

# ROLLINGSUM Function

## Contents:

- *Basic Usage*
  - *Syntax and Arguments*
    - *col\_ref*
    - *rowsBefore\_integer, rowsAfter\_integer*
  - *Examples*
    - *Example - Rolling window functions*
- 

Computes the rolling sum of values forward or backward of the current row within the specified column.

- If an input value is missing or null, it is not factored in the computation. For example, for the first row in the dataset, the rolling sum of previous values is the value in the first row.
- The row from which to extract a value is determined by the order in which the rows are organized based on the `order` parameter.
- If you are working on a randomly generated sample of your dataset, the values that you see for this function might not correspond to the values that are generated on the full dataset during job execution.
- The function takes a column name and two optional integer parameters that determine the window backward and forward of the current row.
  - The default integer parameter values are `-1` and `0`, which computes the rolling average from the current row back to the first row of the dataset.
- This function works with the following transforms:
  - *Window Transform*
  - *Set Transform*
  - *Derive Transform*

**Wrangle vs. SQL:** This function is part of Wrangle , a proprietary data transformation language. Wrangle is not SQL. For more information, see *Wrangle Language*.

## Basic Usage

### Column example:

```
rollingsum(myCol)
```

**Output:** Returns the rolling sum of all values in the `myCol` column.

### Rows before example:

```
rollingsum(myNumber, 3)
```

**Output:** Returns the rolling sum of the current row and the three previous row values in the `myNumber` column.

### Rows before and after example:

```
rollingsum(myNumber, 3, 2)
```

**Output:** Returns the rolling sum of the three previous row values, the current row value, and the two rows after the current one in the `myNumber` column.

## Syntax and Arguments

```
rollingsum(col_ref, rowsBefore_integer, rowsAfter_integer) order: order_col [group: group_col]
```

| Argument           | Required? | Data Type | Description  |
|--------------------|-----------|-----------|--|
| col_ref            | Y         | string    | Name of column whose values are applied to the function              |
| rowsBefore_integer | N         | integer   | Number of rows before the current one to include in the computation. |
| rowsAfter_integer  | N         | integer   | Number of rows after the current one to include in the computation   |

For more information on the `order` and `group` parameters, see *Window Transform*.

For more information on syntax standards, see *Language Documentation Syntax Notes*.

### col\_ref

Name of the column whose values are used to compute the rolling sum.

- Multiple columns and wildcards are not supported.

#### Usage Notes:

| Required? | Data Type  | Example Value |
|-----------|--|---------------|
| Yes       | String (column reference to Integer or Decimal values) | myColumn      |

### rowsBefore\_integer, rowsAfter\_integer

Integers representing the number of rows before or after the current one from which to compute the rolling sum, including the current row. For example, if the first value is 5, the current row and the five rows before it are used in the computation. Negative values for `k` compute the rolling average from rows preceding the current one.

- `rowBefore=0` generates the current row value only.
- `rowBefore=-1` uses all rows preceding the current one.
- If `rowsAfter` is not specified, then the value 0 is applied.
- If a `group` parameter is applied, then these parameter values should be no more than the maximum number of rows in the groups.

#### Usage Notes:

| Required? | Data Type | Example Value |
|-----------|-----------|---------------|
| No        | Integer   | 4             |

### Examples

**Tip:** For additional examples, see *Common Tasks*.

### Example - Rolling window functions

This example describes how to use the rolling computational functions:

- ROLLINGSUM - computes a rolling sum from a window of rows before and after the current row. See *ROLLINGSUM Function*.
- ROLLINGAVERAGE - computes a rolling average from a window of rows before and after the current row. See *ROLLINGAVERAGE Function*.
- ROWNUMBER - computes the row number for each row, as determined by the ordering column. See *ROWNUMBER Function*.

The following dataset contains sales data over the final quarter of the year.

**Source:**

| Date     | Sales |
|----------|-------|
| 10/2/16  | 200   |
| 10/9/16  | 500   |
| 10/16/16 | 350   |
| 10/23/16 | 400   |
| 10/30/16 | 190   |
| 11/6/16  | 550   |
| 11/13/16 | 610   |
| 11/20/16 | 480   |
| 11/27/16 | 660   |
| 12/4/16  | 690   |
| 12/11/16 | 810   |
| 12/18/16 | 950   |
| 12/25/16 | 1020  |
| 1/1/17   | 680   |

**Transformation:**

First, you want to maintain the row information as a separate column. Since data is ordered already by the `Date` column, you can use the following:

|                            |              |
|----------------------------|--------------|
| <b>Transformation Name</b> | Window       |
| <b>Parameter: Formulas</b> | ROWNUMBER( ) |
| <b>Parameter: Order by</b> | Date         |

Rename this column to `rowId` for week of quarter.

Now, you want to extract month and week information from the `Date` values. Deriving the month value:

|                                |                    |
|--------------------------------|--------------------|
| <b>Transformation Name</b>     | New formula        |
| <b>Parameter: Formula type</b> | Single row formula |
| <b>Parameter: Formula</b>      | MONTH(Date)        |

|                                   |         |
|-----------------------------------|---------|
| <b>Parameter: New column name</b> | 'Month' |
|-----------------------------------|---------|

Deriving the quarter value:

|                                   |                            |
|-----------------------------------|----------------------------|
| <b>Transformation Name</b>        | New formula                |
| <b>Parameter: Formula type</b>    | Single row formula         |
| <b>Parameter: Formula</b>         | (1 + FLOOR(((month-1)/3))) |
| <b>Parameter: New column name</b> | 'QTR'                      |

Deriving the week-of-quarter value:

|                            |              |
|----------------------------|--------------|
| <b>Transformation Name</b> | Window       |
| <b>Parameter: Formulas</b> | ROWNUMBER( ) |
| <b>Parameter: Group by</b> | QTR          |
| <b>Parameter: Order by</b> | Date         |

Rename this column WOQ (week of quarter).

Deriving the week-of-month value:

|                            |              |
|----------------------------|--------------|
| <b>Transformation Name</b> | Window       |
| <b>Parameter: Formulas</b> | ROWNUMBER( ) |
| <b>Parameter: Group by</b> | Month        |
| <b>Parameter: Order by</b> | Date         |

Rename this column WOM (week of month).

Now, you perform your rolling computations. Compute the running total of sales using the following:

|                            |                          |
|----------------------------|--------------------------|
| <b>Transformation Name</b> | Window                   |
| <b>Parameter: Formulas</b> | ROLLINGSUM(Sales, -1, 0) |
| <b>Parameter: Group by</b> | QTR                      |
| <b>Parameter: Order by</b> | Date                     |

The -1 parameter is used in the above computation to gather the rolling sum of all rows of data from the current one to the first one. Note that the use of the QTR column for grouping, which moves the value for the 01/01/2017 into its own computational bucket. This may or may not be preferred.

Rename this column QTD (quarter to-date). Now, generate a similar column to compute the rolling average of weekly sales for the quarter:

|                            |                                     |
|----------------------------|-------------------------------------|
| <b>Transformation Name</b> | Window                              |
| <b>Parameter: Formulas</b> | ROUND(ROLLINGAVERAGE(Sales, -1, 0)) |

|                            |      |
|----------------------------|------|
| <b>Parameter: Group by</b> | QTR  |
| <b>Parameter: Order by</b> | Date |

Since the ROLLINGAVERAGE function can compute fractional values, it is wrapped in the ROUND function for neatness. Rename this column avgWeekByQuarter.

**Results:**

When the unnecessary columns are dropped and some reordering is applied, your dataset should look like the following:

| Date     | WOQ | Sales | QTD  | avgWeekByQuarter |
|----------|-----|-------|------|------------------|
| 10/2/16  | 1   | 200   | 200  | 200              |
| 10/9/16  | 2   | 500   | 700  | 350              |
| 10/16/16 | 3   | 350   | 1050 | 350              |
| 10/23/16 | 4   | 400   | 1450 | 363              |
| 10/30/16 | 5   | 190   | 1640 | 328              |
| 11/6/16  | 6   | 550   | 2190 | 365              |
| 11/13/16 | 7   | 610   | 2800 | 400              |
| 11/20/16 | 8   | 480   | 3280 | 410              |
| 11/27/16 | 9   | 660   | 3940 | 438              |
| 12/4/16  | 10  | 690   | 4630 | 463              |
| 12/11/16 | 11  | 810   | 5440 | 495              |
| 12/18/16 | 12  | 950   | 6390 | 533              |
| 12/25/16 | 13  | 1020  | 7410 | 570              |
| 1/1/17   | 1   | 680   | 680  | 680              |