UNIT 2. Relational Data Model.

- **Relational Data Model :-** The Relational data model is most commonly used database model in modern system. In 1970 E F codd's has been developed it. Hence he is known as father of relational database.

The relational database is collection of table and was an attempt to simplify database structere. Relational database is based on tables it known as "relations" These tables have diffrent column are known as "attributes" and rows are known as "tuple".

It is method for organizing and managing data in database using structered tables with rows (tuples) and columns (attributes).

- **Table (Relations) :** Relations is a structere usually represented as a table, organized into rows and columns. Each table in database has unique table name.

Characteristics of Relation :-

1. Table name must be unique.
2. Rows should not be duplicated.
3. order of rows and columns does not matter.
4. Each column contains atomic values.
5. Each column has unique name.

Example :- In college database there is student table which contain student details.

Stud_NO	S_Name	Phone_NO	Address	Geender
100	sita	12345678	Mumbai	F
101	Ram	56789023	Pune	M

Tuple (Row) :- The rows of relation (Table) that contain the values corresponding to attribute are called Tuples. A table can have any number of rows and A row represents a single record in table.

Example :- In above table or in previous table there are 2 tuples.

Attribute / fields :- In previous table a column represent a data item such column in database is called attribute of the table. (Stud_NO, S_Name, Geender ...).

Every table must have one column and it is not possible to have same column name.

In short an Attribute is a column in table. It describes a property of an entity.

Types of Attributes :-

1. Simple Attribute
2. Composite Attribute
3. Single Value Attribute
4. Multi-Valued Attribute
5. Stored Attribute
6. Derived Attribute
7. Null Attribute
8. Key Attribute.

Example :- Stud_NO, S_Name, Phone_NO, Address, Geender are attribute of student table (Relation).

Entities :- An entity is nothing but real world object like student, faculty, subject that can be uniquely identify and this object may be a physical existence or it may have logical existence.

Each entity has it properties. e.g. If student is entity then student_Id, Student_Name, Student_age are its properties and such properties are known as attributes.

- It is basic component of ER Model.
- In short An Entity is a real-world object about which data is stored.

Example :- Student, Employee, Customer, product.

Entity set :- Entity set is collection of all entities of same type.

Types of Entity Set :- (1) Strong entity set.
(2) Weak entity set.

① Domain :- A Domain is the set of Valid Values allowed for an attribute. A domain can have a single value or no value (NULL value).

Example :- ① Attribute :- age → Possible domain :- 18 to 60.
② Attribute :- Geender → Possible domain :- Male or Female.

2.2 keys :-

keys are nothing but column value that uniquely identify the single record in the table called as key of table.

In short keys are used to identify records uniquely in a table.

There is different types of keys :-

① Super key
② Composite key
③ candidate key
④ primary key
⑤ Foreign key.

① Super key :- The set of one or more attributes (column) that can uniquely identify tuple (row) is known as super key.

- It can be existing both individually and in combinations. Every candidate key is super key.

Example :- Possible super key in student table.
Roll_No, Email, Roll_No + Name, Roll_No + Email.

In short a super key is any attribute (or set of attribute which can uniquely identify a record (row) in the table.

② Candidate key :- The minimum super key that can be uniquely identify tuple (row) is known as candidate key.

- It avoid unecessary combination of attribute unlike super key.
- It may contain unique value, ensuring that no two row have same value in candidate key column.

Example :- In given student table stud_NO can be Candidate key because it uniquely identifies each record.

STUD_NO	SNAME	ADDRESS	PHONE
1	Tom	New york	1234567
2	Jhon	Los Angeles	98765432
3	Ram	India	12456798.

③ primary key :- It is chosen from set of candidate key to uniquely identify each record of table.

Example :- In student table both STUD_NO & STUD_PHONE Can be candidate key but STUD-NO is selected as primary key.

- primary key cannot be NULL and unique. each record must have a valid identifier.
- It may be single column or composite
- Database organized data using primary key so, for fast data access and searching.

④ Foreign key :- A foreign key is an attribute (column) is one table that refers the primary key of another table.

- The table that contains the foreign key is called referencing table.
- The table that is refenced is called refenced table.
- It helps connect two or more tables, enabaling you to create relationships between them. This is important for maintaining data integrity (the data contained in the database is both correct and consistent) and preventing data redundancy (preventing duplication of data).

primary key

ID	Name	Course
2041	Tom	Java
2204	John	C++
2043	Alice	Python
2032	Bob	oracle

Student Details

foreign key

ID	Marks
2041	65
2204	55
2043	73
2032	62

Student Marks.

Example :- ID in Student Marks table is foreign key that refers to the ID primary key of student details table.

- unlike a primary key, a foreign key can contain dublicate values and may be NULL.

For example, STUD-NO appears multiple times in student_course because a student can enroll in more than one course.

student_course table.

STUD-NO	TEACHER_NO	COURSE_NO
1	001	C001
2	056	C005

(5) Composite key :- when a single column is not enough to uniquely identify all records in table, a combination of multiple attributes is used to ensure that every row is uniquely identifiable.

Example :- In the STUDENT_COURSE table, [STUD-NO, COURSE_NO] can form a composite key to uniquely identify each record.

Primary key VS	Foreign key
uniquely identifies record	create relationship
cannot be NULL	Can be NULL
one per table	Multiple allowed.

2.3 Data Constraints :-

Data constraints are rules applied to database content to ensure data integrity & accuracy. They are preventing invalid data entry by applying rules to the type or range of values allowed.

- It Validates the quality of the database.

Types of constraints :- (1) Domain constraints
(2) Referential Integrity constraints.

① Domain constraints :- Domain constraints are used to specify that, every column (attribute) in database must contain only values from a predefined set known as Domain.

Domain constraints is used to ensure that every attribut (column) in database must have only valid data. Rule definitions that restrict the values, datatypes and nullability entring a column. (e.g salary > 0 or Geender IN ('M', 'F')).

For example :- consider an STUDENT table with an age column that only allows integer values betn 18 and 35. This domain constraint ensure that inserting a value such as "17" or "70" would violate the rule and be rejected by the system.

Types of constrainsts :-

① Data Type constraint :-
② Range constraint
③ Format constraint
④ Size constraint
⑤ NULL constraint
⑥ Default constraint
⑦ Check constraint.
⑧ Unique constraint.

① Data type constraint :- It restricts the type of data that can be stored.

Column	Data Type.
Roll NO	INT
Name	VARCHAR
DOB	DATE

query :

```
CREATE TABLE Student (
    Roll_NO INT,
    Name VARCHAR(50)
);
```

(2) Range constraints :- Restricts values within a minimum and maximum range.

Example :- Marks should betn 0 to 100 or age should between 18 to 60.

query →
```
CREATE TABLE Student (
    Marks INT CHECK(Marks BETWEEN 0 AND 100)
);
```

(3) Format constraint :- Ensure data follows a specific pattern or structure.

Example :- Mobile Number should be 10 digit or pin code should be 6 digit.

Query :- `CHECK (LENGTH(Mobile) = 10)`

(4) Size constraint :- Restrict the maximum lenght of data.

Example :- Name should be a maximum of 50 characters.

Query :- `Name VARCHAR(50)`

(5) Null constraints :- In database some attributes are mandatory like student name, student email, student phone_no and this attributes cannot be NULL or blank.

Example :-

```
STU_FIRST_NAME char(100) NOT NULL
STU_MIDDLE_NAME char(100) NULL
```

Controls whether column can store NULL (empty value). NULL value means missing value or unassigned value.

TYPE	Meaning
NULL	Empty value allowed.
NOT NULL	Empty value not allowed.

(6) Default constraints :- By using default constraint we can assign default value automatically if user does not enter value.

Query :-
```
CREATE TABLE Student (
    city VARCHAR(20) DEFAULT 'pune'
);
```

(7) check constraints :- It check a specific condition before inserting a data.

Example :- Salary must be positive or age must be 18.

Query :-
```
CREATE TABLE Employee (
    Salary INT CHECK (salary > 0)
);
```

(8) unique constraint :- It prevent duplication of values.

Example :- Email Id should be unique.

Query :-
```
CREATE TABLE Student (
    Email VARCHAR(100) UNIQUE
);
```

Referential Integrity constraints :- The Referential Integrity constraint is specified between two tables (relation) used to maintain consistency among the records of two tables.

Example :

we have 2 tables student and department table.

Student table.

Stud_id	Stud_Name	dept_id.
1	Siya	10
2	Ram	30
3	Madhav	20

Department table.

Dept_id	Dept_name
10	comp
30	ENTC
~~20~~	~~EE~~

← if This never exist in table.

- In above student table has "Dept_id" foreign key reference this called Referential integrity constraints.

we are forcing database to check the Dept_id Value from department table while inserting any Value in student table in Dept_id column there is no value existing in department table of that Dept-id then we cannot insert that value in student table. This help to maintain consistency in database.

Example :

```
Create table student (
    Stud_id integer,
    Stud_name varchar(100) NOT NULL
    Dept_id as integer references department(dept_id)
);
```

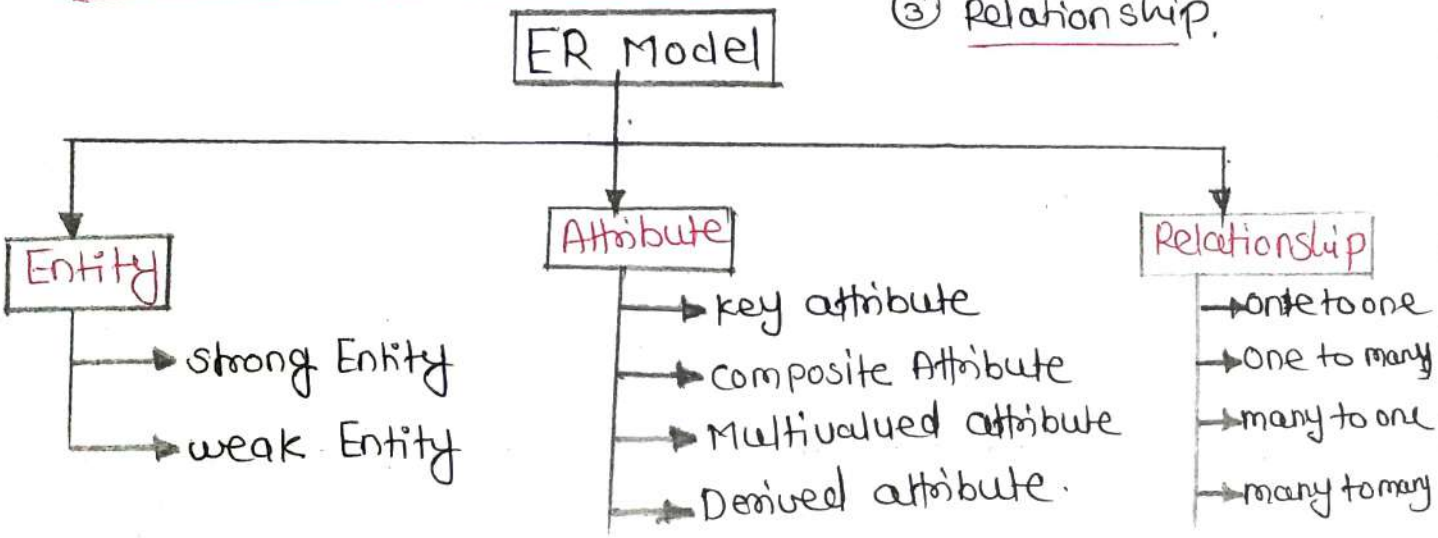
```
Create table Department (
    dept_id integer,
    Dept_name varchar(100) NOT NULL,
    Primary key (dept_id));
```

2.4 Entity Relationship Model

Entity Relationship model also known as ER Model which is developed by scientist Chas Hen in 1976. It is high level conceptual data model.

In short ER Model is used to design database structure visually. It show entities, attributes and relationships.

Components of ER Model : ① Entity ② Attribute ③ Relationship.



- **Entities** : An object that is stored as data. It may be with physical existence - particular person, car, house. or it may be with conceptual existence - a company, a job, university name.
  e.g Student, Department etc.

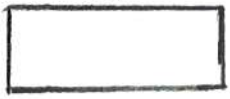
- **Entity set** : An entity is a object of entity type and set of all entities is called as entity set.

Types of entity set :-

- **Strong entity set :-** An entity set that has a key attribute by which we can identify the entity uniquely is called strong entity set.

for example, In the case of college student, each student has a unique student_id which is primary key of the student entity.

- A single rectangle represents a strong entity type.



- **Weak entity set :-** An entity set which are not able to create a unique key attribute by using its attribute and take help from the corresponding strong entity set called as weak entity set.
  - This type of entities are dependent on strong entities for primary key.
  - A double rectangle represent a ~~we~~ weak entity set.

Example :- Employee table

emp_ID	emp_name
E101	Amit
E102	Ram

Dependent table.

Dependent name	Relation	empID
Rahul	son	E101
Priya	wife	E101

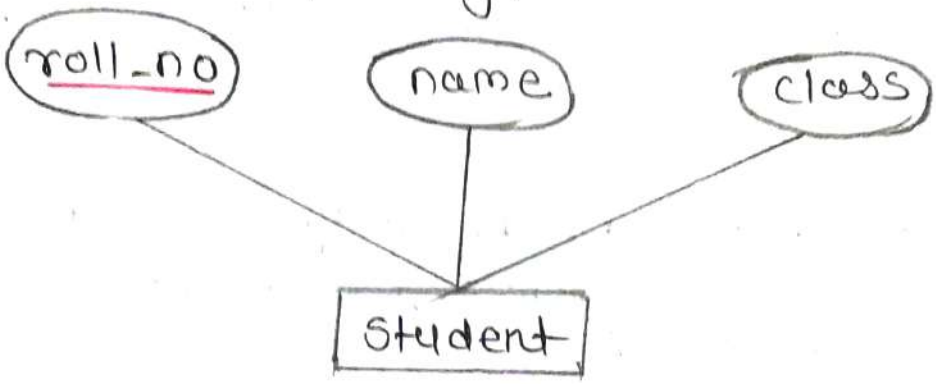
Strong Entity	Weak Entity.
Independent	Dependent
Has own primary key	No full primary key
Can exist alone	Cannot exist alone.

Attributes :- Attributes is nothing but column of table or Attribute are properties that define the entity type.
For example; roll no, Name, Address, mobile-no are attribute of define entity type Student.

Types of Attribute :-

① Simple attribute :- An attribute cannot be divided further is called simple attribute.

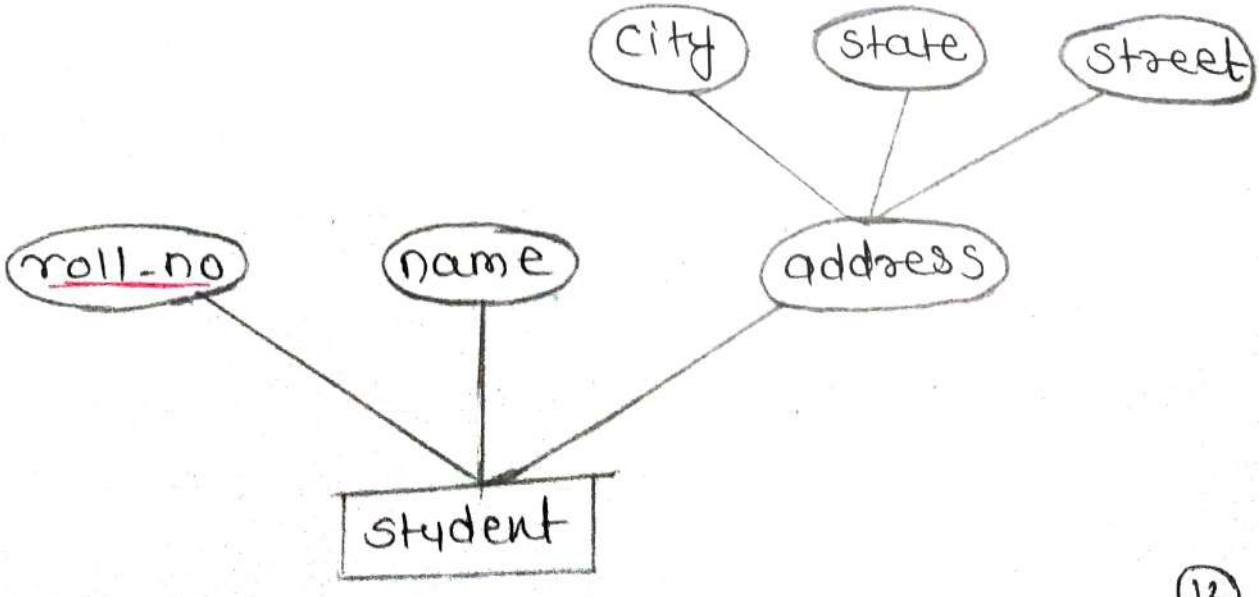
Example :- Age, salary.



② Composite Attribute :- can be divided into smaller parts. An attribute that can be split into components is a composite attribute.

Example :- The address can be further split into house number, street number, city, state, country, and pin code, the name can also be split into first name, middle name and last name.

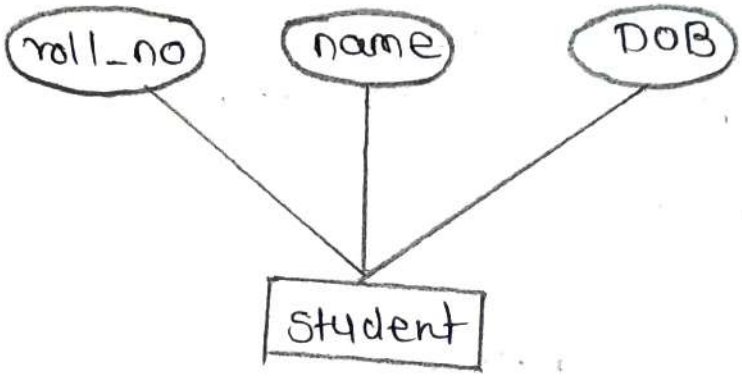
Address ⟶ city, state, PIN.



③ Single Valued attribute :-

It contains one value. The attribute which take up only a single value for each entity instance is a single valued attribute.

Example :- The age of a student, Aadhar card number.

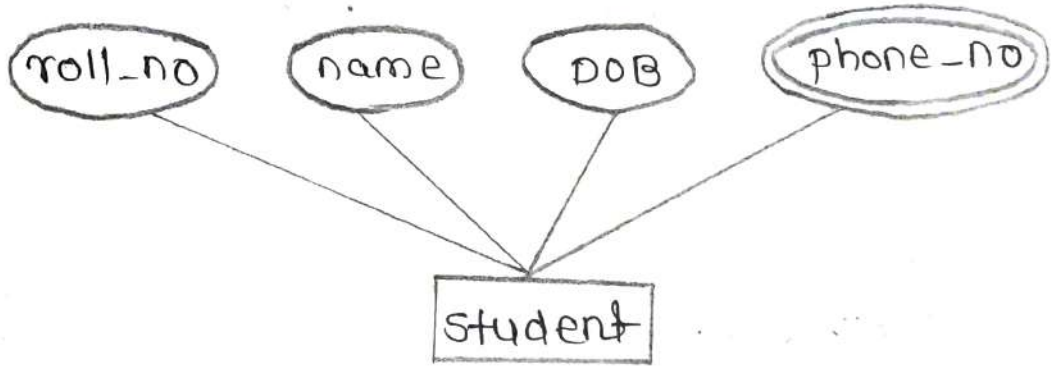


④ Multi-Valued attribute :-

It contains multiple values. The attribute which take up more than a single value for each entity instance is a multi-valued attribute.

It is represented by double oval shape.

Example :- Phone number of student : Landline & mobile.

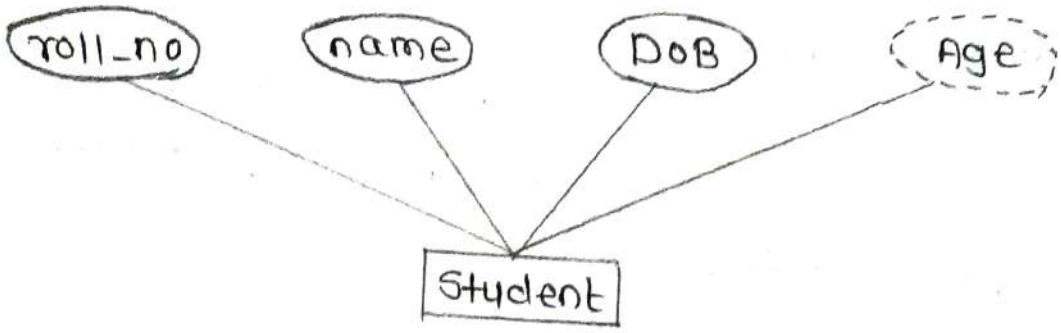


⑤ Derived Attribute :-

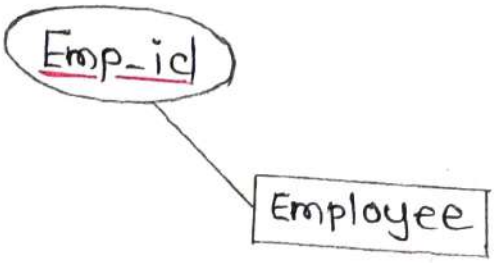
Calculated from another attribute. An attribute that can be derived from other attributes is known as derived attribute.

It is represented by dotted oval shape.

Example : Age derived from Date of Birth.



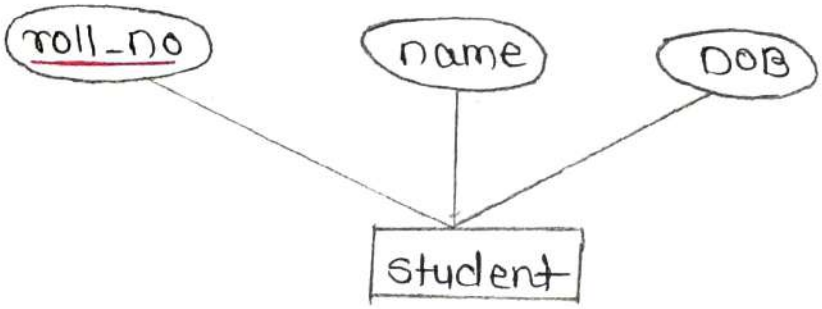
⑥ Key attribute :- This is the attribute of entity that must have unique value by which tuple (row) is identify uniquely called as key attribute.



⑦ 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.

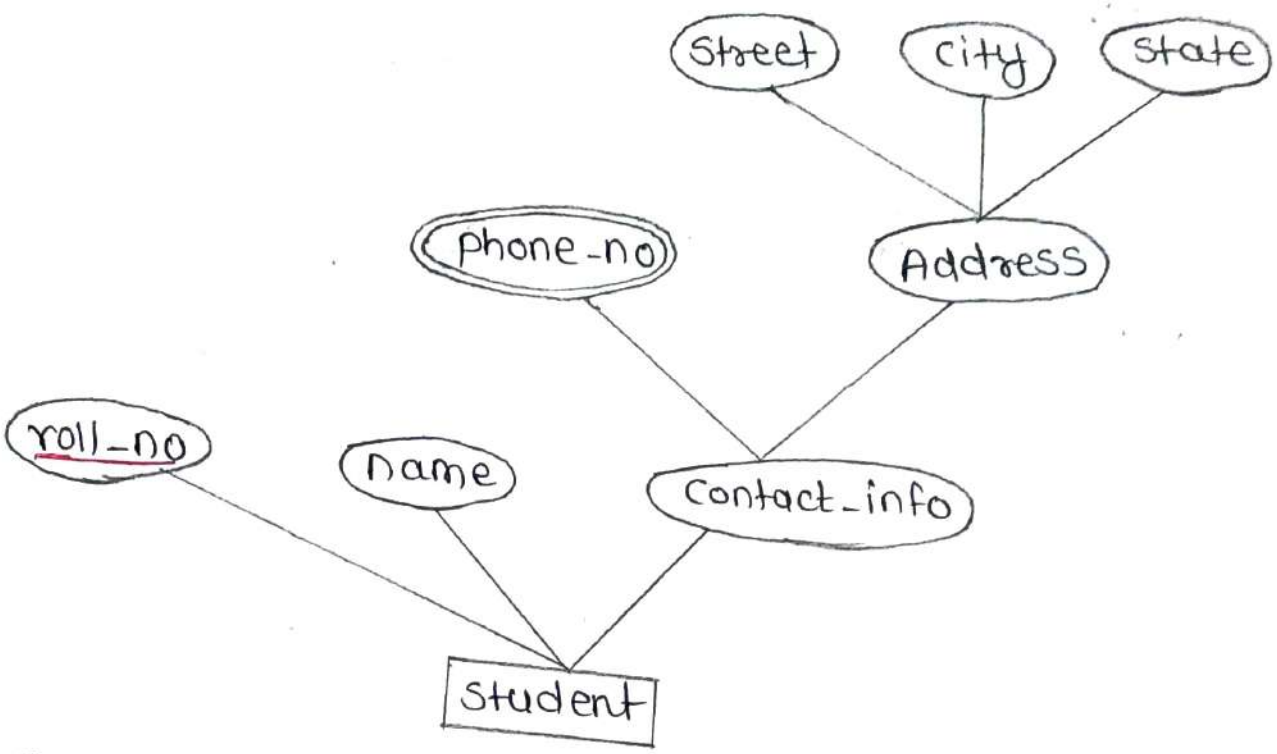


⑧ Complex Attribute :-

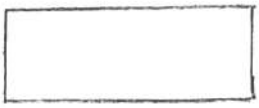
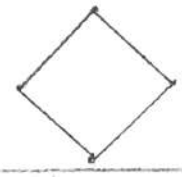
Those attributes, which can be formed by the nesting of composite and multi-valued attributes, are called complex attribute.

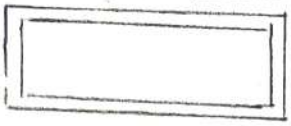
These attributes are rearely used in DBMS (Database Mangement System). That's why they are not so popular.

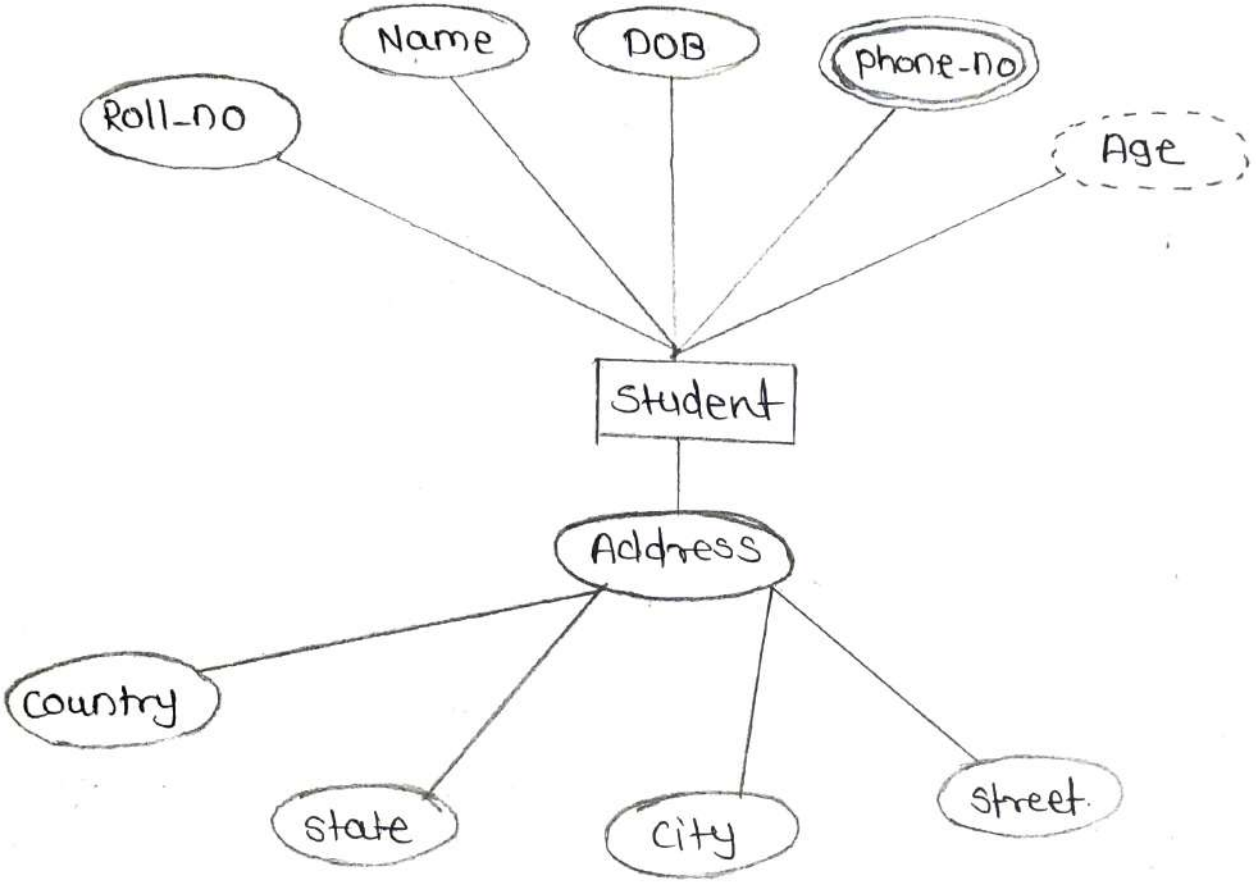
Example :- Address because address contain composite value like street, city, state, PIN code and also multivalued because one people has more than one house address.



Symbols for ER Diagram :-

Name	Symbols	Represents
Rectangle		It represents Entity in ER Diagram.
Ellipse		It represents Attribute in ER Diagram.
Diamond		It represents Relationship among entities.
Line		Represents link between Attribute to Entites and Entity set with other relationship type.
Double Ellipse		It represents Multi-valued Attribute.

Name	Symbol	Represents
Double Rectangle		It represents weak entity.
Dotted Ellipse		It represents Derived attribute.

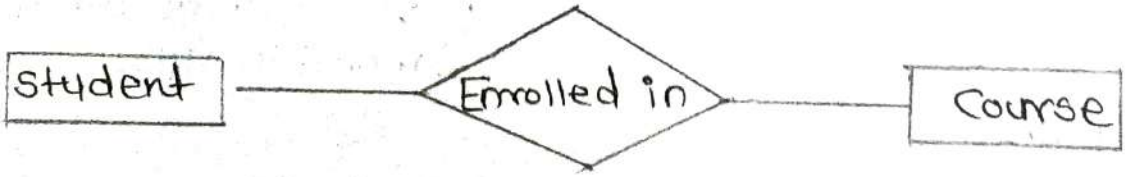


Relationship type and Relationship set :-

Relation type represents the association between entity type.

For example, "Enrolled in" is relationship type exists between entity type student and course.

e.g student enrolled in specific course.

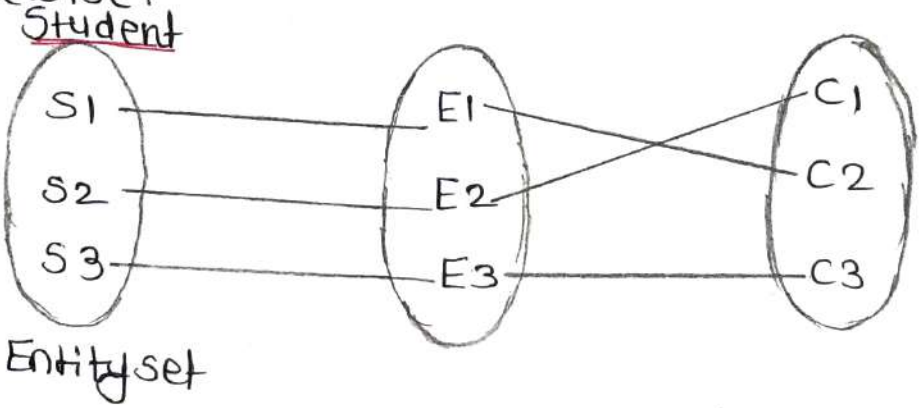


In the ER diagram the relation tuple represented by diamond & connecting the entities with line.

Example 2:

A set of relationships of same type is known as relationship set.

Relationship set show S1 enrolled in C2, S2 enrolled in C1 likewise.



Degree of relationship set :-

The number of different entity set participating in relationship set called Degree of relationship set.

① Unary relationship : when there is only one entity set participating in a relation its known as unary relation-ship.

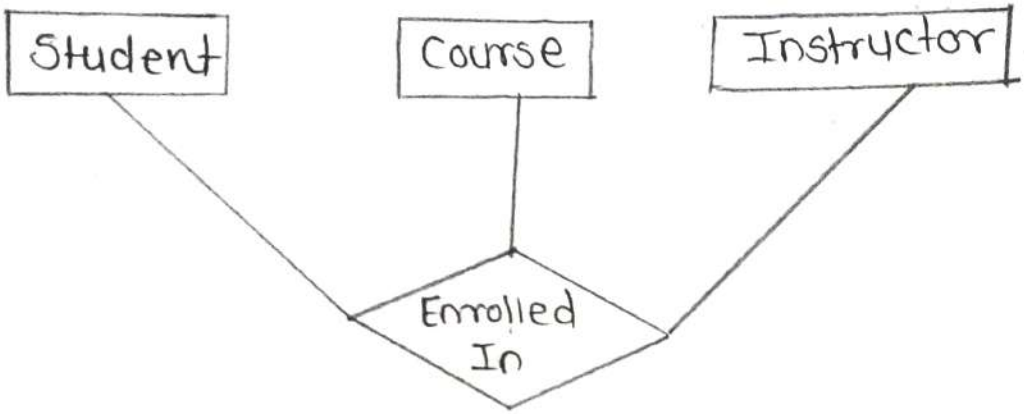
Example : one person is married to only on person.

② Binary Relationship :

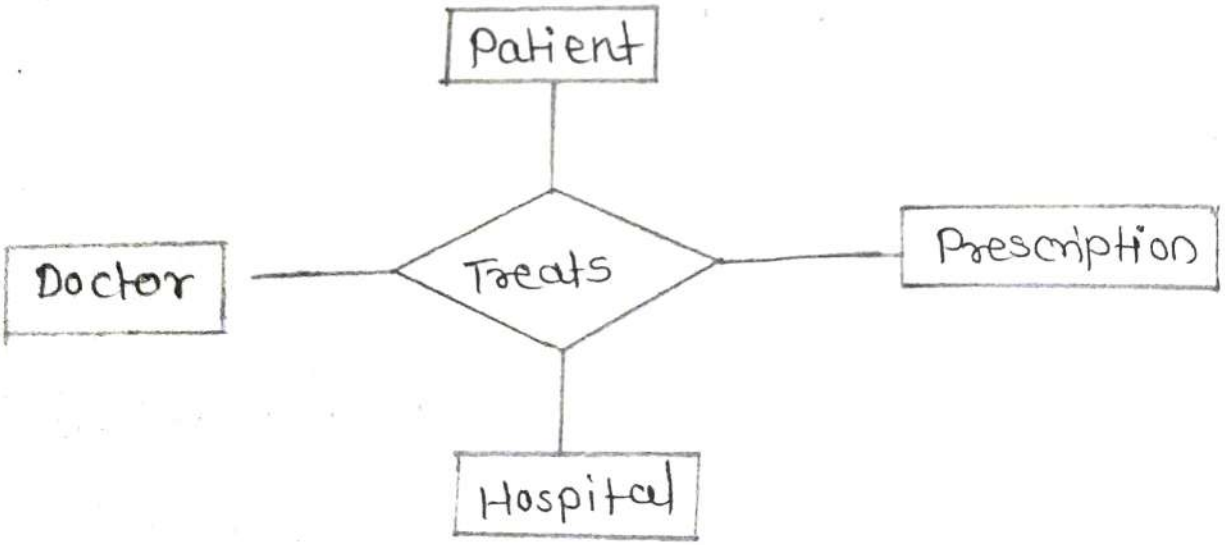
when there are two entities set participating in relationship.

For example, A student is enrolled in a course.

③ Ternary Relationship : when there are three entity sets participating in a relationship.



④ N-ary Relationship : when there are n entities set participating in a relationship, the relationship is called an n-ary relationship.



Cardinality in ER Model :-

The maximum number of times an entity of an entity set participate in relationship set known as cardinality.

Types of mapping cardinality :-

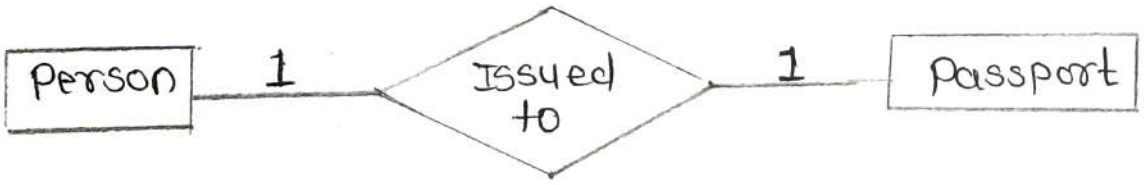
① one to one (1:1) :-

when one tuple (row) in entity is related to only one tuple (row) in other entity set is called one to one cardinality.

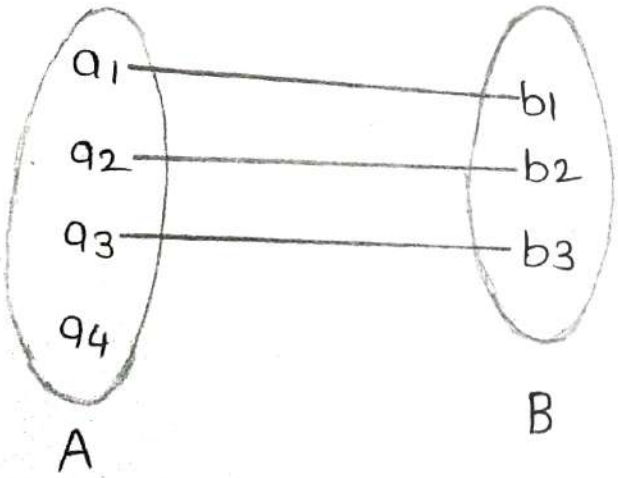
In short one entity is related to only one entity.

Example :- one person has one passport.

one passport belongs to one person.



Using sets it can be represented as :



set represents one to one cardinality.

② one to many (1:M) :-

when one tuple entity can related to many tuples(row) in another entity is called one to many cardinality.

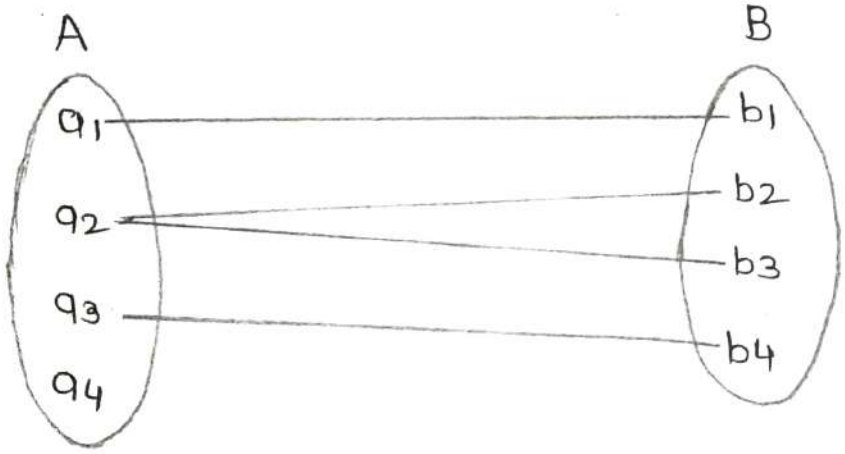
In short one entity is related to many. entity.

Example :-

one department has many employees

one employee belongs to one department.

Using sets, one to many cardinality can be represented as :-



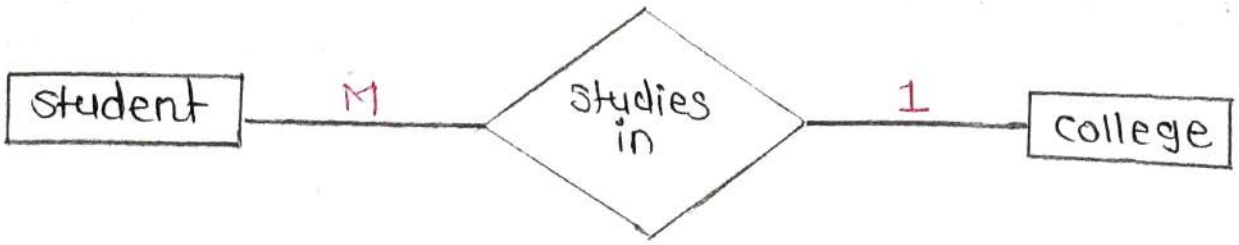
Set representation of one-to-many.

③ Many to one (M:1) :-

when many tuples in entity can be related to only one tuple (row) in another entity is called many to one cardinality.

In short many entities are related to one entity.

Example : Many students study in one college.



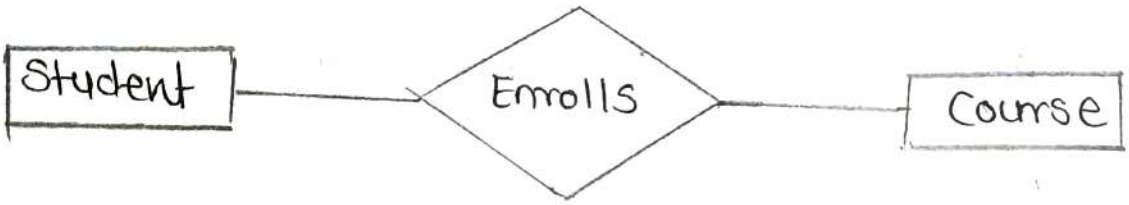
④ Many to many :- (M:M)

when many tuples entity can be related to many tuples in another entity is called many to many cardinality.

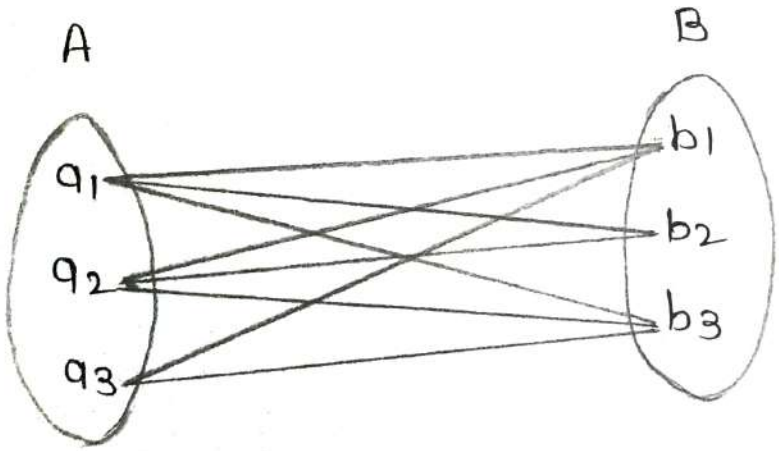
In short many entities are related to many entites.

Example :

one student can study many courses
one course can have many students.



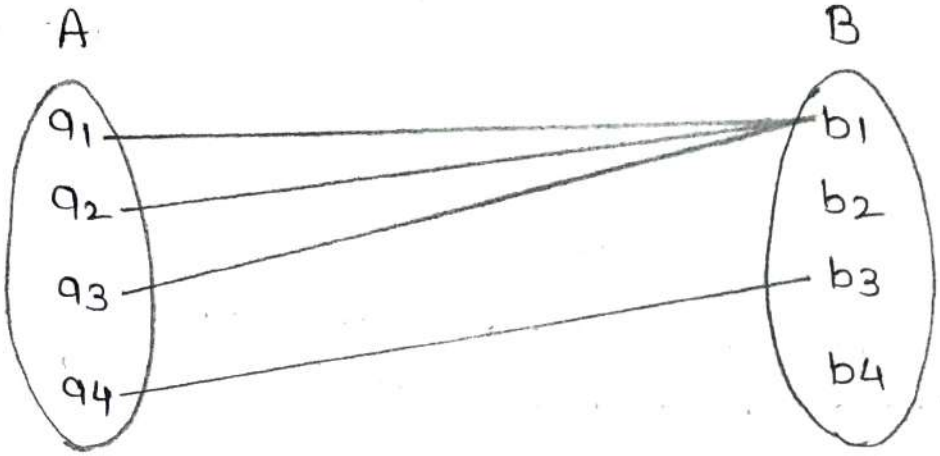
using sets, it can be represented as :



In this example student A1 is enrolled in B1, B2 & B3. and course B3 is taken by A1, A2 & A3.

Therefore, this represents a many to many relationship

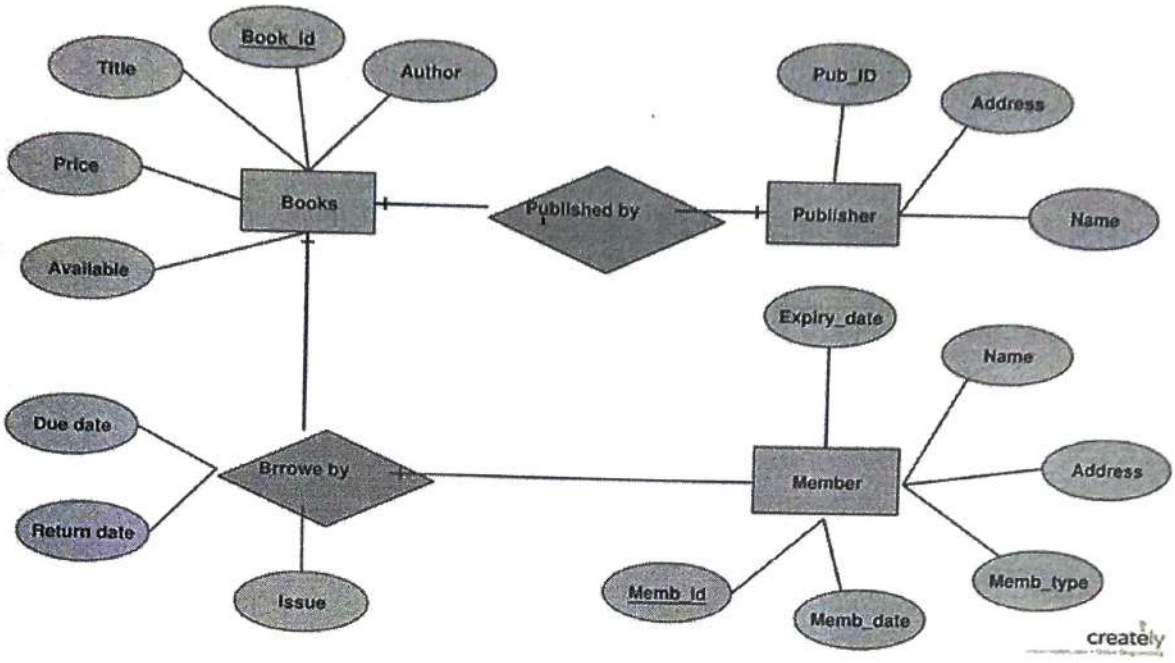
Using set, many to one represented as :



Set representation of many to one.

Mapping Type	Meaning	Example.
1 : 1	one to one	Person – Passport
1 : M	one to many	Department – Employees
M : 1	many to one	Student - College
M : M	many to many	Students - Courses

E-R Diagram for Library Management System



Draw ER diagram college Management System

Fig. 2.29

Example 7: Draw E-R diagram for College Management System.

Fig. 2.30 shows E-R model for College Management System.

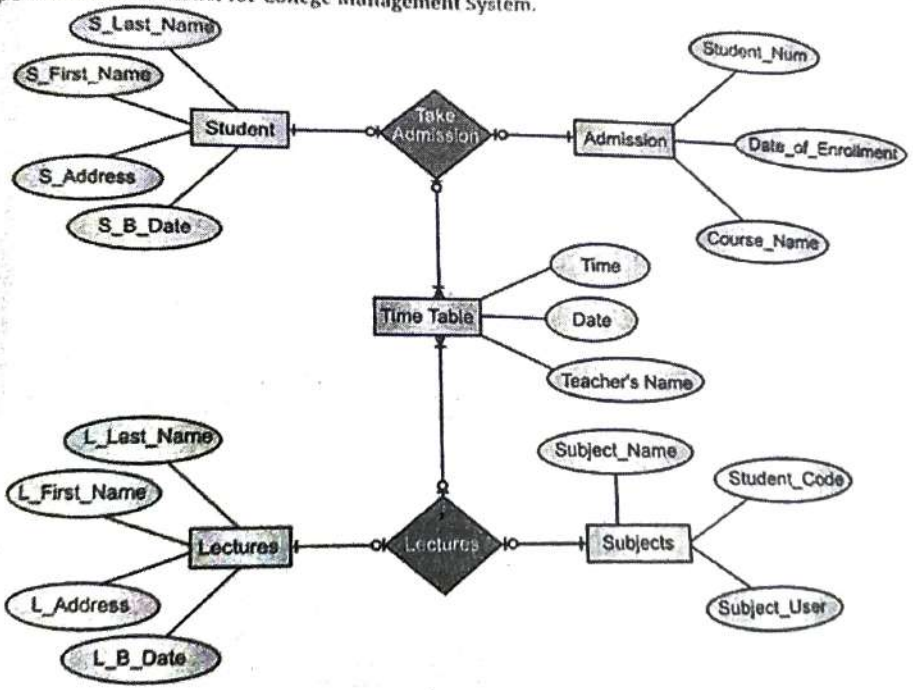


Fig. 2.30

Normalization :-

Normalization provide a systematic approach to determine (define) a table structure through a set of simple techniques by using we can achive the desired table structure.

Normalization is process of organizing the data in database to avoid data redundancy (duplication of data), insertion anomaly, update anomaly and delete anomaly. It divides larger tables in smaller tables and link them using relationships.

Anomalies in DBMS :-

Types of Anomalies

1. Insertion Anomaly
2. update Anomaly
3. Deletion Anomaly.

① Insertion Anomaly :-

Insertion Anomaly :- Insert Anomaly is something when we are not able to insert data into tables due to some constraints.

Example :- Suppose new employee joins the company who under training and currently department is not assign to him then we would not able to insert the data into table if emp_dept field doesn't allow nulls.

② Delete Anomaly :-

delete anomaly is something when we delete some data from the table, and due to that delete operation we loss some other useful data.

③ Update Anomaly :-

Update Anomaly is something when we are trying to update some records in table, and that update causing data inconsistency.

Funcational dependencies :-

The concept of functional dependency given by E. F. codd.

- A funcational dependency occurs when one attribute uniquely determine another attribute within relation.
- If attribute A functionally determined attribute B. we can write it like A → B.

we can say column A is functionally dependent on another column B. if data value in column A change when data value in another column B is modified.

- Functional dependencies are used to mathematically express relations among database entities and very important to understand advanced concept in relational database system.

Roll_no	name	dept_name	course
1	Ram	CO	C
2	jhon	IT	C++
3	rita	CO	Java
4	Ravi	IT	ML

- Valid functional dependencies :-

Roll_no → {name, dept_name, course}

① Roll_no determine value of field name, dept_name, course hence valid functional dependency.

② Roll_no → dept_name.

① First normal form (1NF) :-

All attributes (column) contain only atomic value.

atomic value → Indivisible value or simple value.

- Each table cell should contain a single value.
- Each record needs to be unique.

Example :- Table not in 1 NF

In table 1 course has a multi-valued attribute so not in 1NF we have to remove multivalued attribute from table.

ID	NAME	COURSE
1	Laxman	ME/CO
2	Ram	CO
3	Sita.	CO

Table 1

Solution :- Now in table 2 there is 1NF as there is no multi-valued attribute present in table Table 2.

ID	NAME	COURSE
1	Laxman	ME
1	Laxman	CO
2	Ram	CO
3	Sita	CO

Table 2

— This normal form given by E.F. Codd (1970) and the later version by C.J. Date (2003)

(2) Second normal form (2NF):-

A table is 2NF when it is in 1NF and all its attributes fully depend on the primary key or candidate key.

That is, it has no partial dependencies.

Partial dependency is when the non-prime attribute is determined by a part of candidate or primary key, even if the primary key is composite.

Rules of 2NF :-

- Table must be in 1NF
- all the non-prime attributes should be fully functionally depends on candidate key.
- NO partial dependency.

customer_id	Store-id	Location
1	1	Delhi
1	3	Mumbai
2	1	Delhi
3	2	Banglore
4	3	Mumbai

Here Candidate key :- customer_id, Store_id.

Prime attributes :- customer_id, store_id.

non-prime attribute :- Location.

In above table there is non-prime attribute is location. And this location attribute is depends on Store_id attribute but it should be dependent on both candidate key not on only one. there is Partial dependency.

So, this table is not in 2NF.

So, how to make it in 2NF. we can divide this table.

In 2NF

Customer_id	Store_id
1	1
1	3
2	1
3	2
4	3

- Here customer_id & store_id will be candidate key combinally.
- In this table there is no non-prime attribute.

In 2NF.

Store_id	Location
1	Delhi
2	Bangalore
3	Mumbai.

In this table primary key is store_id.

Now we can say that location is fully dependent on store_id which is a candidate key so, now this table is 2NF.

e.g. There is candidate keys and one non-prime attribute C.

AB is candidate key.

$A \rightarrow C$ (A determined C)

Here C is non-prime attribute and it is determined by part of candidate key. So, it is not in 2NF.

③ Third Normal form (3NF) :-

- The table should be in 2NF.
- There should be no transitive dependency.

for functional dependency.

LHS must be candidate key or super key

OR

RHS is a prime attribute.

NO Non prime attribute is transitively depends on primary key attribute.

- what is transitivity?

In the table we have some column (attribute) that act as primary key and another column depends on this column. But what if a column that is not the primary key depends in anothe column that is also not a primary key or part of it? Then we have Transitive dependency in our table.

Example :-

Roll_no	State	City
1	Maharashtra	Pune
2	Haryana	Ambala
3	Bihar	Patana
4	Maharashtra	Mumbai
5	Haryana	Ambala

Here city attribute is depends on state attribute because first we get state and thence we get known city and state in non-prime attribute and primary key is roll_no so, there is a transitive dependency here.

Advantages of Normalization :-

- A smaller database can be maintained as, normalization eliminates the duplicates data. Overall size of the database is reduced as a result.
- Fewer indexes per table ensure faster maintenance task
- Also realizes the option of joining only the table that are needed.

Disadvantages of Normalization :-

- More tables of joins as by spreading out data into more tables, the need to join table's increases and the task become more tedious (boaring).
- The database become harder to realize as well.
- As the normal form type progresses, the performance become slower and slower.

Needs of Normalization :-

(1) Ensure Data integrity :-

Data integrity means to check the correct data stored within the database.

It can be achived by integrity constraints.

(2) Prevents redundancy in data :-

data redundancy means remove duplicate data.

If data in stored in two location, but the data is updated in only one location then data become incosistent. this refered to "update Anomaly"!

(3) To avoid data anomaly:-

There is 3 anomaly:

Insertion anomaly
Deletion anomaly
update anomaly.

A relational database table should avoid all data anomalies.

21/5/26
VSP

21/5/26
Mr A.V. Pixit.

20/5/26