Understanding Data Types and Normalisation: Foundations of Database Management

Understanding data types and normalisation are foundational concepts in database management. Entry-level data analytics courses in Hyderabad will cover some common data types and normalisation processes, which this article outlines.

Data Types and Normalisation

- **Data Types**

  Data types define the value format a column can store in a database. They ensure that only appropriate data is stored, which helps maintain data integrity and optimise storage space.

  Integer: Used for whole numbers.

  Float/Double: Used for floating-point numbers (numbers with decimal points).

  Char/NVARCHAR/VARCHAR: Used for storing strings of characters.

  Date/Time: Used for storing date and time values.

  Boolean: Used for storing true/false values.

  Binary: Used for storing binary data such as images or files.

  Choosing the right data type is important for efficient data storage and retrieval. For example, using an integer data type for a column that only needs to store numbers without decimal points can save storage space compared to using a float data type.

- **Normalisation**

  Normalisation involves organising the tables and columns of a relational database to reduce redundancy and dependencies, which is a core topic in data analytics courses in Hyderabad. It ensures that each piece of data is stored in only one place, which helps prevent inconsistencies and anomalies.

  Normalisation is typically divided into several normal forms, each building on the previous one:

  First Normal Form (1NF): Ensures that each column contains atomic values, i.e., cannot be further divided.

  Second Normal Form (2NF): Guarantees that all non-key columns depend entirely on the primary key.

  Third Normal Form (3NF): Ensures no transitive dependencies, i.e., non-key columns are not dependent on other non-key columns.

  Boyce-Codd Normal Form (BCNF): An enhanced form of 3NF, requiring every determinant to be a candidate key.

  Normalisation helps maintain data integrity, reduces redundancy, and simplifies database maintenance and updates. However, as data may be spread across multiple tables, it can sometimes increase query complexity.

Understanding these concepts is crucial for designing efficient and maintainable databases. It allows database designers to make informed decisions about data modelling, storage, and retrieval strategies.