# DATE:

Replace, len, left , right :

Replace(id, “”, “’)

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

Extract values from the mail id:

Eg: rakeshkamboj26@gmail.com

Left(email, len(email), 11)

Output will be rakeshkamboj26

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

Right(‘00000000000’ + phone\_no, 10) : it will add zero in the right side if there lengthh is less then 10.

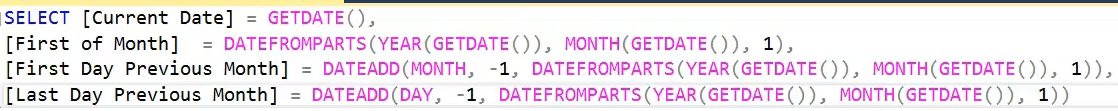
----

## Datefromparts:

Datefromparts(2013,1,1) --> get date : 2013-1-1

MONTH(GETDATE()) 🡪 return month

## Dateadd





select GETDATE(),

DATEFROMPARTS(2016,1,1),

DATEADD(YEAR, 1, DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), DAY(GETDATE()))),

DATEADD(YEAR, -11, DATEFROMPARTS(YEAR(GETDATE()), MONTH(GETDATE()), DAY(GETDATE())))

## DateDiff

DateDiff(DAY, orderdate, shipdate)

## Cast

Cast(GetDate() – 2222 as DATE)

ISNULL([Title], ‘No Title’)

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

Select first\_name =

case

when ID = 2 then 'A'

when ID = 3 then 'b'

else 'd'

end

# union

union only gives answer in unique result.

Select o\_id, date from order

Union

Select order\_id, date from purhase

* We will get unique based on the both column
* Both query should have same number of the column, order of column and also data type
* In final output we will get o\_id as column

Union all : also allow to add duplicate in the final output

Always try to avoid the inner join where we have already applied left join because it will remove common null.

# Over:

* It will be used with aggregate function
* It will maintain row level vis ibility.
* Window function : allow us to include aggregate calculation in your queriees without otherwise changing the output in any way.
* It just add new column in the table also don’t collapse the reaminng row as in group by.

Select sum(amount) over() order:

**It will return same amount for all the row which is sum of all amount**.

## Partition by

* Partition is work as group by

Select sum(amount) over(partition by id) order:

It will return same amount for same id which is sum of all amount for that id

## Row\_number

* Row\_number we rank the record but we have to careful to add order by
* Order by in mandatory
* It will different rank for same value apperaing multiple time

Select row\_number() over(partition by id order by name) order

1 100

2 100

3 100

4 110

## Rank

It wil give same rank to same values but it will skip next number until it will reach next value

Like

1 100

1 100

1 100

4 110

## Dense rank

It will resolve above issur and give rank in the order

1 100

1 100

1 100

2 110

# Correlated query:

Select Id, Salary From table

# outer query retrun id, salary for each row of the table

# which can be used directly in the subquery

# JOIN

Good topic to cover

[SQL for Data Analysis - with SQL Server | Udemy](https://www.udemy.com/course/sql-basics-crash-course-with-sql-server/learn/lecture/21829016#overview)

# Pivot

Select \* from

(

SELECT color, ReorderPoint

FROM [AdventureWorks2019].[Production].[Product]

) A

PIVOT(

SUM(ReorderPoint)

for color in([Black], [Silver])

) B



## Group by with pivot:

Additional columns in subquery is working as group by in below query makeflag works as groupby.

Adding extra column in subquery can lead to wrong result.

/\*\*\*\*\*\* Script for SelectTopNRows command from SSMS \*\*\*\*\*\*/

Select

[Make Flag] = MakeFlag,

Silver,

Black

from

(

SELECT color, ReorderPoint, MakeFlag

FROM [AdventureWorks2019].[Production].[Product]

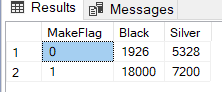
) A

PIVOT(

SUM(ReorderPoint)

for color in([Black], [Silver])

) B



Do query : [The Advanced SQL Server Masterclass For Data Analysis | Udemy](https://www.udemy.com/course/advanced-sql-server-masterclass-for-data-analysis/learn/lecture/26665464#overview)

# CTE

## Top 10 total due of month and previous month in two columns

select A.OrderOfMonth, [Current Month] = A.TotalDue, [Previous Month] = B.TotalDue from

(

select

OrderOfMonth,

TotalDue = sum(TotalDue)

from(

SELECT

TotalDue,

OrderDate

,OrderOfMonth = DATEFROMPARTS(YEAR(OrderDate), MONTH(OrderDate), 1)

,rank1 = ROW\_NUMBER() over(PARTITION BY DATEFROMPARTS(YEAR(OrderDate), MONTH(OrderDate), 1) order by TotalDue DESC)

FROM [AdventureWorks2019].[Sales].[SalesOrderHeader]

) X where rank1 <= 10 group by OrderOfMonth

) A

LEFT JOIN

(

select

OrderOfMonth,

TotalDue = sum(TotalDue)

from(

SELECT

TotalDue,

OrderDate

,OrderOfMonth = DATEFROMPARTS(YEAR(OrderDate), MONTH(OrderDate), 1)

,rank1 = ROW\_NUMBER() over(PARTITION BY DATEFROMPARTS(YEAR(OrderDate), MONTH(OrderDate), 1) order by TotalDue DESC)

FROM [AdventureWorks2019].[Sales].[SalesOrderHeader]

) Y where rank1 <= 10 group by OrderOfMonth

) B on A.OrderOfMonth = DATEADD(MONTH, 1, b.OrderOfMonth)

Order by OrderOfMonth

Above query can be shorten by the CTE.

WITH Sales AS(

SELECT

TotalDue,

OrderDate

,OrderOfMonth = DATEFROMPARTS(YEAR(OrderDate), MONTH(OrderDate), 1)

,rank1 = ROW\_NUMBER() over(PARTITION BY DATEFROMPARTS(YEAR(OrderDate),

MONTH(OrderDate), 1) order by TotalDue DESC)

FROM [AdventureWorks2019].[Sales].[SalesOrderHeader]

),

Top10Sales AS

(

select

OrderOfMonth,

TotalDue = sum(TotalDue)

from Sales

where rank1 <= 10 group by OrderOfMonth

)

Select

A.OrderOfMonth,

[Current Month] = A.TotalDue,

[Previous Month] = B.TotalDue

from Top10Sales as A

Left join

Top10Sales as B

on A.OrderOfMonth = DATEADD(MONTH, 1, B.OrderOfMonth)

Order by A.OrderOfMonth

## CTE Recursion:

With NumberSeries as

(

Select 1 As MyNumber – 1 as output

UNION ALL

Select MyNumber + 1

From NumberSeries

Where MyNumber < 10 --- it will repeat call to NumberSeries until condition statisfied.

)

Select MyNumber from NumberSeries

## Condition

Can max recursion condition to overcome above issue.

OPTION(MAXRECURSION 365)