Constraints in SQL

Constraints in SQL are rules enforced on **columns** or **tables** to maintain data integrity and accuracy.

Types of Constraints

1. **NOT NULL** – Ensures a column cannot have NULL values.
2. **UNIQUE** – Ensures all values in a column are unique.
3. **PRIMARY KEY** – Uniquely identifies each row in a table (combines NOT NULL + UNIQUE).
4. **FOREIGN KEY** – Ensures referential integrity between tables.
5. **CHECK** – Enforces a condition on column values.
6. **DEFAULT** – Assigns a default value if no value is provided.

1. NOT NULL Constraint

Ensures a column cannot have NULL values.

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL  -- Name cannot be NULL
);
```

-- Valid Insert

```
INSERT INTO Employees (EmployeeID, Name) VALUES (1, 'John Doe');
```

-- Invalid Insert (Fails because Name is NULL)

```
INSERT INTO Employees (EmployeeID, Name) VALUES (2, NULL);
```

2. UNIQUE Constraint

Ensures all values in a column are unique.

```
CREATE TABLE Customers (
   CustomerID INT PRIMARY KEY,
   Email VARCHAR(255) UNIQUE  -- Email must be unique
);
```

-- Valid Inserts

```
INSERT INTO Customers (CustomerID, Email) VALUES (1, 'alice@example.com');
INSERT INTO Customers (CustomerID, Email) VALUES (2, 'bob@example.com');
```

-- Invalid Insert (Fails because Email is duplicate)

```
INSERT INTO Customers (CustomerID, Email) VALUES (3, 'alice@example.com');
```

3. PRIMARY KEY Constraint

Combines NOT NULL + UNIQUE to uniquely identify each row.

```
CREATE TABLE Products (
   ProductID INT PRIMARY KEY,  -- Ensures uniqueness and not null
   ProductName VARCHAR(100) NOT NULL
);

-- Valid Insert
INSERT INTO Products (ProductID, ProductName) VALUES (101, 'Laptop');

-- Invalid Insert (Fails because ProductID is duplicate)
INSERT INTO Products (ProductID, ProductName) VALUES (101, 'Smartphone');
```

4. FOREIGN KEY Constraint

Links two tables to maintain referential integrity.

```
CREATE TABLE Departments (
   DepartmentID INT PRIMARY KEY,
   DepartmentName VARCHAR(100) NOT NULL
);

CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   DepartmentID INT,
   CONSTRAINT fk_department FOREIGN KEY (DepartmentID) REFERENCES
Departments(DepartmentID) -- Ensures DepartmentID exists
);

-- Valid Insert
INSERT INTO Departments (DepartmentID, DepartmentName) VALUES (1, 'HR');
INSERT INTO Employees (EmployeeID, Name, DepartmentID) VALUES (1, 'Alice', 1);

-- Invalid Insert (Fails because DepartmentID 2 does not exist)
INSERT INTO Employees (EmployeeID, Name, DepartmentID) VALUES (2, 'Bob', 2);
```

5. CHECK Constraint

Restricts values based on a condition.

```
CREATE TABLE Students (
   StudentID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   Age INT CHECK (Age >= 18)  -- Age must be 18 or older
);

-- Valid Insert
INSERT INTO Students (StudentID, Name, Age) VALUES (1, 'John', 20);

-- Invalid Insert (Fails because Age is less than 18)
INSERT INTO Students (StudentID, Name, Age) VALUES (2, 'Jane', 16);
```

6. DEFAULT Constraint

Assigns a default value if no value is provided.

```
CREATE TABLE Orders (
   OrderID INT PRIMARY KEY,
   OrderDate DATE DEFAULT CURRENT_DATE  -- Default value is the current date
);

-- Insert without OrderDate (Uses default)
INSERT INTO Orders (OrderID) VALUES (1);

-- Insert with a specific OrderDate
INSERT INTO Orders (OrderID, OrderDate) VALUES (2, '2024-02-13');
```

ENUM in SQL

An **ENUM** (short for *enumeration*) is a data type in SQL that allows a column to have a predefined set of values. It ensures that only valid, specified values are stored in the column, improving data integrity.

Syntax (MySQL Example)

```
CREATE TABLE table_name (
   column_name ENUM('value1', 'value2', 'value3', ...) NOT NULL
);
```

Example: Creating a Table with ENUM

```
CREATE TABLE Employees (
   EmployeeID INT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   Gender ENUM('Male', 'Female', 'Other') NOT NULL, -- Gender must be one of the specified values
   EmploymentStatus ENUM('Full-Time', 'Part-Time', 'Contract', 'Intern') DEFAULT 'Full-Time' -- Default value set
);
```

Inserting Data

```
-- ☑ Valid insert
INSERT INTO Employees (EmployeeID, Name, Gender, EmploymentStatus)
VALUES (1, 'John Doe', 'Male', 'Part-Time');

-- ☑ Insert using default EmploymentStatus
INSERT INTO Employees (EmployeeID, Name, Gender)
VALUES (2, 'Jane Smith', 'Female');

-- ✖ Invalid insert (Fails because 'Temporary' is not a valid ENUM value)
INSERT INTO Employees (EmployeeID, Name, Gender, EmploymentStatus)
VALUES (3, 'Bob Brown', 'Male', 'Temporary');
```