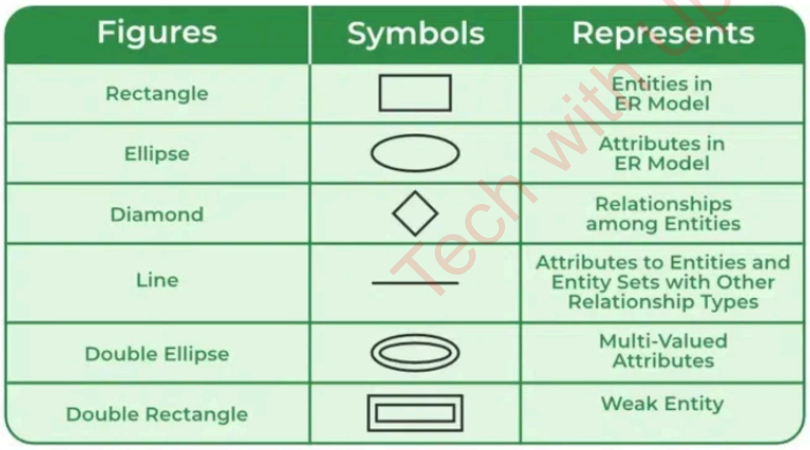
DBMS NOTES (Source - Apna College, GFG)

Database Management System is a software or technology used to manage data from a database. Some popular databases are MySQL, Oracle, MongoDB, etc.

ER MODEL

The Entity Relationship Diagram explains the **relationship among the entities** present in the database.
ER models are used to **model real-world objects like a person, a car, or a company** and the relation between these real-world objects.
In short, the ER Diagram is the **structural format of the database.**

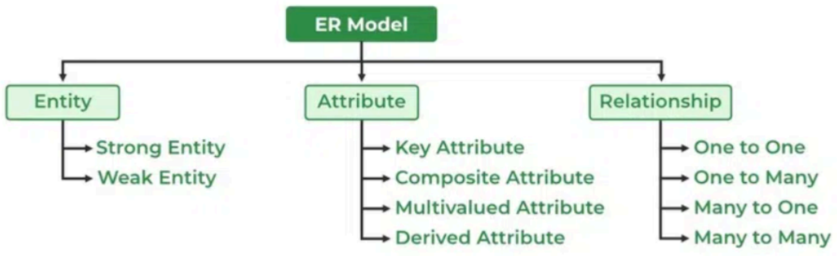
It gives a standard solution for visualizing the data logically.



Figures	Symbols	Represents
Rectangle	▭	Entities in ER Model
Ellipse	⬭	Attributes in ER Model
Diamond	◇	Relationships among Entities
Line	—	Attributes to Entities and Entity Sets with Other Relationship Types
Double Ellipse	◎	Multi-Valued Attributes
Double Rectangle	▣	Weak Entity

Components of ER Diagram

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



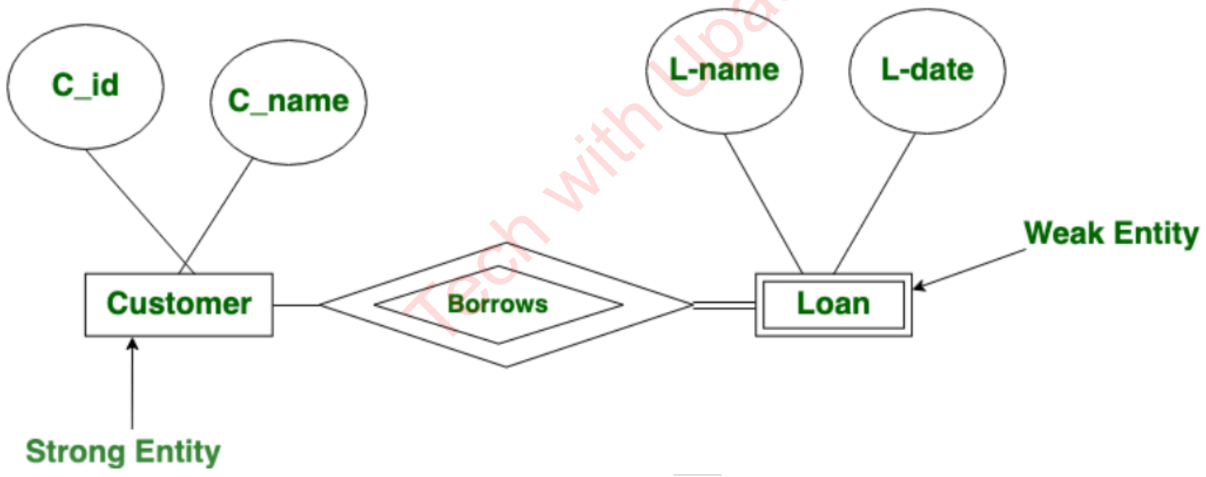
An Entity may be an object with a **physical existence** – a particular person, car, house, or employee –
or it may be an object with a **conceptual existence** – a company, a job, or a university course.

What is Strong Entity?

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.

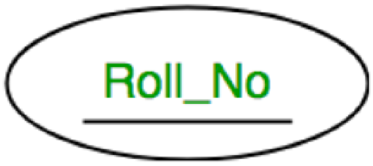
What is Weak Entity?

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 **identifying relationship.**

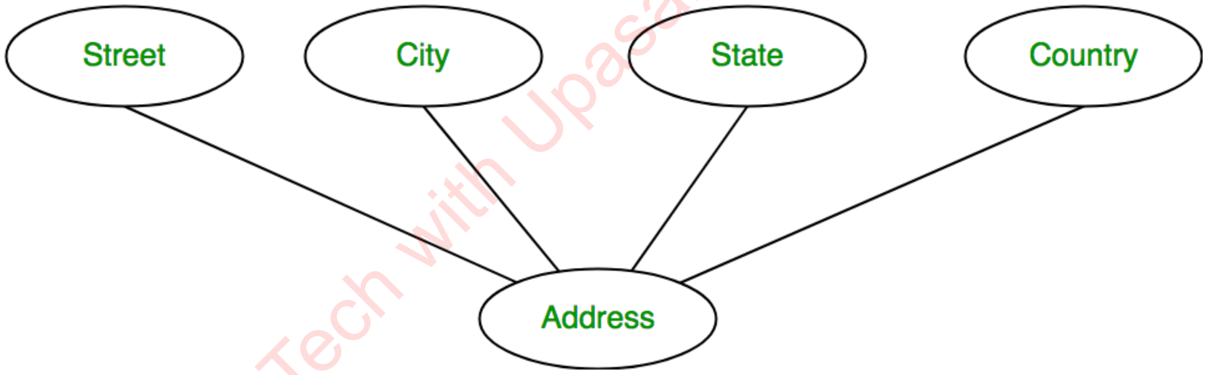


Attributes: Attributes are the properties that define the entity type. For example, Roll_No, Name, DOB, Age, Address, and Mobile_No are the attributes that define entity type Student. In ER diagram, the attribute is represented by an oval.

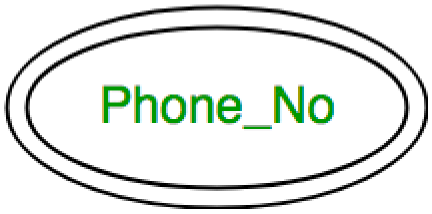
1. Key Attributes - The attribute in an entity set which is unique. Like in Student table, roll no. will be unique and hence called key attribute.



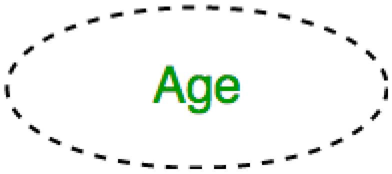
2. **Composite Attribute**
   An attribute **composed of many other attributes** is called a composite attribute. For example, the Address attribute of the student Entity type consists of Street, City, State, and Country. In ER diagram, the composite attribute is represented by an oval comprising of ovals.

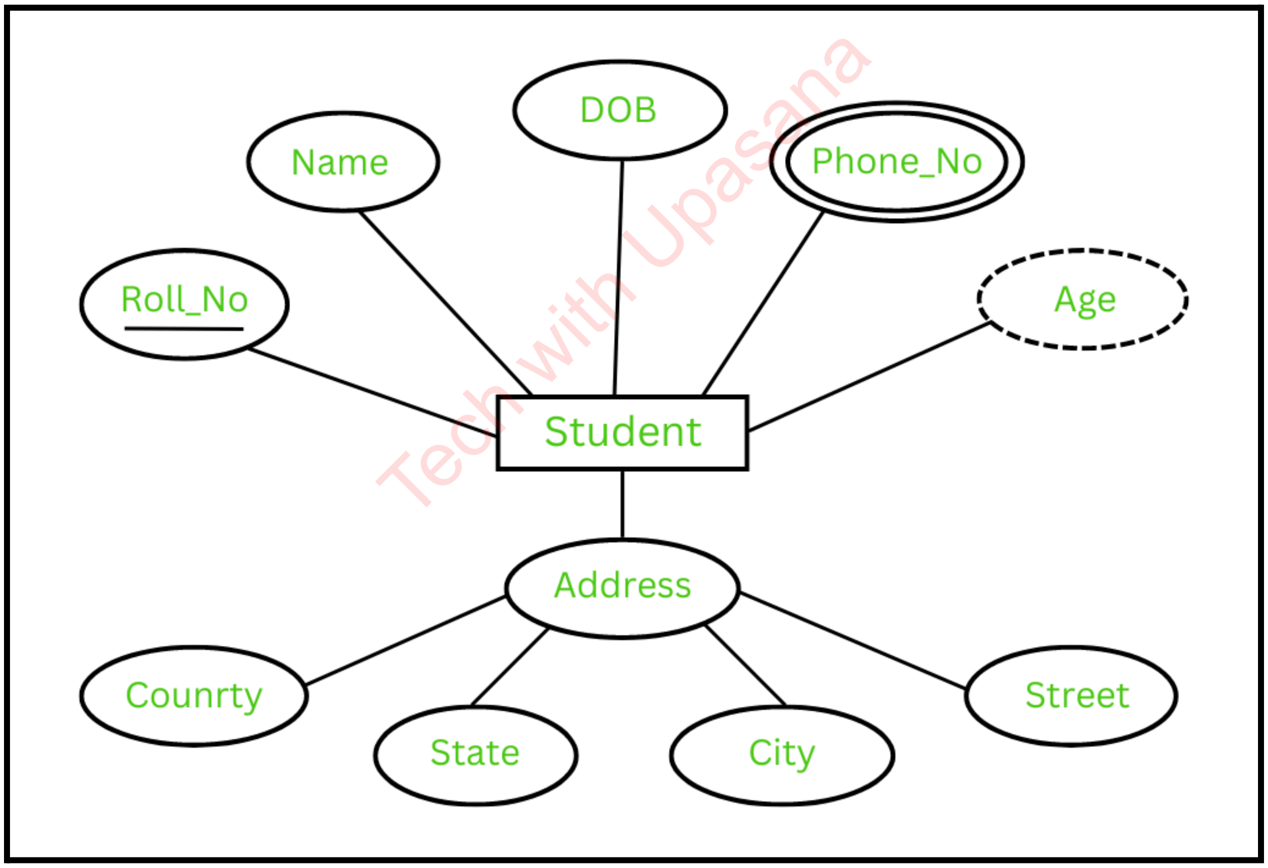


3. **Multivalued Attribute**
   An attribute consisting of more than one value for a given entity. For example, Phone_No (can be more than one for a given student). In ER diagram, a multivalued attribute is represented by a double oval.



4. **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.

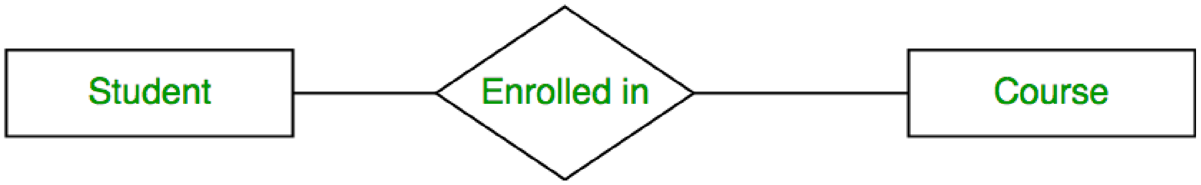




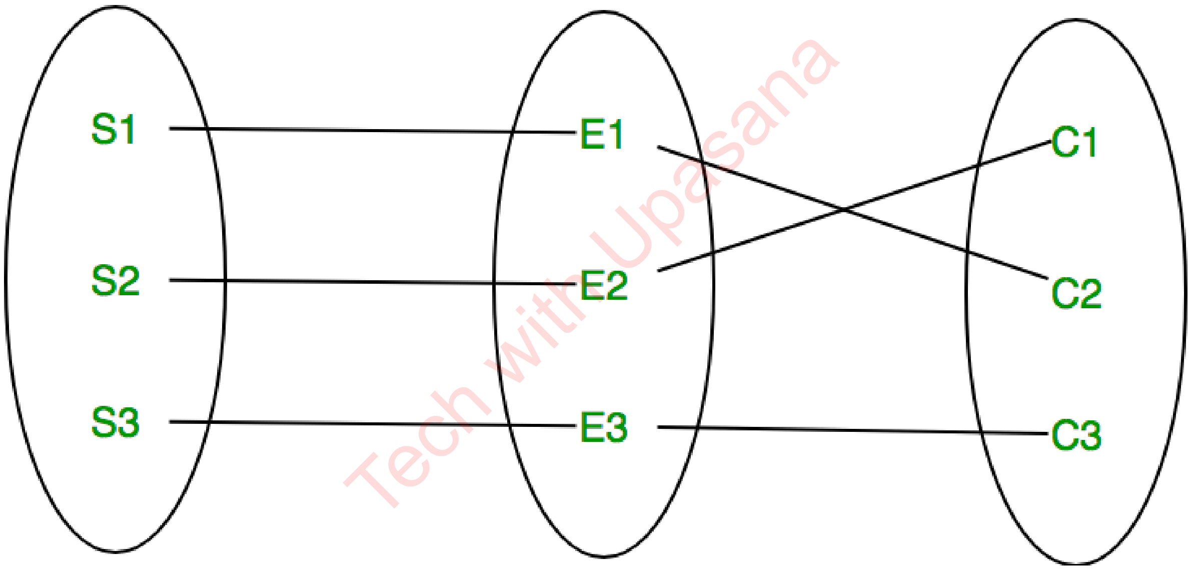
Relationship Type and Relationship Set

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



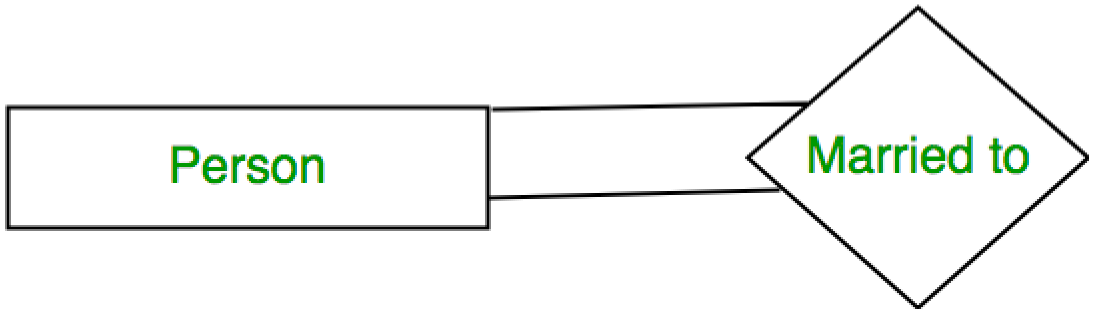
A set of relationships of the same type is known as a relationship set. The following relationship set depicts S1 as enrolled in C2, S2 as enrolled in C1, and S3 as registered in C3.



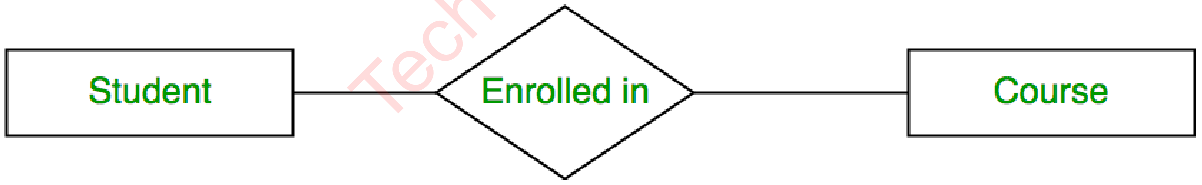
Degree of a Relationship Set

The number of different entity sets participating in a relationship set is called the 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.



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

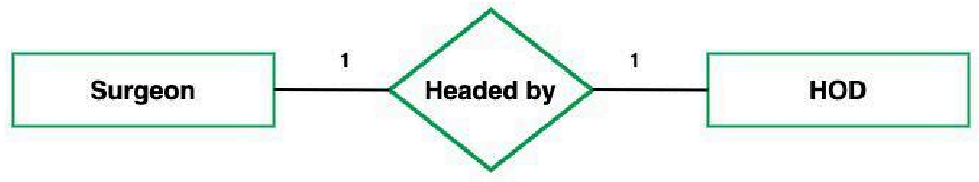
3. Ternary Relationship: When there are n entities set participating in a relation, the relationship is called an n-ary relationship.

Cardinality

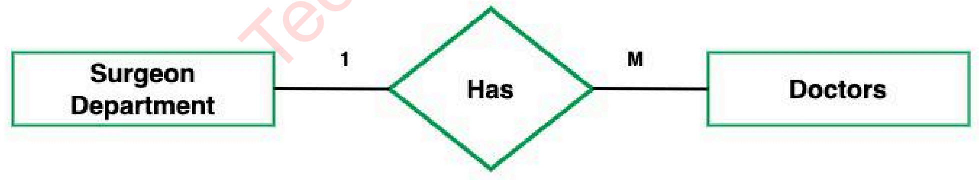
The number of times an entity of an entity set participates in a relationship set is known as cardinality.

1. One-to-One: When each entity in each entity set can take part only once in the relationship, the cardinality is one-to-one. Let us assume that a male can marry one female and a female can marry one male. So the relationship will be one-to-one.

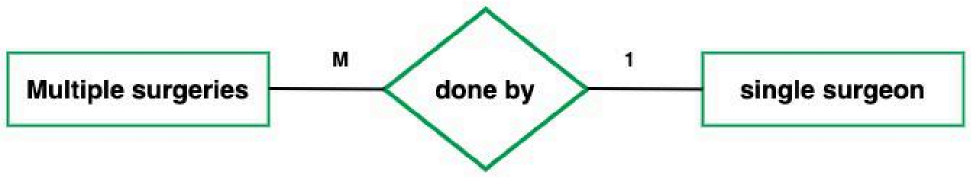
the total number of tables that can be used in this is 2.



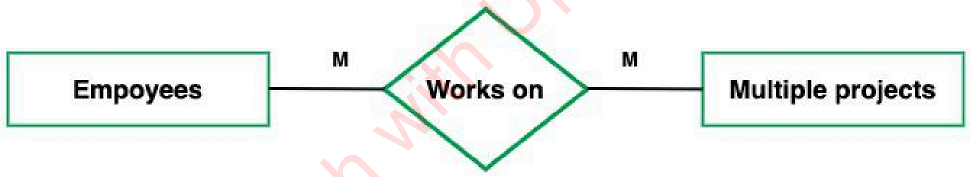
2. One-to-Many: In one-to-many mapping as well where each entity can be related to more than one entity and the total number of tables that can be used in this is 2. Let us assume that one surgeon department can accommodate many doctors. So the Cardinality will be 1 to M.



3. Many-to-One: When entities in one entity set can take part only once in the relationship set and entities in other entity sets can take part more than once in the relationship set, cardinality is many to one. Let us assume that a student can take only one course but one course can be taken by many students. So the cardinality will be n to 1.



4. Many-to-Many: When entities in all entity sets can take part more than once in the relationship cardinality is many to many. Let us assume that a student can take more than one course and one course can be taken by many students. So the relationship will be many to many.



RELATIONAL MODEL

It uses the concept of relations to represent each and every file. Relations are Two-Dimensional Tables.
It uses the primary key and secondary key to connect any two files.
Relational Algebra and Relational Calculus are used to process the relations manually.
A Relational Database Model consists of relations to connect them by key fields. A relation has some attributes.
The relation is represented in rows and columns. Each column of the relation is called an attribute.
Each row in the relation is called a tuple. Each relation can have one unique column i.e. primary key.

Instances

An Instance is the state of an operational database with data at any given time. It contains a snapshot of the database.

Example:
Let’s say a table teacher in our database whose name is School, suppose the table has 50 records so the instance of the database has 50 records for now and tomorrow we are going to add another fifty records so tomorrow the instance has a total of 100 records. This is called an instance.

Schema

Schema is the overall description of the database. The schema is same for the whole database.

Schema is of three types: Logical Schema, Physical Schema and view Schema.

- **Logical Schema** – It describes the database designed at a logical level.
- **Physical Schema** – It describes the database designed at the physical level.
- **View Schema** – It defines the design of the database at the view level.

RDBMS Vendors

There are several vendors that offer Relational Database Management Systems (RDBMS).

Microsoft SQL Server, IBM DB2, Oracle

Relational Algebra

It is a procedural Language. It consists of a set of operators that can be performed on relations.

Union, Intersection, MINUS, Cartesian Product

SELECTION - Select particular data from db (denoted by sigma)

PROJECTION - Displays the elected columns (denoted by pi)

JOINS

DIVIDES

RENAME - p(old relation, new relation)

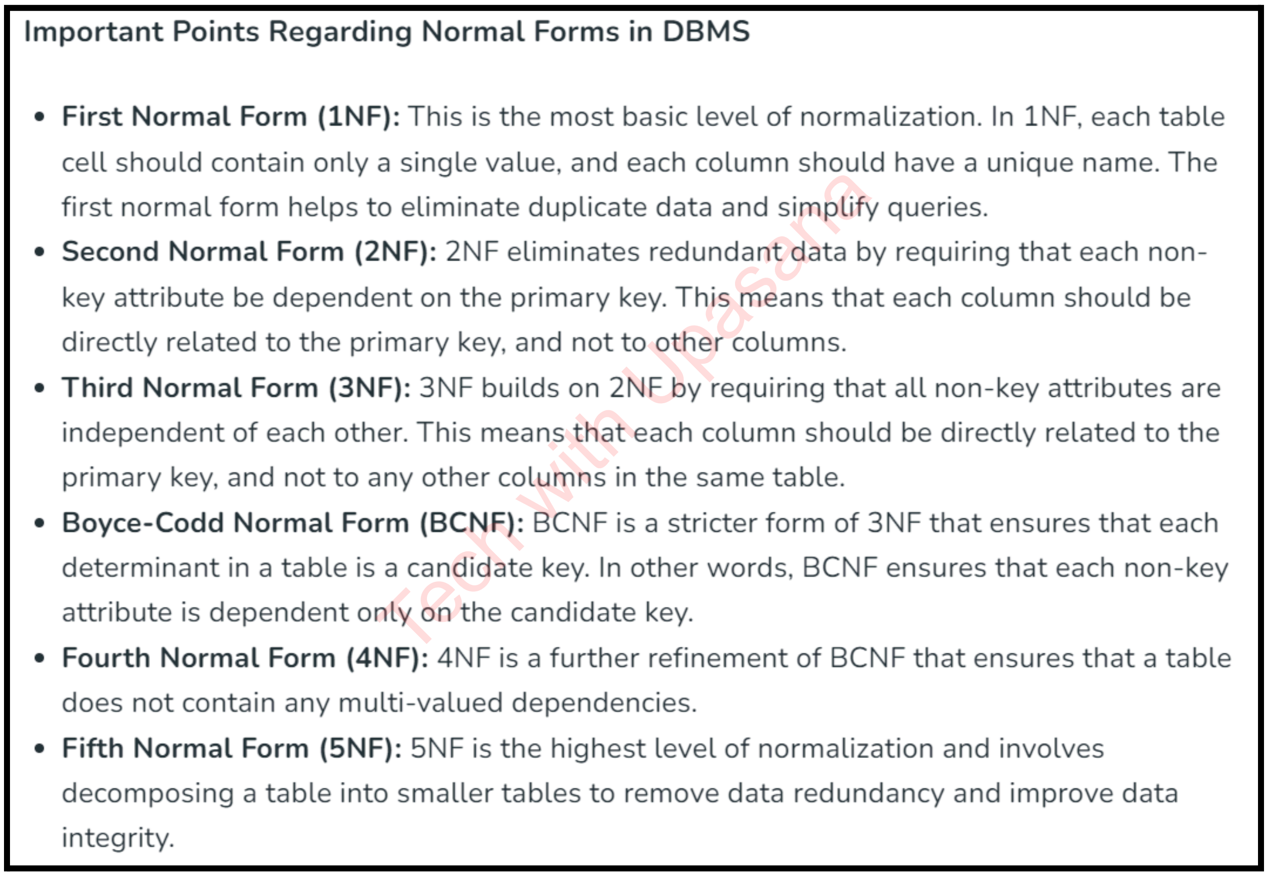
SQL Joins (Inner, Left, Right and Full Join)

SQL Join operation combines data or rows from two or more tables based on a common field between them.

It can access data from multiple tables simultaneously using common key values shared across different tables.

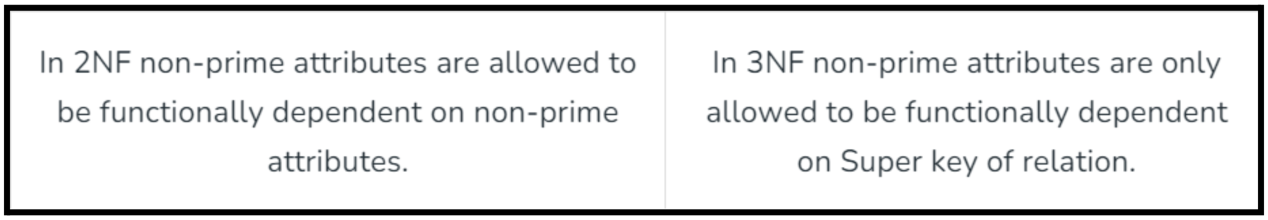
Normalization

Normalization is the process of minimizing **redundancy** from a relation or set of relations. Redundancy in relation may cause insertion, deletion, and update anomalies



Important Points Regarding Normal Forms in DBMS

- **First Normal Form (1NF):** This is the most basic level of normalization. In 1NF, each table cell should contain only a single value, and each column should have a unique name. The first normal form helps to eliminate duplicate data and simplify queries.
- **Second Normal Form (2NF):** 2NF eliminates redundant data by requiring that each non-key attribute be dependent on the primary key. This means that each column should be directly related to the primary key, and not to other columns.
- **Third Normal Form (3NF):** 3NF builds on 2NF by requiring that all non-key attributes are independent of each other. This means that each column should be directly related to the primary key, and not to any other columns in the same table.
- **Boyce-Codd Normal Form (BCNF):** BCNF is a stricter form of 3NF that ensures that each determinant in a table is a candidate key. In other words, BCNF ensures that each non-key attribute is dependent only on the candidate key.
- **Fourth Normal Form (4NF):** 4NF is a further refinement of BCNF that ensures that a table does not contain any multi-valued dependencies.
- **Fifth Normal Form (5NF):** 5NF is the highest level of normalization and involves decomposing a table into smaller tables to remove data redundancy and improve data integrity.



In 2NF non-prime attributes are allowed to be functionally dependent on non-prime attributes.	In 3NF non-prime attributes are only allowed to be functionally dependent on Super key of relation.

For a dependency A -> B, if for a single value of A, multiple values of B exist, then the table may have a multi-valued dependency.

Example: Consider the database table of a class that has two relations R1 contains student ID(SID) and student name (SNAME) and R2 contains course id(CID) and course name (CNAME).

Table R1

SID	SNAME
S1	A
S2	B

Table R2

CID	CNAME
C1	C
C2	D

When their cross-product is done it resulted in multivalued dependencies.

Table R1 X R2

SID	SNAME	CID	CNAME
S1	A	C1	C
S1	A	C2	D
S2	B	C1	C
S2	B	C2	D

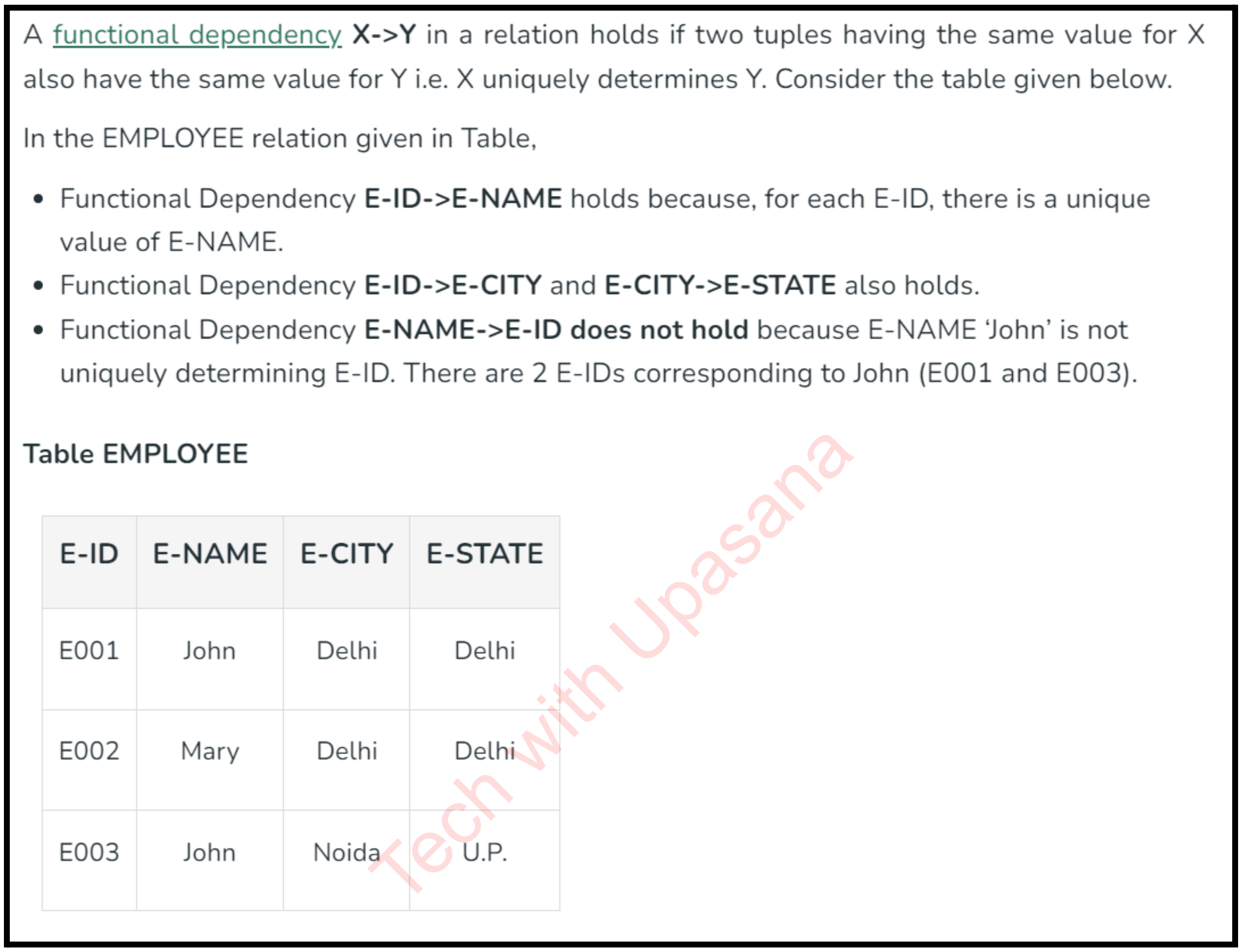
Multivalued dependencies (MVD) are:

`SID->->CID; SID->->CNAME; SNAME->->CNAME`

Functional dependency, join dependency, find no. of candidate keys, dependency preserving mechanism,

Functional Dependency

A functional dependency A->B in a relation holds if two tuples having the same value of attribute A also have the same value for attribute B.



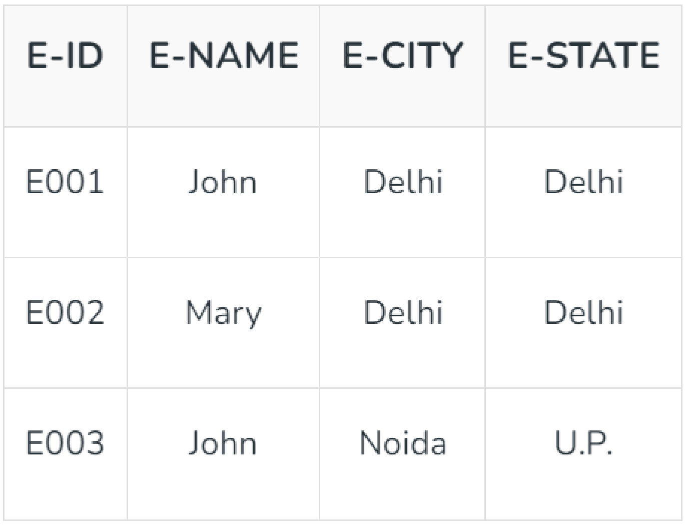
A functional dependency **X->Y** in a relation holds if two tuples having the same value for X also have the same value for Y i.e. X uniquely determines Y. Consider the table given below.

In the EMPLOYEE relation given in Table,

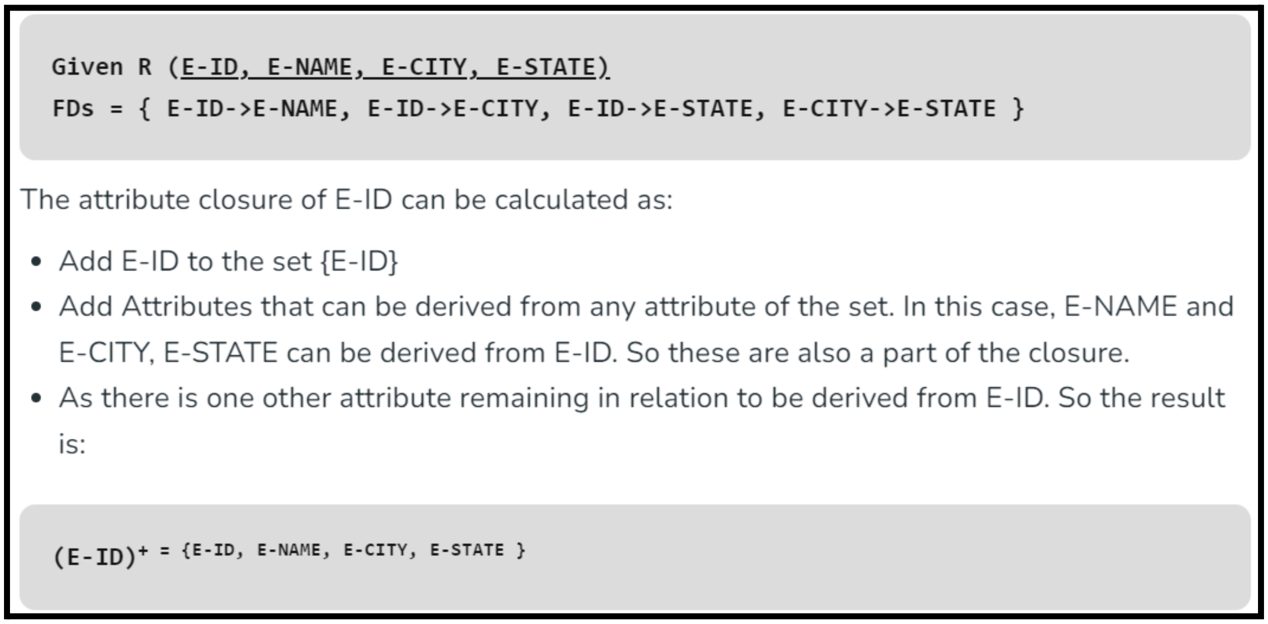
- Functional Dependency **E-ID->E-NAME** holds because, for each E-ID, there is a unique value of E-NAME.
- Functional Dependency **E-ID->E-CITY** and **E-CITY->E-STATE** also holds.
- Functional Dependency **E-NAME->E-ID does not hold** because E-NAME 'John' is not uniquely determining E-ID. There are 2 E-IDs corresponding to John (E001 and E003).

Table EMPLOYEE

E-ID	E-NAME	E-CITY	E-STATE
E001	John	Delhi	Delhi
E002	Mary	Delhi	Delhi
E003	John	Noida	U.P.



E-ID	E-NAME	E-CITY	E-STATE
E001	John	Delhi	Delhi
E002	Mary	Delhi	Delhi
E003	John	Noida	U.P.

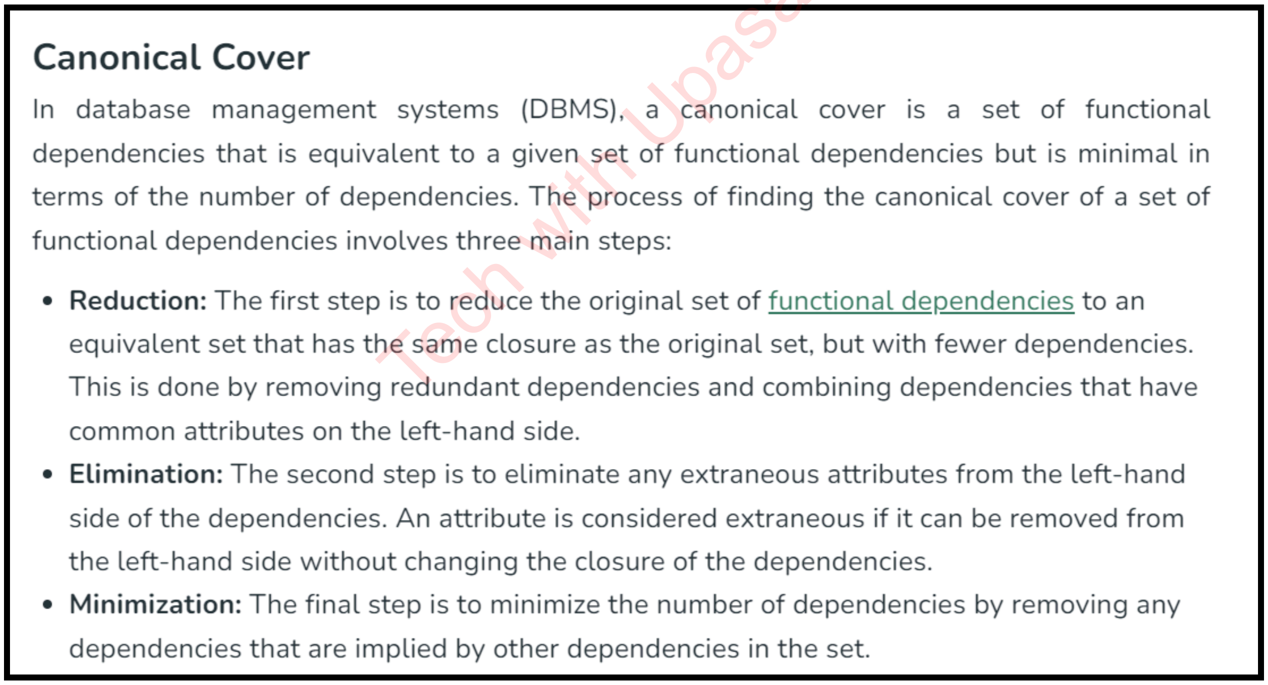


```
Given R (E-ID, E-NAME, E-CITY, E-STATE)
FDs = { E-ID->E-NAME, E-ID->E-CITY, E-ID->E-STATE, E-CITY->E-STATE }
```

The attribute closure of E-ID can be calculated as:

- Add E-ID to the set {E-ID}
- Add Attributes that can be derived from any attribute of the set. In this case, E-NAME and E-CITY, E-STATE can be derived from E-ID. So these are also a part of the closure.
- As there is one other attribute remaining in relation to be derived from E-ID. So the result is:

```
(E-ID)+ = {E-ID, E-NAME, E-CITY, E-STATE }
```



Canonical Cover

In database management systems (DBMS), a canonical cover is a set of functional dependencies that is equivalent to a given set of functional dependencies but is minimal in terms of the number of dependencies. The process of finding the canonical cover of a set of functional dependencies involves three main steps:

- **Reduction:** The first step is to reduce the original set of functional dependencies to an equivalent set that has the same closure as the original set, but with fewer dependencies. This is done by removing redundant dependencies and combining dependencies that have common attributes on the left-hand side.
- **Elimination:** The second step is to eliminate any extraneous attributes from the left-hand side of the dependencies. An attribute is considered extraneous if it can be removed from the left-hand side without changing the closure of the dependencies.
- **Minimization:** The final step is to minimize the number of dependencies by removing any dependencies that are implied by other dependencies in the set.

Concurrency Control is the management procedure that is required for controlling concurrent execution of the operations that take place on a database.

Concurrent Execution in DBMS

- In a multi-user system, multiple users can access and use the same database at one time, which is known as the concurrent execution of the database. It means that the same database is executed simultaneously on a multi-user system by different users.

In a database transaction, the two main operations are **READ** and **WRITE** operations. So, there is a need to manage these two operations in the concurrent execution of the transactions as if these operations are not performed in an interleaved manner, and the data may become inconsistent.

Concurrency Control

Concurrency Control is the working concept that is required for controlling and managing the concurrent execution of database operations and thus avoiding the inconsistencies in the database.

Concurrency Control Protocols

The concurrency control protocols ensure the *atomicity, consistency, isolation, durability* and *serializability* of the concurrent execution of the database transactions. Therefore, these protocols are categorized as:

- Lock Based Concurrency Control Protocol
- Time Stamp Concurrency Control Protocol
- Validation Based Concurrency Control Protocol

Lock-Based Protocol

In this type of protocol, any transaction cannot read or write data until it acquires an appropriate lock on it. There are two types of lock:

1. Shared lock: It is also known as a Read-only lock. In a shared lock, the data item can only read by the transaction.

- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

2. Exclusive lock:

- In the exclusive lock, the data item can be both reads as well as written by the transaction.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

The Timestamp Ordering Protocol is used to order the transactions based on their Timestamps. The order of transaction is nothing but the ascending order of the transaction creation.

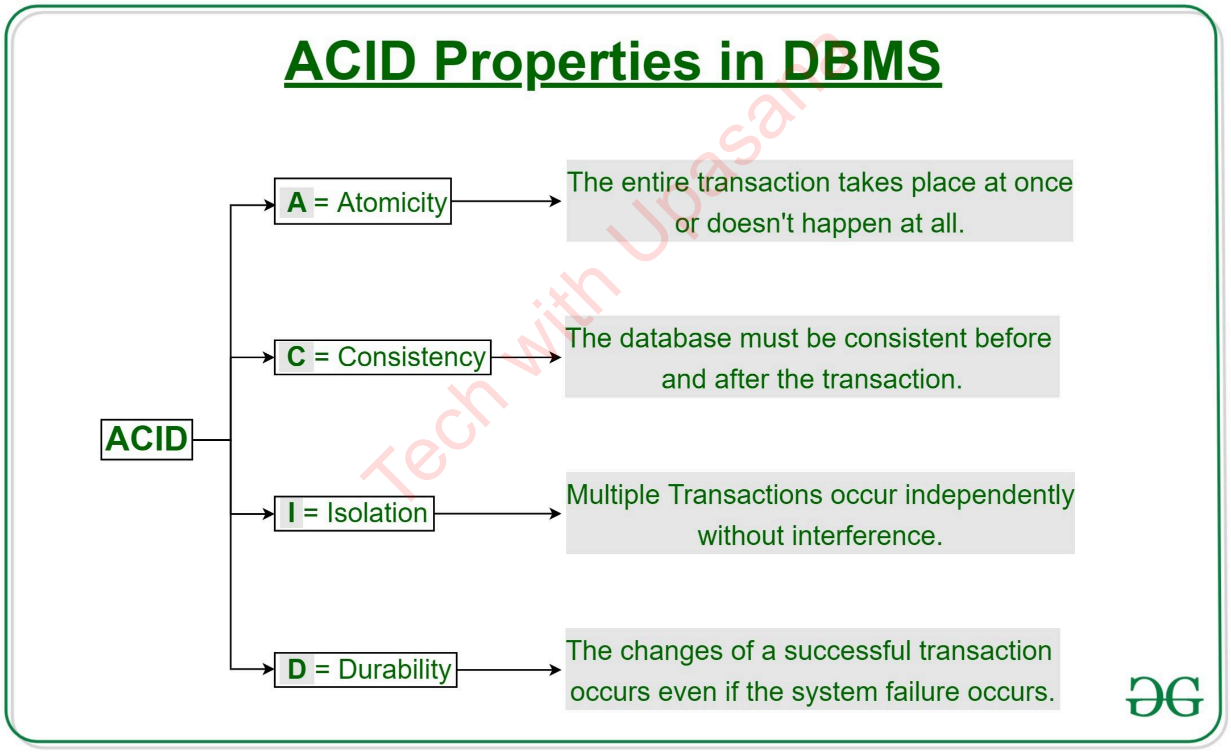
Validation phase is also known as optimistic concurrency control technique. In the validation based protocol, the transaction is executed in the following three phases:

1. **Read phase:** In this phase, the transaction T is read and executed. It is used to read the value of various data items and stores them in temporary local variables. It can perform all the write operations on temporary variables without an update to the actual database.
2. **Validation phase:** In this phase, the temporary variable value will be validated against the actual data to see if it violates the serializability.
3. **Write phase:** If the validation of the transaction is validated, then the temporary results are written to the database or system otherwise the transaction is rolled back.

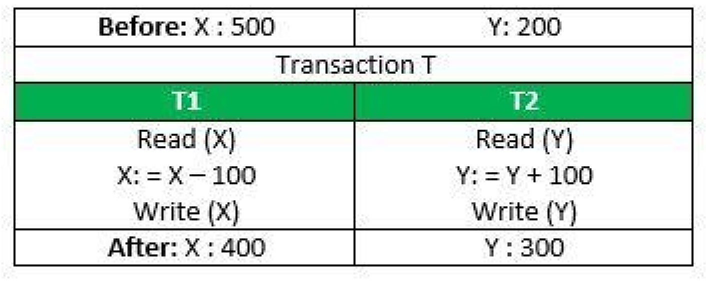
ACID PROPERTIES

A **transaction** is a single logical unit of work that accesses and possibly modifies the contents of a database. Transactions access data using read and write operations.

In order to maintain consistency in a database, before and after the transaction, certain properties are followed. These are called **ACID** properties.


ACID Properties in DBMS
ACID
A = Atomicity
The entire transaction takes place at once or doesn't happen at all.
C = Consistency
The database must be consistent before and after the transaction.
I = Isolation
Multiple Transactions occur independently without interference.
D = Durability
The changes of a successful transaction occurs even if the system failure occurs.
Tech with Upasana

Consider the following transaction **T** consisting of **T1** and **T2** : Transfer of 100 from account **X** to account **Y** .



Before: X : 500	Y: 200
Transaction T	
T1	T2
Read (X) X: = X − 100 Write (X)	Read (Y) Y: = Y + 100 Write (Y)
After: X : 400	Y : 300

If the transaction fails after completion of **T1** but before completion of **T2** .(say, after **write(X)** but before **write(Y)**), then the amount has been deducted from **X** but not added to **Y** . This results in an inconsistent database state. Therefore, the transaction must be executed in its entirety in order to ensure the correctness of the database state.

Consistency:

This means that integrity constraints must be maintained so that the database is consistent before and after the transaction. It refers to the correctness of a database. Referring to the example above,
The total amount before and after the transaction must be maintained.
Total **before T** occurs = **500 + 200 = 700** .
Total **after T occurs** = **400 + 300 = 700** .
Therefore, the database is **consistent** . Inconsistency occurs in case **T1** completes but **T2** fails. As a result, T is incomplete.

Durability:

This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs.

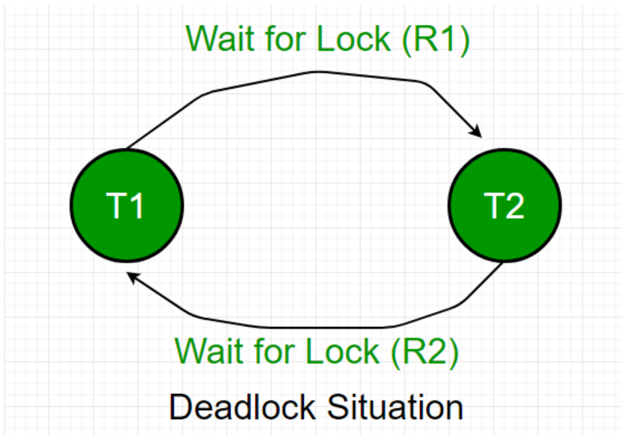
Deadlock

In a database management system (DBMS), a deadlock occurs when two or more transactions are waiting for each other to release resources, such as locks on database objects, that they need to complete their operations.

Deadlock Avoidance: When a database is stuck in a deadlock, It is always better to avoid the deadlock rather than restarting or aborting the database. The deadlock avoidance method is suitable for smaller databases whereas the deadlock prevention method is suitable for larger databases.

Wait-for-graph is one of the methods for detecting the deadlock situation. This method is suitable for smaller databases. In this method, a graph is drawn based on the transaction and its lock on the resource. If the graph created has a closed loop or a cycle, then there is a deadlock.

For the above-mentioned scenario, the Wait-For graph is drawn below:



Deadlock prevention: For a large database, the deadlock prevention method is suitable. A deadlock can be prevented if the resources are allocated in such a way that a deadlock never occurs. The DBMS analyzes the operations whether they can create a deadlock situation or not, If they do, that transaction is never allowed to be executed.

RAID (Redundant Arrays of Independent Disks)

RAID (Redundant Arrays of Independent Disks) is a technique that makes use of a combination of multiple disks for storing the data instead of using a single disk for increased performance, data redundancy, or to protect data in the case of a drive failure.

RAID (Redundant Array of Independent Disks) is like having backup copies of your important files stored in different places on several hard drives or solid-state drives (SSDs). If one drive stops working, your data is still safe because you have other copies stored on the other drives.

What is a RAID Controller?

A RAID controller is like a boss for your hard drives in a big storage system. It works between your computer's operating system and the actual hard drives, organizing them into groups to make them easier to manage. This helps speed up how fast your computer can read and write data, and it also adds a layer of protection in case one of your hard drives breaks down.

Category	OLAP (Online Analytical Processing)	OLTP (Online Transaction Processing)
Definition	It is well-known as an online database query management system.	It is well-known as an online database modifying system.
Data source	Consists of historical data from various Databases.	Consists of only operational current data.
Method used	It makes use of a data warehouse.	It makes use of a standard database management system (DBMS).
Application	It is subject-oriented. Used for Data Mining, Analytics, Decisions making, etc.	It is application-oriented. Used for business tasks.
Normalized	In an OLAP database, tables are not normalized.	In an OLTP database, tables are normalized (3NF).
Usage of data	The data is used in planning, problem-solving, and decision-making.	The data is used to perform day-to-day fundamental operations.

Some more topics you can explore from Geeks for Geeks -
Paging, Segmentation, Indexing, etc....