- Built in Functions

=====================

Inputs (Arguments)

Output (Result)

Two Types:

Single # STUDY

each result for each row

Multiple(Aggregate Functions) # DONE

result for some rows

==================================

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

Will Study ...Single Functions

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

input can be Column or (Expresssion --> Fname + ' ' + Lname).

can use nested functions.

Make Conditions with Where

Character Functions

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

lower() upper() ==> return small case string or capital one

len() ==> return length of eaxh column value

left(column,number) right(column,number) ==> return left or right number of letters of value of column

substring(column1,start index,length to return) ==> return substring

concat(any number of char inputs) ==> return full string after concatination

\*NOTE\* :DO NOT RETURN NULL IF THERE IS A VALUE EXIST AND THE OTHER MISSING

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

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

Number Functions

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

round (value, number of decimal numbers) ==> return value with specific decimal numbers after the decimal point.

floor (value) ==> return integer number

square (value) ==> return the value power 2

sqrt (value) ==> return the square root of the value

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

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

Date Functions

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

day(), month(), year() ==> return each category of date individually.

\*NOTE\* :MONTH FUNCTION RETURN THE NUMBER OF MONTH

getdate() ==> return now date

datename(dw, bdate) ==> return the name of (day, month) in the bdate

dw=date of week

month=date of month

dateadd(day, 25, bdate) ==> return the new date after adding the number of days or months on the bdate

day ==> number of days,

month ==> number of months,

year ==> number of years.

\*NOTE\* :CAN USE BOTH, ADD OR MINUS NUMBERS

\*NOTE\* :

first date --> 1/1/1900

diff between dates --> return number of days from the specific date to the first date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

RankingFunctions

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

row\_number() over(order by 'column') ==> return column numbering of rows based on order by column "Unique Values Without Escaping"

rank () over(order by 'column') ==> return column numbering of rows based on order by column "Duplicate Values With Escaping"

dense\_rank() over(order by 'column') ==> return column numbering of rows based on order by column "Duplicate Values Without Escaping"

ntile (number) over()order by salary desc) ==> take 1 Argument, return table with number of groups

can use also 'Partition By' + 'Order By' using it The Order will be based on the Column in Partition by

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

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

Other Functions

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

convert(new data type, column) ==> return a casting column to another data type

can not convert varcahr to numbering or date

isnull(column, updated value) ==> return new value in the column if NULL with the updated value

\*NOTE\* :1) CAN ONLY PUT THE UPDATED VALUE WITH THE SAME DATA TYPE AS THE COLUMN

OR

USE CONVERT() FUNCTION TO CONVERT THE COLUMN INTO UPDATED VALUE DATA TYPE

2) CAN USE THE UPDATED VALUE WITH COLUMN NAME

coalesce(any number of columns, updated value) ==> return the new value in the column if NULL Exist,

and take the columns in order till acheive the updated value

\*NOTE\* :avg(salary) ==> return average of salaries, without counting nulls.

use isnull(), and put updated value to make it count and calc avg correctly

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

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