MOD Function

Returns the modulo value, which is the remainder of dividing the first argument by the second argument. Equivalent to the % operator.

- Each argument can be a literal Integer or Decimal number, a function returning a number, or a reference to a column containing numeric values.

> **NOTE:** Within an expression, you might choose to use the corresponding operator, instead of this function. For more information, see *Numeric Operators*.

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

```
mod(14,3)
```

Output: Returns remainder of the value `14` divided by `3`, which is `2`.

Syntax and Arguments

```
mod(value1, value2)
```

Argument	Required?	Data Type	Description
value1	Y	string	The first value must be an Integer or Decimal literal, column reference, or expression that evaluates to one of those two numeric types.
value2	Y	string	The first value must be an Integer or Decimal literal, column reference, or expression that evaluates to one of those two numeric types.

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

value1, value2

Integer or Decimal expressions, column references or literals.

- Missing or mismatched values generate missing string results.

Usage Notes:

Required?	Data Type	Example Value
Yes	Literal, function, or column reference returning an Integer or Decimal value	`15`

Examples

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

Example - Numeric Functions

This example demonstrates how to use numeric functions to perform computations in your recipe steps.

Functions:

Item	Description
ADD Function	Returns the value of summing the first argument and the second argument. Equivalent to the + operator.
MOD Function	Returns the modulo value, which is the remainder of dividing the first argument by the second argument. Equivalent to the % operator.
NEGATE Function	Returns the opposite of the value that is the first argument. Equivalent to the - operator placed in front of the argument.
SUBTRACT Function	Returns the value of subtracting the second argument from the first argument. Equivalent to the - operator.
MULTIPLY Function	Returns the value of multiplying the first argument by the second argument. Equivalent to the * operator.
DIVIDE Function	Returns the value of dividing the first argument by the second argument. Equivalent to the / operator.
LCM Function	Returns the least common multiple shared by the first and second arguments.

Source:

ValueA	ValueB
8	2
10	4
15	10
5	6

Transformation:

Execute the following transformation steps:

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`ADD(ValueA, ValueB)`
Parameter: New column name	`'add'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`SUBTRACT(ValueA, ValueB)`
Parameter: New column name	`'subtract'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`

Parameter: Formula	`MULTIPLY(ValueA, ValueB)`
Parameter: New column name	`'multiply'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`DIVIDE(ValueA, ValueB)`
Parameter: New column name	`'divide'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`MOD(ValueA, ValueB)`
Parameter: New column name	`'mod'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`NEGATE(ValueA)`
Parameter: New column name	`'negativeA'`

Transformation Name	`New formula`
Parameter: Formula type	`Single row formula`
Parameter: Formula	`LCM(ValueA, ValueB)`
Parameter: New column name	`'lcm'`

Results:

With a bit of cleanup, your dataset results might look like the following:

ValueA	ValueB	lcm	negativeA	mod	divide	multiply	subtract	add
8	2	8	-8	0	4	16	6	10
10	4	20	-10	2	2.5	40	6	14
15	10	30	-15	5	1.5	150	5	25
5	6	30	-5	5	0.833333333	30	-1	11