DBMS

3-tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.

- **Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
- **Application (Middle) Tier** − At this tier reside the application server and the programs that access the database.
- **User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer

---

➢ **Normalisation**

Database normalization is the process of organizing the attributes of database to reduce or eliminate **data redundancy (having same data but at different places)** .

Problems because of data redundancy

Data redundancy unnecessarily increases size of database as same data is repeated on many places. Inconsistency problems also arise during insert, delete and update operations.

➢ **Functional Dependency**

Functional Dependency is a constraint between two sets of attributes in a relation from a database. Functional dependency is denoted by arrow (→). If an attributed A functionally determines B, then it is written as A → B.

Trivial Functional Dependency

X –> Y is trivial only when Y is subset of X.

Non Trivial Functional Dependencies

X –> Y is a non trivial functional dependencies when Y is not a subset of X.

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

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 –

If a relation contain composite or multi-valued attribute, it violates first normal form or a relation is in first normal form if it does not contain any composite or multi-valued attribute. A relation is in first normal form if every attribute in that relation is **singled valued attribute**.

2. Second Normal Form –

be **in first normal form and relation must not contain any partial dependency**. **Partial Dependency –** If proper subset of candidate key determines non-prime attribute, it is called partial dependency.

3. Third Normal Form –

A relation is in third normal form, if there is **no transitive dependency** for non-prime attributes is it is in second normal form.

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.

- **ACID Properties in DBMS**

Atomicity
By this, we mean that either the entire transaction takes place at once or doesn't happen at all.

—**Abort**: If a transaction aborts, changes made to database are not visible.
—**Commit**: If a transaction commits, changes made are visible.
Atomicity is also known as the 'All or nothing rule'.

Consistency
This means that integrity constraints must be maintained so that the database is consistent before and after the transaction. It refers to correctness of a database.

Isolation
Transactions occur independently without interference. Changes occurring in a particular transaction will **not be visible to any other transaction** until that particular change in that transaction is written to memory or has been committed.

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 is system failure occurs**.

- **What is a Schedule?**

A schedule is series of operations from one or more transactions. A schedule can be of two types:

Serial Schedule: When one transaction completely executes before starting another transaction, the schedule is called serial schedule. A serial schedule is always consistent.

Concurrent Schedule: When operations of a transaction are interleaved with operations of other transactions of a schedule, the schedule is called Concurrent schedule.

Conflict Serializable: A schedule is called conflict serializable if it can be transformed into a serial schedule by swapping non-conflicting operations.
Conflicting operations: Two operations are said to be conflicting if all conditions satisfy:

- They belong to different transaction
- They operation on same data item
- At Least one of them is a write operation

- **Keys**

Candidate Key: The minimal set of attribute which can uniquely identify a tuple is known as candidate key.

Super Key: The set of attributes which can uniquely identify a tuple is known as Super Key. For Example, STUD_NO, (STUD_NO, STUD_NAME) etc.

- Adding zero or more attributes to candidate key generates super key.
- A candidate key is a super key but vice versa is not true.

➢ **Joins**

Natural Join (⋈) (intersection)
Natural join is a binary operator. Natural join between two or more relations will result set of all combination of tuples where they have equal common attribute.

Conditional Join
in conditional join we can specify the any **condition** such as **greater than, less than, not equal.**

- **Inner join**: Returns records that have matching values in both tables.
- **Left (Outer) Join**: Return all records from the left table, and the matched records from the right table
- **Right (outer) join**: Return all records from the right table, and the matched records from the left table
- **Full (outer) join**: Return all records when there is a match in either left or right table