ARRAYRIGHTINDEXOF Function

Contents:

Computes the index at which a specified element is first found within an array, when searching right to left. Returned value is based on left-to-right indexing.

- Leftmost index value is `0`. Rightmost index value is the same value returned by ARRAYLEN function.
- If the element is not found, null is returned.
- For left-to-right searching, use ARRAYINDEXOF.
  - If only one element exists in the array, both functions return the same value.
  - For more information, see *ARRAYINDEXOF Function*.

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

Array literal reference example:

```
arrayrightindexof(["A","B","C","D"],"C")
```

Output: Returns the right-index of the element "C" in the array, which is 2 in an 0-based index from left to right.

Column reference example:

```
arrayrightindexof([myValues],myElement)
```

Output: Returns the right-search in the `myValues` arrays for the elements listed in the `myElement` column.

Syntax and Arguments

```
arrayrightindexof(array_ref,my_element)
```

Argument	Required?	Data Type	Description
array_ref	Y	array or string	Name of Array column, Array literal, or function returning an Array to apply to the function
my_element	Y	any	The element to locate in the array, searching from right to left

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

array_ref

Name of the array column, array literal, or function returning an array whose element you want to locate.

- Multiple columns and wildcards are not supported.

Usage Notes:

Required?	Data Type	Example Value
Yes	String (column reference or function) or array literal	`myArray1`

my_element

Element literal that you wish to locate in the array. It can be a value of any data type.

Usage Notes:

Required?	Data Type	Example Value
Yes	Any	`"1st"`

Examples

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

Example - Computing points based on position of finish

This example covers the following functions:

- `ARRAYINDEXOF` - Returns the index value of an array for the specified value, searching from left to right. See *ARRAYINDEXOF Function*.
- `ARRAYRIGHTINDEXOF` - Returns the index value of an array for the specified value, searching from right to left. See *ARRAYRIGHTINDEXOF Function*.

Source:

The following set of arrays contain results, in order, of a series of races. From this list, the goal is to generate the score for each racer according to the following scoring matrix.

Place	Points
1st	30
2nd	20
3rd	10
Last	-10
Did Not Finish (DNF)	-20

Results:

RaceId	RaceResults
1	["racer3","racer5","racer2","racer1","racer6"]
2	["racer6","racer4","racer2","racer1","racer3","racer5"]
3	["racer4","racer3","racer5","racer2","racer6","racer1"]
4	["racer1","racer2","racer3","racer5"]
5	["racer5","racer2","racer4","racer6","racer3"]

Transformation:

Note that the number of racers varies with each race, so determining the position of the last racer depends on the number in the event. The number of racers can be captured using the following:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`ARRAYLEN(RaceResults)`
Parameter: New column name	`'countRacers'`

Create columns containing the index values for each racer. Below is the example for `racer1`:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`ARRAYINDEXOF(RaceResults, 'racer1')`
Parameter: New column name	`'arrL-IndexRacer1'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`ARRAYRIGHTINDEXOF(RaceResults, 'racer1')`
Parameter: New column name	`'arrR-IndexRacer1'`

You can then compare the values in the two columns to determine if they are the same.

> **NOTE:** If ARRAYINDEXOF and ARRAYRIGHTINDEXOF do not return the same value for the same inputs, then the value is not unique in the array.

Since the points awarded for 1st, 2nd, and 3rd place follow a consistent pattern, you can use the following single statement to compute points for podium finishes for `racer1`: computing based on the value stored for the left index value:

Transformation Name	`Conditional column`

Parameter: Condition type	`if...then...else`
Parameter: If	`{arrayL-IndexRacer1} < 3`
Parameter: Then	`(3 - {arrayL-IndexRacer1}) * 10`
Parameter: Else	`0`
Parameter: New column name	`'ptsRacer1'`

The following transform then edits the `ptsRacer1` to evaluate for the Did Not Finish (DNF) and last place conditions:

Transformation Name	`Edit column with formula`
Parameter: Columns	`ptsRacer1`
Parameter: Formula	`IF(ISNULL({arrayL-IndexRacer1}), -20, ptsRacer1))`

You can use the following to determine if the specified racer was last in the event:

Transformation Name	`Edit column with formula`
Parameter: Columns	`ptsRacer1`
Parameter: Formula	`IF(arrR-IndexRacer1 == countRacers, -10, ptsRacer1)`

Results:

RaceId	RaceResults	countRacers	arrR-IndexRacer1	arrL-IndexRacer1	ptsRacer1
1	["racer3","racer5","racer2","racer1","racer6"]	5	3	3	0
2	["racer6","racer4","racer2","racer1","racer3","racer5"]	6	3	3	0
3	["racer4","racer3","racer5","racer2","racer6","racer1"]	6	5	5	-10
4	["racer1","racer2","racer3","racer5"]	4	0	0	20
5	["racer5","racer2","racer4","racer6","racer3"]	5	null	null	-20