CONSTRAINTS: (06-11-2024)
==========

- constraints are used to enforce / restricted unwanted data(i.e invalid data) from a table.
- sqlserver supports the following six types of constraints.
    - UNIQUE
    - NOT NULL
    - CHECK
    - PRIMARY KEY
    - FOREIGN KEY
    - DEFAULT
- constraints can be defined at two levels.
    - i) column level
    - ii) table level

i) column level:
==========

- in this level constraint can be defined on individual columns wise.

syntax:
=======
create table <table name>(<column name1> <datatype>[size] <constraint type>, <column name2> <datatype>[size] <constraint type>,.....);

ii) table level:
==========

- in this level constraint can be defined after all columns definitions i.e the end of table.

syntax:
=======
create table <table name>(<column name1> <datatype>[size], <column name2> <datatype>[size],.....,<constraint type>(column name1,column name2,.....));

UNIQUE:
========

- to restricted duplicate values but allowed nulls into columns.

EX:
column level:
==========
CREATE TABLE TEST1(SNO INT UNIQUE,NAMES VARCHAR(10) UNIQUE)

TESTING:

INSERT INTO TEST1 VALUES(1,'A')-------> allowed
INSERT INTO TEST1 VALUES(1,'A')-------> not allowed
INSERT INTO TEST1 VALUES(NULL,NULL)----> allowed

table level:
=========
CREATE TABLE TEST2(SNO INT,NAMES VARCHAR(10),UNIQUE(SNO,NAMES));

TESTING:
INSERT INTO TEST2 VALUES(1,'A')------allowed
INSERT INTO TEST2 VALUES(1,'A')-----not allowed
INSERT INTO TEST2 VALUES(1,'B)----allowed
INSERT INTO TEST2 VALUES(NULL,NULL)-----allowed

NOT NULL:
==========
- to restricted nulls but allowed duplicate values into columns.
- it can be defined at column level only.

EX:
column level:
===========
CREATE TABLE TEST3(SNO INT NOT NULL,NAMES VARCHAR(10) NOT NULL)

TESTING:
INSERT INTO TEST3 VALUES(1,'A')--------allowed
INSERT INTO TEST3 VALUES(1,'A')-------allowed
INSERT INTO TEST3 VALUES(NULL,NULL)-----not allowed

CHECK:
=======
- to check the values with user defined condition before accepting into a column.

EX:
column level:
===========
CREATE TABLE TEST4
(
REGNO INT UNIQUE NOT NULL,
CNAME VARCHAR(10) NOT NULL,
ENTRY_FEE MONEY NOT NULL CHECK(ENTRY_FEE=500),
AGE INT NOT NULL CHECK(AGE BETWEEN 18 AND 30),
LOC VARCHAR(10) NOT NULL CHECK(LOC IN('HYD','MUMBAI','DELHI'))
)

TESTING:
INSERT INTO TEST4 VALUES(1021,'SMITH',450,31,'HYDERABAD');-----> NOT ALLOWED
INSERT INTO TEST4 VALUES(1021,'SMITH',500,18,'HYD');----------> ALLOWED

table level:
==========
CREATE TABLE TEST5(EID INT,SAL MONEY,LOC VARCHAR(10),
CHECK((SAL>=15000) AND (LOC IN('HYD','DELHI'))));

TESTING:
INSERT INTO TEST5 VALUES(1001,14000,'MUMBAI')-----> NOT ALLOWED
INSERT INTO TEST5 VALUES(1001,15000,'DELHI')----->ALLOWED

PRIMARY KEY:
=============
- it is a combination of "unique and not null" constraints.
- it restricted duplicate values and also nulls.
- a table is having only one primary key constraint.

EX:
colum level:
==========
CREATE TABLE TEST6(PCODE INT PRIMARY KEY,PNAME VARCHAR(10) UNIQUE NOT NULL)

TESTING:
INSERT INTO TEST6 VALUES(1021,'P1');------allowed
INSERT INTO TEST6 VALUES(1021,'P2');-----not allowed
INSERT INTO TEST6 VALUES(NULL,'P2');-----not allowed
INSERT INTO TEST6 VALUES(1022,'P2');----allowed

07-11-2024:
==========
table level: (Composite Primary key)
==============================
- when we apply a primary key constraint on multiple combination of columns in the table is known as "composite primary key".
- in composite primary key individual columns are accepting duplicate values but the combination of columns are not accepting duplicate values.

EX:
CREATE TABLE TEST7(BCODE INT,BNAME VARCHAR(10),LOC VARCHAR(10),
PRIMARY KEY(BCODE,BNAME));

TESTING:
INSERT INTO TEST7 VALUES(1021,'SBI','AMEERPET');-----allowed
INSERT INTO TEST7 VALUES(1021,'SBI','MADHAPUR')----not allowed
INSERT INTO TEST7 VALUES(1022,'SBI','MADHAPUR')----allowed
INSERT INTO TEST7 VALUES(1021,'HDFC','AMEERPET')----allowed

FOREIGN KEY:
==========

- to make relationship between tables for taking referencial data (i.e identity) from one table to another table.

Basic Rules:
==========
1. we should maintain atleast one common column in both tables.
2. this comman column datatypes must be match / same.
3. to make a relationship between tables,one table should have primary key and another table should have foreign key.Here primary key and foreign key column must be common column only.
4. a primary key table is called as "parent table" and a foreign key table is called as "child table".
5. a foreign key column is accepting the values which was found in primary key column only.
6. by default a foreign key column is allowed duplicates and nulls.

syntax:
======
<common column of child table> <datatype>[size] foreign key references <parent table name>(common column of parent table)

EX:
CREATE TABLE DEPT1(DNO INT PRIMARY KEY,DNAME VARCHAR(10)); ---> PARENT TABLE
INSERT INTO DEPT1 VALUES(10,'.NET'),(20,'SQL');

CREATE TABLE EMP1(EID INT PRIMARY KEY,ENAME VARCHAR(10),
DNO INT FOREIGN KEY REFERENCES DEPT1(DNO));------> CHILD TABLE

INSERT INTO EMP1 VALUES(1021,'SMITH',10);-----> allowed
INSERT INTO EMP1 VALUES(1022,'ALLEN',10);-----> allowed
INSERT INTO EMP1 VALUES(1023,'JONES',20);-----> allowed
INSERT INTO EMP1 VALUES(1024,'ADAMS',NULL);----> allowed
INSERT INTO EMP1 VALUES(1025,'JAMES',30);-------> not allowed

- once we establish relationship between tables there are three rules are come

into picture.

i) Insertion rule
ii) Updation rule
iii) Deletion rule

08-11-2024:
==========
i) Insertion rule:
==========

- we cannot insert the values into a foreign key column those values are not existing in primary key column.

i.e no parent = no child

EX:
INSERT INTO EMP1 VALUES(1025,'MILLER',30);------not allowed

ii) Updation rule:
==========

- we cannot update the values of primary key column of parent table those parent rows are having the corresponding child rows in child table without addressing to child.

EX:
UPDATE DEPT1 SET DNO=11 WHERE DNO=10;------not allowed

iii) Deletion rule:
==========

- we cannot delete a row from the parent table those parent rows are having the corresponding child rows in child table without addressing to child.

EX:
DELETE FROM DEPT1 WHERE DNO=20;------not allowed

- if we want update and delete data from a parent table then we must use the following cascade rules:

i) ON UPDATE CASCADE
ii) ON DELETE CASCADE
iii) ON UPDATE SET NULL
iv) ON DELETE SET NULL

i) ON UPDATE CASCADE:
==========

- when we update data in parent table(primary key column) then the corresponding child rows of foreign key column values also updated automatically.

ii) ON DELETE CASCADE:
==========

- when we delete row from a parent table then the corresponding child rows also deleted automatically.

EX:

```
CREATE TABLE DEPT2(DNO INT PRIMARY KEY,DNAME VARCHAR(10));
INSERT INTO DEPT2 VALUES(10,'DBA'),(20,'SAP');

CREATE TABLE EMP2(EID INT PRIMARY KEY,ENAME VARCHAR(10),
DNO INT FOREIGN KEY REFERENCES DEPT2(DNO) ON UPDATE CASCADE ON DELETE
CASCADE);
```

TESTING:

RULE1:

```
INSERT INTO EMP2 VALUES(1021,'SMITH',10);---------> allowed
INSERT INTO EMP2 VALUES(1022,'JONES',20);---------> allowed
INSERT INTO EMP2 VALUES(1023,'ADAMS',30);---------> not allowed
```

RULE2:

```
UPDATE DEPT2 SET DNO=11 WHERE DNO=10;-------allowed
```

RULE3:

```
DELETE FROM DEPT2 WHERE DNO=20;------allowed
```

iii) ON UPDATE SET NULL:
==========

- when we update data in parent table(primary key column) then the corresponding child rows of foreign key column values are converting into NULLS automatically.

iv) ON DELETE SET NULL:
==========

- when we delete row from a parent table then the corresponding child rows of foreign key column values are converting into NULLS automatically.

EX:

```
CREATE TABLE DEPT3(DNO INT PRIMARY KEY,DNAME VARCHAR(10));

INSERT INTO DEPT3 VALUES(10,'DBA'),(20,'SAP');
```

CREATE TABLE EMP3(EID INT PRIMARY KEY,ENAME VARCHAR(10),
DNO INT FOREIGN KEY REFERENCES DEPT3(DNO)
ON UPDATE SET NULL ON DELETE SET NULL);

INSERT INTO EMP3 VALUES(1021,'SMITH',10);---------> allowed
INSERT INTO EMP3 VALUES(1022,'JONES',20);---------> allowed

TESTING:
RULE1:
INSERT INTO EMP3 VALUES(1023,'ADAMS',30);---------> not allowed

RULE2:
UPDATE DEPT3 SET DNO=11 WHERE DNO=10;-------allowed

RULE3:
DELETE FROM DEPT3 WHERE DNO=20;------allowed
==========
==========
How to add constraint on an existing table:
==========
syntax:
======
ALTER TABLE <TABLE NAME> ADD <CONSTRAINT> <CONSTRAINT KEY NAME>
<CONSTRAINT TYPE>(<COLUMN NAME>);

I) Adding Primary key:
==========
EX:
CREATE TABLE PARENT(EID INT,ENAME VARCHAR(10),SALARY MONEY);
- Before apply a primary key constraint on EID column in parent table first we should make it EID column as a NOT NULL.

syntax to apply NOT NULL constraint:
==========
ALTER TABLE <TABLE NAME> ALTER <COLUMN> <COLUMN NAME> <DATATYPE>[SIZE]
NOT NULL;

EX:
ALTER TABLE PARENT ALTER COLUMN EID INT NOT NULL;
ALTER TABLE PARENT ADD CONSTRAINT PK_EID PRIMARY KEY(EID);

09-11-2024:
==========

II) Adding Unique key:
==========
ALTER TABLE PARENT ADD CONSTRAINT UQ_ENAME UNIQUE(ENAME);

III) Adding Check constraint:
==========
ALTER TABLE PARENT ADD CONSTRAINT CHK_SAL CHECK(SALARY>=8000);

IV) Adding Foreign key references:
==========
syntax:
=======
ALTER TABLE <TN> ADD CONSTRAINT <CONSTRAINT KEY NAME>
FOREIGN KEY(COMMON COLUMN OF CHILD TABLE) REFERENCES
<PARENT TABLE NAME>(COMMON COLUMN OF PARENT TABLE)
ON UPDATE CASCADE ON DELETE CASCADE / ON UPDATE SET NULL ON DLEETE SET NULL;

EX:
CREATE TABLE CHILD(DNAME VARCHAR(10),EID INT);

ALTER TABLE CHILD ADD CONSTRAINT FK_EID FOREIGN KEY(EID)
REFERENCES PARENT(EID) ON UPDATE CASCADE ON DELETE CASCADE;

How to drop constraint from an existing table:
==========
syntax:
======
ALTER TABLE <TABLE NAME> DROP <CONSTRAINT> <CONSTRAINT KEY NAME>;

I) Dropping Primary key:
==========
CASE-1: Without relationship:
==========
ALTER TABLE PARENT DROP CONSTRAINT PK_EID;

CASE-2: With relationship:
==========
ALTER TABLE CHILD DROP CONSTRAINT FK_EID---------FIRST
ALTER TABLE PARENT DROP CONSTRAINT PK_EID-----LATER

II) Dropping Unique,Check constraint:
==========

```
ALTER TABLE PARENT DROP CONSTRAINT UQ_ENAME;
ALTER TABLE PARENT DROP CONSTRAINT CHK_SAL;
```

How to make NOT NULL to NULL of a column:
==========
```
ALTER TABLE PARENT ALTER COLUMN EID INT NULL;
```

DEFAULT CONSTRAINT:
==========
- to assign a user defined default values to a column in the table.

syntax:
======
<column name> <datatype>[size] default <value>;

Ex:
```
CREATE TABLE TEST8(SNO INT,NAME VARCHAR(10),LOC VARCHAR(10) DEFAULT 'HYD');
```

TESTING:
```
INSERT INTO TEST8(SNO,NAME,LOC) VALUES(1,'ALLEN','PUNE');
INSERT INTO TEST8(SNO,NAME)VALUES(2,'MILLER');
INSERT INTO TEST8(SNO,NAME)VALUES(3,'JAMES');
```

How to add default constraint value to an existing column in the table:
==========
syntax:
======
```
ALTER TABLE <TN> ADD CONSTRAINT <CONSTRAINT KEY NAME> DEFAULT <value>
FOR <COLUMN NAME>;
```

Ex:
```
CREATE TABLE TEST9(ENAME VARCHAR(10),SALARY MONEY);
ALTER TABLE TEST9 ADD CONSTRAINT DF_SAL DEFAULT 5000 FOR SALARY;
```

How to drop default constraint value from an existing column:
==========
```
ALTER TABLE TEST9 DROP CONSTRAINT DF_SAL;
```