



SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)

COIMBATORE- 641 035

UNIT I



SYLLABUS

23CST201

DATABASE MANAGEMENT SYSTEMS

UNIT I INTRODUCTION

9

Purpose of Database System -- Views of data – Data models, Database Management system - Three-schema architecture of DBMS, Components of DBMS. Entity –Relationship Model - Conceptual data modelling - motivation, entities, entity types, attributes, relationships, relationship types, E/R diagram notations, Examples

INTRODUCTION

UNDERSTANDING DATABASE FUNDAMENTALS

DATA

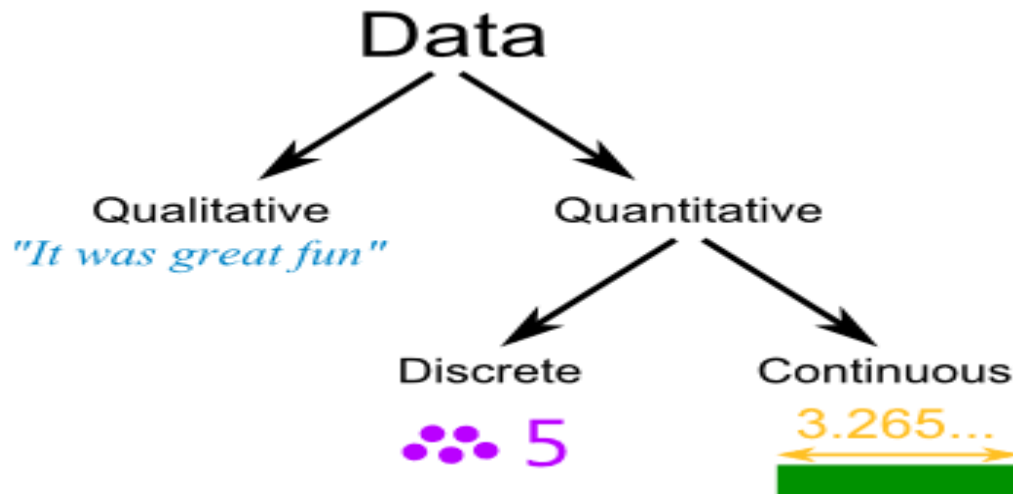
Data is a collection of facts, such as values or measurements.

It can be numbers, words, measurements, observations or even just descriptions of things.

Qualitative vs Quantitative

Data can be qualitative or quantitative.

- **Qualitative data** is descriptive information (it *describes* something)
- **Quantitative data**, is numerical information (numbers).



And **Quantitative data** can also be Discrete or Continuous:

- **Discrete data** can only take certain values (like whole numbers)
- **Continuous data** can take any value (within a range)

Put simply: **Discrete data** is counted, **Continuous data** is measured

INFORMATION

Information is valuable because it can affect behavior, a decision, or an outcome. For example, if a manager is told his/her company's net profit decreased in the past month, he/she may use this information as a reason to cut financial spending for the next month. A piece of information is considered valueless if, after receiving it, things remain unchanged. For a technical definition of information see information theory.

Information is defined as the knowledge of something; particularly, an event, situation, or knowledge derived based on research or experience.



Data is any information related to an organization that should be stored for any purpose according to the requirements of an organization.

DBMS

A database management system (DBMS) is the software that allows a computer to perform database functions of storing, retrieving, adding, deleting and modifying data. Relational database management systems (RDBMS) implement the relational model of tables and relationships.

Examples:

Microsoft Access, MySQL, Microsoft SQL Server, Oracle and FileMaker Pro are all examples of database management systems.

The following are examples of database applications:

- computerized library systems
- automated teller machines
- flight reservation systems
- computerized parts inventory systems

RDBMS

Short for *relational database management system* and pronounced as separate letters, a type of database management system (DBMS) that stores data in the form of related tables. Relational databases are powerful because they require few assumptions about how data is related or how it will be extracted from the database. As a result, the same database can be viewed in many different ways.



An important feature of relational systems is that a single database can be spread across several tables. This differs from flat-file databases, in which each database is self-contained in a single table.

SQL

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database. SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, Oracle, Sybase, Informix, postgres and SQL Server use SQL as standard database language.

Also, they are using different dialects, such as:

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,

History:

- **1970** -- Dr. E. F. "Ted" of IBM is known as the father of relational databases. He described a relational model for databases.
- **1974** -- Structured Query Language appeared.
- **1978** -- IBM worked to develop Codd's ideas and released a product named System/R.
- **1986** -- **IBM** developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software and its later becoming Oracle.



SQL is pronounced as “S-Q-L” or “see-quill”.

SQL uses -- character sequence as a single line comment identifier.

SQL commands are not case sensitive and the following SQL queries are equivalent:

SELECT * FROM Users

select * from Users

Characteristics of Database Approach

1. Represent Some Aspects of real world applications

A database represents some features of real world applications. Any change in the real world is reflected in the database. If we have some changes in our real applications like railway reservation system then it will be reflected in database too.

For example, let us take railway reservation system; we have in our mind some certain applications of maintaining records of attendance, waiting list, train arrival and departure time, certain day etc. related to each train.

2. Manages Information

A database always takes care of its information because information is always helpful for whatever work we do. It manages all the information that is required to us. By managing information using a database, we become more deliberated user of our data.

Also See: What is Database?

3. Easy Operation implementation

All the operations like insert, delete, update, search etc. are carried out in a flexible and easy way. Database makes it very simple to implement these operations. A user with



little knowledge can perform these operations. This characteristic of database makes it more powerful.

4. Multiple views of database

Basically, a view is a **subset of the database**. A view is defined and devoted for a particular user of the system. Different users of the system may have different views of the same system.

Every view contains only the data of interest to a user or a group of users. It is the responsibility of users to be aware of how and where the data of their interest is stored.

5. Data for specific purpose

A database is designed for data of specific purpose. **For example**, a database of student management system is designed to maintain the record of student's marks, fees and attendance etc. This data has a specific purpose of maintaining student record.

Also See: Advantages Of Database Management System

6. It has Users of Specific interest

A database always has some indented group of users and applications in which these user groups are interested.

For example, in a library system, there are three users, official administration of the college, the librarian, and the students.

7. Self Describing nature

A database is of self describing nature; it always describes and narrates itself. It contains the description of the whole data structure, the constraints and the variables.



It makes it different from traditional file management system in which definition was not the part of application program. These definitions are used by the users and DBMS software when needed.

8. Logical relationship between records and data

A database gives a logical relationship between its records and data. So a user can access various records depending upon the logical conditions by a single query from the database.

Data Model

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

Types of database models

There are many kinds of data models. Some of the most common ones include:

- Hierarchical database model
- Relational model
- Network model
- Object-oriented database model
- Entity-relationship model
- Document model
- Entity-attribute-value model
- Star schema
- The object-relational model, which combines the two that make up its name

Flat Data Model



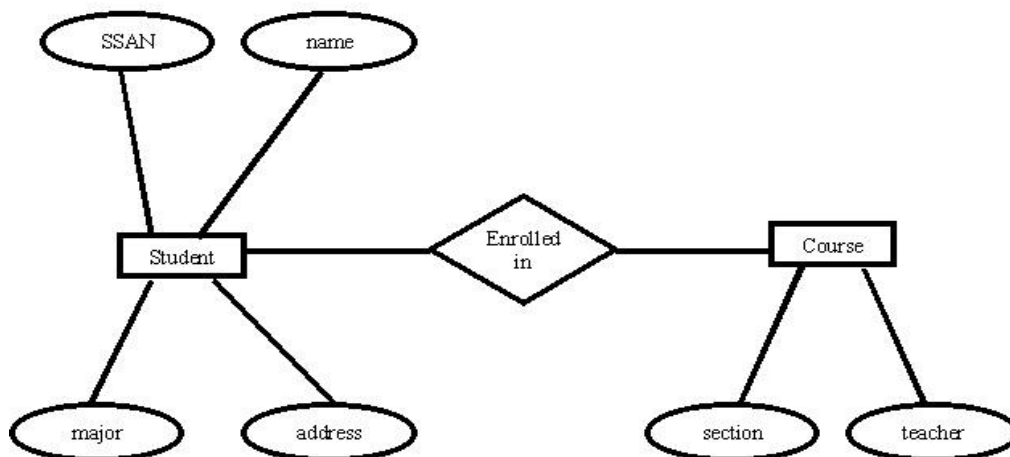
Flat data model is the first and foremost introduced model and in this all the data used is kept in the same plane. Since it was used earlier this model was not so scientific.

Roll No	Name	Course
5482	Mark	Web Designing
5486	Steve	Java
5496	Smith	Oracle

Entity Relationship Data Model

Entity relationship model is based on the notion of the real world entities and their relationships. While formulating the real world scenario in to the database model an entity set is created and this model is dependent on two vital things and they are :

- Entity and their attributes
- Relationships among entities

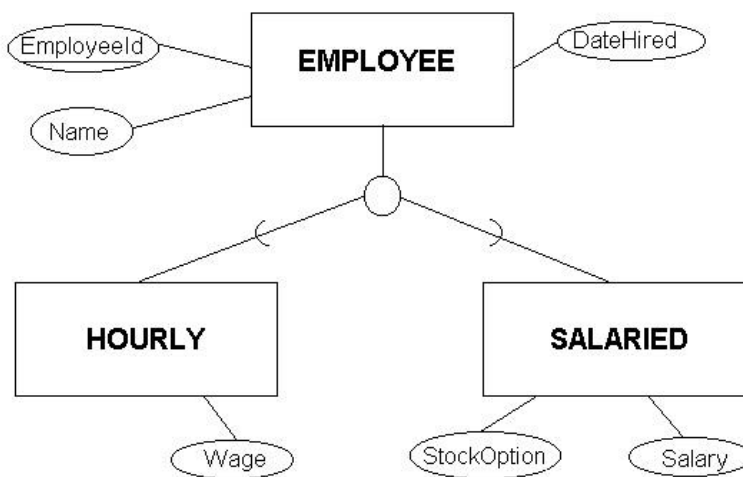




An entity has a real world property called attribute and attribute define by a set of values called domain. For example, in a university a student is an entity, university is the database, name and age and sex are the attributes. The relationships among entities define the logical association between entities.

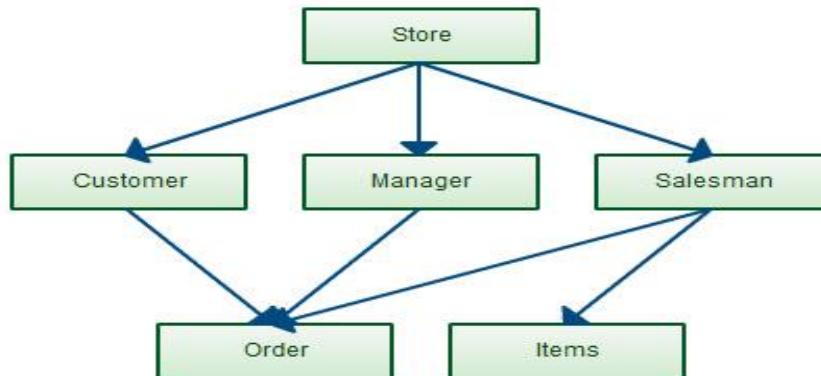
Relational Data Model

Relational model is the most popular model and the most extensively used model. In this model the data can be stored in the tables and this storing is called as relation, the relations can be normalized and the normalized relation values are called atomic values. Each row in a relation contains unique value and it is called as tuple, each column contains value from same domain and it is called as attribute.



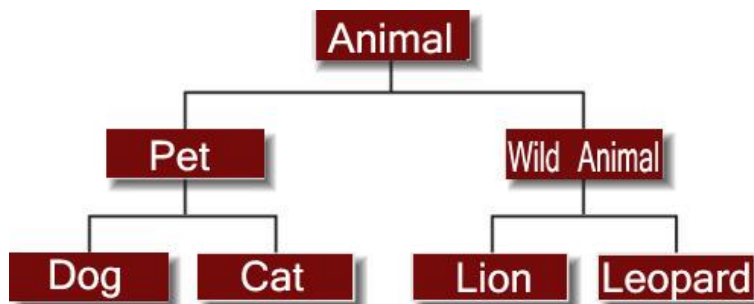
Network Data Model

Network model has the entities which are organized in a graphical representation and some entities in the graph can be accessed through several paths.



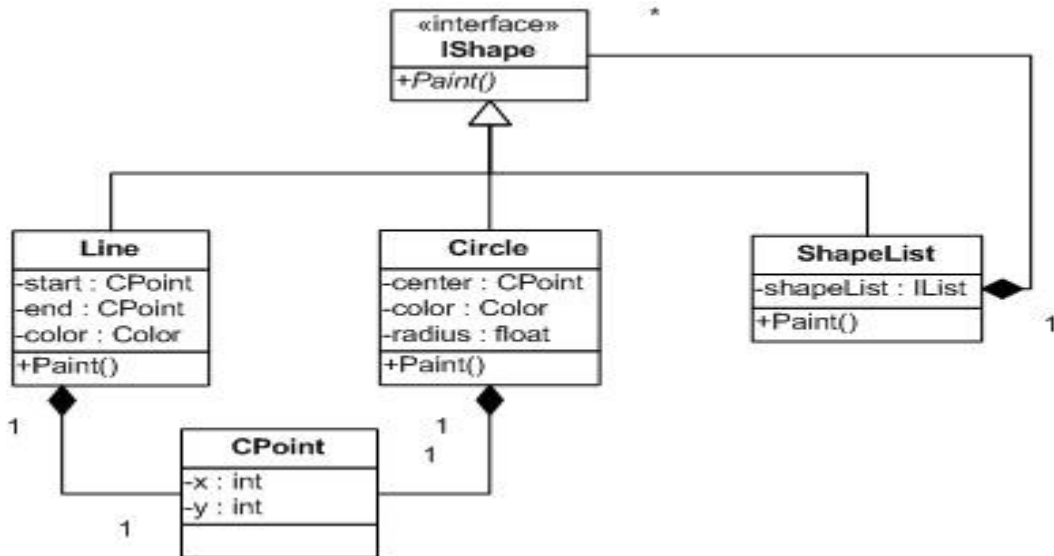
Hierarchical Data Model

Hierarchical model has one parent entity with several children entity but at the top we should have only one entity called root. For example, department is the parent entity called root and it has several children entities like students, professors and many more.



Object oriented Data Model

Object oriented data model is one of the developed data model and this can hold the audio, video and graphic files. These consist of data piece and the methods which are the DBMS instructions.



Record base Data Model

Record base model is used to specify the overall structure of the database and in this there are many record types. Each record type has fixed no. of fields having the fixed length.

Object relation Data Model

Object relation model is a very powerful model but coming to it's design it is quiet complex. This complexity is not problem because it gives efficient results and widespread with huge applications. It has a feature which allows working with other models like working with the very known relation model.

Semi structured Data Model

Semi structured data model is a self describing data model, in this the information that is normally associated with a scheme is contained within the data and this property is called as the self describing property.



Associative Data Model

Associative model has a division property, this divides the real world things about which data is to be recorded in two sorts i.e. between entities and associations. Thus, this model does the division for dividing the real world data to the entities and associations.

Database Architecture

Three Level Architecture of DBMS

Following are the three levels of database architecture,

1. Physical Level
2. Conceptual Level
3. External Level

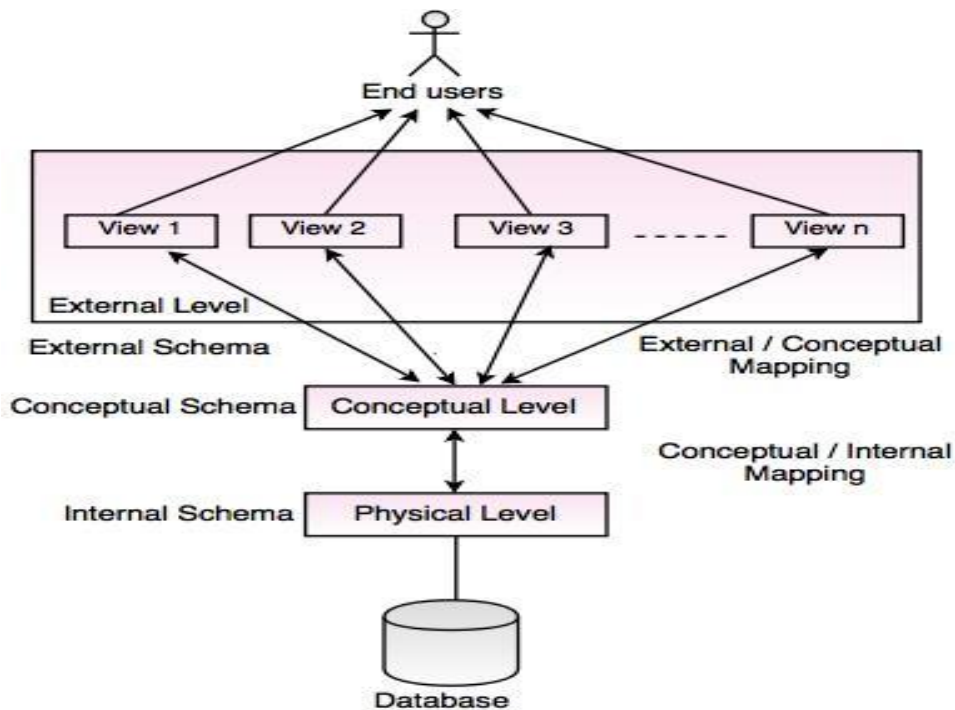


Fig. Three Level Architecture of DBMS



In the above diagram,

- It shows the architecture of DBMS.
- Mapping is the process of transforming request response between various database levels of architecture.
- Mapping is not good for small database, because it takes more time.
- In External / Conceptual mapping, DBMS transforms a request on an external schema against the conceptual schema.
- In Conceptual / Internal mapping, it is necessary to transform the request from the conceptual to internal levels.

1. Physical Level

- Physical level describes the physical storage structure of data in database.
- It is also known as Internal Level.
- This level is very close to physical storage of data.
- At lowest level, it is stored in the form of bits with the physical addresses on the secondary storage device.
- At highest level, it can be viewed in the form of files.
- The internal schema defines the various stored data types. It uses a physical data model.

2. Conceptual Level

- Conceptual level describes the structure of the whole database for a group of users.
- It is also called as the data model.
- Conceptual schema is a representation of the entire content of the database.
- These schema contains all the information to build relevant external records.
- It hides the internal details of physical storage.

3. External Level



- External level is related to the data which is viewed by individual end users.
- This level includes a no. of user views or external schemas.
- This level is closest to the user.
- External view describes the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

Data Abstraction and Data Independence

Database systems comprise of complex data-structures. In order to make the system efficient in terms of retrieval of data, and reduce complexity in terms of usability of users, developers use abstraction i.e. hide irrelevant details from the users. This approach simplifies database design.

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

Physical: This is the lowest level of data abstraction. It tells us how the data is actually stored in memory. The access methods like sequential or random access and file organisation methods like B+ trees, hashing used for the same. Usability, size of memory, and the number of times the records are factors which we need to know while designing the database.

Suppose we need to store the details of an employee. Blocks of storage and the amount of memory used for these purposes is kept hidden from the user.

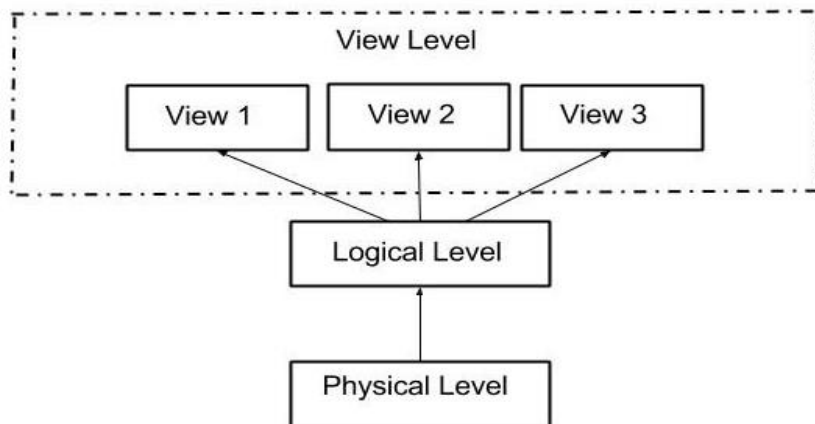
Logical: This level comprises of the information that is actually stored in the database in the form of tables. It also stores the relationship among the data entities in relatively simple structures. At this level, the information available to the user at the view level is unknown.

We can store the various attributes of an employee and relationships, e.g. with the manager can also be stored.

View: This is the highest level of abstraction. Only a part of the actual database is viewed by the users. This level exists to ease the accessibility of the database by an individual user. Users view data in the form of rows and columns. Tables and relations are used to store data. Multiple views of the same database may exist. Users can just



view the data and interact with the database, storage and implementation details are hidden from them.



The main purpose of data abstraction is achieving data independence in order to save time and cost required when the database is modified or altered. We have namely two levels of data independence arising from these levels of abstraction :

Physical level data independence :

It refers to the characteristic of being able to modify the physical schema without any alterations to the conceptual or logical schema, done for optimisation purposes, e.g., Conceptual structure of the database would not be affected by any change in storage size of the database system server. Changing from sequential to random access files is one such example. These alterations or modifications to the physical structure may include:

- Utilising new storage devices.
- Modifying data structures used for storage.
- Altering indexes or using alternative file organisation techniques etc.

Logical level data independence:

It refers characteristic of being able to modify the logical schema without affecting the external schema or application program. The user view of the data would not be affected by any changes to the conceptual view of the data. These changes may include



insertion or deletion of attributes, altering table structures entities or relationships to the logical schema etc.

ENTITY RELATIONSHIP (ER) MODELING

Introduction to ER Model

ER Model is a high-level data model, developed by Chen in 1976. This model defines the data elements and relationships for a specified system. It is useful in developing a conceptual design for the database & is very simple and easy to design logical view of data.

Importance of ER Model


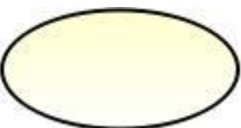
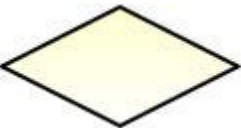

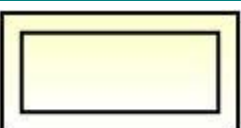
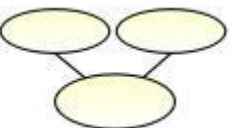
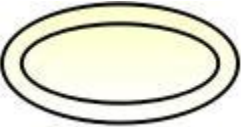

- ER Model is plain and simple for designing the structure.
- It saves time.
- Without ER diagrams you cannot make a database structure & write production code.
- It displays the clear picture of the database structure.

ER Diagrams

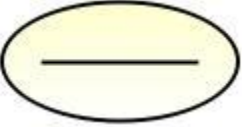
- ERD stands for Entity Relationship diagram.
- It is a graphical representation of an information system.
- ER diagram shows the relationship between objects, places, people, events etc. within that system.
- It is a data modeling technique which helps in defining the business process.
- It used for solving the design problems.



Following are the components of ER Diagram,

Notations	Representation	Description
	Rectangle	It represents the Entity.
	Ellipse	It represents the Attribute.
	Diamond	It represents the Relationship.
	Line	It represents the link between attribute and entity set to relationship set.
	Double Rectangle	It represents the weak entity.
	Composite Attribute	It represents composite attribute which can be divided into subparts. For eg. Name can be divided into First Name and Last Name
	Multi valued Attribute	It represents multi valued attribute which can have many values for a particular entity. For eg. Mobile Number.
	Derived Attribute	It represents the derived attribute which can be derived from the value of related attribute.

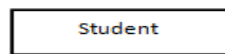


	Key Attribute	It represents key attribute of an entity which have a unique value in a table. For eg. Employee → EmpId (Employee Id is Unique).
---	---------------	--

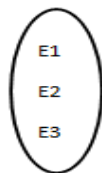
Entity, Entity Type, Entity Set

An Entity may be an object with a physical existence – a particular person, car, house, or employee – or it may be an object with a conceptual existence – a company, a job, or a university course.

An Entity is an object of Entity Type and set of all entities is called as entity set. e.g.; E1 is an entity having Entity Type Student and set of all students is called Entity Set. In ER diagram, Entity Type is represented as:



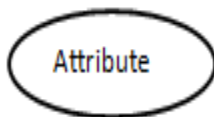
Entity Type



Entity Set

Attribute

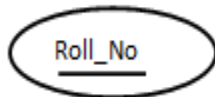
Attributes are the **properties which define the entity type**. For example, Roll_No, Name, DOB, Age, Address, Mobile_No are the attributes which defines entity type Student. In ER diagram, attribute is represented by an oval.



Key Attribute

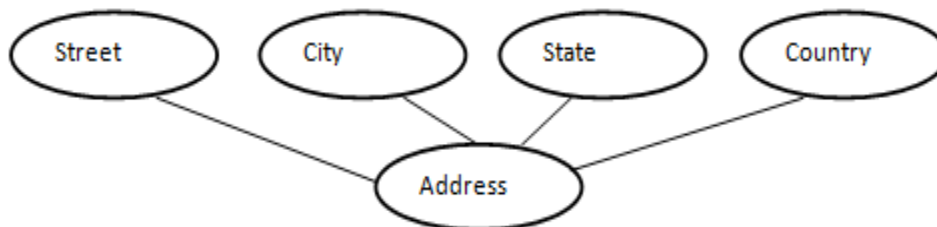


The attribute which **uniquely identifies each entity** in the entity set is called key attribute. For example, Roll_No will be unique for each student. In ER diagram, key attribute is represented by an oval with underlying lines.



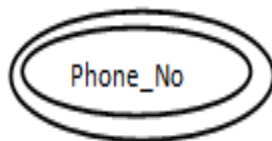
Composite Attribute

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.



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, multivalued attribute is represented by double oval.

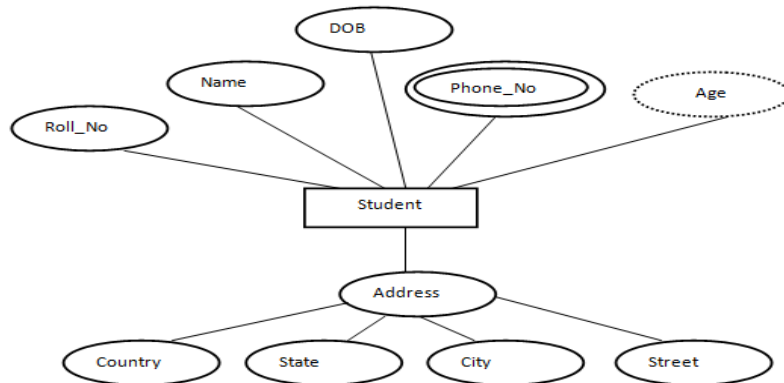


Derived Attribute

An attribute which can be **derived from other attributes** of the entity type is known as derived attribute. e.g.; Age (can be derived from DOB). In ER diagram, derived attribute is represented by dashed oval.

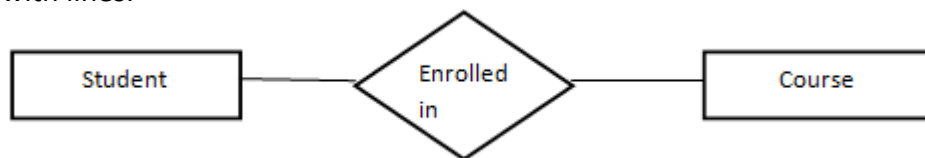


The complete entity type **Student** with its attributes can be represented as:

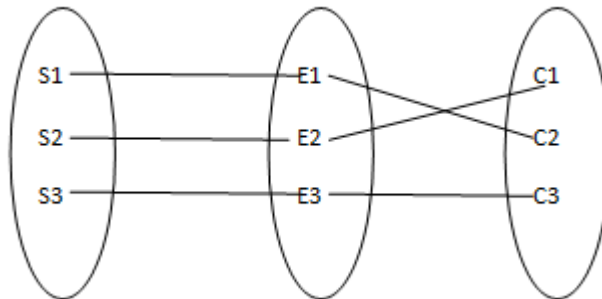


Relationship Type and Relationship Set

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, relationship type is represented by a diamond and connecting the entities with lines.



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

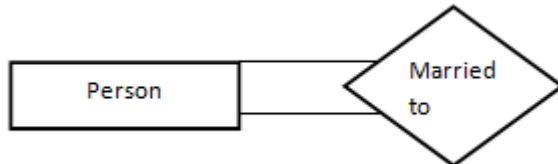


Degree of a relationship set

The number of different entity sets **participating in a relationship** set is called as degree of a relationship set.

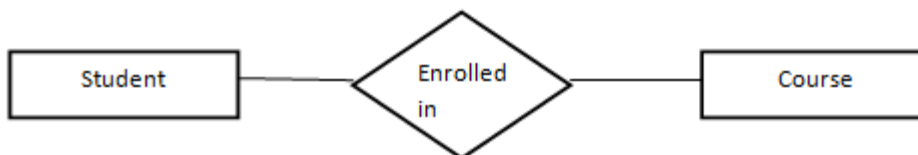
Unary Relationship

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



Binary Relationship

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



n-ary Relationship

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

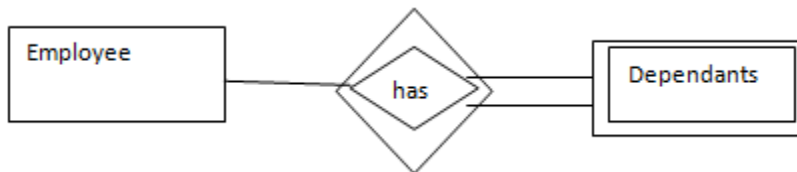
Weak Entity Type and Identifying Relationship



As discussed before, an entity type has a key attribute which uniquely identifies each entity in the entity set. But there exists **some entity type for which key attribute can't be defined**. These are called Weak Entity type.

For example, A company may store the information of dependants (Parents, Children, Spouse) of an Employee. But the dependents don't have existence without the employee. So Dependent will be weak entity type and Employee will be Identifying Entity type for Dependant.

A weak entity type is represented by a double rectangle. The participation of weak entity type is always total. The relationship between weak entity type and its identifying strong entity type is called identifying relationship and it is represented by double diamond.



Relationship Mapping in ER Diagram of Databases

Cardinality

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

Cardinality can be of different types:

Following are the types of Relationship Mapping,

1. One - to - One Relationship
2. One - to - Many Relationship
3. Many - to - One Relationship
4. Many - to - Many Relationship



1. One - to - One Relationship

- In One - to - One Relationship, one entity is related with only one other entity.
- One row in a table is linked with only one row in another table and vice versa.

For example: A Country can have only one Capital City.

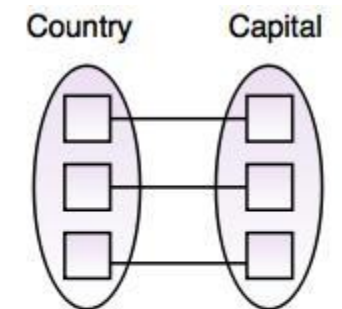


Fig. One to One Mapping

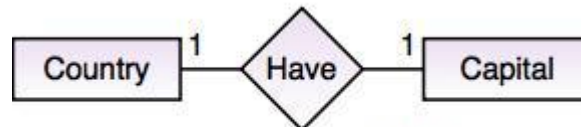
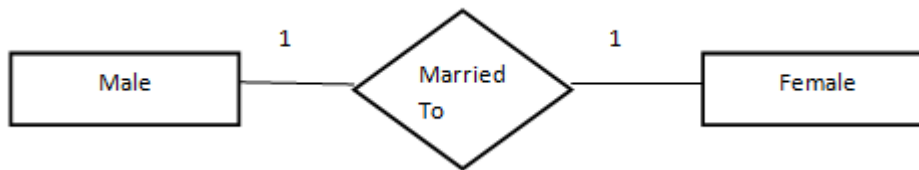


Fig. Representation in ER Diagram



2. One - to - Many Relationship

- In One - to - Many Relationship, one entity is related to many other entities.
- One row in a table A is linked to many rows in a table B, but one row in a table B is linked to only one row in table A.

For example: One Department has many Employees.



Department Employee

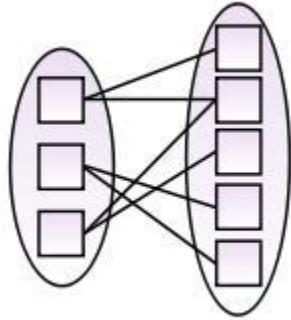


Fig. One to Many Mapping

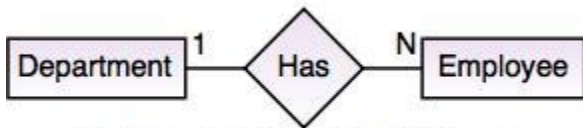


Fig. Representation in ER Diagram

3. Many - to - One Relationship

- In Many - to - One Relationship, many entities can be related with only one other entity.

For example: No. of Employee works for Department.

- Multiple rows in Employee table is related with only one row in Department table.



Employee Department

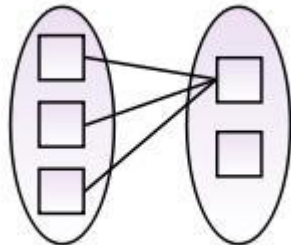


Fig. Many to One Mapping



Fig. Representation in ER Diagram

4. Many - to - Many Relationship

- In Many - to - Many Relationship, many entities are related with the multiple other entities.
- This relationship is a type of cardinality which refers the relation between two entities.

For example: Various Books in a Library are issued by many Students.

Book Student

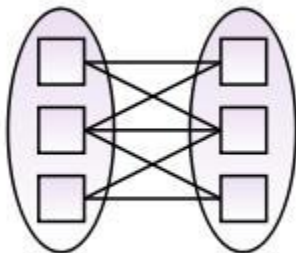


Fig. Many to Many Mapping

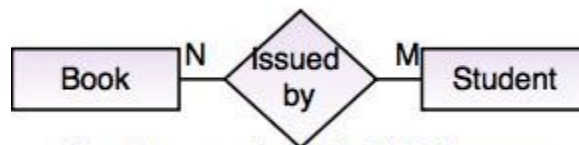


Fig. Representation in ER Diagram

Participation Constraint

Participation Constraint is applied on the entity participating in the relationship set.

Total Participation:



- Each entity in the entity set **must participate** in the relationship. If each student must enroll in a course.
- In Total Participation, every entity in the set is involved in some association of the relationship.
- It is indicated by a double line () between entity and relationship.

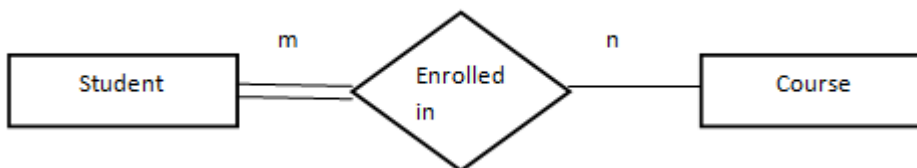
For example: Every Department must have a Manager.



Fig. Total Participation

Partial Participation:

- The entity in the entity set **may or may NOT participate** in the relationship.
- If some courses are not enrolled by any of the student, the participation of course will be partial.
- The diagram depicts the 'Enrolled in' relationship set with Student Entity set having total participation and Course Entity set having partial participation.



- In Partial Participation, not all entities in the set are involved in association of the relationship.
- It is indicated by a single line () between entity and relationship.



Fig. Partial Participation

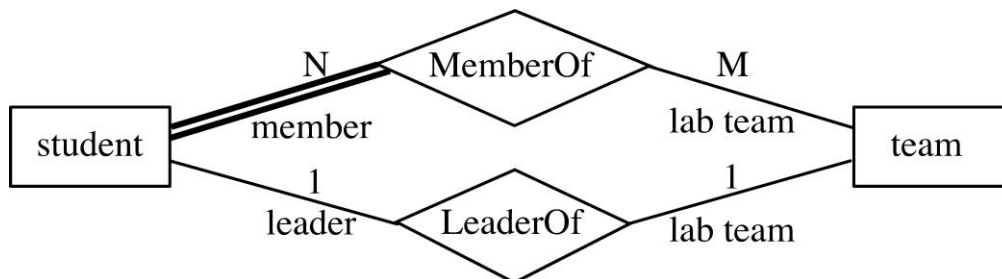
Example : Participation Constraints

Every student must be a member of a team, or, in other words, a student entity is of interest only if it participates in a *MemberOf* relationship. Thus, we can include in an ER diagram a **participation constraint** in which participation

of *student* in *MemberOf* is **total**. A double line indicates the total participation constraint in an ER model

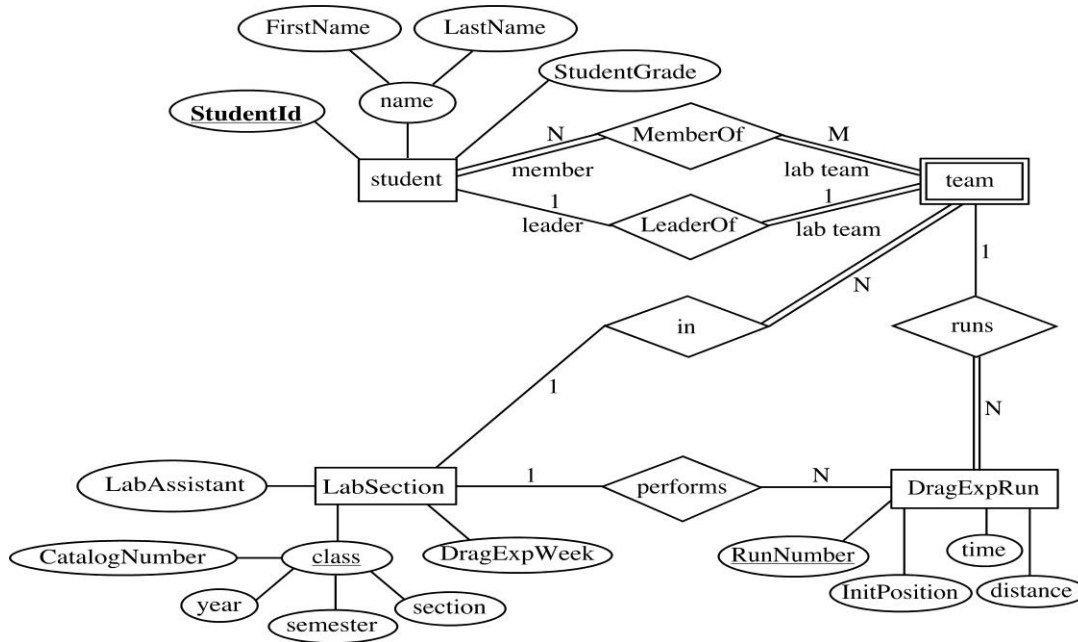
of *student* in *LeaderOf* is **partial**, because a student might be a team leader.

Figure 17. ER diagram notation for total participation constraint



Using the above components,

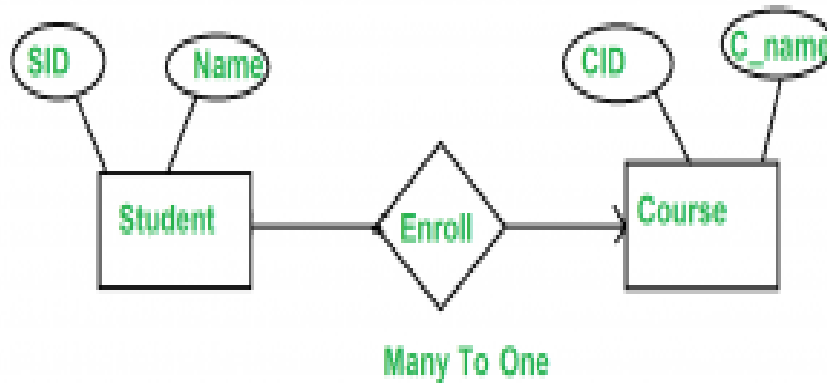
Figure 18. ER diagram notation for total participation constraint



Entity Relationship (ER) Diagram – Relationship

Example

1) When there is One to Many cardinality in ER diagram. For example, a student can be enrolled only in one course, but a course can be enrolled by many students



For Student(SID, Name), SID is the primary key. For Course (CID, C_name), CID is the primary key

Student
(SID Name)

1 A
2 B
3 C
4 D

Course
(CID C_name)

c1 Z
c2 Y
c3 X

Enroll
(SID CID)

1 C1
2 C1
3 c3
4 C2

Now the question is, what should be the primary key for Enroll SID or CID or combined. We can't have CID as primary key as you can see in enroll for the same CID we have multiples SID. (SID , CID) can distinguish table uniquely, but it is not minimum. So SID is the primary key for the relation enroll.

For above ER diagram, we considered three tables in database

Student

Enroll



Course

But we can combine Student and Enroll table renamed as Student_enroll.

Student_Enroll
(SID Name CID)

1	A	c1
2	B	c1
3	C	c3
4	D	c2

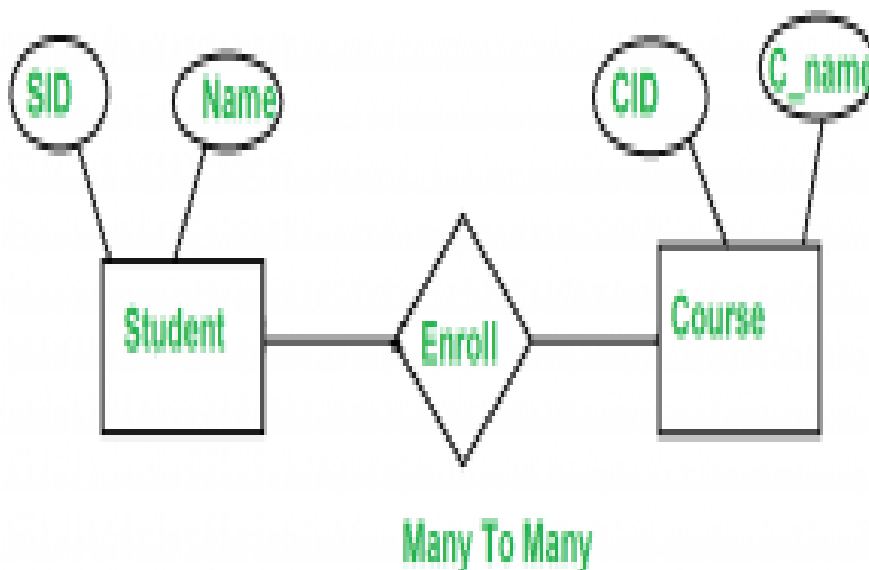
Student and enroll tables are merged now .

So require minimum two DBMS tables for Student_enroll and Course.

Note: In One to Many relationship we can have minimum two tables.

2. When there is Many to Many cardinality in ER Diagram.

Let us consider above example with the change that now student can also enroll more than 1 course.



Student
(SID Name)

Course
(CID C_name)



1	A	c1	Z
2	B	c2	Y
3	C	c3	X
4	D		

Enroll
(SID CID)

1	C1
1	C2
2	C1
2	C2
3	c3
4	C2

Now, same question what is the primary key of Enroll relation, if we carefully analyse the Enroll primary key for Enroll table is (SID , CID).

But in this case we can't merge Enroll table with any one of Student and Course. If we try to merge Enroll with any one of the Student and Course it will create redundant data.

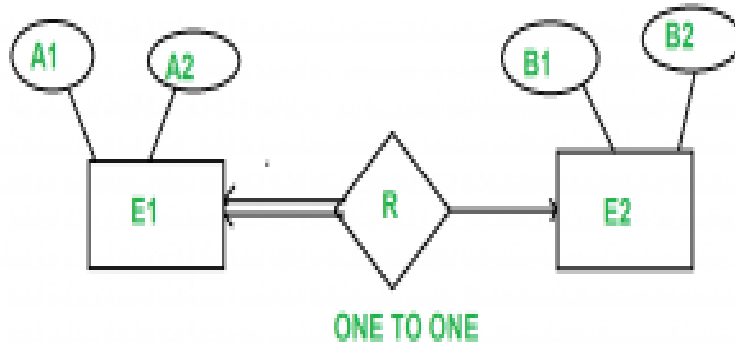
Note: Minimum three tables are required in Many to Many relationship.

3. One to One Relationship

There are two possibilities

A) If we have One to One relationship and we have total participation at at-least one end.

For example, consider the below ER diagram.



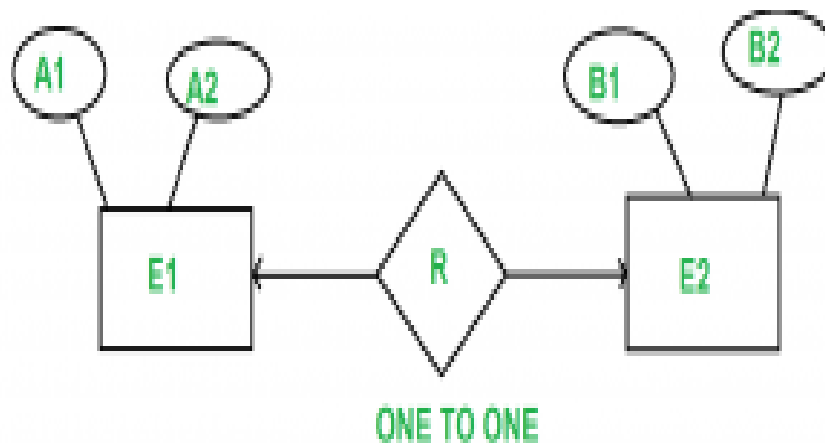
A1 and B1 are primary keys of E1 and E2 respectively.

In the above Diagram we have total participation at E1 end.

Only the primary key of E1, which is in total participation should be allowed as the primary key of the reduced table, since if the primary key of E2 is used, it might have null values for many of its entries, since its participation is only partial and may not have corresponding entries for all its values.

Note – Only one table required.

B) One to One relationship with no total participation.



A1 and B1 are primary keys of E1 and E2 respectively.

Primary key of R can be A1 or B1, but we can't still combine all the three table into one. if we do, so some entries in combined table may have NULL entries. So idea of merging all three table into one is not good.

But we can merge R into E1 or E2. So minimum 2 tables are required.

POSSIBLE QUESTIONS

2 MARK QUESTIONS

1. List the types of database models.
2. List the data types used in DBMS.
3. What is the difference between char and varchar2?
4. Mention the three levels of database schema.
5. Define primary key with example.

8 MARK QUESTIONS

1. Define database and explain various database models with neat diagram.
2. Draw and explain data base architecture.
3. What are the typical ERD symbols and draw a neat ERD for an employee table.
b) Explain i) Entity types ii) Relationships
4. Discuss i) Data Independence ii) Constraints
5. What are the different data models present and explain briefly?