In other words, by using the LAG() function, from the current row, you can access data of the previous row, or the row before the previous row, and so on.

Syntax:

LAG(return\_value ,offset [,default])

OVER (

[PARTITION BY partition\_expression, ... ]

ORDER BY sort\_expression [ASC | DESC], ...

)

Return Value: return value of the previous row based on the specified offset.

Offset: number of rows back from the current row to access data.

Default value =1

Default : value to be returned if it is out of rage for a particular offset/

Partition by Clause:

Distributes the row into partitions

Example:

**SELECT**

\*

**FROM**

sales.vw\_netsales\_brands

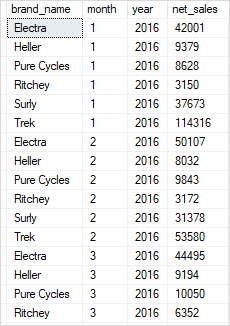
**ORDER** **BY**

**year**,

**month**,

brand\_name,

net\_sales;



**WITH** cte\_netsales\_2018 **AS**(

**SELECT**

**month**,

**SUM**(net\_sales) net\_sales

**FROM**

sales.vw\_netsales\_brands

**WHERE**

**year** = 2018

**GROUP** **BY**

**month**

)

**SELECT**

**month**,

net\_sales,

LAG(net\_sales,1) **OVER** (

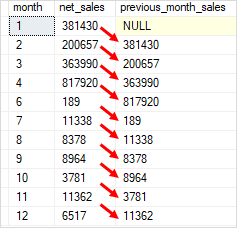
**ORDER** **BY** **month**

) previous\_month\_sales

**FROM**

cte\_netsales\_2018;

Output



For example, by using the LEAD() function, from the current row, you can access data of the next row, or the row after the next row, and so on

LEAD(return\_value ,offset [,default])

OVER (

[PARTITION BY partition\_expression, ... ]

ORDER BY sort\_expression [ASC | DESC], ...

)

**WITH** cte\_netsales\_2017 **AS**(

**SELECT**

**month**,

**SUM**(net\_sales) net\_sales

**FROM**

sales.vw\_netsales\_brands

**WHERE**

**year** = 2017

**GROUP** **BY**

**month**

)

**SELECT**

**month**,

net\_sales,

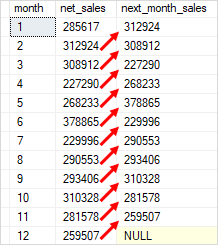
**LEAD**(net\_sales,1) **OVER** (

**ORDER** **BY** **month**

) next\_month\_sales

**FROM**

cte\_netsales\_2017;



SELECT [Year],

       [Quarter],

       Sales,

       LEAD(Sales, 1, 0) OVER(PARTITION BY [Year]

       ORDER BY [Year],

                [Quarter] DESC) AS [NextQuarterSales]

FROM dbo.ProductSales;

