Unit-04

Integrity Constraints

Data integrity

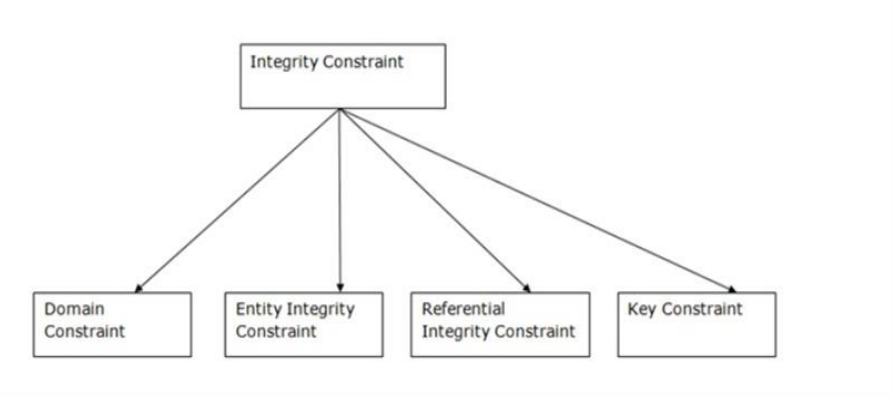
Data Integrity is having correct and accurate data in your database. When we are storing data in the database, we don't want repeating values, we don't want incorrect values or broken relationships between tables.

Integrity Constraints

Integrity constraints are a set of rules. It is used to maintain the quality of information. Integrity constraints ensure that the data insertion, updating, and other processes must be performed in such a way that data integrity is not affected. Thus, integrity constraint is used to guard against accidental damage to the database.

Integrity Constraints

1. Domain constraint
2. Entity Intrigity Constraints
3. Referential constraints
4. Key Constraints
5. Trigger
6. Assertion

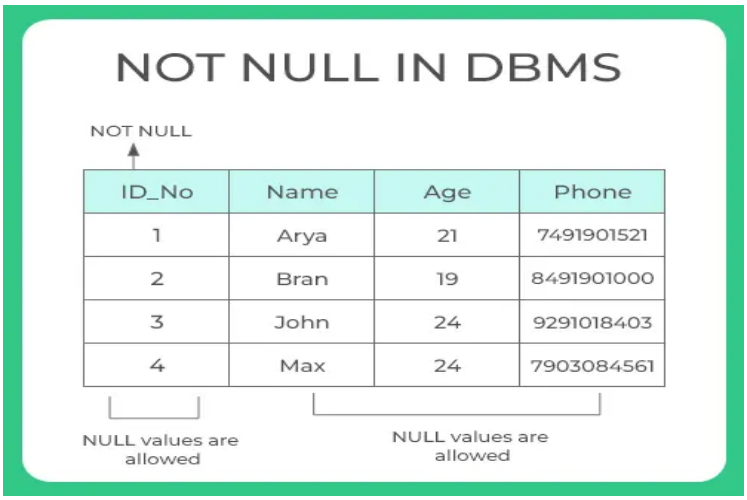


Domain constraints

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.

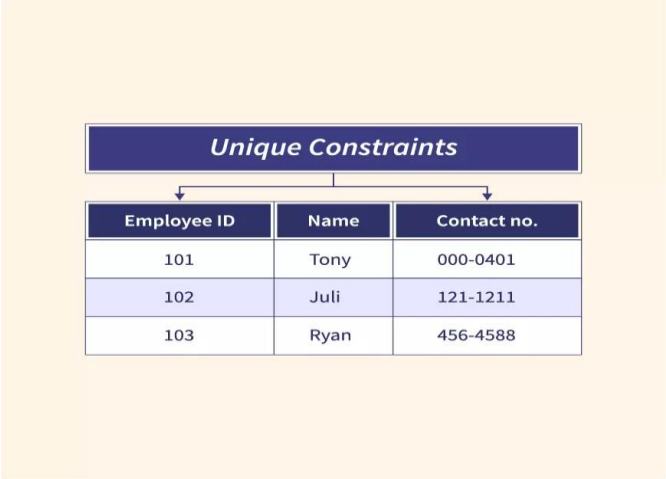
1. **Not Null Constraints**

→ A "NOT NULL" constraint is a database constraint used to ensure that a column in a relational database table does not contain any NULL values. In the context of a relational database, a NULL value represents the absence of data or an unknown value in a particular column.



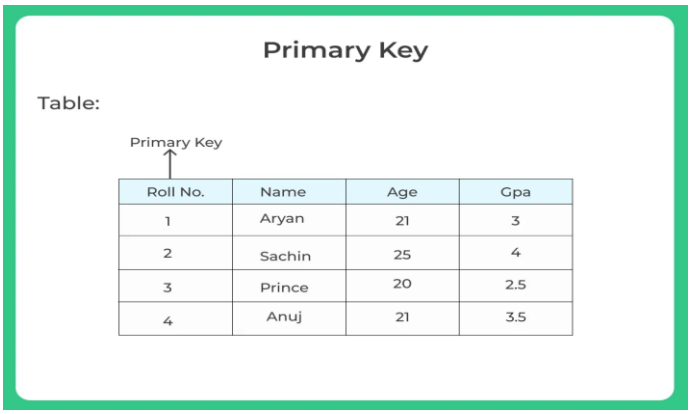
2. **Unique Constraints**

→ A "UNIQUE" constraint in a relational database is used to ensure that all values in a particular column (or a combination of columns) are unique across the rows in a table. Unlike the "PRIMARY KEY" constraint, which implies uniqueness and also implies that the column (or set of columns) is a key for identifying records, a "UNIQUE" constraint allows for the uniqueness of values without necessarily being the primary means of identifying records.



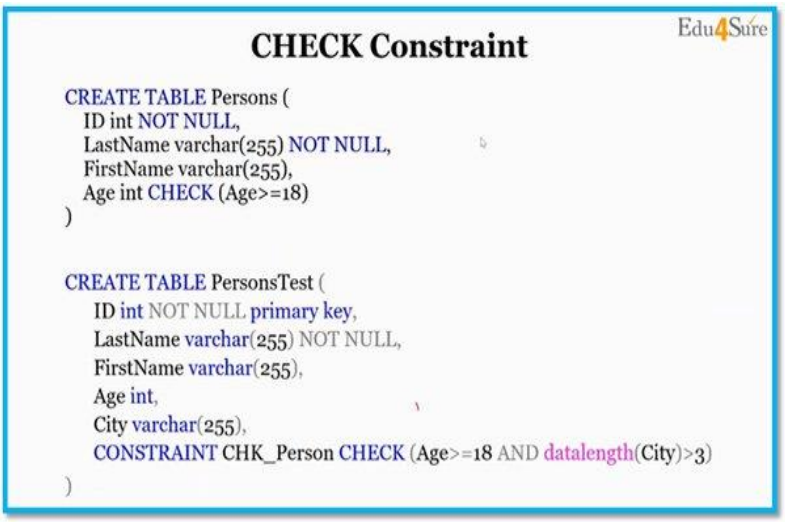
3. Primary key Constraints

→ A "PRIMARY KEY" constraint in a relational database is used to uniquely identify each record in a table. The primary key serves as a unique identifier for each row and provides a way to ensure data integrity and facilitate efficient data retrieval. The main features of primary key are **Uniqueness, Not Null, Implicit Index, Uniquely Identifies Records.**



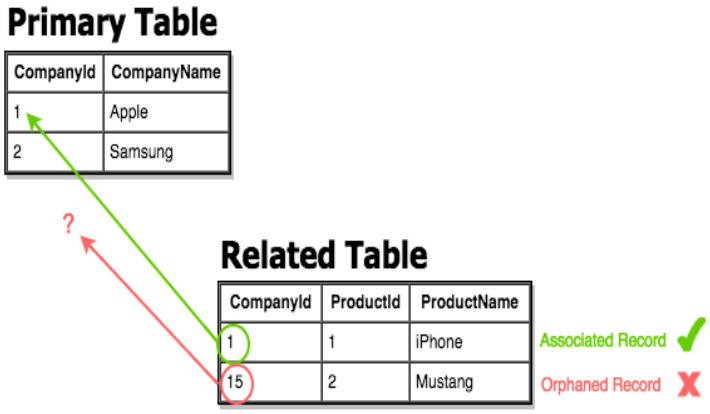
4. Check Constraints

→ A "CHECK" constraint in a relational database is used to define a condition that must be satisfied for the values in a column or a combination of columns. It allows you to specify a condition that each row in the table must meet, ensuring data integrity. If a row violates the specified condition, the database system rejects the operation that would cause the violation.



Referential Integrity

→ Referential integrity is a concept in relational databases that ensures the consistency and accuracy of relationships between tables. It is enforced using foreign key constraints, which establish a link between two tables based on a related column. In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.



Cascading Actions

→ Cascading actions, in the context of referential integrity in a relational database, refer to the automatic propagation of changes from one table to another when certain actions are performed on the referenced table. These actions are defined in the foreign key constraint and dictate how changes to the referenced primary key are handled in the table containing the foreign key.

Assertions and Triggers

Assertions

→ Assertions are different from check constraints in the way that check constraints are rules that relate to one single row only. Assertions, on the other hand, can involve any number of other tables, or any number of other rows at the same table. Assertions also check a condition, which must return a Boolean value. We can take an illustrative example. Let us imagine that we have the following table, which contains employees in a company — we then also store an attribute containing their salary.

ID	Name	Age	DepNo	Salary
0	Hannah	18	10	1000$
1	Gavin	45	2	25.000$
2	Bobby	70	21	700$

Example: Creating an Assertion

CREATE ASSERTION salary_constraint

CHECK (salary > 0 AND salary < 100000);

Deleting Assertions:

To delete an assertion, you use the DROP ASSERTION statement.

DROP ASSERTION salary_constraint;

Triggers

A trigger is a database object that is associated with the table, it will be activated when a defined action is executed for the table. The trigger can be executed when we run the following statements:

• INSERT

• UPDATE

• DELETE

Triggers can be configured to execute either before or after the specified event, allowing for the enforcement of business rules, data validation, and automatic logging of changes.

Example: Creating a Trigger

```
CREATE TRIGGER before_insert_trigger
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
   //Trigger body//
   :NEW.creation_date := SYSDATE;
END;
```

Example: Deleting a Trigger

```
DROP TRIGGER before_insert_trigger;
```

Difference between Assertions and Triggers

S.N	Assertions	Triggers
1	We can use Assertions when we know that the given particular condition is always true.	We can use Triggers even particular condition may or may not be true
2	When the SQL condition is not met then there are chances to an entire table or even Database to get locked up.	Triggers can catch errors if the condition of the query is not true.
3	Assertions are not linked to specific table or event. It performs task specified or defined by the user.	It helps in maintaining the integrity constraints in the database tables, especially when the primary key and foreign key constraint are not defined.
4	Assertions do not maintain any track of changes made in table.	Triggers maintain track of all changes occurred in table.
5	Assertions have small syntax compared to Triggers.	They have large Syntax to indicate each and every specific of the created trigger.
6	Modern databases do not use Assertions.	Triggers are very well used in modern databases.