Integrity Constraints in DBMS

Integrity Constraints in DBMS

- Constraints are the rules that the table must satisfy. These can be specified at the time of creating table or can also be added to table by using alter table statement. - Constraints prevent the table from deletion, if there is any dependency

Types of Integrity Constraints in DBMS are :

- DEFAULT Constraint - NOT NULL Constraint - UNIQUE Constraint - CHECK Constraint PRIMARY KEY Constraint] FOREIGN KEY Constraint]

Syntax of Constraints :

```
Create table table_name
   (
    column1 datatype column1 constraint,
    column2 datatype column2 constraint,
    column3 datatype
   );
```

Constraints can be defined at 2 levels :

Column level - At column level the constraints are referred for a single column and is defined within the specification.
Table Level - At table level when constraints are defined at table level then it can be referred to one or more columns and it can be defined separately from the definitions of the columns in the table.

DEFAULT CONSTRAINT
It is used to define a default value for an attribute.
Example :

```
CREATE TABLE employee
 (
  EID int,
  Name varchar(20),
  Address varchar(50),
  Salary int,
  Sex char(1) DEFAULT='M'
 );
```

NOT NULL Constraint - NOT NULL constraint is used when we do not want null values to be entered. It ensures that null values are not permitted in the column. - The columns without NOT NULL constraint can have null values. By default, a column can contain NULL.

Example :

```
CREATE TABLE employee
       (
        EmpID int NOT NULL,
        Name varchar(30) NOT NULL,
        Address varchar(50),
        Salary int,
        Sex char(1) DEFAULT='M'
       );
```

The column "EmpID" and "Name" cannot include NULL, But the column "Address" can.

```
   INSERT INTO employee (EmpID, Address, Salary)
```

```
VALUES ("1000","Bhiwani",29000)
```

This query would returned an error because it will lead to column Name being NULL, which violates the NOT NULL constraint.

UNIQUE Constraint - The UNIQUE constraint is used to uniquely identify each record in a database. It requires that each value in a column should be unique,i.e. no 2 rows of same column should have duplicate values. - It provides you the guarantee for uniqueness in a column or set of columns and allows us to input the NULL values, unless there is not defined NOT NULL constraint for the same column. - As in UNIQUE constraint, there is a fact that there could be and number of rows which can include NULL for columns without NOT NULL constraints, reason because nulls are not considered equal to anything. - It can be defined at table level or at column level.

Example :

```
CREATE TABLE employee
 (
  EID int NOT NULL,
  Name varchar(20) NOT NULL,
  Address varchar(50),
  Salary int,
  Sex char(1) DEFAULT='M',
  UNIQUE (EID)
 );
```

The column "EmpID" cannot have null values and the values in it are distinct.

CHECK constraint : - The check constraint is used to check the value which is entered into a record. It is used to define condition which each row must satisfy. - If the condition value evaluates to fall, then the record violates the constraint and you cannot enter it into the table. - We cannot define check constraint in SQL view.

Syntax :

```
CREATE TABLE table_name
  (
   column1 datatype,
   column2 datatype,
   ....
   CONSTRAINT constraint_name CHECK
   (column-name_contition));
```

The check constraint can be defined by a create table statement or alter table statement.

A single column can contain a number of check constraints point it can be defined at the table level or column level

Example :

```
CREATE TABLE employee
 (
  EmpID int NOT NULL,
  Name varchar(30) NOT NULL,
  Address char(50),
  Sex char(1) DEFAULT='M',
  Salary int,
  PRIMARY KEY(EmpID),
  CONSTRAINT chk_Salary CHECK (Salary>20000) );
```

The above query will check the chk_Salary constraint from table employee. The constraint will ensure that Salary contains value greater than 20000.

We can also define CHECK constraint as :

```
CREATE TABLE employee
 (
  EmpID int NOT NULL,
```

```
  Name varchar(30) NOT NULL,
  Address char(50),
  Sex char(1) DEFAULT='M',
  Salary int CHECK(Salary>20000),
  PRIMARY KEY(EmpID),
 );
```

CHECK Constraint using ALTER TABLE Statement :
Add a CHECK Constraint :
Syntax :

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name CHECK
(column_name condition);
```

Example :

```
ALTER TABLE employee
ADD CHECK (EmpID>999);
```

This CHECK Constraint on EmpID column can be added only when the table is already created.

```
ALTER TABLE employee
ADD CONSTRAINT chk_EmpID CHECK (EmpID>999 AND Salary>17000);
```

The above CHECK Constraint is defined for multiple columns.
DROP a CHECK Constraint
Syntax :

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

Example :

```
ALTER TABLE employee
DROP CONSTRAINT chk_EmpID;
```

ENABLE a CHECK Constraint
Syntax :

```
ALTER TABLE table_name
ENABLE CONSTRAINT constraint_name;
```

Example :

```
ALTER TABLE employee
ENABLE CONSTRAINT chk_Salary;
```

DISABLE a CHECK Constraint
Syntax :

```
ALTER TABLE table_name
DISABLE CONSTRAINT constraint_name;
```

Example :

```
ALTER TABLE employee
DISABLE CONSTRAINT chk_Salary;
```

CLICK for Primary and Foreign Key Constraint Previous Home Next DCL Commands Primary Key and Foreign Key Constraint