26-05-2023

**DOMAIN CONSTRAINTS IN RELATIONAL DATABASE MANAGEMENT SYSTEMS**

* Domain constraints can be defined as the definition of a valid set of values for an attribute.
* The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain

In relational database management systems (RDBMS), domain constraints are rules or restrictions applied to the values that can be stored in a specific column or attribute of a table. These constraints define the valid data types, formats, and ranges of values that can be entered into a particular field.

Or in other words, it is an attribute that specifies all the possible values that the attribute can hold like integer, character, date, time, string, etc. It defines the domain or the set of values for an attribute and ensures that the value taken by the attribute must be an atomic value (Can’t be divided) from its domain.

Some of the common types of domain constraints in RDBMS includes:

* **Data Type Constraints:**

Data type constraints define the allowed data types for a column. Each column in a table has a specific data type associated with it, such as integers, strings, dates, or Booleans. These constraints ensure that only values of the specified data type can be stored in the column. For example, a column defined as an integer data type cannot store string values or dates.

* **Length Constraints:**

Length constraints specify the maximum number of characters or digits allowed in a column. They are commonly applied to string or numeric columns. For instance, if a column is defined with a length constraint of 50 characters, any value exceeding that length will be rejected. This ensures that the data fits within the specified length and prevents data truncation or overflow.

* **Range Constraints:**

Range constraints define the valid range of values that can be stored in a column. They are typically used with numeric or date columns. For example, a column representing age may have a range constraint of 0 to 120, meaning that only age values within that range are accepted. Range constraints help ensure that the stored data is within the expected boundaries.

* **Not Null Constraint:**

The not null constraint ensures that a column cannot contain null values. It enforces the requirement for a column to always have a non-null value. When this constraint is applied to a column, any attempt to insert or update a row with a null value in that column will result in an error. Not null constraints help maintain data integrity by preventing the storage of missing or incomplete data.

* **Default Value Constraint:**

Default value constraints assign a default value to a column if no explicit value is provided during data insertion. When a row is inserted into a table and a specific column is not provided with a value, the default value constraint automatically assigns a predefined value to that column. This ensures that the column always contains a valid value even if one is not explicitly specified.

* **Unique Constraint:**

Unique constraints enforce the uniqueness of values in a column or set of columns. They ensure that no duplicate values are allowed in the specified column(s). Typically, unique constraints are applied to columns that act as primary keys or other unique identifiers. When a unique constraint is in place, any attempt to insert a duplicate value will result in an error, maintaining the uniqueness and integrity of the data.

* **Check Constraint:**

Check constraints allow you to define custom validation rules based on logical or comparison operators. They specify a condition that must be satisfied for the values in a column. For example, a check constraint can ensure that a numeric column only accepts positive values or that a string column only allows certain characters or patterns. Check constraints help enforce business rules and ensure the validity of the data being stored.