Class\_15

--Exist and Not Exist

--EXIST is used to check whether the result of co-related nested query is emptyor not.

--Exist(S)

--TRUE: S has atleast one row/record

--FALSE : S has no row/record.

--NOT EXIST(S)

--TRUE:S has no row/record.

--FALSE :S has atleast one row/record

Create Table customer(C\_ID varchar(5) primary key ,CNAME varchar(20),Loc varchar(20))

insert into customer values('C1','AMIT','PUNE')

insert into customer values('C2','Sumit','Delhi')

insert into customer values('C3','varun','Mumbai')

insert into customer values('C4','snehal','Latur')

insert into customer values('C5','Raj','Sangli')

insert into customer values('C6','Mohit','Satara')

select \* from customer

create table orders(OID int primary key, CID varchar(5),groceries varchar(20))

insert into orders values(1,'C2','almonds')

insert into orders values(2,'C3','deo')

insert into orders values(3,'C1','milk')

insert into orders values(4,'C2','soap')

insert into orders values(5,'C4','dishes')

insert into orders values(6,'C2','rice')

select \* from orders

select \* from customer C where exists (select \* from orders O where C.C\_ID =O.CID )

select \* from customer C where not exists (select \* from orders O where C.C\_ID =O.CID )

--Sub query and Co-Relational Query

--Sub query(Nested subquery)

--Query within query i.e outer query(OQ) and inside inner query(IQ).

--OQ and IQ is independent.

--It follows bottom up approach

--Inside Subquery, IQ always execute only once.

select \* from customer where C\_ID in (select CID from orders) --(C2,C3,C1,C2,C4,C2)

--Co-relational query

--Query within query i.e outer query(OQ) and inside inner query(IQ).

--IQ is dependent on outer query.

--It follows top down up approach.

select \* from customer C where exists (select \* from orders O where C.C\_ID =O.CID )

--Q.What is the diffrence between Sub query and Co-relational query.

--------------------------------------------------------------------

Class\_15

ROW NUMBER

--It will return the sequential number of row starting at 1

--Order by clause is required.

--PARTITION BY clause is optional

--When the data is partitioned, row number reset to 1 when the partition changes.

--syantx

--ROW\_NUMBER() OVER(ORDER BY Col1,col2)

select \*, ROW\_NUMBER() over (order by Esal ) as RowNo from Info\_3;

---RANK,DENSE RANK AND ROW NUMBER

--Rank() and Dense\_Rank()

--It will return a rank starting at 1 based on ordering of rows and imposed by order by clause.

--Order by clause is required mandatory.

--PARTITION BY Clause is optional.

--Rank Syntax: RANK() OVER (ORDER BY col1,col2,....coln ASC/DESC [PARTITION BY Col1,col2...coln])

--Dense\_Rank Syntax: DENSE\_RANK() OVER (ORDER BY col1,col2,....coln ASC/DESC [PARTITION BY Col1,col2...coln])

--example

--Marks =496,496,495,494,494,490

--rank = 1,1,3,4,4,6

--Dense\_rank = 1,1,2,3,3,4

--Example:

--[sal] = [1000,1000,2000,3000,4000]

--Rank() -- [1,1,3,4,5]

--Dense\_rank() --[1,1,2,3,4] -- school level mark inside the class

select \* ,rank() over (order by Esal) as rank1 from Info\_3;

select \* ,dense\_rank() over (order by Esal) as denserank from info\_3;

--Q. What is the difference between Rank() and Dense\_Rank()

--Rank() -- Rank function skips ranking if there is same value or number.

--Dense\_Rank() --It will not skips ranking if there is same value or number.

--2nd highest salary by using rank()

select \* ,rank() over (order by salary) as rank1 from over\_Test where rank() over (order by salary) = 2

select \* ,dense\_rank() over (order by salary) as denserank from over\_Test where dense\_rank() over (order by salary)=2

--The above query will through an exception in

--i.e. Windowed functions can only appear in the SELECT or ORDER BY clauses.

--In order to avoid this kind of exception or Error in SQl we have to use CTE i.e. COMMON TBALE EXPRESSION

--CTE (Common Table Expression)

--It is temporary result set.

--It will store the temporary results to make use of that in your main query.

--It can be referred within a SELECT, INSERT,UPDATE and DELETE statements that immediately follows the CTE.

--Only DML type of operation we can perform on CTE

--Syntax

--With CTE\_NAME (COL1, COL2 ...etc)

--AS

--CTE\_Query

Q. Find out 4th max sal

with new\_rank as

(select Esal, dense\_rank() over (order by Esal desc) as ranking from info\_3)

select \* from new\_rank where ranking=4;

Q.Find out 3rd min sal

with new\_rank as

(select Esal, dense\_rank() over (order by Esal asc) as ranking from info\_3)

select \* from new\_rank where ranking=3;

Q. Find out 65th max salary.

Q. Find out 30th min salary.

Q. Display top 5 salary from table.

Q. Display salary from 3rd max to 6th max.

Q. Display salary other than 2nd min to 4th min.

# Partition by

select \*, ROW\_NUMBER() over (partition by salary order by salary ) as RowNo from over\_Test

Q. create table over\_Test(EMPID int, FirstName varchar(20),Gender varchar(2),salary int)

insert into over\_Test values(1,'Mohini','F',1000)

insert into over\_Test values(2,'Rohit','M',2000)

insert into over\_Test values(3,'Amit','M',4000)

insert into over\_Test values(4,'Sonal','F',5000)

insert into over\_Test values(5,'Minal','F',6000)

insert into over\_Test values(6,'Amar','M',3600)

insert into over\_Test values(7,'Shital','F',4500)

insert into over\_Test values(8,'Sohil','M',6000)

insert into over\_Test values(9,'praveen','F',9000)

insert into over\_Test values(10,'Mithali','F',9000)

insert into over\_Test values(11,'seema','F',9000)

insert into over\_Test values(12,'meena','F',10000)

select \* from over\_Test

Q. How to find and how to delete duplicate records from table using row\_number?

------------------------------------------------------------------------------------------------

Class\_16

--SQL SERVER FUNCTIONS

--1.UPPER()

--UPPER() Function converts the value of field/Column to upper case.

--The upper case function requires 1 argument

--syntax :upper (text/column name)

select UPPER('scodeen global')

select \* from INFORMATION\_SCHEMA.tables

select \*,UPPER(firstname) as UPPERCASE from employee

--2.LOWER()

--Lower() Function converts the value of field/Column to lower case.

--The lower case function requires 1 argument

--syntax :lower (text/column name)

select LOWER('SCODEEN GLOBAL')

select \*,LOWER(LastName) as lowercase from employee

--3.Substring

--The substring function used to extract charecter from text field

--Synatx : substring(Column\_Name,Start,end[lenth]) from table\_Name

--Ex: substring ( 'varchar',int,int)

select SUBSTRING('SCODEEN',3,2)

select \*,SUBSTRING(FirstName,1,1) as FisrtLetter from employee

select \*,SUBSTRING(FirstName,5,len(firstname)) as FisrtLetter ,len(firstname) as lenths from employee

--4.DATALENGTH() and LEN()

--This function returns the number of bytes used to reprsent the expression.

--Syntax : DATALENGTH(string),LEN(String/Column\_name)

create table length\_check (Lid int, Lname char(20))

insert into length\_check values(1,'Praveen')

insert into length\_check values(2,'Amit')

insert into length\_check values(3,'Meena')

insert into length\_check values(4,'Sohan')

insert into length\_check values(5,'Rajni')

select \*,LEN(Lname) as lengths from length\_check

select \*,datalength(Lname) as datalengths from length\_check

select \*,LEN(Lid) as lengths from length\_check

select \*,datalength(Lid) as datalengths from length\_check

--5.Reverse()

--The REVERSE() function reverses a string and returns the result.

--synatx : REVERSE(string)

select REVERSE('PUNE')

select REVERSE('MITHALI')

--6.REPLACE()

--The REPLACE() function replaces all occurrences of a substring within a string, with a new string.

--Note: The search is not case-insensitive.

--Syntax - REPLACE(string, old\_string, new\_string)

-- A-a , B-b meaning is same in replace function.

select replace ('SCODEen','E','M')

--7.REPLICATE()

--The REPLICATE() function repeats a string a specified number of times.

--Syntax :REPLICATE(string, integer)

select replicate ('SODEEN ',4)

--8.CONCAT() , CONCAT with

--The CONCAT() function adds two or more strings together.

--Syntax: CONCAT(string1,string2....)

--The + operator allows you to add two or more strings together.

--syntax:string1 + string2 + string\_n

select 'Scodeen' +' ' +'Global' + ' '+'PUNE'

select \* from employee

select EID,concat(firstName,' ',Lastname) as Full\_name from employee

select EID,firstName +' '+ Lastname as Full\_name from employee

--9.LTRIM(), RTRIM() and TRIM()

--The LTRIM() function removes leading spaces from a string.

--The RTRIM() function removes trailing spaces from a string.

--TRIM() function removes leading as well as trailing spaces from string.

select len(' LTRIM')

select len(LTRIM(' LTRIM') )

select len(' LTRIM ')

select Rtrim(' SCODEEN ') --'LTRIM '

select ltrim(' LTRIM ')

select trim(' LTRIM ')

10.Round

--The ROUND() function rounds a number to a specified number of decimal places.

--Syntax : ROUND(NUMERIC\_EXPRESSION, length, [(function)])

--NUMERIC\_EXPRESSION : it takes the number to be roundoff.

--Length : the number of digits that we want to round off.

-- if length is +ve then rounding is applied after decimal and if length is -ve the before decimal

--function : is used to indicate rounding or truncation operation. 0 -indicates rounding and non-zero indicates truncation, by default it is 0.

select ROUND('value',1) -- Exception

select ROUND(74.4,0)

select ROUND(789.56,-2)

select ROUND(78.56,1)

select ROUND(78.56,2)

select ROUND(78.467,0)

select round(750.556,2) -- Round to 2 places after the decimal point

select round(750.556,2,1) --Truncate anything after 2 places after the decimal point.

select round(750.4456666,-2,1)

select round(750.4456666,3)

select round(750.4456666,3,1)

select round(750.4454666,3)

11.CONVERT()

--The CONVERT() function converts a value (of any type) into a specified datatype.

--Syntax :CONVERT(data\_type[(length)], expression/Col\_NAME, [(style)])

create table DOJ (id int, NAME varchar(20),DOJ datetime)

insert into DOJ values (1,'Mansa','2020-01-01 10:10:10')

insert into DOJ values (2,'Vasavi','2015-06-01 10:20:10')

insert into DOJ values (3,'Pravlika','2014-04-01 11:10:10')

insert into DOJ values (4,'Jyoti','2017-08-01 12:10:10')

insert into DOJ values (5,'Pushpa','2016-05-01 01:23:10')

insert into DOJ values (6,'Seema',GETDATE())

select \* from DOJ

select GETDATE()

select convert(varchar,getdate(),2)

select \*, convert(varchar,DOJ) as NewCreatedDOJ from DOJ

select \*, convert(varchar(11),DOJ,102) as NewCreatedDOJ from DOJ

select \*, convert(varchar(11),DOJ,2) as NewCreatedDOJ from DOJ