POW Function

Contents:

Computes the value of the first argument raised to the value of the second argument. Each argument can be a Decimal or Integer literal or a reference to a column containing numeric values.

Basic Usage

Numeric literal example:

```
derive type:single value: POW(10,3)
```

Output: Generates a column containing the value of 10^3, which is `1000`.

Column reference example:

```
derive type:single value: POW(MyValue,2) as: 'sqred_MyValue'
```

Output: Generates the new `sqred_myValue` column containing the value of the `MyValue` column raised to the power of 2 (squared).

Syntax and Arguments

```
derive type:single value: POW(base_numeric_value, exp_numeric_value)
```

Argument	Required?	Data Type	Description
base_numeric_value	Y	string, decimal, or integer	Name of column or Decimal or Integer literal that is the base value to be raised to the power of the second argument
exp_numeric_value	Y	string, decimal, or integer	Name of column or Decimal or Integer literal that is the power to which to raise the base value

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

base_numeric_value

Name of the column or numeric literal whose values are used as the bases for the exponential computation.

- Missing input values generate missing results.
- Literal numeric values should not be quoted.
- Multiple columns and wildcards are not supported.

Usage Notes:

Required?	Data Type	Example Value
Yes	String (column reference) or Integer or Decimal literal	`2.3`

exp_numeric_value

Name of the column or numeric literal whose values are used as the power to which the base-numeric value is raised.

- Missing input values generate missing results.
- Literal numeric values should not be quoted.
- Multiple columns and wildcards are not supported.

Usage Notes:

Required?	Data Type	Example Value
Yes	String (column reference) or Integer or Decimal literal	5

Examples

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

Example - Exponential functions

The following example demonstrates how the exponential functions work together. These functions include the following:

- `EXP` - e^X . See *EXP Function*.
- `LN` - natural logarithm of the above. See *LN Function*.
- `LOG` - 10^X. See *LOG Function*.
- `POW` - X^Y. The value X raised to the power Y. See *POW Function*.

Source:

rowNum	X
1	-2
2	1
3	0
4	1
5	2
6	3
7	4
8	5

Transform:

```
derive type:single value: EXP (X) as: 'expX'
```

```
derive type:single value: LN (expX) as: 'ln_expX'
```

```
derive type:single value: LOG (X) as: 'logX'
```

```
derive type:single value: POW (10,logX) as: 'pow_logX'
```

Results:

In the following, `(null value)` indicates that a null value is generated for the computation.

rowNum	X	expX	ln_expX	logX	pow_logX
1	-2	0.1353352832366127	-2	(null value)	(null value)
2	-1	0.1353352832366127	-0.9999999999999998	(null value)	(null value)
3	0	1	0	(null value)	0
4	1	2.718281828459045	1	0	1
5	2	7.3890560989306495	2	0.30102999566398114	1.9999999999999998
6	3	20.085536923187668	3	0.47712125471966244	3
7	4	54.59815003314423	4	0.6020599913279623	3.999999999999999
8	5	148.41315910257657	5	0.6989700043360187	4.999999999999999

Example - Pythagorean Theorem

The following example demonstrates how the `POW` and `SQRT` functions work together to compute the hypotenuse of a right triangle using the Pythagorean theorem.

- `POW` - X^Y. In this case, 10 to the power of the previous one. See *POW Function* .
- `SQRT` - computes the square root of the input value. See *SQRT Function*.

The Pythagorean theorem states that in a right triangle the length of each side (x,y) and of the hypotenuse (z) can be represented as the following:

$z^2 = x^2 + y^2$

Therefore, the length of z can be expressed as the following:

$z = sqrt(x^2 + y^2)$

For more information on the Pythagorean theorem, see *https://en.wikipedia.org/wiki/Pythagorean_theorem*.**Source:**

The dataset below contains values for x and y:

X	Y
3	4
4	9
8	10

30	40

Transform:

You can use the following transform to generate values for z^2.

> **NOTE:** Do not add this step to your recipe right now.

```
derive type:single value:(POW(x,2) + POW(y,2)) as:'Z'
```

You can see how column Z is generated as the sum of squares of the other two columns. Now, wrap the value computation in a SQRT function:

```
derive type:single value:SQRT((POW(x,2) + POW(y,2))) as: 'Z'
```

Results:

X	Y	Z
3	4	5
4	9	9.848857801796104
8	10	12.806248474865697
30	40	50