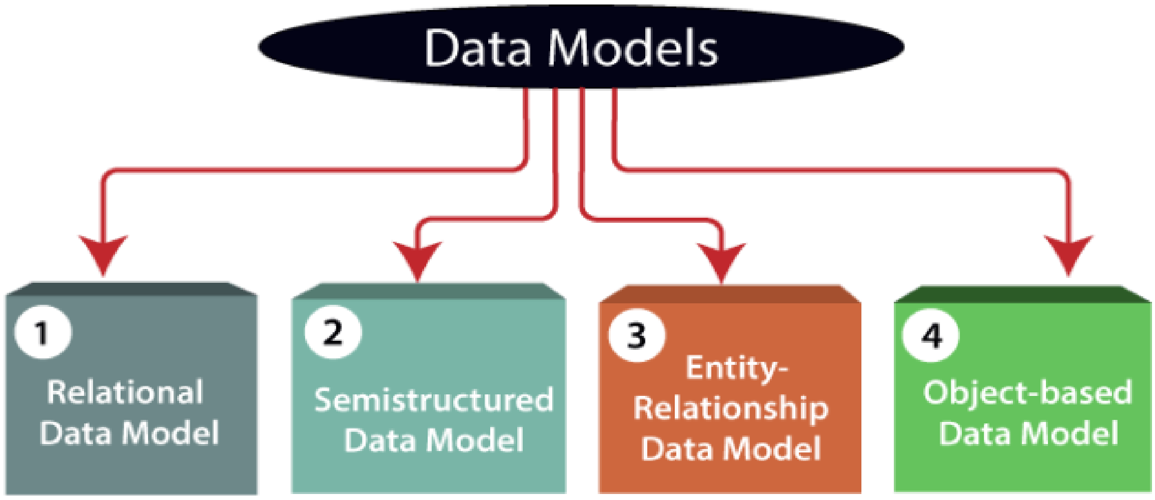
TCL is used to run the changes made by the DML statement. TCL can be grouped into a logical transaction.

Here are some tasks that come under TCL:

- **Commit:** It is used to save the transaction on the database.
- **Rollback:** It is used to restore the database to original since the last Commit.

DATA MODELS

Data Model is the modeling of the data description, data semantics, and consistency constraints of the data.



Relational Data Model: This type of model designs the data in the form of rows and columns within a table. Thus, a relational model uses tables for representing data and in-between relationships. Tables are also called relations. The relational data model is the widely used model which is primarily used by commercial data processing applications.

Entity-Relationship Data Model: An ER 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. A set of attributes describe the entities.

For example, student_name, student_iddescribes the 'student' entity. A set of the same type of entities is known as an 'Entity set', and the setof the same type of relationships is known as 'relationship set'.

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.

Semi Structured Data Model: This type of data model is different from the other three data models (explained above). The semi structured data model allows the data specifications at places where the individual data items of the same type may have different attributes sets.

SCHEMA AND INSTANCE

Schema

The overall design of a database is called **schema.** A database schema is the skeleton structure of the database. It represents the logical view of the entire database.

A schema contains schema objects like table, foreign key, primary key, views, columns, data types, stored procedure, etc. A database schema can be represented by using the visualdiagram. That diagram shows the database objects and relationship with each other.

A database schema is designed by the database designers to help programmers whose software will interact with the database. The process of database creation is called **data modeling.**

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 INDEPENDENCE

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

Data independence refers to modify the schema at one level of the database system withoutaltering the schema at the next higher level.

There are two types of data independence:

1. Logical Data Independence

Logical data independence refers 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 wouldnot 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**.

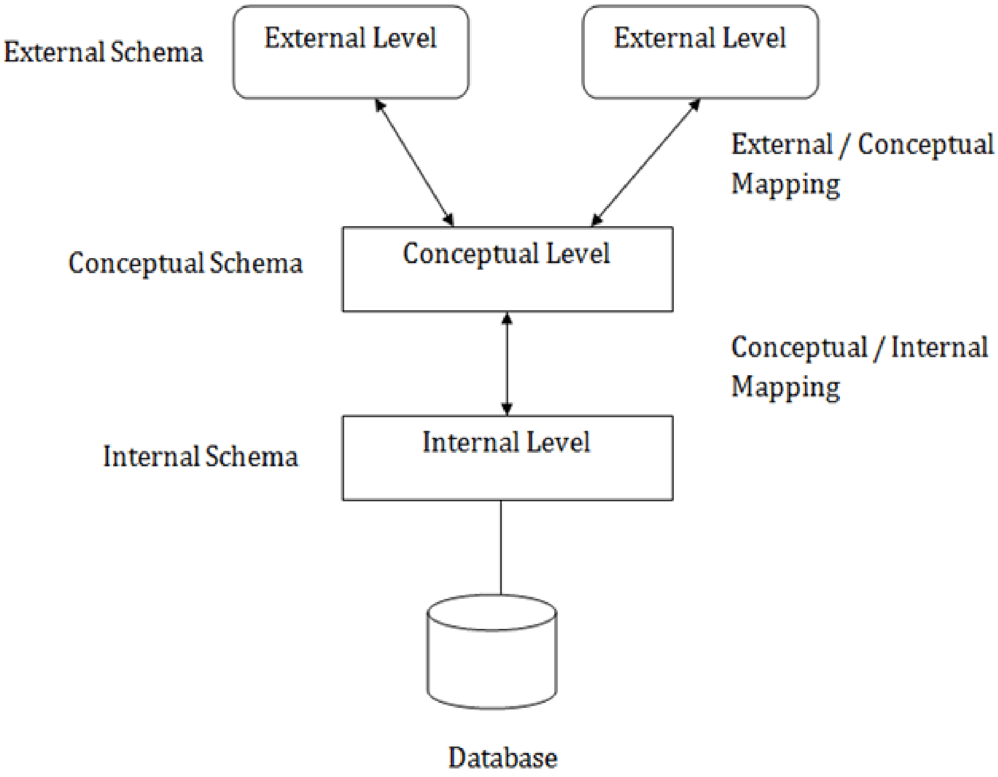
If we do any changes in the storage size of the database system server, then the Conceptual structure of the database will not be affected.

Physical data independence is used to separate conceptual levels from the internal levels.
Physical data independence occurs at the logical interface level.

3-TIER SCHEMA ARCHITECTURE

Physical Level:

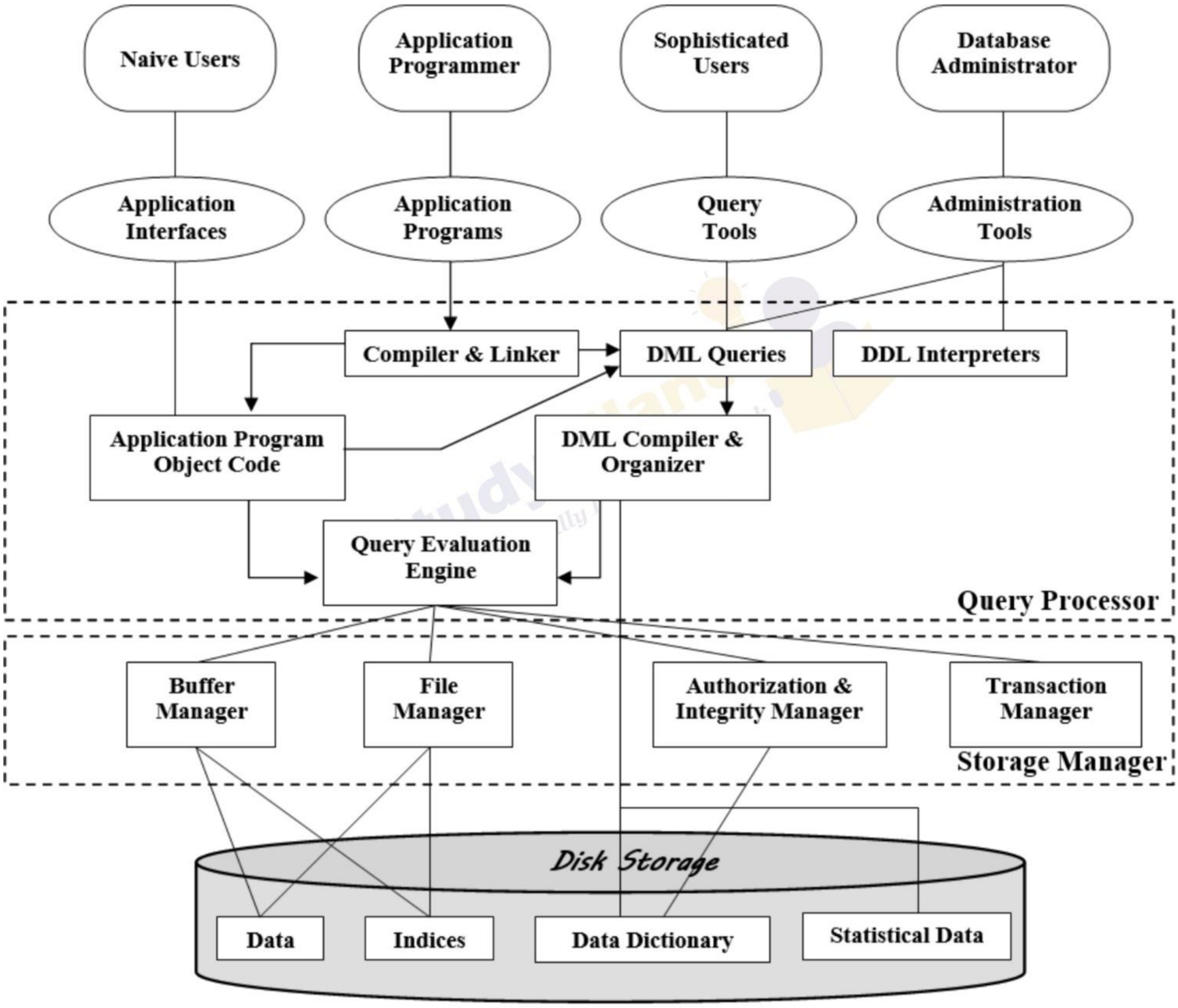
At the physical level, the information about the location of database objects in the data store is kept. Various users of DBMS are unaware of the locations of these objects. In simple terms, physical level of a database describes how the data is being stored in secondary storage devices like disks and tapes and also gives insights on additional storage details.



Conceptual Level:
At conceptual level, data is represented in the form of various database tables. For Example, STUDENT database may contain STUDENT and COURSE tables which will be visible to users but users are unaware of their storage. Also referred as logical schema, it describes what kind of data is to be stored in the database.

External Level:
An external level specifies a view of the data in terms of conceptual level tables. Each external level view is used to cater to the needs of a particular category of users. For Example, FACULTY of a university is interested in looking course details of students, STUDENTS are interested in looking at all details related to academics, accounts, courses and hostel details as well. So, different views can be generated for different users. The main focus of external level is data abstraction.

STRUCTURE OF DATABASE MANAGEMENT SYSTEM



DBMS means Database Management System, which is a tool or software used to create the database or delete or manipulate the database. A software programme created to store, retrieve, query, and manage data is known as a Database Management System (DBMS). Data can be generated, read, updated, and destroyed by authorized entities thanks to user interfaces (UIs).

Users of DBMSs include application programmers, Database Administrators (DBAs), and end users.

Database Administrators are typically the only people who work directly with a DBMS. Today, end users read and write to databases using front-end interfaces made by programmers, while programmers use cloud APIs to connect with DBMSs.

Three Parts that make up the Database System are:.

- Query Processor
- Storage Manager
- Disk Storage

1. Query Processor

The query processing is handled by the query processor, as the name implies. It executes the user's query, to put it simply. In this way, the query processor aids the database system in making data access simple and easy. The query processor's primary duty is to successfully execute the query. The Query Processor transforms (or interprets) the user's application program-provided requests into instructions that a computer can understand.

Components of the Query Processor

DDL Interpreter:

Data Definition Language is what DDL stands for. As implied by the name, the DDL Interpreter interprets DDL statements like those used in schema definitions (such as create, remove, etc.). This interpretation yields a set of tables that include the meta-data (data of data) that is kept in the data dictionary. Metadata may be stored in a data dictionary. In essence, it is a part of the disc storage that will be covered in a later section of this article.

DML Compiler:

Compiler for DML Data Manipulation Language is what DML stands for. In keeping with its name, the DML Compiler converts DML statements like select, update, and delete into low-level instructions or simply machine-readable object code, to enable execution. The optimization of queries is another function of the DML compiler. Since a single question can typically be translated into a number of evaluation plans. As a result, some optimization is needed to select the evaluation plan with the lowest cost out of all the options. This process, known as query optimization, is exclusively carried out by the DML compiler. Simply put, query optimization determines the most effective technique to carry out a query.

Embedded DML Pre-compiler:

Before the query evaluation, the embedded DML commands in the application program (such as SELECT, FROM, etc., in SQL) must be pre-compiled into standard procedural calls (program instructions that the host language can understand). Therefore, the DML statements which are embedded in an application program must be converted into routine calls by the Embedded DML Pre-compiler.

Query Optimizer:

It starts by taking the evaluation plan for the question, runs it, and then returns the result. Simply said, the query evaluation engine evaluates the SQL commands used to access the database's contents before returning the result of the query. In a nutshell, it is in charge of analyzing the queries and running the object code that the DML Compiler produces. Apache Drill, Presto, and other Query Evaluation Engines are a few examples.

2. Storage Manager:

An application called Storage Manager acts as a conduit between the queries made and the data kept in the database. Another name for it is Database Control System. By applying the restrictions and running the DCL instructions, it keeps the database's consistency and integrity. It is in charge of retrieving, storing, updating, and removing data from the database.

Components of Storage Manager

Following are the components of Storage Manager:

Integrity Manager:

Whenever there is any change in the database, the Integrity manager will manage the integrity constraints.

Authorization Manager:

Authorization manager verifies the user that he is valid and authenticated for the specific query or request.

File Manager:

All the files and data structure of the database are managed by this component.

Transaction Manager:

It is responsible for making the database consistent before and after the transactions. Concurrent processes are generally controlled by this component.

Buffer Manager:

The transfer of data between primary and main memory and managing the cache memory is done by the buffer manager.

3. Disk Storage

A DBMS can use various kinds of Data Structures as a part of physical system implementation in the form of disk storage.

Components of Disk Storage

Following are the components of Disk Manager:

Data Dictionary:

It contains the metadata (data of data), which means each object of the database has some information about its structure. So, it creates a repository which contains the details about the structure of the database object.

Data Files:

This component stores the data in the files.

Indices:

These indices are used to access and retrieve the data in a very fast and efficient way.

CLIENT SERVER ARCHITECTURE FOR THE DATABASE

A Database Architecture is a representation of DBMS design. It helps to design, develop, implement, and maintain the database management system. It also helps to understand the components of a database.

Client – The client can be any computer that requests something from the server.

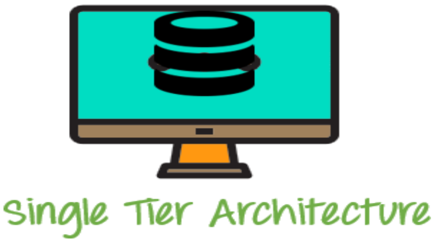
Server – On the other hand, the Server is the computer that is designed to serve the requests to the client

There are mainly three types of DBMS architecture

- **One Tier Architecture (Single Tier Architecture)**
- **Two Tier Architecture**
- **Three Tier Architecture**

One-Tier Architecture:

It is the simplest architecture of Database in which the client, server, and Database all reside on the same machine. A simple one tier architecture example would be anytime you install a Database in your system and access it to practice SQL queries. But such architecture is rarely used in production.



Two-Tier Architecture:

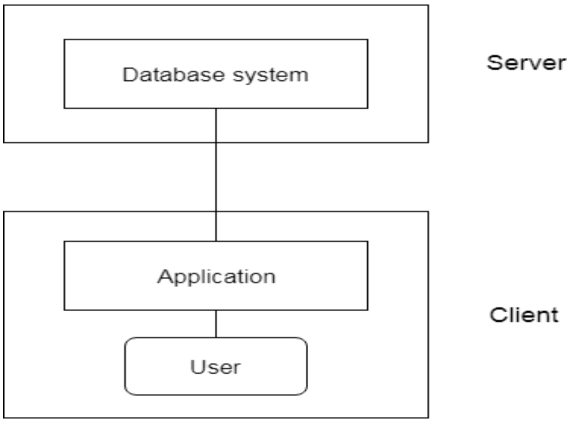
The 2-Tier architecture is same as basic client-server. In the two-tier architecture, applications on the client end can directly communicate with the database at the server side. For this interaction, API's like: ODBC, JDBC are used.

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 serverside..

2-Tier Architecture



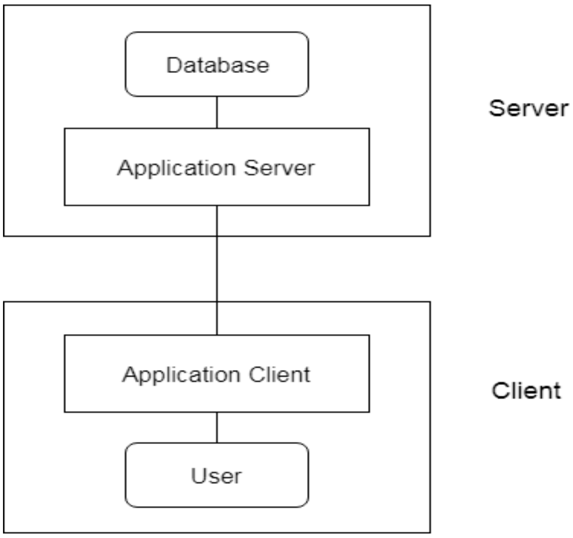
Three-Tier Architecture:

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.



3-Tier Archite**cture**

ENTITY RELATIONSHIP MODEL

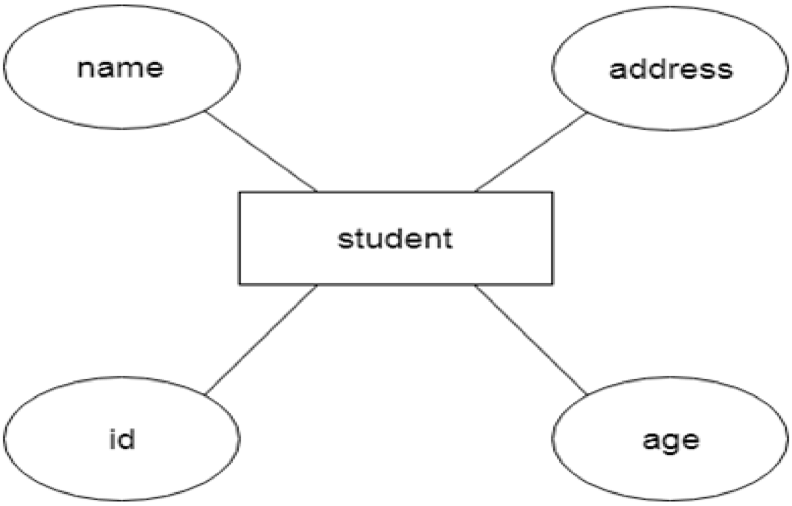
ER model stands for an Entity-Relationship model. It is a high-level data model. The Entity Relational Model is a model for identifying entities to be represented in the database and representation of how those entities are related.

The Entity Relationship Diagram explains the relationship among the entities present in the database.

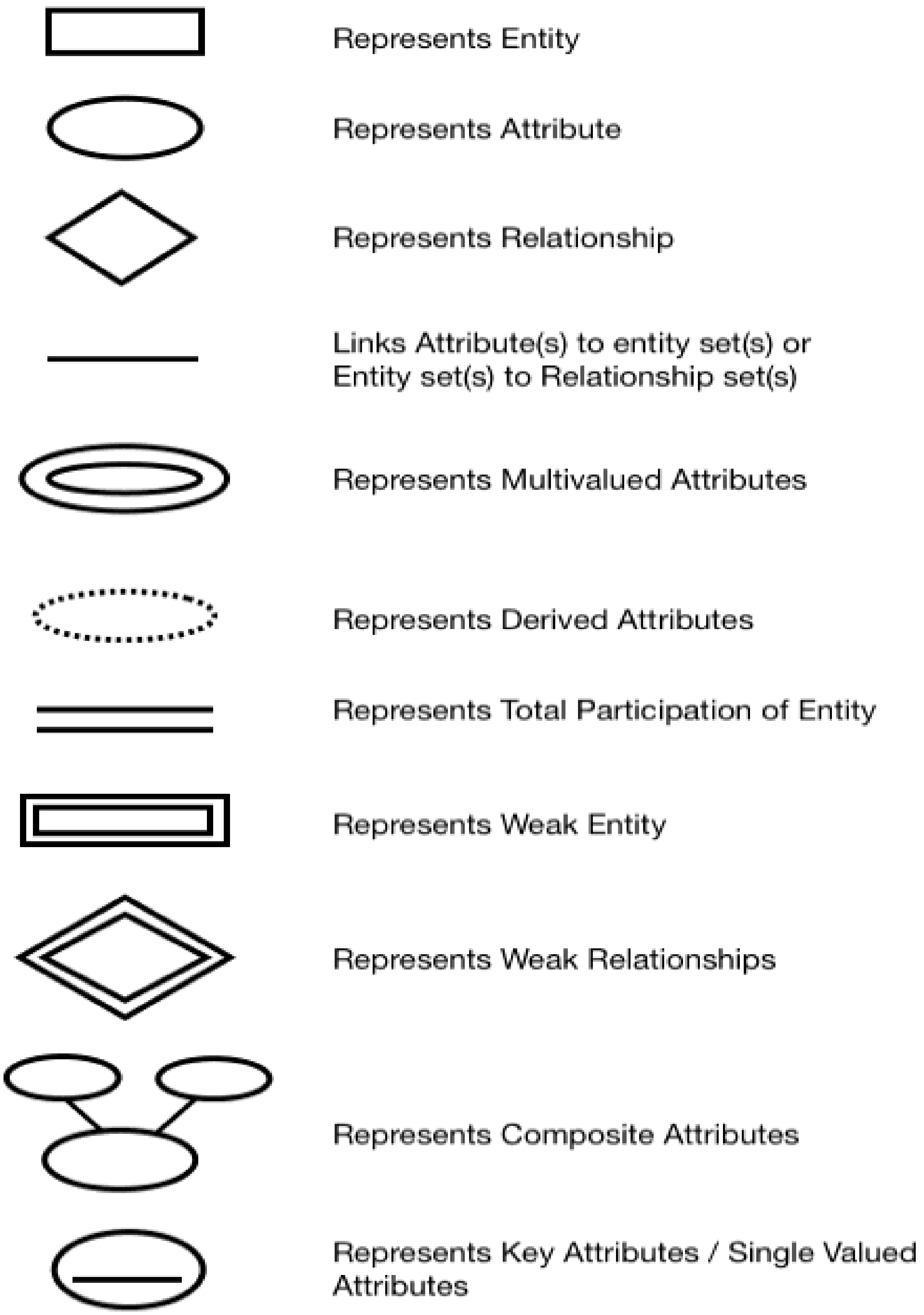
Why Use ER Diagrams In DBMS?

ER diagrams are used to represent the E-R model in a database, which makes them easy to be converted into relations (tables).
It gives a standard solution for visualizing the data logically.

For example, Suppose we design a school database. In this database, the student will be
an entity with attributes like address, name, id, age, etc.

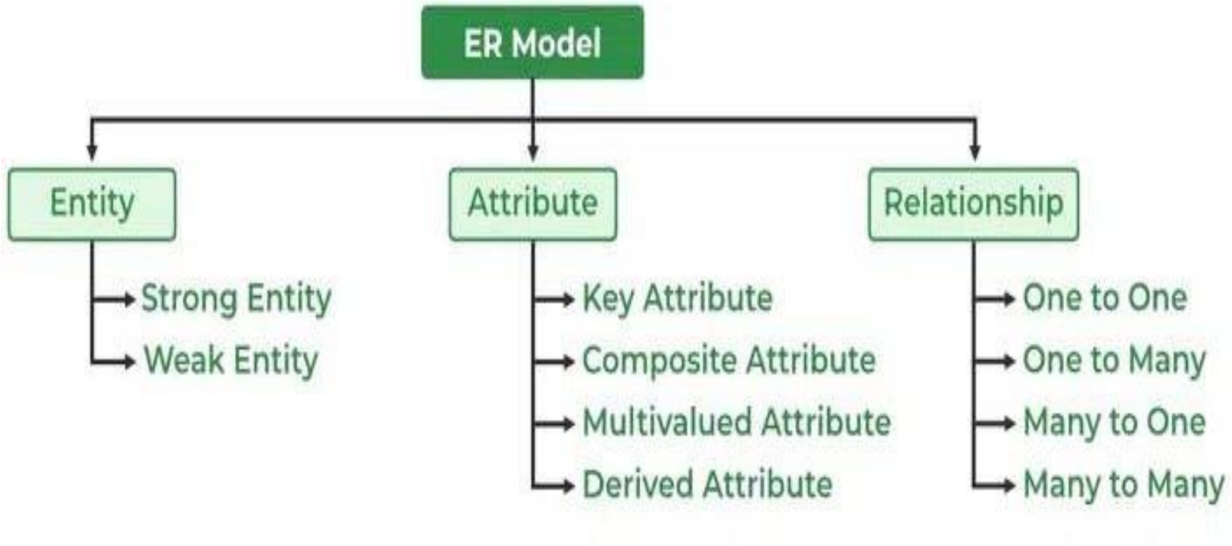


Graphical Notations to develop ER Diagram:


Represents Entity
Represents Attribute
Represents Relationship
Links Attribute(s) to entity set(s) or Entity set(s) to Relationship set(s)
Represents Multivalued Attributes
Represents Derived Attributes
Represents Total Participation of Entity
Represents Weak Entity
Represents Weak Relationships
Represents Composite Attributes
Represents Key Attributes / Single Valued Attributes

COMPONENTS OF ER-MODEL

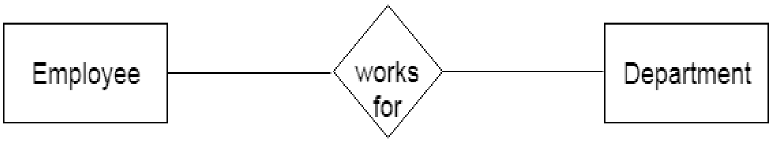
ER Model consists of Entities, Attributes, and Relationships among Entities in a Database System.



Entity:

An Entity may be an object with a physical existence.EX: a particular person ,car ,house ,or employee.

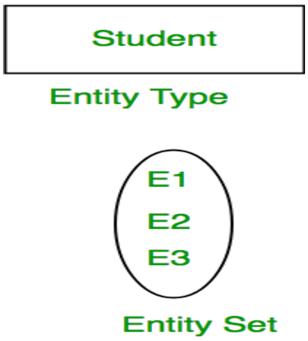
In the ER diagram, an entity can be represented as rectangles.



Entity Set:

An Entity is an object of Entity Type and a set of all entities is called as an entity .

e.g.;E1 is an entity having Entity Type Student and set of all students is called Entity Set.



Entity Types:

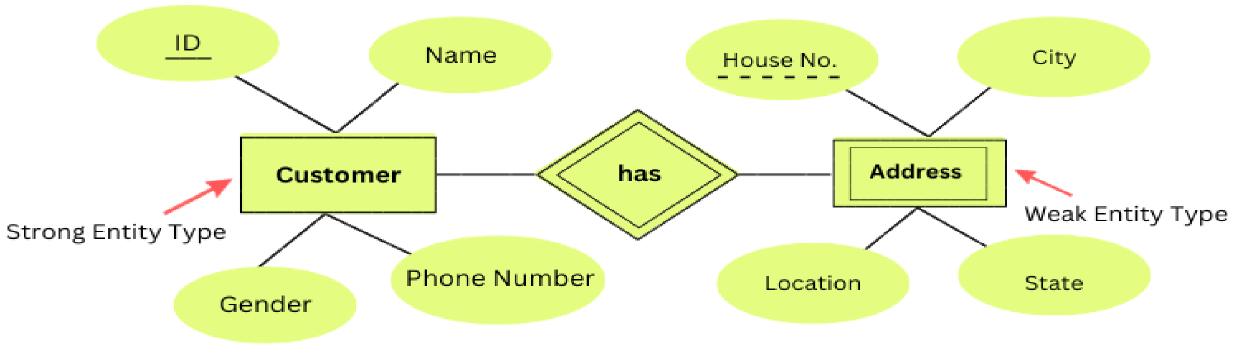
1. Strong Entity Set:

A strong entity is not dependent on any other entity in the schema. A strong entity will always have a primary key. Strong entities are represented by a single rectangle. The relationship of two strong entities is represented by a single diamond. Various strong entities, when combined together, create a strong entity set.

2. Weak Entity Set:

A weak entity is dependent on a strong entity to ensure its existence. Unlike a strong entity, a weak entity does not have any primary key. It instead has a partial discriminator key. A weak entity is represented by a double rectangle. The relation between one strong and one weak entity is represented by a double diamond. This relationship is also known as an identifying relationship.

Example:



Entity Relationship Diagram between Strong Entity Type & Weak Entity Type

In the above ER Diagram the weak entity 'address' represents using double rectangles and strong entity 'customer' represents using single rectangle.

In the above example, the "Customer" is the entity type with attributes such as ID, Name, Gender, and Phone Number. Customer is a strong entity type as it has a unique ID for each customer. "Address" is a weak entity type with attributes such as House No., City, Location, and State.
The relationship between a strong and a weak entity type is known as an identifying relationship.

Using a double diamond, the Entity-Relationship Diagram represents a relationship between the strong and the weak entity type.

Attribute(s):

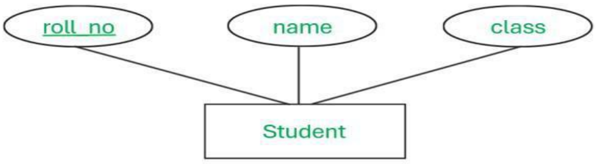
Attributes are the properties that define the entity type.

For example, Roll_ No, Name, DOB, Age, Address, Mobile_ No are the attributes that define entity type (E1, E2, E3) Student. In ER diagram, the attribute is represented by an oval.

TYPES OF ATTRIBUTES

Simple Attribute:

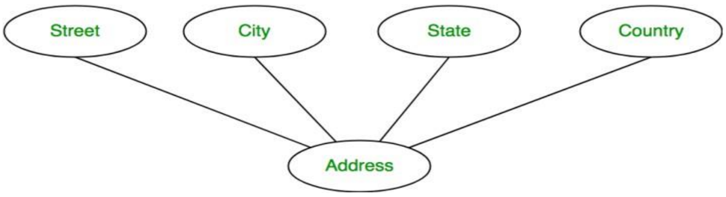
An attribute that cannot be further subdivided into components is a simple attribute.
Example: The roll number of a student, the ID number of an employee, gender, and many more.



Composite Attribute:

An attribute composed of many other attribute is called as composite attribute.

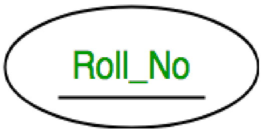
For example, Address attribute of student Entity type consists of Street, City, State, and Country. In ER diagram, composite attribute is represented by an oval comprising of ovals.



Key Attribute:

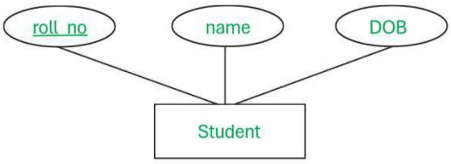
The attribute which uniquely identifies each entity in the entity set is called key attribute.

For example ,Roll No will be unique for each student .In ER diagram, key attribute is represented by an oval with underlying lines.



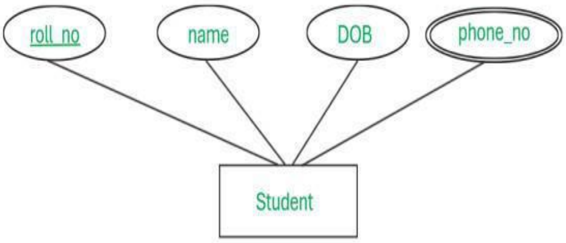
Single-Valued Attribute:

The attribute which takes up only a single value for each entity instance is a single-valued attribute.
Example**:** The age of a student, Aadhar card number.



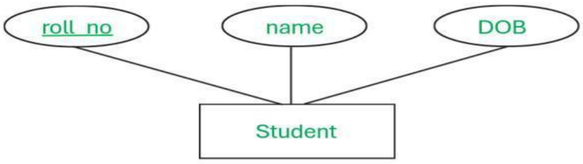
Multivalued Attribute:

An attribute consisting more than one value for a given entity.
For example, Phone_ No (can be more than one for a given student). In ER diagram, a multivalued attribute is represented by a double oval.



Stored Attribute:

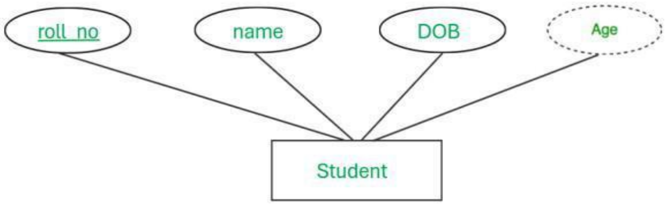
The stored attribute are those attribute which doesn’t require any type of further update since they are stored in the database.
Example**:** DOB(Date of birth) is the stored attribute.



Derived Attribute:

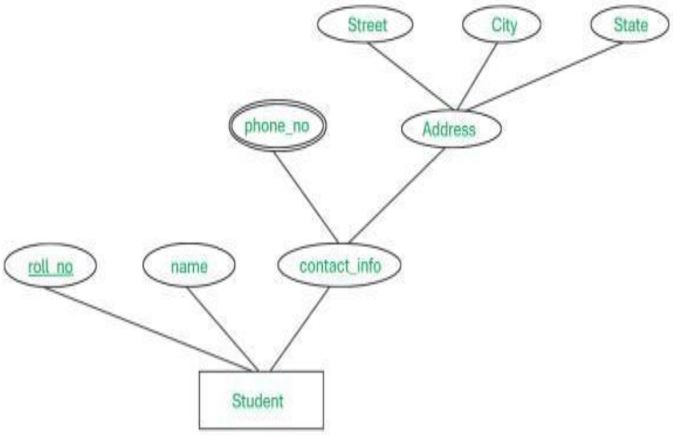
An attribute that can be derived from other attributes of the entity type is known as a derived attribute.

E.g.: Age (can be derived from DOB). In ER diagram, the derived attribute is represented by a dashed oval.

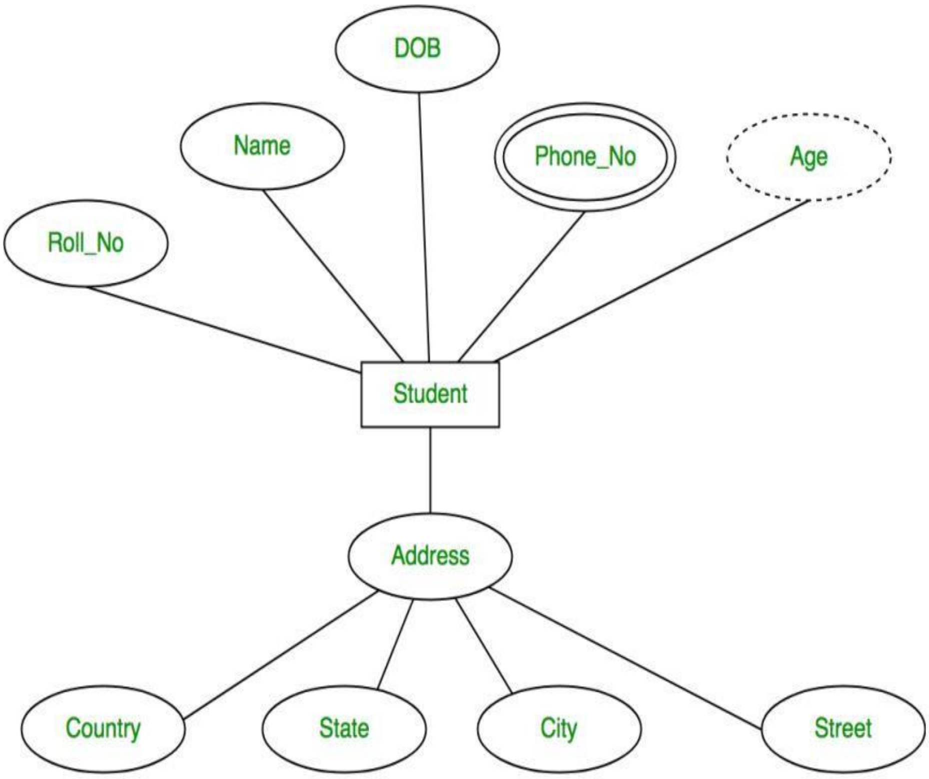


Complex Attribute :

Those attributes, which can be formed by the nesting of composite and multi-valued attributes, are called “Complex Attributes“.



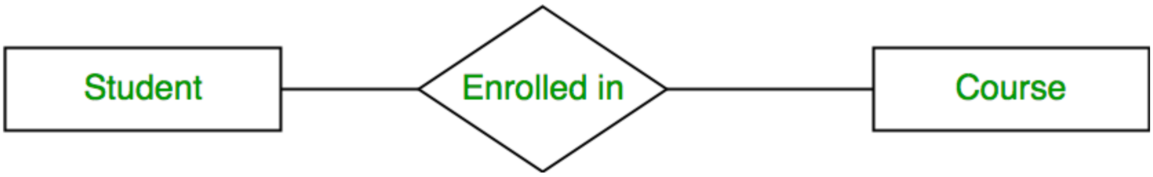
The complete entity type **Student** with its attributes can be represented as:



Relationship Type and Relationship Set:

Relationship Type:

A relationship type represents the **association between entity types**. For example, ‘Enrolled in’ is a relationship type that exists between entity type Student and Course .In ER diagram, the relationship type is represented by a diamond and connecting the entities with lines.



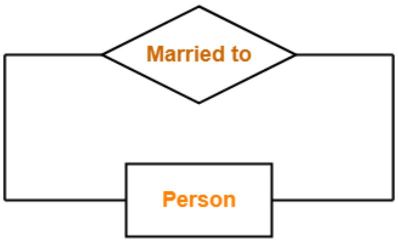
Relationship Set:

A set of relationships of the same type is known as a relationship set.
The following relationship set depicts S1 as enrolled in C2,S2 is enrolled in C1,and S3 is enrolled in C3.

Degree of a relationship set:

1.Unary Relationship:

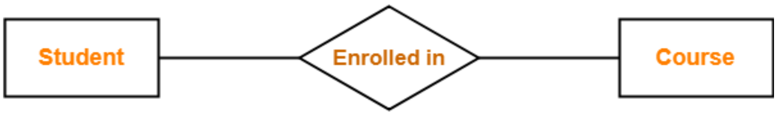
When there is **only ONE entity set participating in a relation**, the relationship is called a unary relationship. For example, one person is married to only one person.



Unary Relationship Set

2.Binary Relationship:

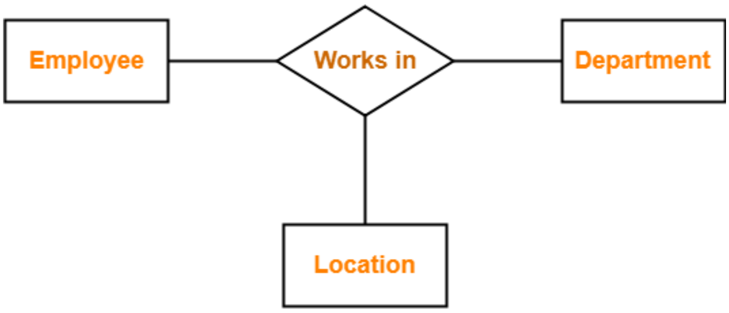
When there are **TWO entities set participating in a relationship** ,the relationship is called a binary relationship. For example, a Student is enrolled in a Course.



Binary Relationship Set

3.n-ary Relationship:

When there are n entities set participating in a relation ,the relationship is called an n-ary relationship.



Ternary Relationship Set

CONSTRAINTS:

Constraints are used for modeling limitations on the relations between entities.

Various types of constraints on the Entity Relationship(ER) model.

1. Mapping cardinality or cardinality ratio.
2. Key Constraints
3. Participation constraints

1. Mapping Cardinality:

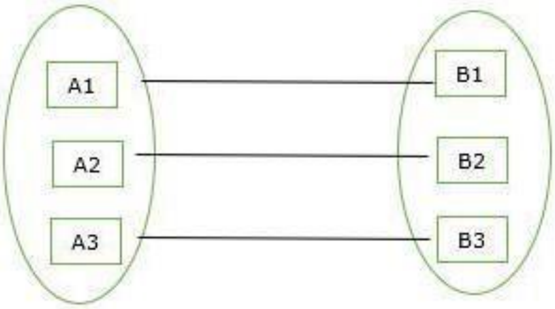
It is expressed as the number of entities to which another entity can be associated via a relationship set.

For binary relationship set there are entity set A and B then the mapping cardinality can be one of the following−

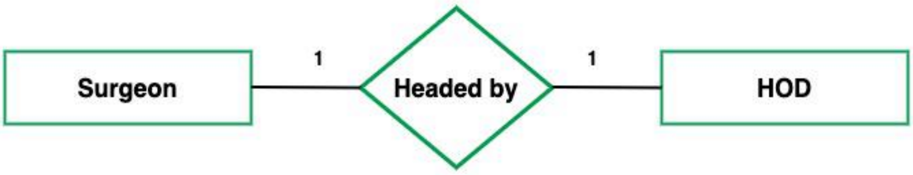
1. One-to-one
2. One-to-many
3. Many-to-one
4. Many-to-many

1.One-to-one relationship: An entity of A is associated with atmost one entity in B and entity in B is associated with atmost one entity of A.

.

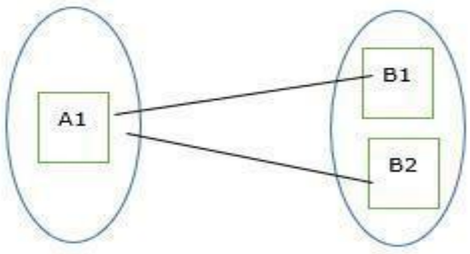


In a particular hospital, the surgeon department has one head of department.

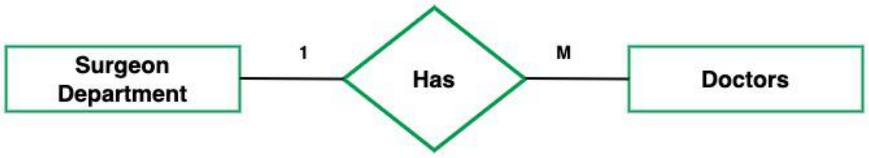


2.One-to-many relationship

An entity set A is associated with any number of entities in B with a possibility of zero and entity in B is associated with at most one entity

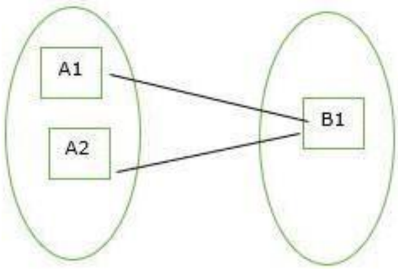


In a particular hospital, the surgeon department has multiple doctors.

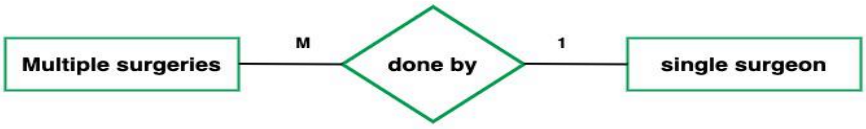


3.Many-to-one relationship

An entity set A is associated with at most one entity in B and an entity set in B can be associated with any number of entities in A with a possibility of zero.



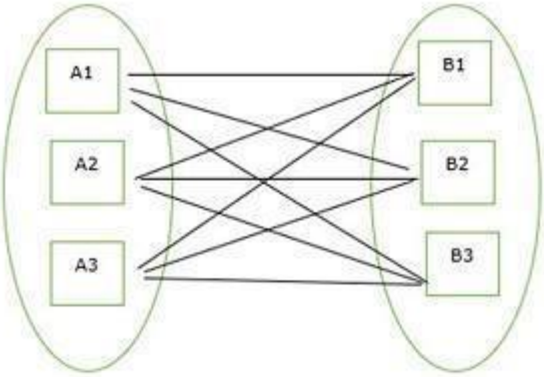
In a particular hospital, multiple surgeries are done by a single surgeon.



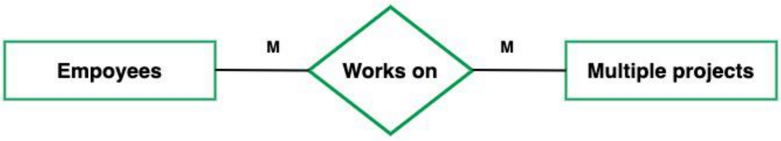
4. **Many-to-many relationship**

Many entities of A are associated with many entities of B.

An entity in A is associated with many entities of B and an entity in B is associated with many entities of A.



In a particular company, multiple people work on multiple projects.

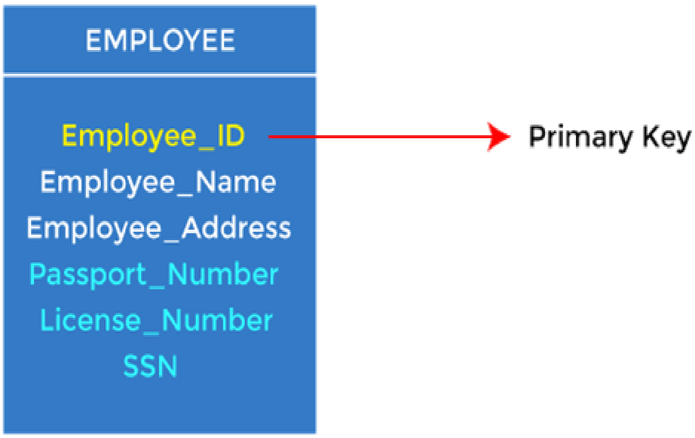


Types of keys: (Key Constraints):

1.Primary key:

It is a key which uniquely identifies a Tuple. It cannot be NULL.

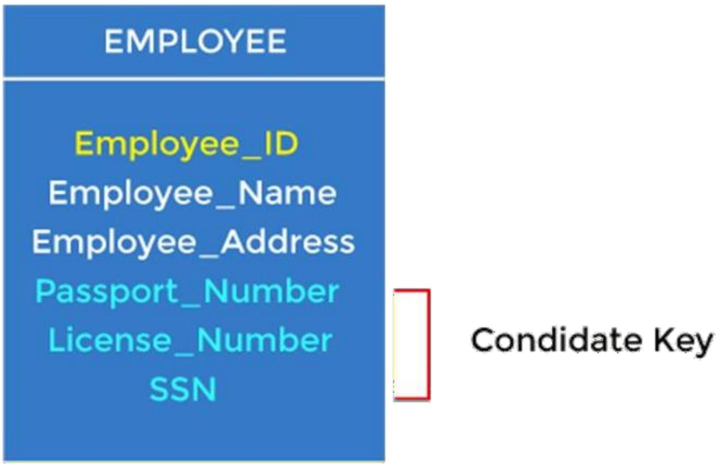
In the EMPLOYEE table, ID can be the primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License_Number and Passport_Number as primary keys since they are also unique, but they may contain null values also. That is the reason for taking ID as Primary key.



2.Candidate key:

A candidate key is an attribute or set of attributes that can uniquely identify a tuple. It can contain NULL values.

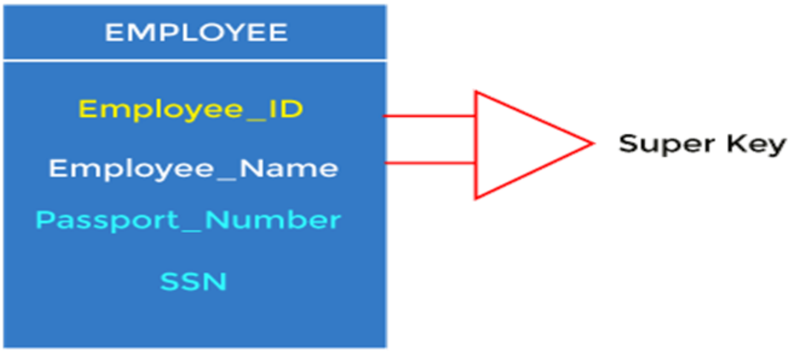
Except for the primary key, the remaining attributes are considered a candidate key. The candidate keys are as strong as the primary key.



For example: In the EMPLOYEE table, id is best suited for the primary key. The rest of the attributes, like SSN, Passport_Number, License_Number, etc., are considered a candidate key.

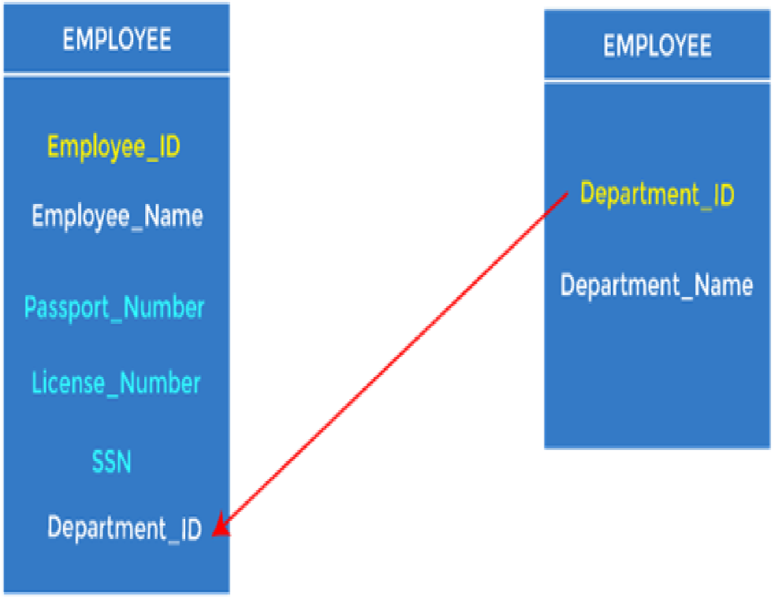
3.Super Key:

Super key is an attribute set that can uniquely identify a tuple. A super key is a superset of a candidate key.



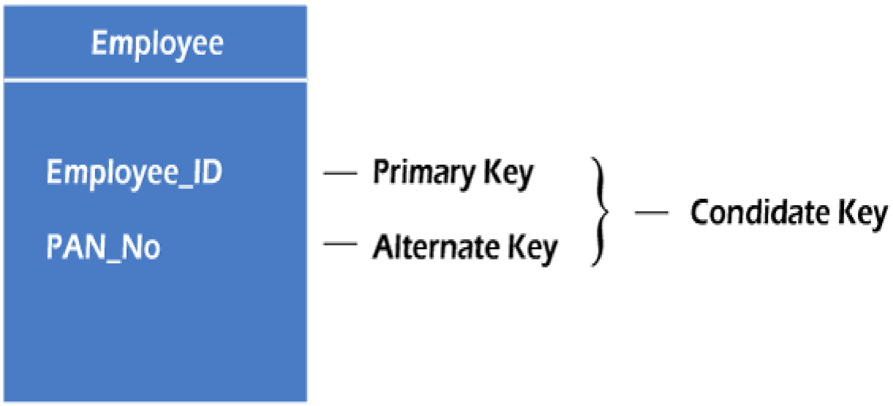
For example: In the above EMPLOYEE table, for(EMPLOEE_ID, EMPLOYEE_NAME), the name of two employees can be the same, but their EMPLYEE_ID can't be the same .Hence, this combination can also be a key
The super key would be EMPLOYEE-ID(EMPLOYEE_ID,EMPLOYEE-NAME), etc.

4.Foreign key:
Foreign keys are the column of the table used to point to the primary key of another table.
Every employee works in a specific department in a company, and employee and department are two different entities. So we can't store the department's information in the employee table. That's why we link these two tables through the primary key of one table.
We add the primary key of the DEPARTMENT table, Department_Id, as a new attribute in the EMPLOYEE table.
In the EMPLOYEE table ,Department_Id is the foreign key ,and both the tables are related.



5. Alternate key:

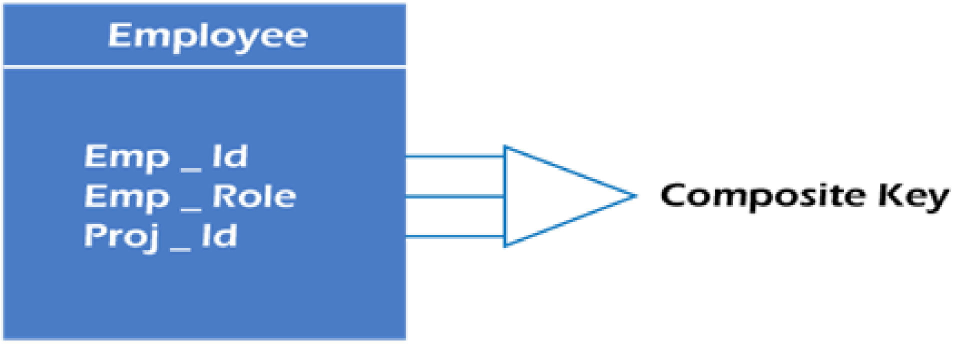
There may be one or more attributes or a combination of attributes that uniquely identify each tuple in a relation. These attributes or combinations of the attributes are called the candidate keys. One key is chosen as the primary key from these candidate keys, and the remaining candidate key, if it exists, is termed the alternate key.



For example, employee relation has two attributes, Employee_Id and PAN_No, that act as candidate keys. In this relation, Employee_Id is chosen as the primary key, so the other candidate key, PAN_No, acts as the Alternate key.

6.Composite key:

Whenever a primary key consists of more than one attribute, it is known as a composite key. This key is also known as Concatenated Key.



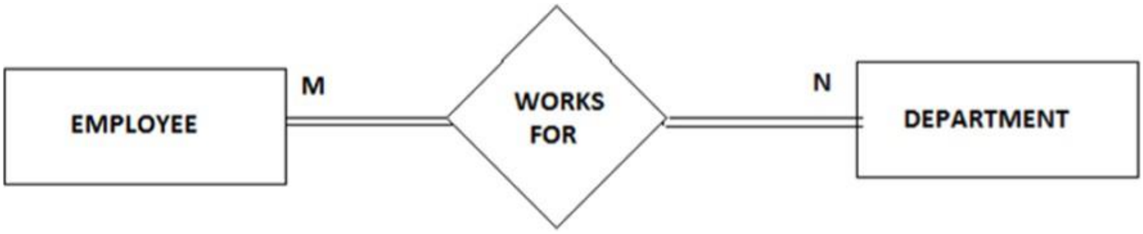
For example, in employee relations, we assume that an employee may be assigned multiple roles, and an employee may work on multiple projects simultaneously. So the primary key will be composed of all three attributes, namely Emp_ID, Emp_role, and Proj_ID in combination. So these attributes act as a composite key since the primary key comprises more than one attribute.

Entity Relationship Participation in Database (Participation constraints)

In a Relationship, Participation constraint specifies the existence of an entity when it is related to another entity in a relationship type. It is also called minimum cardinality constraint.
There are two types of Participation constraints −

Total Participation

Each entity in the entity set is involved in atleast one relationship in a relationship set i.e. the number of relationship in every entity is involved is greater than 0.



Consider two entities Employee and Department related via Works_For relationship. Now, every Employee works in at least one department therefore an Employee entity exist if it has at least one Work_For relationship with Department entity. Thus the participation of Employee in Works_For is total relationship.

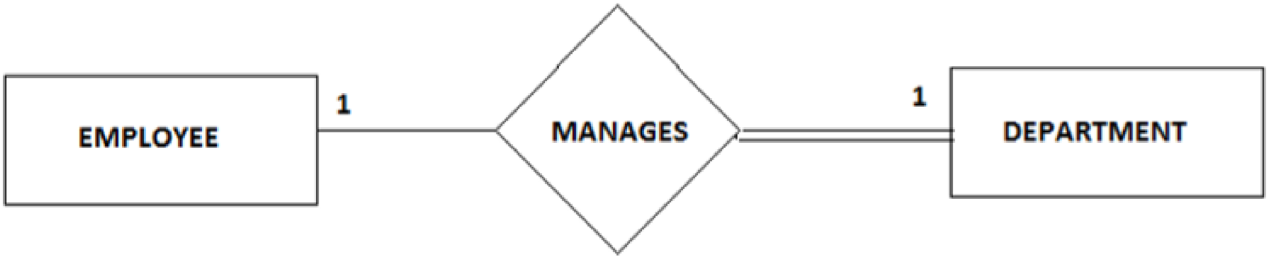
Total Participation is represented by double line in ER diagram.

Partial Participation

Each entity in entity set may or may not occur in at least one relationship in a Relationship model. In addition to all that, it also contains features of Sub classes, Super classes and Inheritance.

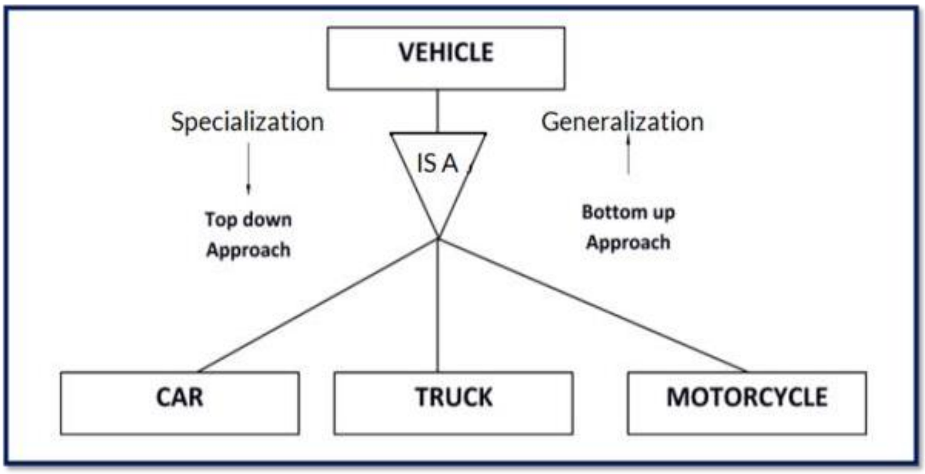
For example: Consider two entities Employee and Department and they are related to each other via Manages relationship. An Employee must manage a Department, he or she could be the head of the department. But not every Employee in the company manages the department. So, participation of employee in the Manages relationship type is partial i.e. only a particular set of Employees will manage the Department but not all.

Partial Participation is represented by single line in ER diagram.



Subclass:

A subclass is a class derived from the super class. It inherits the properties of the super class and also contains attributes of its own.



Car, Truck and Motorcycle are all sub classes of the super class Vehicle. They all inherit common attributes from vehicle such as speed, colour etc. while they have different attributes also i.e Number of wheels in Car is 4 while in Motorcycle is 2.

Super class:

A super class is the class from which many subclasses can be created. The subclasses inherit the characteristics of a super class. The super class is also known as the parent class or base class.

Inheritance

Inheritance is basically the process of basing a class on another class i.e to build a class on a existing class. The new class contains all the features and functionalities of the old class in addition to its own.

Additional Features of ER Diagram

Generalization and Specialization

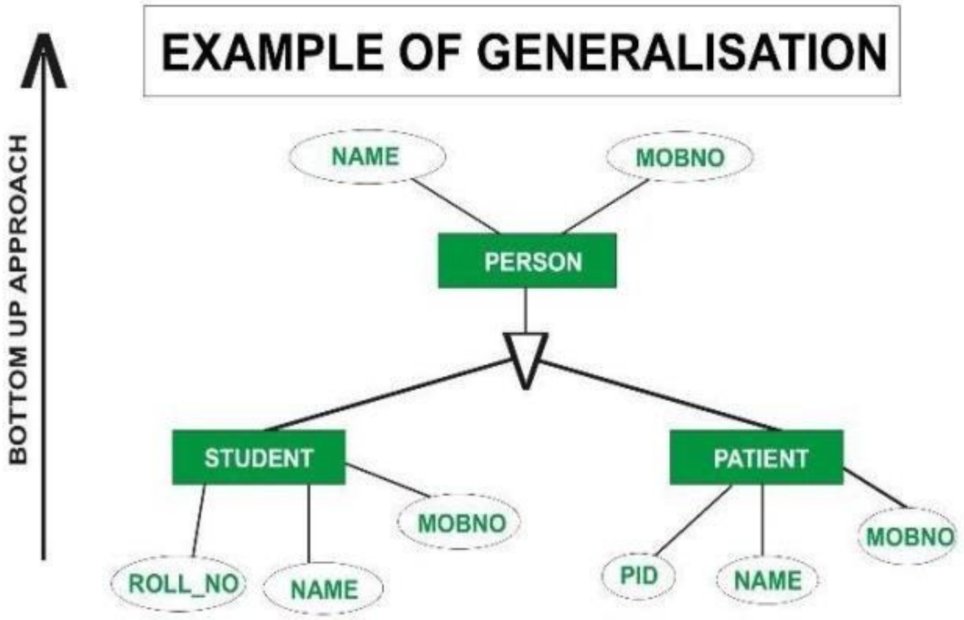
Generalization and specialization are the Enhanced Entity Relationship diagram (EER- diagram)

1.Generalization:

It works on the principle of bottom up approach .In Generalization lower level functions are combined to form higher level function which is called as entities. This process is repeated further to make advanced level entities.
In the Generalization process properties are drawn from particular entities and thus we can create generalized entity. We can summarize the Generalization process as it combines subclasses to form super class.

Example of Generalization–

Consider two entities Student and Patient. These two entities will have some characteristics of their own. For example Student entity will have Roll_No, Name and Mob_No while patient will have PId, Name and Mob_No characteristics. Now in this example Name and Mob_No of both Student and Patient can be combined as a Person to form one higher level entity and this process is called as Generalization Process.

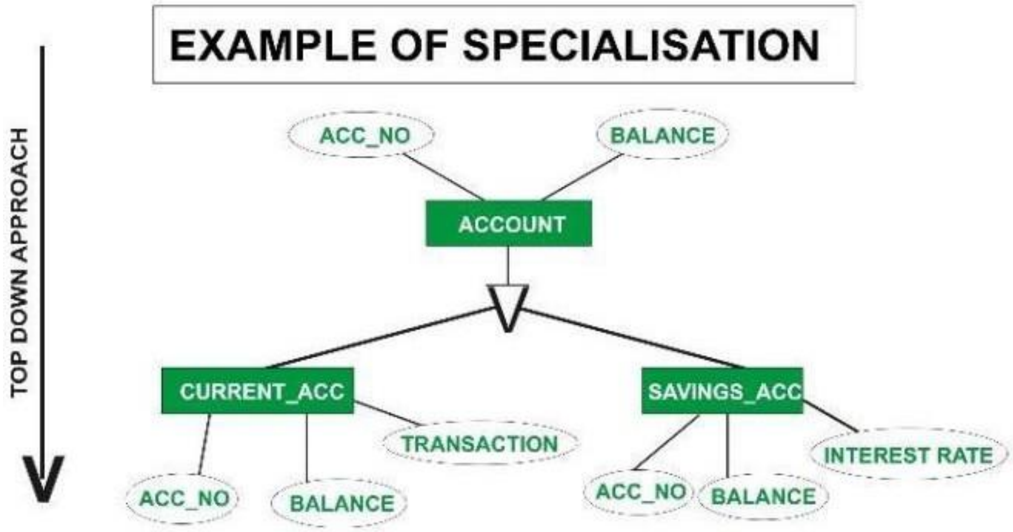


2. Specialization:

We can say that Specialization is opposite of Generalization. In Specialization things are broken down into smaller things to simplify it further. We can also say that in Specialization a particular entity gets divided into sub entities and it's done on the basis of it's characteristics. Also in Specialization Inheritance takes place.

Example of Specialization:

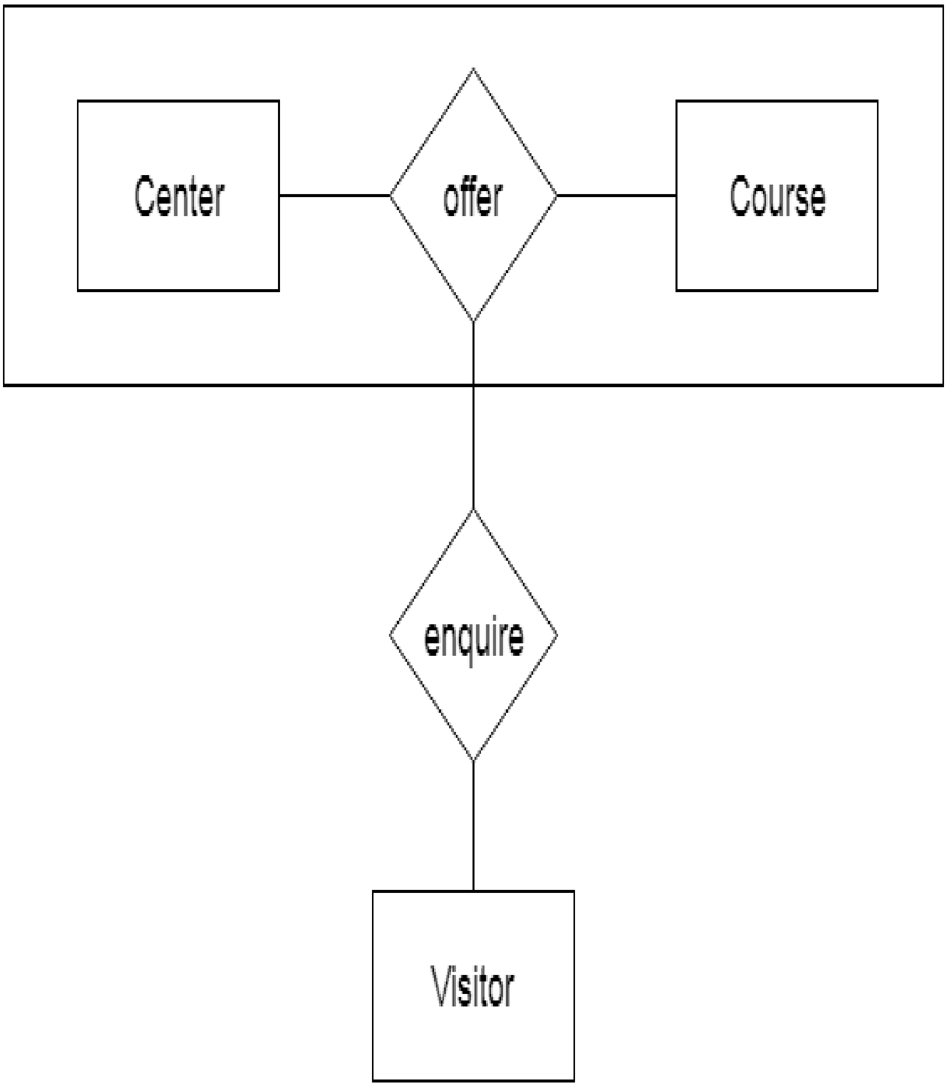
Consider an entity Account. This will have some attributes consider them Acc_No and Balance. Account entity may have some other attributes like Current_Acc and Savings_Acc. Now Current_Acc may have Acc_No, Balance and Transactions while Savings_Acc may have Acc_No, Balance and Interest_Rate hence forth we can say that specialized entities inherits characteristics of higher level entity.



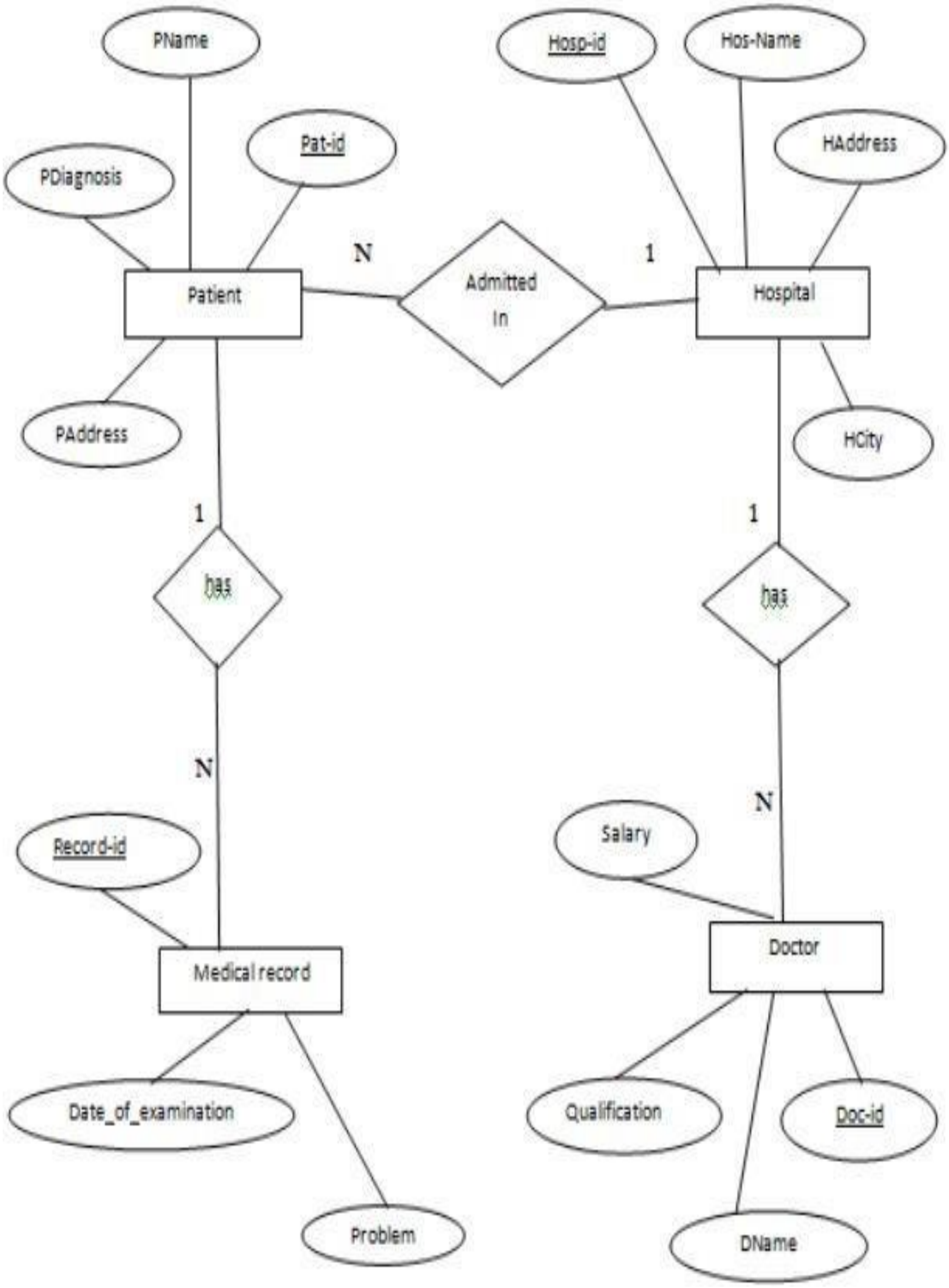
Aggregation:

In aggregation, the relation between two entities is treated as a single entity. In aggregation ,relationship with its corresponding entities is aggregated into a higher level entity.

For example: Center entity offers the Course entity act as a single entity in the relationship which is in a relationship with another entity visitor. In the real world, if a visitor visits a coaching center then he will never enquiry about the Course only or just about the Center instead he will ask the enquiry about both.



Hospital Management System:


PName
Hosp-id
Hos-Name
Pat-id
HAddress
PDiagnosis
N
1
Admitted In
Patient
Hospital
PAddress
HCity
1
1
has
has
N
N
Record-id
Salary
Doctor
Medical record
Date_of_examination
Qualification
Doc-id
Problem
DName