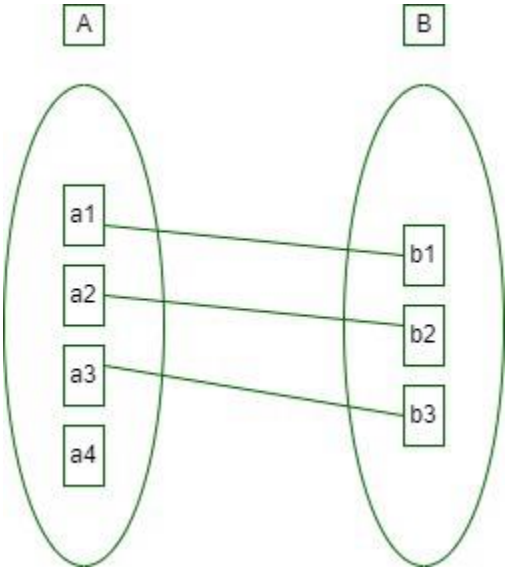
1. CARDINALITY

In database management, cardinality represents the number of times an entity of an entity set participates in a relationship set. The cardinality of a relationship is the number of tuples (rows) in a relationship. Types of cardinality in between tables are:

- one-to-one
- one-to-many
- many-to-one
- many-to-many

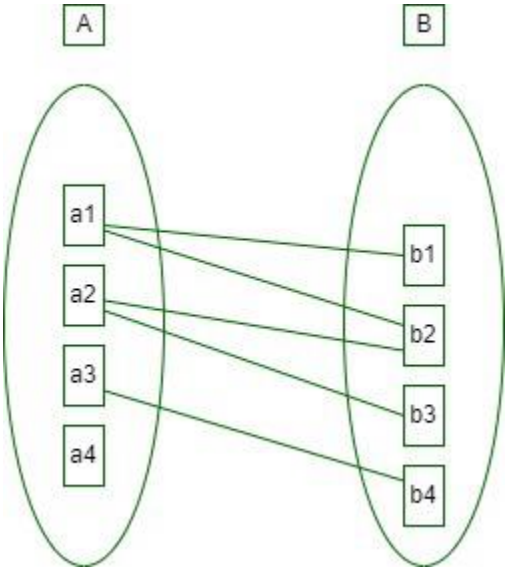
1. One-to-one: An entity in A is connected to at most one entity in B. Or a unit or item in B is connected to at most one unit or item in A.



Example:

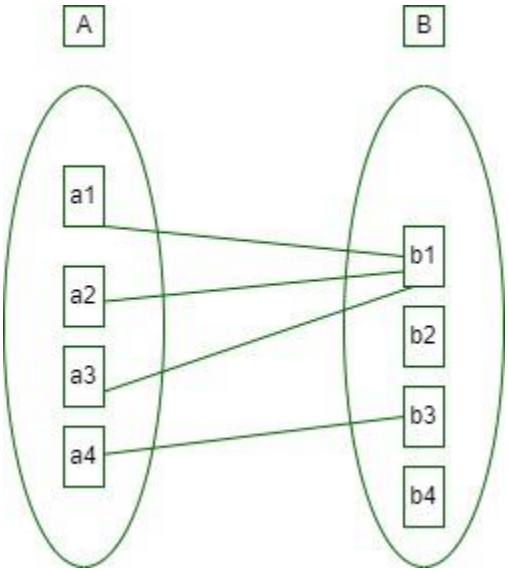
In a particular hospital, the surgeon department has one head of department. They both serve one-to-one relationships.

2. One-to-many: An entity in A is associated with any number of entities in B. Or that one unit or item in B can be connected to at most one unit or item in A.



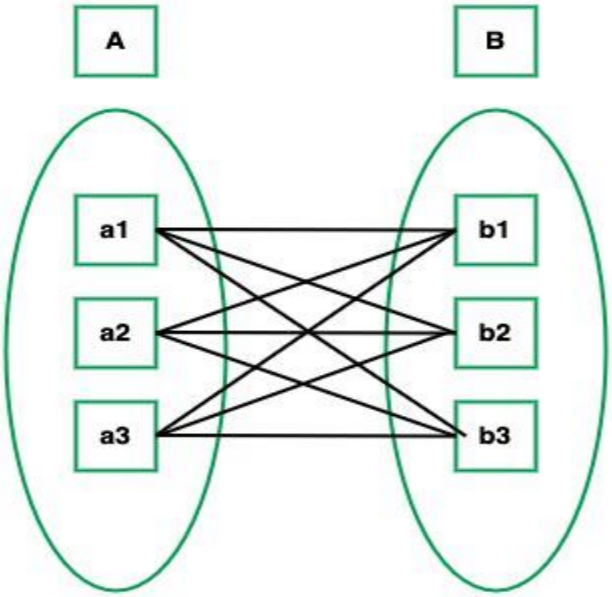
Example:
In a particular hospital, the surgeon department has multiple doctors. They serve one-to-many relationships.

3. Many-to-one: An entity in A is connected to at most one entity in B. Or we can say a unit or item in B can be associated with any number (zero or more) of entities or items in A.



Example:
In a particular hospital, multiple surgeries are done by a single surgeon. Such a type of relationship is known as a many-to-many relationship.

4. Many-to-many: An entity in A is associated with any number of entities in B, and an entity in B is associated with any number of entities in A.



Example:
In a particular company, multiple people work on multiple projects. They serve many-to-many relationships.

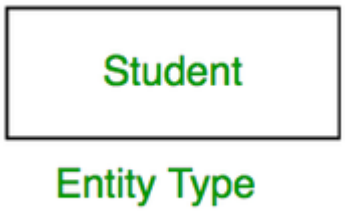
2. ENTITY, ENTITY SET AND ENTITY TYPE

The entity, Entity Set, and Entity Type are very important concepts of ER Model.

Entity : An entity is a thing in a real-world with independent existence. An entity can exist independently and is distinguishable from other objects. It can be identified uniquely.

Example :

- A student with a particular roll number is an entity.
- A company with a particular registration number is an entity.



- .

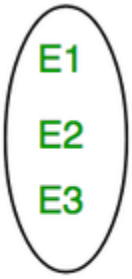
Entity Type : It refers to the category that a particular entity belongs to. It is represented by the name of the table and its schema

Example :

- A table named student in a university database.
- A table named employee in a company database.
-

.

Entity Set : An entity set is a collection or set of all entities of a particular entity type at any point in time. The type of all the entities should be the same.



Entity Set

Example :

- The collection of all the students from the student table at a particular instant of time is an example of an entity set.
- The collection of all the employees from the employee table at a particular instant of time is an example of an entity set.

Relation With Table :

Consider a table student as follows :

Table Name : Student

Student_ID	Student_Name	Student_Age	Student_Gender
1	Avi	19	M
2	Ayush	23	M
3	Nikhil	21	M

Student_ID	Student_Name	Student_Age	Student_Gender
4	Riya	16	F

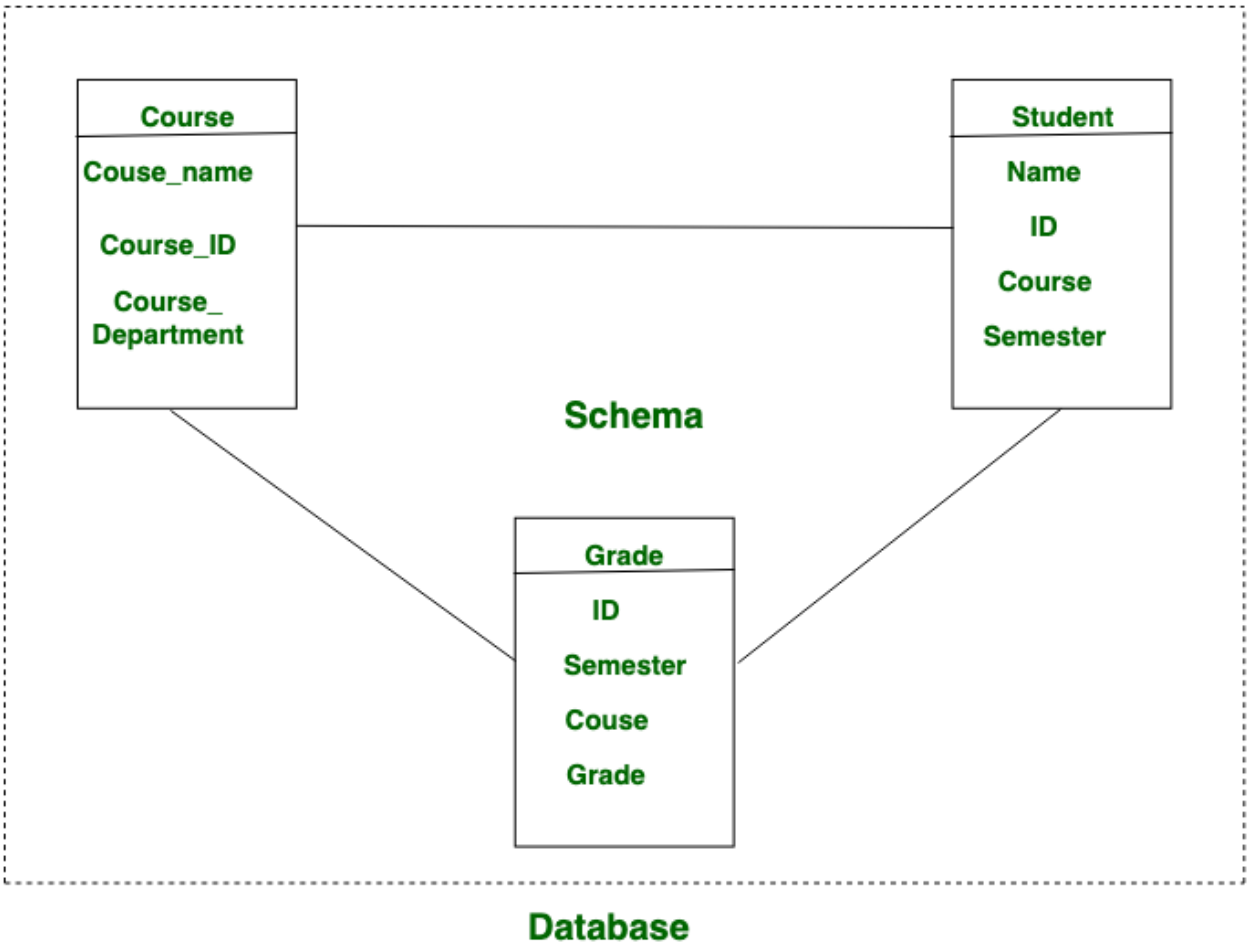
Entity : Each row is an entity.
Example :

1 Avi 19 M

Entity Type : Each entity belongs to the student type. Hence, the type of entity here is a student.
Entity Set : The complete data set of all entities is called entity set. For the above table, the records with student id 1, 2, 3, 4 are the entity set.

3. **SCHEMA**

This **is** the organization of data as a blueprint of how the **database** is constructed .The formal definition of a **database schema** is a set of formulas (sentences) called integrity constraints imposed on a **database .** Schema is such that describes the structural read of a information that confirms the tables that will be concerned in making a information, the table's attributes and their association



4. **DBMS**

Database Management System: The software which is used to manage databases is called Database Management System (DBMS). For Example, MySQL, Oracle, etc. are popular commercial DBMS used in different applications. DBMS allows users the following tasks:

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

5. **RDBMS**

RDBMS stands for **Relational DataBase Management Systems**. It is basically a program that allows us to create, delete, and update a relational

database. Relational Database is a database system that stores and retrieves data in a tabular format organized in the form of rows and columns. It is a smaller subset of DBMS which was designed by E.F Codd in the 1970s. The major DBMS like SQL, My-SQL, ORACLE are all based on the principles of relational DBMS.

Relational Database Management Systems maintains data integrity by simulating the following features:

- **Entity Integrity:** No two records of the database table can be completely duplicate.
- **Referential Integrity:** Only the rows of those tables can be deleted which are not used by other tables. Otherwise, it may lead to data inconsistency.
- **User-defined Integrity:** Rules defined by the users based on confidentiality and access.
- **Domain integrity:** The columns of the database tables are enclosed within some structured limits, based on default values, type of data or ranges.

6. **NORMALIZATION**

Database normalization is the process of organizing the attributes of the database to reduce or eliminate **data redundancy (having the same data but at different places)** . Data redundancy unnecessarily increases the size of the database as the same data is repeated in many places. Inconsistency problems also arise during insert, delete and update operations.

Normalization is the process of minimizing redundancy from a relation or set of relations. Redundancy in relation may cause insertion, deletion, and update anomalies. So, it helps to minimize the redundancy in relations. Normal forms are used to eliminate or reduce redundancy in database tables.

1. First Normal Form –A relation is in first normal form if every attribute in that relation is **singled valued attribute**.

- **Example 1 –** Relation STUDENT in table 1 is not in 1NF because of multi-valued attribute STUD_PHONE. Its decomposition into 1NF has been shown in table 2.

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721, 9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 1

Conversion to first normal form

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY
1	RAM	9716271721	HARYANA	INDIA
1	RAM	9871717178	HARYANA	INDIA
2	RAM	9898297281	PUNJAB	INDIA
3	SURESH		PUNJAB	INDIA

Table 2

- **Example 2 –**
-
- ID Name Courses
- ----------
- 1 A c1, c2
- 2 E c3
- 3 M C2, c3

In the above table Course is a multi-valued attribute so it is not in 1NF.

Below Table is in 1NF as there is no multi-valued attribute

ID Name Course

---

1 A c1

1 A c2

2 E c3

3 M c2

3 M c3

2. Second Normal Form A relation must be in first normal form and relation must not contain any partial dependency. A relation is in 2NF if it has **No Partial Dependency,** i.e., no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Partial Dependency – If the proper subset of candidate key determines non-prime attribute, it is called partial dependency.

- **Example 1 –** Consider table-3 as following below.

STUD_NO	COURSE_NO	COURSE_FEE
1	C1	1000
2	C2	1500
1	C4	2000
4	C3	1000
4	C1	1000
2	C5	2000

{Note that, there are many courses having the same course fee. }

Here,
COURSE_FEE cannot alone decide the value of COURSE_NO or STUD_NO;
COURSE_FEE together with STUD_NO cannot decide the value of COURSE_NO;
COURSE_FEE together with COURSE_NO cannot decide the value of STUD_NO;
Hence,
COURSE_FEE would be a non-prime attribute, as it does not belong to the one only candidate key {STUD_NO, COURSE_NO} ;
But, COURSE_NO -> COURSE_FEE, i.e., COURSE_FEE is dependent on COURSE_NO, which is a proper subset of the candidate key. Non-prime attribute COURSE_FEE is dependent on a proper subset of the candidate key, which is a partial dependency and so this relation is not in 2NF.

To convert the above relation to 2NF,
we need to split the table into two tables such as :
Table 1: STUD_NO, COURSE_NO
Table 2: COURSE_NO, COURSE_FEE

Table 1 **Table 2**

STUD_NO	COURSE_NO	COURSE_NO	COURSE_FEE
1	C1	C1	1000
2	C2	C2	1500
1	C4	C3	1000
4	C3	C4	2000
4	C1	C5	2000
2	C5		

NOTE: 2NF tries to reduce the redundant data getting stored in memory. For instance, if there are 100 students taking C1 course, we don't need to store its Fee as 1000 for all the 100 records, instead, once we can store it in the second table as the course fee for C1 is 1000.

- **Example 2 –** Consider following functional dependencies in relation R (A, B , C, D)
- AB -> C [A and B together determine C]

  BC -> D [B and C together determine D]

  In the above relation, AB is the only candidate key and there is no partial dependency, i.e., any proper subset of AB doesn't determine any non-prime attribute.

3. Third Normal Form –

A relation is in third normal form, if there is **no transitive dependency** for non-prime attributes as well as it is in second normal form. A relation is in 3NF if **at least one of the following condition holds** in every non-trivial function dependency X –> Y

1. X is a super key.
2. Y is a prime attribute (each element of Y is part of some candidate key).

STUD_NO	STUD_NAME	STUD_STATE	STUD_COUNTRY	STUD_AGE
1	RAM	HARYANA	INDIA	20
2	RAM	PUNJAB	INDIA	19
3	SURESH	PUNJAB	INDIA	21

Table 4

Transitive dependency – If A->B and B->C are two FDs then A->C is called transitive dependency.

3. **Example 1 –** In relation STUDENT given in Table 4,
   FD set: {STUD_NO -> STUD_NAME, STUD_NO -> STUD_STATE, STUD_STATE -> STUD_COUNTRY,

STUD_NO -> STUD_AGE}
Candidate Key: {STUD_NO}

For this relation in table 4, STUD_NO -> STUD_STATE and STUD_STATE -> STUD_COUNTRY are true. So STUD_COUNTRY is transitively dependent on STUD_NO. It violates the third normal form. To convert it in third normal form, we will decompose the relation STUDENT (STUD_NO, STUD_NAME, STUD_PHONE, STUD_STATE, STUD_COUNTRY_STUD_AGE) as:
STUDENT (STUD_NO, STUD_NAME, STUD_PHONE, STUD_STATE, STUD_AGE)
STATE_COUNTRY (STATE, COUNTRY)

4. **Example 2 –** Consider relation R(A, B, C, D, E)
A -> BC,
CD -> E,
B -> D,
E -> A
All possible candidate keys in above relation are {A, E, CD, BC}
All attributes are on right sides of all functional dependencies are prime.

4. Boyce-Codd Normal Form (BCNF) –

A relation R is in BCNF if R is in Third Normal Form and for every FD, LHS is super key. A relation is in BCNF iff in every non-trivial functional dependency X –> Y, X is a super key.

Example 1 – Find the highest normal form of a relation R(A,B,C,D,E) with FD set as {BC->D, AC->BE, B->E}
Step 1. As we can see, (AC)+ ={A,C,B,E,D} but none of its subset can determine all attribute of relation, So AC will be candidate key. A or C can't be derived from any other attribute of the relation, so there will be only 1 candidate key {AC}.
Step 2. Prime attributes are those attributes that are part of candidate key {A, C} in this example and others will be non-prime {B, D, E} in this example.
Step 3. The relation R is in 1st normal form as a relational DBMS does not allow multi-valued or composite attribute.
The relation is in 2nd normal form because BC->D is in 2nd normal form (BC is not a proper subset of candidate key AC) and AC->BE is in 2nd normal form (AC is candidate key) and B->E is in 2nd normal form (B is not a proper subset of candidate key AC).
The relation is not in 3rd normal form because in BC->D (neither BC is a super key nor D is a prime attribute) and in B->E (neither B is a super key nor E is a prime

attribute) but to satisfy 3rd normal for, either LHS of an FD should be super key or RHS should be prime attribute.
So the highest normal form of relation will be 2nd Normal form.

1. **Example 2** –For example consider relation R(A, B, C)
   A -> BC,
   B ->
   A and B both are super keys so above relation is in BCNF.

Key Points –

2. BCNF is free from redundancy.
3. If a relation is in BCNF, then 3NF is also satisfied.
4. If all attributes of relation are prime attribute, then the relation is always in 3NF.
5. A relation in a Relational Database is always and at least in 1NF form.
6. Every Binary Relation (a Relation with only 2 attributes) is always in BCNF.
7. If a Relation has only singleton candidate keys(i.e. every candidate key consists of only 1 attribute), then the Relation is always in 2NF(because no Partial functional dependency possible).
8. Sometimes going for BCNF form may not preserve functional dependency. In that case go for BCNF only if the lost FD(s) is not required, else normalize till 3NF only.
9. There are many more Normal forms that exist after BCNF, like 4NF and more. But in real world database systems it's generally not required to go beyond BCNF.

Exercise 1: Find the highest normal form in R (A, B, C, D, E) under following functional dependencies.

ABC --> D

CD --> AE

Important Points for solving above type of question.
1) It is always a good idea to start checking from BCNF, then 3 NF, and so on.
2) If any functional dependency satisfied a normal form then there is no need to check for lower normal form. For example, ABC –> D is in BCNF (Note that ABC is a superkey), so no need to check this dependency for lower normal forms.
Candidate keys in the given relation are {ABC, BCD}

BCNF: ABC -> D is in BCNF. Let us check CD -> AE, CD is not a super key so this dependency is not in BCNF. So, R is not in BCNF.

3NF: ABC -> D we don't need to check for this dependency as it already satisfied BCNF. Let us consider CD -> AE. Since E is not a prime attribute, so the relation is not in 3NF.
2NF: In 2NF, we need to check for partial dependency. CD is a proper subset of a candidate key and it determines E, which is non-prime attribute. So, given relation is also not in 2 NF. So, the highest normal form is 1 NF.

7. DDL,DML, DCL

DDL or Data Definition Language consists of the SQL commands used to define the database schema DDL is used to create, modify, and delete database structures but not data.
List of DDL commands:

- **CREATE**: This command is used to create the database or its objects (like table, index, function, views, store procedure, and triggers).
- **DROP**: This command is used to delete objects from the database.
- **ALTER:** This is used to alter the structure of the database.
- **TRUNCATE:** This is used to remove all records from a table, including all spaces allocated for the records are removed.
- **COMMENT**: This is used to add comments to the data dictionary.
- **RENAME:** This is used to rename an object existing in the database.

DML(Data Manipulation Language):
The SQL commands deals with the manipulation of data present in the database and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.

List of DML commands:

- **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 records from a database table.
- **LOCK:** Table control concurrency.
- **CALL:** Call a PL/SQL or JAVA subprogram.
- **EXPLAIN PLAN:** It describes the access path to data.

DCL (Data Control Language):
DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

List of DCL commands:

- **GRANT:** This command gives users access privileges to the database.

- **REVOKE:** This command withdraws the user's access privileges given by using the GRANT command.

8, FIELD , RECORD TUPLE AND TABLE

A field is single unit of data that is unique within each entry/row, but the overall data category is common to all entries. For instance, "address" is a field that is common to all named entries in the phone book; however, the content of the address field will be unique for each named entry. As a guideline, a database field refers to the columns, or data categories, that are used by all entries/rows.

A record is a row of a database, a horizontal grouping of fields; the content of those fields is unique to that row. In the phone book example, each last name begins a record/row which contains data in the name, address and phone number fields. A record is often used in a database search because each record has a unique, identifying quality (or value). A database record is, basically, a row that contains unique data in each of the fields. A database will usually contain a large number of records but only a small number of fields.

A tuple is one record (one row)

In a relational database, a tuple contains all the data for an individual record. For example, in a database containing client contact information, the fields may be categories such as name, phone number, email address and mailing address, while a tuple for that database could be:

Bill Gates	206-555-1234	billg@microsoft.com	PO Box 123, Seattle, WA 98100

a *table* is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect.[1] A table has a specified number of columns, but can have any number of rows.[2] Each row is identified by one or more values appearing in a particular column subset. A specific choice of columns which uniquely identify rows is called the primary key.

9 . ER MODEL

ER Model is used to model the logical view of the system from data perspective which consists of these components: **Entity, Entity Type, Entity Set**

10 ATTRIBUTES AND RELATIONS

An attribute is a property or characteristic of an entity. An entity may contain any number of attributes. One of the attributes is considered as the primary key. In an Entity-Relation model, attributes are represented in an elliptical shape

Attribute(s):
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.

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.

11 DATABASE TRANSACTION

Transactions are a set of operations used to perform a logical set of work. A transaction usually means that the data in the database has changed. One of the major uses of DBMS is to protect the user's data from system failures. It is done by ensuring that all the data is restored to a consistent state when the computer is restarted after a crash. The transaction is any one execution of the user program in a DBMS. Executing the same program multiple times will generate multiple transactions

Uses of Transaction Management :

- The DBMS is used to schedule the access of data concurrently. It means that the user can access multiple data from the database without being interfered with each other. Transactions are used to manage concurrency.
- It is also used to satisfy ACID properties.
- It is used to solve Read/Write Conflict.
- It is used to implement Recoverability, Serializability, and Cascading.
- Transaction Management is also used for Concurrency Control Protocols and Locking of data.