**Introduction to Database Normalization**

Normalization is an important process in database design that helps improve the database's efficiency, consistency, and accuracy. It makes it easier to manage and maintain the data and ensures that the database is adaptable to changing business needs.

* 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.
* In the relational model, there exist standard methods to quantify how efficient a databases is. These methods are called [normal forms](https://www.geeksforgeeks.org/dbms/normal-forms-in-dbms/) and there are algorithms to covert a given database into normal forms.
* Normalization generally involves splitting a table into multiple ones which must be linked each time a query is made requiring data from the split tables.

**Why do we need Normalization?**

The primary objective for normalizing the relations is to eliminate the below anomalies. Failure to reduce anomalies results in data redundancy, which may threaten data integrity and cause additional issues as the database increases. Normalization consists of a set of procedures that assist you in developing an effective database structure.

* **Insertion Anomalies:** Insertion anomalies occur when it is not possible to insert data into a database because the required fields are missing or because the data is incomplete. For example, if a database requires that every record has a primary key, but no value is provided for a particular record, it cannot be inserted into the database.
* **Deletion anomalies:** Deletion anomalies occur when deleting a record from a database and can result in the unintentional loss of data. For example, if a database contains information about customers and orders, deleting a customer record may also delete all the orders associated with that customer.
* **Updation anomalies:**Updation anomalies occur when modifying data in a database and can result in inconsistencies or errors. For example, if a database contains information about employees and their salaries, updating an employee’s salary in one record but not in all related records could lead to incorrect calculations and reporting.

*Read more about* [*Anomalies in Relational Model.*](https://www.geeksforgeeks.org/dbms/anomalies-in-relational-model/)

Before Normalization: The table is prone to redundancy and anomalies (insertion, update, and deletion).  
After Normalization: The data is divided into logical tables to ensure consistency, avoid redundancy and remove anomalies making the database efficient and reliable.

**Features of Database Normalization**

* **Elimination of Data Redundancy:**One of the main features of normalization is to eliminate the data redundancy that can occur in a database. Data redundancy refers to the repetition of data in different parts of the database. Normalization helps in reducing or eliminating this redundancy, which can improve the efficiency and consistency of the database.

*Read More about* [*The Problem of Redundancy in Database*](https://www.geeksforgeeks.org/dbms/the-problem-of-redundancy-in-database/)

* **Ensuring Data Consistency:**Normalization helps in ensuring that the data in the database is consistent and accurate. By eliminating redundancy, normalization helps in preventing inconsistencies and contradictions that can arise due to different versions of the same data.
* **Simplification of Data Management:** Normalization simplifies the process of managing data in a database. By breaking down a complex data structure into simpler tables, normalization makes it easier to manage the data, update it, and retrieve it.
* **Improved Database Design:** Normalization helps in improving the overall design of the database. By organizing the data in a structured and systematic way, normalization makes it easier to design and maintain the database. It also makes the database more flexible and adaptable to changing business needs.
* **Avoiding Update Anomalies:**Normalization helps in avoiding update anomalies, which can occur when updating a single record in a table affects multiple records in other tables. Normalization ensures that each table contains only one type of data and that the relationships between the tables are clearly defined, which helps in avoiding such anomalies.
* **Standardization:**Normalization helps in standardizing the data in the database. By organizing the data into tables and defining relationships between them, normalization helps in ensuring that the data is stored in a consistent and uniform manner.

**Normal Forms in DBMS**

| **Normal Forms** | **Description of Normal Forms** |
| --- | --- |
| **First Normal Form (1NF)** | A relation is in [first normal form](https://www.geeksforgeeks.org/dbms/first-normal-form-1nf/) if every attribute in that relation is single-valued attribute. |
| **Second Normal Form (2NF)** | A relation that is in First Normal Form and every non-primary-key attribute is fully functionally dependent on the primary key, then the relation is in [Second Normal Form (2NF).](https://www.geeksforgeeks.org/dbms/second-normal-form-2nf/) |
| **Third Normal Form (3NF)** | A relation is in the [third normal form](https://www.geeksforgeeks.org/dbms/third-normal-form-3nf/), if there is no transitive dependency for non-prime attributes as well as it is in the second normal form. A relation is in 3NF if at least one of the following conditions holds in every non-trivial function dependency X –> Y.   * X is a super key. * Y is a prime attribute (each element of Y is part of some candidate key). |
| **Boyce-Codd Normal Form (BCNF)** | For BCNF the relation should satisfy the below conditions   * The relation should be in the 3rd Normal Form. * X should be a super-key for every functional dependency (FD) X−>Y in a given relation. |
| **Fourth Normal Form (4NF)** | A relation R is in [4NF](https://www.geeksforgeeks.org/dbms/introduction-of-4th-and-5th-normal-form-in-dbms/) if and only if the following conditions are satisfied:   * It should be in the [Boyce-Codd Normal Form (BCNF)](https://www.geeksforgeeks.org/dbms/boyce-codd-normal-form-bcnf/). * The table should not have any Multi-valued Dependency. |
| **Fifth Normal Form (5NF)** | A relation R is in[5NF](https://www.geeksforgeeks.org/dbms/what-is-fifth-normal-form-5nf-in-dbms/)if and only if it satisfies the following conditions:   * R should be already in 4NF. * It cannot be further non loss decomposed (join dependency). |

*Read more about* [*Normal Forms in DBMS.*](https://www.geeksforgeeks.org/dbms/normal-forms-in-dbms/)

**Advantages of Normalization**

* Normalization eliminates data redundancy and ensures that each piece of data is stored in only one place, reducing the risk of data inconsistency and making it easier to maintain data accuracy.
* By breaking down data into smaller, more specific tables, normalization helps ensure that each table stores only relevant data, which improves the overall data integrity of the database.
* Normalization simplifies the process of updating data, as it only needs to be changed in one place rather than in multiple places throughout the database.
* Normalization enables users to query the database using a variety of different criteria, as the data is organized into smaller, more specific tables that can be joined together as needed.
* Normalization can help ensure that data is consistent across different applications that use the same database, making it easier to integrate different applications and ensuring that all users have access to accurate and consistent data.

**Disadvantages of Normalization**

* Normalization can result in increased performance overhead due to the need for additional[join operations](https://www.geeksforgeeks.org/dbms/joins-in-dbms/) and the potential for slower query execution times.
* Normalization can result in the loss of data context, as data may be split across multiple tables and require additional joins to retrieve.
* Proper implementation of normalization requires expert knowledge of database design and the normalization process.
* Normalization can increase the complexity of a database design, especially if the data model is not well understood or if the normalization process is not carried out correctly.