identity

CREATE DATABASE STUDENTS

CREATE TABLE students( **-----WITHOUT IDENTITY**

SNO int NOT NULL,

Name nvarchar(50),

Standard int ,

DOB date,

)

Insert into students(SNO, Name,standard,DOB) Values(1,'abinsh',9,'04-jan-2000')

Insert into students(SNO, Name,standard,DOB) Values(2,'naveen',9,'04-feb-2001')

Insert into students(SNO, Name,standard,DOB) Values(3,'vijayaragavan',9,'04-mar-2003')

Insert into students(SNO, Name,standard,DOB) Values(4,'keerthi',9,'04-apr-2003')

Insert into students(SNO, Name,standard,DOB) Values(5,'arivahagan',9,'04-may-2004')

SELECT \* FROM students

CREATE TABLE students2( **-----WITH IDENTITY** default (1,1)

SNO int Identity,

Name nvarchar(50),

Standard int ,

DOB date,

)

Insert into students2( Name,standard,DOB) Values('abinsh',9,'04-jan-2000')

Insert into students2( Name,standard,DOB) Values('naveen',10,'04-feb-2001')

Insert into students2( Name,standard,DOB) Values('vijayaragavan',11,'04-mar-2003')

Insert into students2( Name,standard,DOB) Values('keerthi',12,'04-apr-2003')

Insert into students2( Name,standard,DOB) Values('arivahagan',1,'04-may-2004')

select \* from students2

CREATE TABLE students3( -----seedvalue,increment

SNO int identity(101,1),

Name nvarchar(50),

Standard int ,

DOB date,

)

Insert into students3( Name,standard,DOB) Values('abinsh',9,'04-jan-2000')

Insert into students3( Name,standard,DOB) Values('naveen',10,'04-feb-2001')

Insert into students3( Name,standard,DOB) Values('vijayaragavan',11,'04-mar-2003')

Insert into students3( Name,standard,DOB) Values('keerthi',12,'04-apr-2003')

Insert into students3( Name,standard,DOB) Values('arivahagan',1,'04-may-2004')

Select \* from students3

SELECTION AND WHERE CONDITION

create table employee

(

ID int,

Name nvarchar(50),

Gender nvarchar(50),

salary int

)

Insert into employee values (1,'Ahamed','Male',14000)

Insert into employee values (2,'harini','female',15000)

Insert into employee values (3,'diviya','female',16000)

Insert into employee values (4,'priya','female',17000)

Insert into employee values (4,'dinesh','Male',24000)

select\* from employee

select ID,name,salary from employee

select \* from employee where ID = 1 ------------ID 1

select \* from employee where ID<>1 -----------not equal

select \* from employee where ID in(1,2) ---------------ID IN(!<@) will be shown 1,2

select \* from employee where name in ('priya','dinesh') -----------call by column name too

select \* from employee where ID not in (1,2) ------------------------not in avoid the both 1,2 shows other data

select \* from employee where salary >15000 -------------------------------15,000 above

select \* from employee where salary <15000-----------------------------------15,000 below

select \* from employee where salary <=15000-------------------------------15,000 equal and below

select \* from employee where salary >=15000----------------------------------greater than equal 15,000 and above

select \* from employee where salary between 15000 and 17000 ------------------<15 between 17

select distinct name from employee-------------------------------------distict will avoid the duplicate data and shows the data

select distinct ID,name from employee--------------------------it vary in id but it tells that it different person

select\* from employee where name like 'p%' -------------------------select the name that start with P

select\* from employee where name like '%i'------------------------ the name finish with I

select\* from employee where name like '%di%'---------------------- the character that combined with di

select\* from employee where name not like '%di%'------------- with the combination of di

select\* from employee where name like '[oh][AO]%'-------------------IT WHETEHR D OR P SECOND LETTER A OR O

select\* from employee where name like '[oh][^A]%'-------------- the second letter should not be a

**DELETE,DROP AND TRUNCATE**

create table employee555

(

ID int identity,

Name nvarchar(50),

Gender nvarchar(50),

salary int

)

Insert into employee555 values ('Ahamed','Male',14000)

Insert into employee555 values ('harini','female',15000)

Insert into employee555 values ('diviya','female',16000)

Insert into employee555 values ('priya','female',17000)

Insert into employee555 values ('dinesh','Male',24000)

select \* from employee555 where ID=3 ----------select the third id

delete from employee555 where ID=3----------- delete the id -3

delete from employee555 ------------total data will be deleted in table (delete will continue the identity)

truncate table employee555-------------truncate will enter the fresh id start from 1----

drop table employee555-------------drop will delete the table totatlly

update table

create table emp

(

ID int identity,

Name nvarchar(50),

Gender nvarchar(50),

salary int

)

Insert into emp values ('Ahamed','Male',14000)

Insert into emp values ('harini','female',15000)

Insert into emp values ('diviya','female',16000)

Insert into emp values ('priya','female',17000)

Insert into emp values ('dinesh','Male',24000)

select \* from emp

select \* from emp where ID=3

update emp set name = 'diviyajothi' where ID=3

update emp set name = 'priyadinesh' ,Gender = 'femalemale' where ID=4

update emp set name = 'diviyajothi'-------------total damage name changes in every column

update employee set Salary = salary -20/100+salary

select\* from employee

**constraints**

1.unique

2.not null

3.primary key

4.default

5.check

6foreign key

create table constraints

(Id int identity(100,1),

Name nvarchar(50),

Email nvarchar(50) unique, ---------- email should be unique

MobileNo nvarchar(20)not null,-------------------not should be empty

status bit

)

insert into constraints(Name,Email,MobileNo,status)

values('vignesh','vignesh@gmail.com','9094479321',1)

select\*from constraints

insert into constraints(Name,Email,MobileNo,status)

values('vinoth','vignesh@gmail.com','9094479321',1) -------------- unique as been seted to the setting

insert into constraints(Name,Email,status)

values('vignesh','gignesh@gmail.com',1)-------------------- mobile number column should not be empty

Primary key :

Primary key is a constrain it is unique and not null combination

-------one primary key for two column is posibble but one table two primary is not posibble

-------one primary key with same combination is to be known

create table Department

(

ID int primary key , -------tools /designers/prevent saving-- grphically

Departement nvarchar(50),

Manager nvarchar(50)

)

Insert into Department values (1,'IT','Ramya')

Insert into Department values (2,'IT','Radhika')

Insert into Department values (3,'IT','Imbarhim')

Insert into Department values (4,'IT','Suresh')

select \* from Department

Set up the two key for the columns

Create table tblDepartment

(

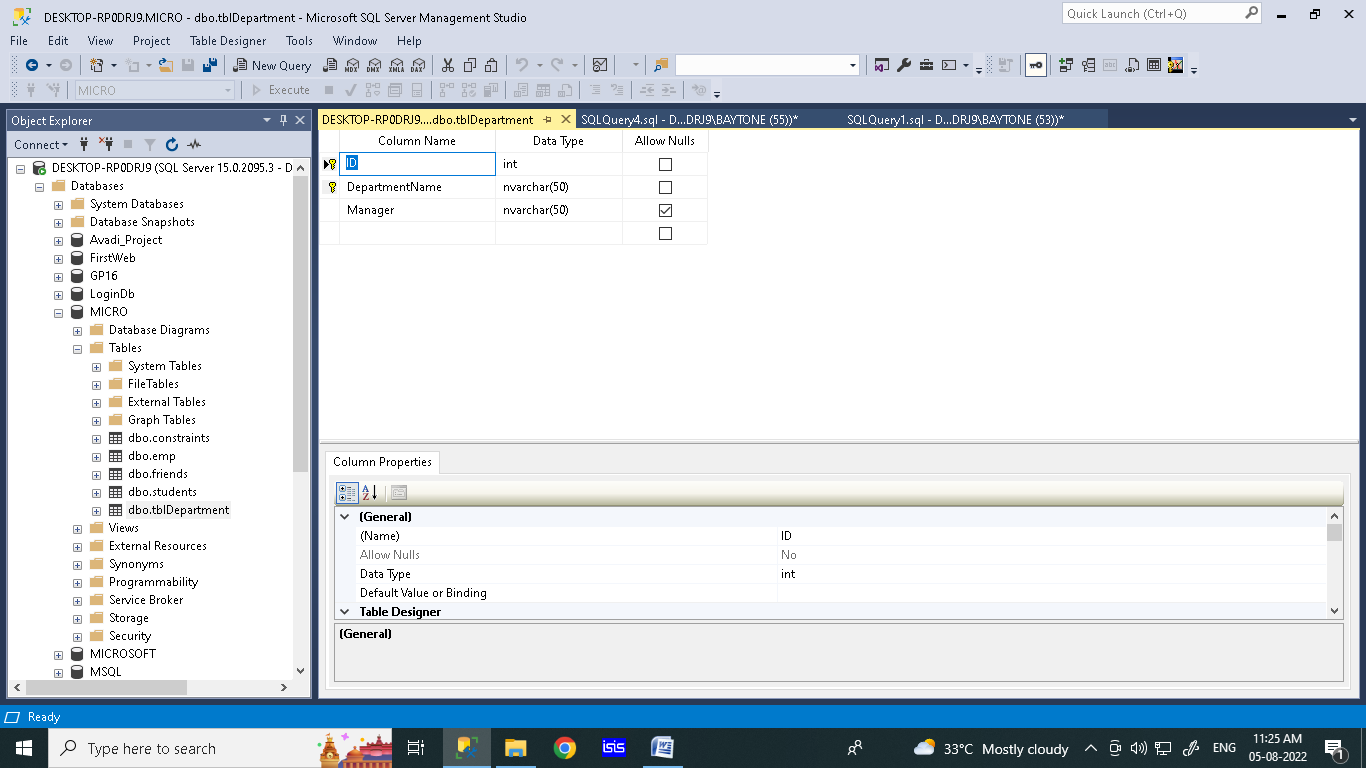
ID int, -------tools /designers/prevent saving-- grphically

DepartmentName nvarchar(50),

Manager nvarchar(50),

Constraint pk\_tblDepartments\_ID primary Key(ID,Departmentname)

)



Right click set the primary key

Default constraints

Default constrain is used for giving a default value that the data is not entered in the particular column is called default contrain

create table tblstaff

(Id int identity primary key ,----identity= values increment & decrement primary key= duplicate

Name nvarchar(50),

Email nvarchar(50),

MobileNo nvarchar(20),

status bit default 1 --------------------bit = true or false (default -1 is makes the empty data entry takes 1 as value

)

insert into tblstaff(Name,Email,MobileNo,status) values('harini','harini@gmail.com','123456789',0)--it takes as per given

insert into tblstaff(Name,Email,MobileNo) values('harini','harini@gmail.com','123456789')--default

insert into tblstaff(Name,Email,mobileno,status) values('harini','harini@gmail.com','123456789',null)--

-------------null is given at boolean it enter as null

select \* from tblstaff

when you create a table without defaut constrain . you can update the table with query

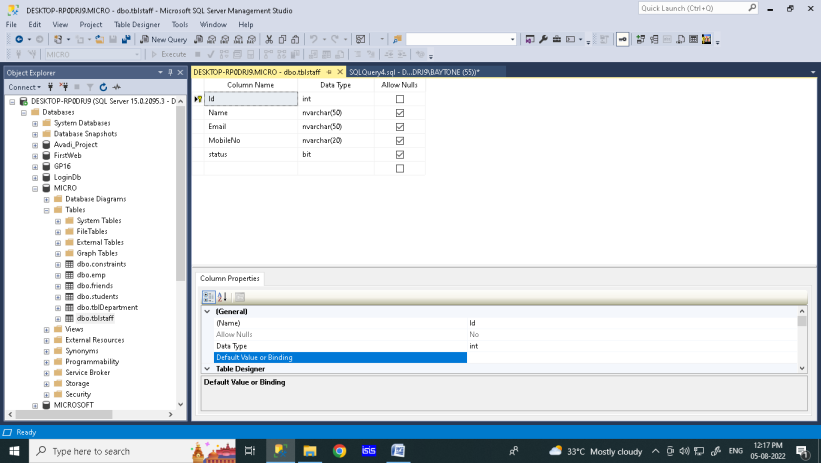
ALTER TABLE tblstaff

ADD CONSTRAINT DF\_tblstaff\_status

DEFAULT 0 for Status

**graphically** can we can set up through settings

right click on the table . Go to design of the table you can see the option **defaut value binding** in below screen shot



create table tblstaff

(Id int identity primary key ,--------------------1

Name nvarchar(50),

Email nvarchar(50),

MobileNo nvarchar(20),

status bit ---------------------without defalt

)

select \* from tblstaff

ALTER TABLE tblstaff

ADD CONSTRAINT DF\_tblstaff\_status-----------------2

DEFAULT 0 For status

insert into tblstaff(Name,Email,MobileNo,status) values('harini','harini@gmail.com','123456789',0)

insert into tblstaff(Name,Email,MobileNo) values('harini','harini@gmail.com','123456789')-------------------3

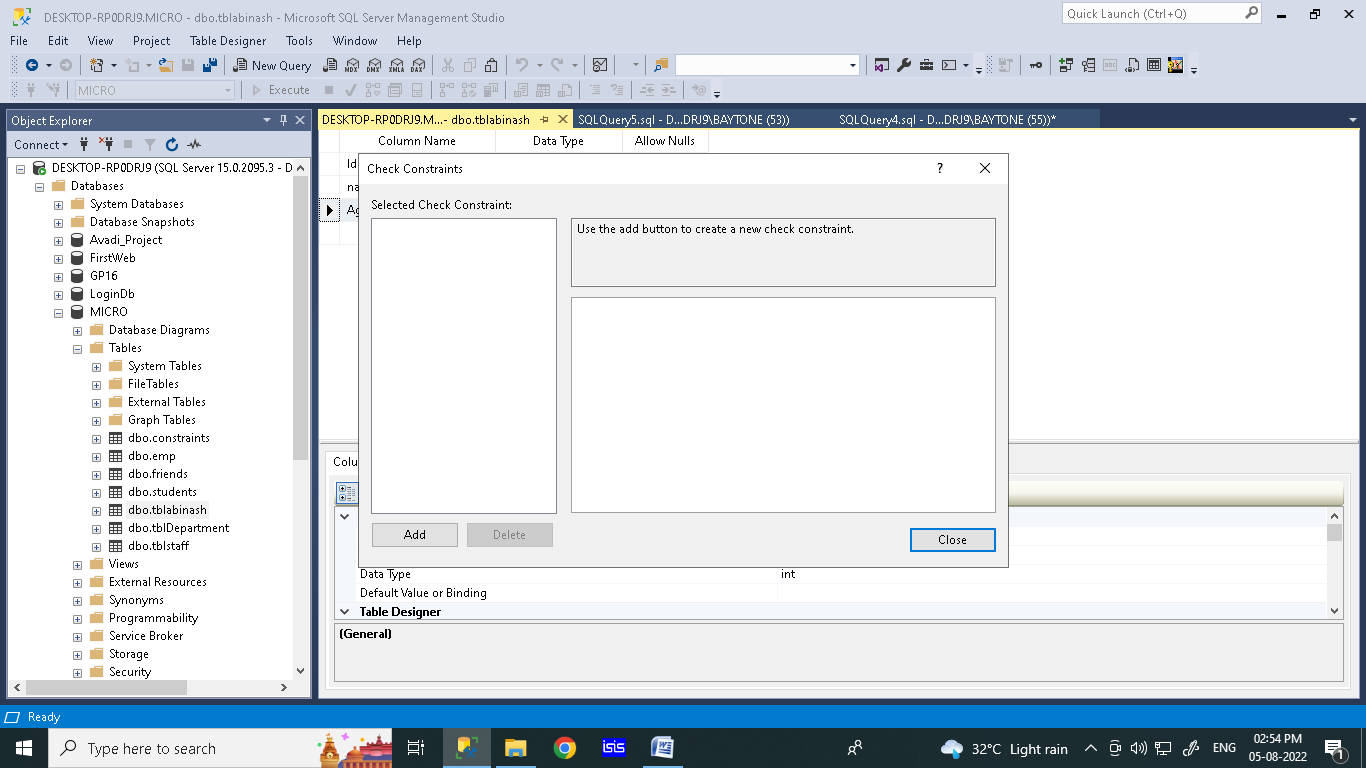
insert into tblstaff(Name,Email,mobileno,status) values('harini','harini@gmail.com','123456789',null)

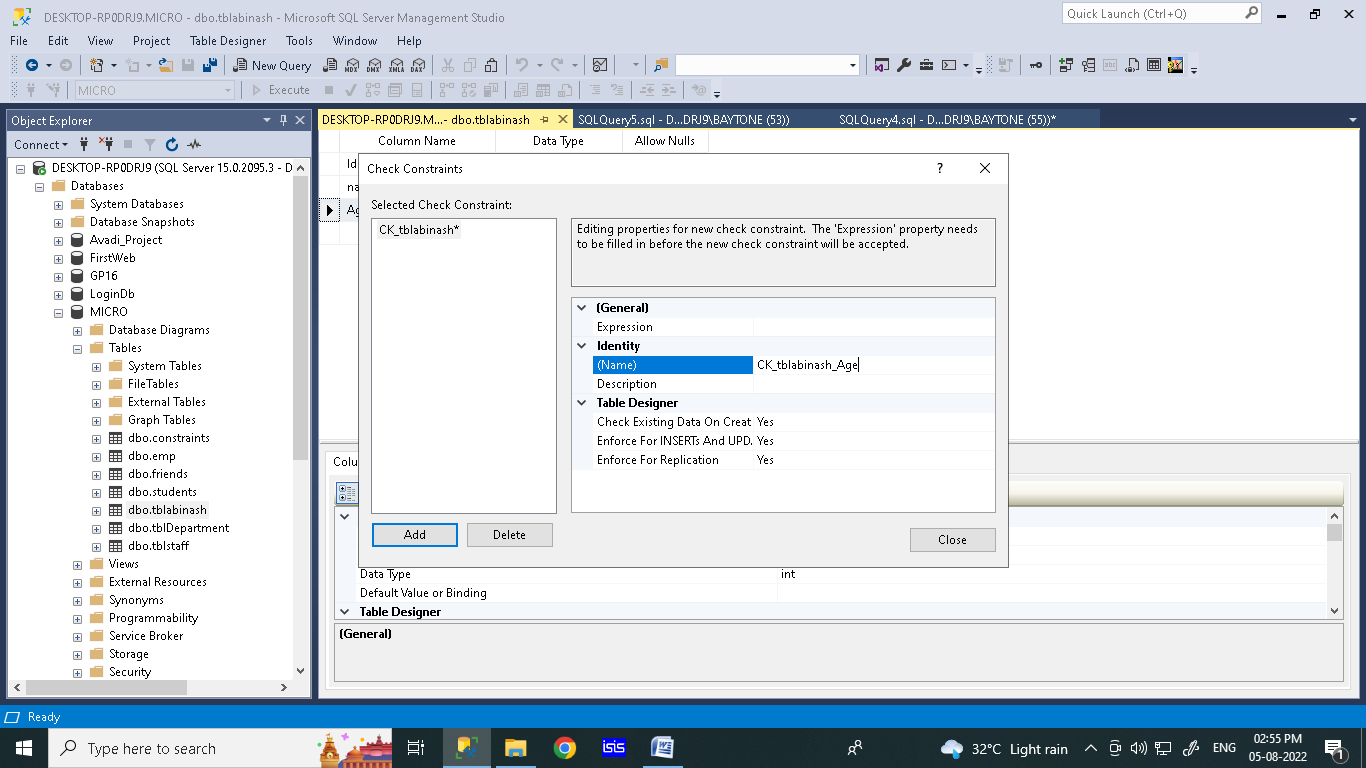
-------------null is given at boolean it enter as null

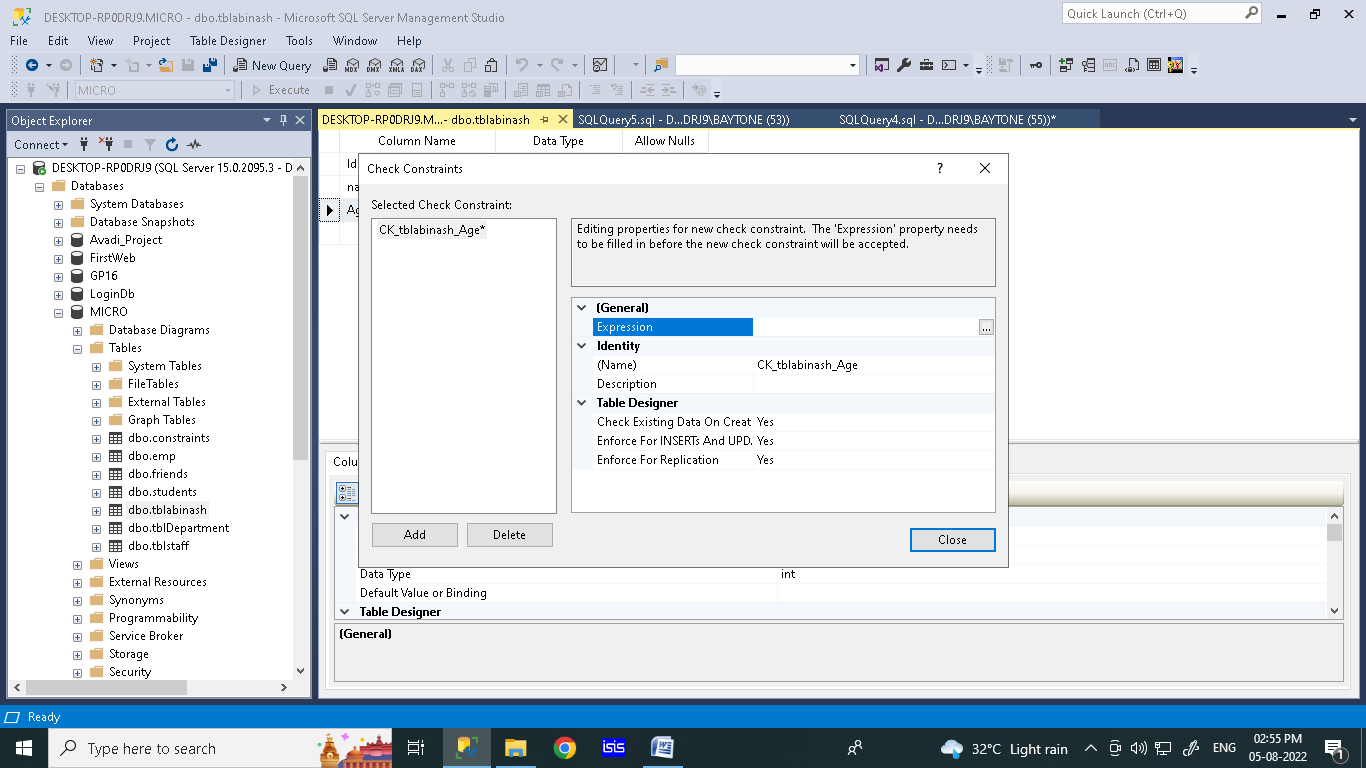
Check constraints

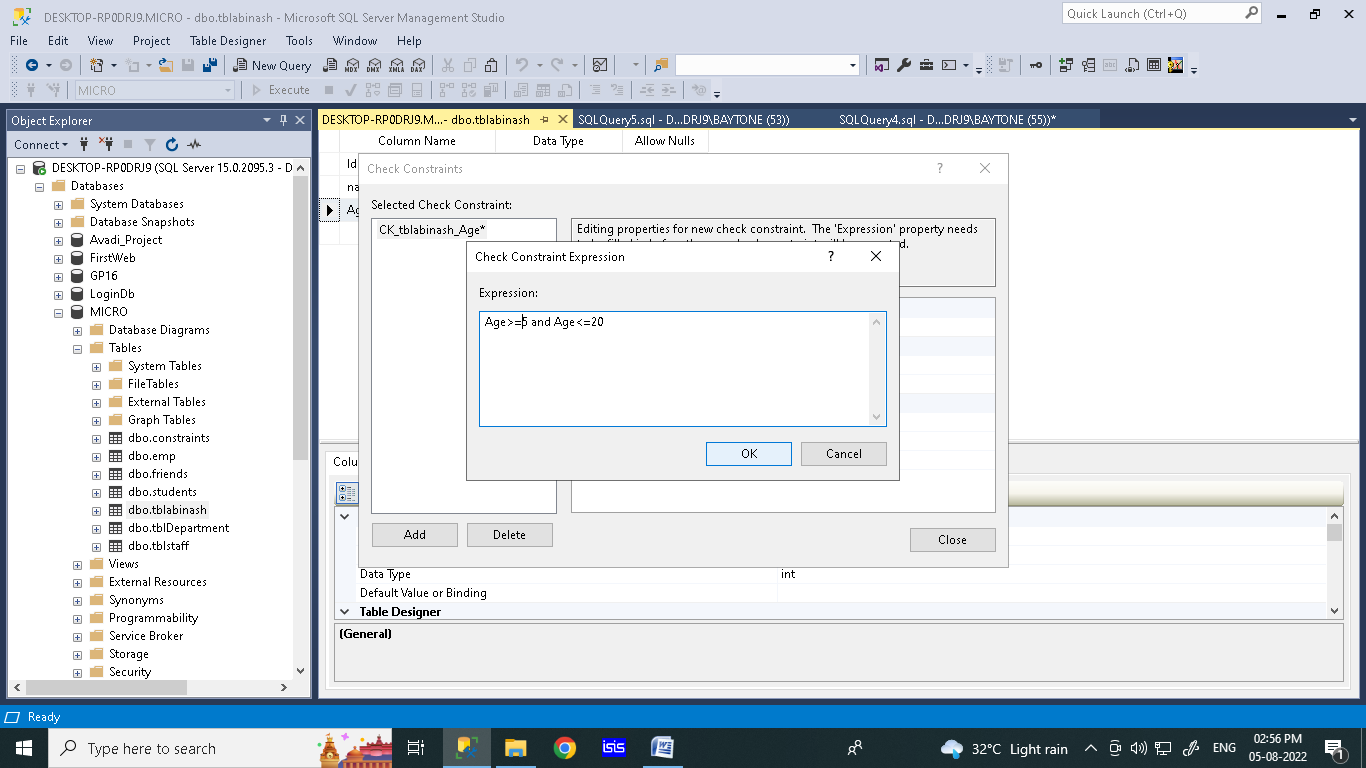
Check constraint is created before creating a table . it used to say setting a limit a particular column .

Example; Age-limit,Salary-maximum,Character-length









create table tblabinash

(Id int identity,

name nvarchar(50),

Age int

)

insert into tblabinash(name,Age) values ('ram',15)---------possible

insert into tblabinash(name,Age) values('ABI',21)----------not possible because the age limit is blocked

query to set check constraint

create table tblabinash

(Id int identity,

name nvarchar(50),

Age int

)

alter table tblabinash

add constraint ck\_tbleabinash\_Age check(Age > 5 AND Age < 20)

insert into tblabinash(name,Age) values ('ram',15)---------possible

insert into tblabinash(name,Age) values('ABI',21)----------not possible because the age limit is blocked

select \* from tblabinash

Foreign Key

The foreign key creating between the two tables . making a relationship with two table

Graphically method

create table tbldepartemnt

(

ID int primary key,

Departmentname nvarchar(50),

manager nvarchar (50)

)

create table tblemployee

(

ID int primary key,

name varchar(50),

gender nvarchar(50),

salary int,

department int

)

Insert into tbldepartemnt values (1,'IT','Ramya')

Insert into tbldepartemnt values (2,'csc','Radhika')

Insert into tbldepartemnt values (3,'EEE','Imbarhim')

Insert into tbldepartemnt values (4,'MECH','Suresh')

select \* from tbldepartemnt

Insert into tblemployee values (1,'Ahamed','Male',14000,1)------------------1

Insert into tblemployee values (2,'harini','female',15000,3)

Insert into tblemployee values (3,'diviya','female',16000,2)

Insert into tblemployee values (4,'priya','female',17000,4)

Insert into tblemployee values (5,'priya','female',17000,9)-------------------- after etting changes 2 not executed because of foreign key

select \* from tblemployee

truncate table tblemployee

**SQL | Join (Inner, Left, Right and Full Joins)**

**SQL Join** statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are as follows:

* INNER JOIN
* LEFT JOIN
* RIGHT JOIN
* FULL JOIN

### ****A. INNER JOIN****

The INNER JOIN keyword selects all rows from both the tables as long as the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

**Syntax**:

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

INNER JOIN table2

ON table1.matching\_column = table2.matching\_column;

**table1**: First table.

**table2**: Second table

**matching\_column**: Column common to both the tables.

***Note****: We can also write JOIN instead of INNER JOIN. JOI*



**Example Queries(INNER JOIN)**

This query will show the names and age of students enrolled in different courses.

SELECT StudentCourse.COURSE\_ID, Student.NAME, Student.AGE FROM Student

INNER JOIN StudentCourse

ON Student.ROLL\_NO = StudentCourse.ROLL\_NO;

### ****B. LEFT JOIN****

This join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain *null*. LEFT JOIN is also known as LEFT OUTER JOIN.

**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

LEFT JOIN table2

ON table1.matching\_column = table2.matching\_column;

table1: First table.

table2: Second table

matching\_column: Column common to both the tables.

***Note****: We can also use LEFT OUTER JOIN instead of LEFT JOIN, both are the same.*



**Example Queries(LEFT JOIN)**:

SELECT Student.NAME,StudentCourse.COURSE\_ID

FROM Student

LEFT JOIN StudentCourse

ON StudentCourse.ROLL\_NO = Student.ROLL\_NO;

### ****C. RIGHT JOIN****

RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain *null*. RIGHT JOIN is also known as RIGHT OUTER JOIN.

**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

RIGHT JOIN table2

ON table1.matching\_column = table2.matching\_column;

table1: First table.

table2: Second table

matching\_column: Column common to both the tables.

***Note****: We can also use RIGHT OUTER JOIN instead of RIGHT JOIN, both are the same.*



**Example Queries(RIGHT JOIN)**:

SELECT Student.NAME,StudentCourse.COURSE\_ID

FROM Student

RIGHT JOIN StudentCourse

ON StudentCourse.ROLL\_NO = Student.ROLL\_NO;

### ****D. FULL JOIN****

FULL JOIN creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain *NULL* values.



**Syntax:**

SELECT table1.column1,table1.column2,table2.column1,....

FROM table1

FULL JOIN table2

ON table1.matching\_column = table2.matching\_column;

table1: First table.

table2: Second table

matching\_column: Column common to both the tables.

**Example Queries(FULL JOIN)**:

SELECT Student.NAME,StudentCourse.COURSE\_ID

FROM Student

FULL JOIN StudentCourse

ON StudentCourse.ROLL\_NO = Student.ROLL\_NO;

# Union and union all

create table batch1

(

ID int primary key,

name varchar(50),

gender nvarchar(50),

salary int,

department int

)

Insert into batch values (1,'IT','Ramya')

Insert into batch1 values (2,'csc','Radhika')

Insert into batch1 values (3,'EEE','Imbarhim')

Insert into batch1 values (4,'MECH','Suresh')

select \* from batch1