DOUBLEMETAPHONE Function

Returns a two-element array of primary and secondary phonetic encodings for an input string, based on the Double Metaphone algorithm.

The Double Metaphone algorithm processes an input string to render a primary and secondary spelling for it. For English language words, the algorithm removes silent letters, normalizes combinations of characters to a single definition, and removes vowels, except from the beginnings of words. In this manner, the algorithm can normalize inconsistencies between spellings for better matching. For more information, see *https://en.wikipedia.org/wiki/Metaphone*.

Tip: This function is useful for performing fuzzy matching between string values, such as between potential join key values.

Source values can be string literals, column references, or expressions that evaluate to strings.

Basic Usage

String literal reference example:

```
derive type:single DOUBLEMETAPHONE('My String') as:'double_metaphone'
```

Output: See below.

```
["MSTRNK","MSTRNK"]
```

Column reference example:

```
derive type:single value:DOUBLEMETAPHONE(string1) as:'double_metaphone'
```

Output: Generates a new `double_metaphone` column containing the evaluation of `string1` column values through the Double Metaphone algorithm.

Syntax and Arguments

```
derive type:single value:DOUBLEMETAPHONE(string_ref)
```

Argument	Required?	Data Type	Description
string_ref	Y	string	Name of column or string literal to apply to the function

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

string_ref1

String literal, column reference, or expression whose elements you want to filter through the Double Metaphone algorithm.

Usage Notes:

Required?	Data Type	Example Value

Yes	String literal, column reference, or expression evaluating to a string	`myString1`

Examples

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

Example - Phonetic string comparisons

This example illustrates how the following Double Metaphone algorithm functions operate in Trifacta.

- `DOUBLEMETAPHONE` - Computes a primary and secondary phonetic encoding for an input string. Encodings are returned as a two-element array. See *DOUBLEMETAPHONE Function*.
- `DOUBLEMETAPHONEQUALS` - Compares two input strings using the Double Metaphone algorithm. Returns `true` if they phonetically match. See *DOUBLEMETAPHONEEQUALS Function*.

Source:

The following table contains some example strings to be compared.

string1	string2	notes
My String	my string	comparison is case-insensitive
judge	juge	typo
knock	nock	silent letters
white	wite	missing letters
record	record	two different words in English but match the same
pair	pear	these match but are different words.
bookkeeper	book keeper	spaces cause failures in comparison
test1	test123	digits are not compared
the end.	the end….	punctuation differences do not matter.
a elephant	an elephant	a and an are treated differently.

Transform:

You can use the `DOUBLEMETAPHONE` function to generate phonetic spellings, as in the following:

```
derive type: single value: DOUBLEMETAPHONE(string1) as: 'dblmeta_s1'
```

You can compare `string1` and `string2` using the `DOUBLEMETAPHONEEQUALS` function:

```
derive type: single value: DOUBLEMETAPHONEEQUALS(string1, string2, 'normal') as: 'compare'
```

Results:

The following table contains some example strings to be compared.

string1	dblmeta_s1	string2	compare	Notes

My String	["MSTRNK","MSTRNK"]	my string	TRUE	comparison is case-insensitive
judge	["JJ","AJ"]	juge	TRUE	typo
knock	["NK","NK"]	nock	TRUE	silent letters
white	["AT","AT"]	wite	TRUE	missing letters
record	["RKRT","RKRT"]	record	TRUE	two different words in English but match the same
pair	["PR","PR"]	pear	TRUE	these match but are different words.
bookkeeper	["PKPR","PKPR"]	book keeper	FALSE	spaces cause failures in comparison
test1	["TST","TST"]	test123	TRUE	digits are not compared
the end.	["0NT","TNT"]	the endâ€¦.	TRUE	punctuation differences do not matter.
a elephant	["ALFNT","ALFNT"]	an elephant	FALSE	a and an are treated differently.