Formatting Calculations in Tableau

Applies to: Tableau Cloud, Tableau Desktop, Tableau Public, Tableau Server

This article describes how to create and format calculations in Tableau. It lists the basic components of calculations and explains the proper syntax for each.

Calculation building blocks

There are four basic components to calculations in Tableau:

- **Functions**: Statements used to transform the values or members in a field.

  - Functions require *arguments*, or specific pieces of information. Depending on the function, arguments can be fields, literals, parameters, or nested functions.
- **Fields**: Dimensions or measures from your data source.
- **Operators**: Symbols that denote an operation.
- **Literal expressions**: Constant values that are hardcoded, such as "High" or 1,500.

Not all calculations need to contain all four components. Additionally, calculations can contain:

- **Parameters**: Placeholder variables that can be inserted into calculations to replace constant values. For more information on parameters, see Create Parameters (parameters_create.htm).
- **Comments**: Notes about a calculation or its parts, not included in the computation of the

calculation.

For more information about how to use and format each of these components in a calculation, see the following sections.

Example calculation explained

For example, consider the following calculation, which adds 14 days to a date ([Initial Visit]). A calculation like this could be useful for automatically finding the date for a two-week followup.

```
DATEADD
(
'day'
,
14
,
[Initial Visit
)
```

The components of this calculation can be broken down as:

- Function: `DATEADD`, which requires three arguments.
  - date_part ('day')
  - interval (14)
  - date ([Initial Visit]).
- Field: [Initial Visit]
- Operators: n/a
- Literal expressions:
  - String literal: 'day'

  - Numeric literal: 14

In this example, the hardcoded constant 14 could be replaced with a parameter, which would allow the user to select how many days out to look for a followup appointment.

```
DATEADD
(
'day'
,
[How many days out?]
,
[Initial Visit
)
```

At a glance: calculation syntax

Components	Syntax	Example

Functions	See Tableau Functions (Alphabetical) or Tableau Functions (by Category) (functions_all_categories.htm) for examples of how to format all functions in Tableau.	`SUM(expression)`

Fields	A field in a calculation is often surrounded by brackets [ ]. See Field syntax for more information.	`[Category]`

Operators	`+, -, *, /, %, ==, =, >, <, >=, <=, !=, <>, ^, AND, OR, NOT, ( ).` See Operator syntax for information on the types of operators you can use in Tableau calculation and the order they are performed in a formula.	`[Price]*(1-[discount])`

Literal expressions	Numeric literals are written as numbers.
	String literals are written with quotation marks.
	Date literals are written with the # symbol.
	Boolean literals are written as either true or false.
	Null literals are written as null.
		`1.3567`
		`"Unprofitable"`
		`#August 22, 2005#`
		`true`
		`Null`

	See Literal expression syntax for more information.	
Parameters	A parameter in a calculation is surrounded by brackets [ ], like a field. See Create Parameters (parameters_create.htm#Parameters) for more information.	`[Bin Size]`

Comments	To enter a comment in a calculation, type two forward slashes //. See Add comments to a calculation for more information.
	Multi-line comments can be added by typing /* to start the comment and */ to end it.
		`SUM([Sales]) / SUM([Profit])`
		`/*John's calculation`
		`To be used for profit ratio`
		`Do not edit*/`

Calculation syntax in detail

See the following sections to learn more about the different components of Tableau calculations and how to format them to work in Tableau.

Function syntax

Functions are the main components of a calculation and can be used for various purposes.

Every function in Tableau requires a particular syntax. For example, the following calculation uses two functions, LEN and LEFT, as well as several logical operators (IF, THEN, ELSE, END, and >).

```
IF LEN([Name])> 5 THEN LEFT([Name],5)
ELSE [Name] END
```

- LEN takes a single argument, such as `LEN([Name])` which returns the number of characters (that is, the length) for each value in the Name field.
- LEFT takes two arguments, a field and a number, such as `LEFT([Name], 5)` which returns the first five characters from each value in the Name field starting from the left.
- The logical operators IF , THEN, ELSE, and END work together to create a logical test.

This calculation evaluates the length of a name and, if it's more than five characters, returns only the first five. Otherwise, it returns the entire name.

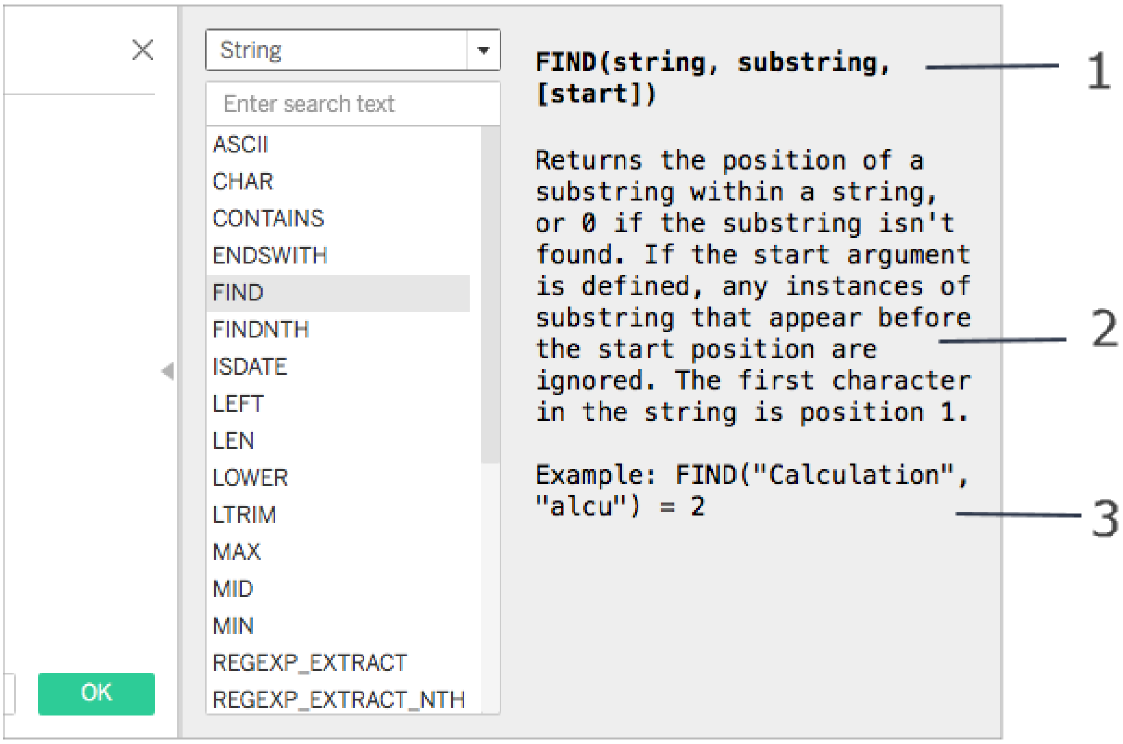
In the calculation editor, functions are colored blue.

Use the calculation editor reference pane

You can look up how to use and format a particular function at any time. To open the list of functions in Tableau:

1. Select **Analysis** > **Create Calculated Field**.
2. In the Calculation Editor that opens, click the expand (triangle) icon located on the right edge of the editor.

A list of functions appears for you to browse. When you select a function from the list, the section on the far right updates with information about that function's required syntax (**1**), its description (**2**), and one or more examples (**3**).



```
FIND(string, substring,
[start])

Returns the position of a
substring within a string,
or 0 if the substring isn't
found. If the start argument
is defined, any instances of
substring that appear before
the start position are
ignored. The first character
in the string is position 1.

Example: FIND("Calculation",
"alcu") = 2
```

Using multiple functions in a calculation

You can use more than one function in a calculation. For example:

```
ZN(SUM([Order Quantity])) -
WINDOW_AVG(SUM([Order Quantity]))
```

There are three functions in the calculation: ZN, SUM, and WINDOW_AVG. The ZN function and the WINDOW_AVG function are separated with the subtraction operator (-).

A function can also be part of another function (or, nested), as is the case with the `ZN(SUM([Order Quantity]))` portion of the example above. In this case, the SUM of Order Quantity is computed before the ZN function because it is inside parentheses. For more information on why, see Parentheses.

Field syntax

Fields can be inserted into your calculations. Often, a function's syntax indicates where a field should be inserted into the calculation. For example: `SUM(expression)`.

Field names should be encompassed by brackets [ ] in a calculation when the field name contains a space or is not unique. For example, [Sales Categories].

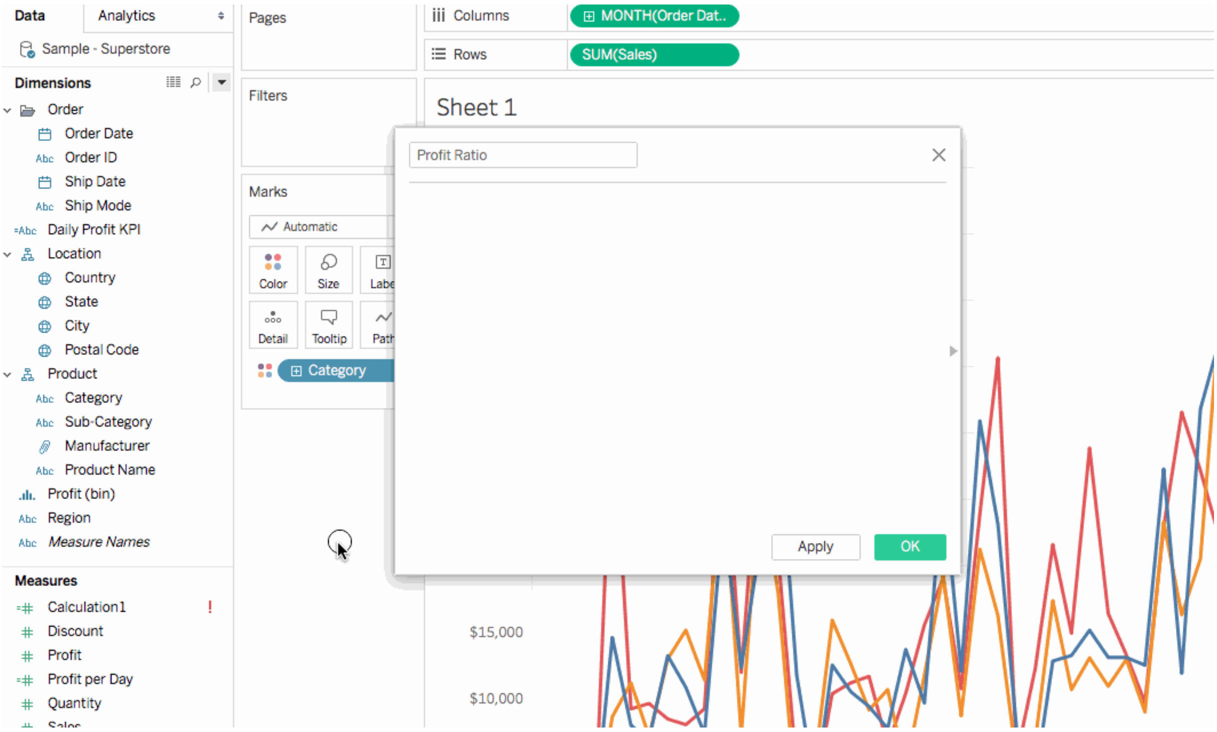
The type of function you use determines the type of field you use. For example, with the SUM function, you can insert a numerical field, but you cannot insert a date field. For more information, see Understanding data types in calculations.

The fields you choose to include in your calculations also depend on the purpose of calculation. For example, if you want to calculate profit ratio your calculation will use the Sales and Profit fields from your data source:

```
SUM([Sales])/SUM([Profit])
```

To add a field to a calculation, do one of the following:

- Drag it from the Data pane or the view and drop it in the calculation editor.
- In the Calculation Editor, type the field name. Note: The Calculation Editor attempts to auto-complete field names.



Fields are colored orange in Tableau calculations.

Operator syntax

To create calculations, you need to understand the operators supported by Tableau. This section discusses the basic operators that are available and the order (precedence) they are performed.

Operators are colored black in Tableau calculations.

+ (addition)

The + operator means addition when applied to numbers and concatenation when applied to strings. When applied to dates, it can be used to add a number of days to a date.

For example:

- `7 + 3`
- `Profit + Sales`
- `'abc' + 'def' = 'abcdef'`
- `#April 15, 2024# + 15 = #April 30, 2024#`

– (subtraction)

The - operator means subtraction when applied to numbers and negation if applied to an expression. When applied to dates, it can be used to subtract a number of days from a date. Hence, it can also be used to calculate the difference in days between two dates.

For example:

- `7 - 3`
- `Profit - Sales`
- `-(7+3) = -10`
- `#April 16, 2024# - 15 = #April 1, 2024#`
- `#April 15, 2024# - #April 8, 2024# = 7`

* (multiplication)

The * operator means numeric multiplication.

For example: `5 * 4 = 20`

/ (division)

The / operator means numeric division.

For example: `20 / 4 = 5`

% (modulo)

The % operator returns the remainder of a division operation. Modulo can only operate on integers.

For example: `9 % 2 = 1`. (Because 2 goes into 9 four times with a remainder of 1.)

==, =, >, <, >=, <=, !=, <> (comparisons)

These are the basic comparison operators that can be used in expressions. Their meanings are as follows:

- **`==`** or **`=`** (equal to)
- **`>`** (greater than)
- **`<`** (less than)
- **`>=`** (greater than or equal to)

- **<=** (less than or equal to)
- **!=** or **<>** (not equal to)

Each operator compares two numbers, dates, or strings and returns either TRUE, FALSE, or NULL.

^ (power)

This symbol is equivalent to the POWER function. It raises a number to the specified power.

For example: `6^3 = 216`

AND

This is a logical operator. An expression or a boolean must appear on either side of it.

For example: `IIF(Profit =100 AND Sales =1000, "High", "Low")`

See **AND** in Logical Functions (functions_functions_logical.htm) for more information.

OR

This is a logical operator. An expression or a boolean must appear on either side of it.

For example: `IIF(Profit =100 OR Sales =1000, "High", "Low")`

See **OR** in Logical Functions (functions_functions_logical.htm) for more information.

NOT

This is a logical operator. It can be used to negate another boolean or an expression. For example,

```
IIF(NOT(Sales = Profit),"Not Equal","Equal")
```

Other Operators

CASE, ELSE, ELSEIF, IF, THEN, WHEN, and END are also operators used for Logical Functions (functions_functions_logical.htm).

Operator precedence

All operators in a calculation are evaluated in a specific order. For example, `2*1+2` is equal to 4 and not equal to 6, because multiplication is performed before addition (the * operator is always evaluated before the + operator).

If two operators have the same precedence (such as addition and subtraction (+ or -) they are evaluated from left to right in the calculation.

Parentheses can be used to change the order of precedence. See the Parentheses section for more information.

PRECEDENCE	OPERATOR
1	– (negate)
2	^ (power)
3	*, /, %
4	+, –
5	==, =, >, <, >=, <=, !=, <>
6	NOT
7	AND
8	OR

Parentheses

Parentheses can be used as needed to force an order of precedence. Operators that appear within parentheses are evaluated before those outside of parentheses, starting from the innermost parentheses and moving outward.

For example, (1 + (2*2+1)*(3*6/3)) = 31 because the operators within the innermost parentheses are performed first. The calculation is calculated in the following order:

1. (2*2+1) = 5
2. (3*6/3) = 6
3. (1+ 5*6) = 31

Literal expression syntax

This section describes the proper syntax for using literal expressions in Tableau calculations. A literal expression signifies a constant value that is represented as is. When you are using functions you will sometimes want to use literal expressions to represent numbers, strings, dates, and more.

For example, you may have a function where your input is a date. Rather than type "May 1, 2005", which would be interpreted a string, you would type #May 1, 2005#. This is equivalent to using a date function to convert the argument from a string to a date (refer to Date Functions (functions_functions_date.htm)).

You can use numeric, string, date, boolean, and null literals in Tableau calculations. Each type, and how to format them, are described below.

Literal expressions are colored black and gray in Tableau calculations.

Numeric Literals

A numeric literal is written as a number. For example, to input the number one as a numeric literal, enter `1`. If you want to input the number 0.25 as a numeric literal, enter `0.25`.

String Literals

A string literal can be written either using 'single quote' or "double quote".

If your string has a single or double quote within it, use the other option for the outermost string literals.

For example, to input the string `"cat"` as a string literal, type`'"cat"'`. For `'cat'` type `"'cat'"`. If you want to type the string `She's my friend` as a string literal, use double quotes for the literals, as in `"She's my friend."`

Date Literals

Date literals are signified by the pound symbol (#). To input the date "August 22, 2005" as a literal date, enter the ISO formatted date, `#2005-08-22#`.

Boolean Literals

Boolean literals are written as either true or false. To input "true" as a boolean literal, enter `true`.

Null Literals

Null literals are written as Null. To input "Null" as a Null literal, enter `Null`.

Add parameters to a calculation

Parameters are placeholder variables that can be inserted into calculations to replace constant values. When you use a parameter in a calculation, you can then expose a parameter control in a view or dashboard to allow users to dynamically change the value.

For details, see Use a parameter (parameters_create.htm#Calc).

Parameters are colored purple in Tableau calculations.

Add comments to a calculation

You can add comments to a calculation to make notes about it or its parts. Comments are not included in the computation of the calculation.

To add a comment to a calculation, type two forward slash (//) characters.

For example:

```
SUM([Sales])/SUM([Profit]) //My calculation
```

In this example, `//My calculation` is a comment.

A comment starts at the two forward slashes (//) and goes to the end of the line. To continue with your calculation, you must start a new line.

A multi-line comment can be written by starting the comment with a forward slash followed by an asterisk (/*), and closed with an asterisk followed by a forward slash (*/). For example:

```
SUM([Sales])/SUM([Profit])
/* This calculation is
used for profit ratio.
Do not edit */
```

Comments are colored gray in Tableau calculations.

Understanding data types in calculations

If you create calculated fields, you need to know how to use and combine the different data types in calculations. Many functions that are available to you when you define a calculation only work when they are applied to specific data types.

For example, the `DATEPART()` function can accept only a date/datetime data type as an argument. You can enter `DATEPART('year',#2024-04-15#)` and expect a valid result: 2024. You cannot enter

`DATEPART('year',"Tom Sawyer")` and expect a valid result. In fact, this example returns an error because "`Tom Sawyer`" is a string, not a date/datetime.

Note: Although Tableau attempts to fully validate all calculations, some data type errors cannot be found until the query is run against the database. These issues appear as error dialogs at the time of the query rather than in the calculation dialog box.

The data types supported by Tableau are described below. Refer to Type Conversion (functions_functions_typeconversion.htm) to learn about converting from one data type to another.

String

A sequence of zero or more characters. For example, "`Wisconsin`", "`ID-44400`", and "`Tom Sawyer`" are all strings. Strings are recognized by single or double quotes. The quote character itself can be included in a string by repeating it. For example, `'O''Hanrahan'`.

Date/Datetime

A date or a datetime. For example "`January 23, 1972`" or "`January 23, 1972 12:32:00 AM`". If you would like a date written in long-hand style to be interpreted as a date/datetime, place the # sign on either side of it. For instance, "`January 23, 1972`" is treated as a string data type but `#January 23, 1972#` is treated as a date/datetime data type.

Number

Numerical values in Tableau can be either *integers* or *floating-point numbers*.

With floating-point numbers, results of some aggregations may not always be exactly as expected. For example, you may find that the SUM function returns a value such as -1.42e-14 for a field of numbers that you know should sum to exactly 0. This happens because the Institute of Electrical and Electronics Engineers (IEEE) 754 floating-point standard requires that numbers be stored in binary format, which means that numbers are sometimes rounded at extremely fine levels of precision. You can eliminate this potential distraction by formatting the number to show fewer decimal places. For more information, see ROUND in Number functions available in Tableau (functions_functions_number.htm#type).

Operations that test floating point values for equality can behave unpredictably for the same reason. Such comparisons can occur when using level of detail expressions as dimensions, in categorical filtering, creating ad-hoc groups, creating IN/OUT sets, and with data blending.

Note: The largest signed 64-bit integer is 9,223,372,036,854,775,807. When connecting to a new data source, any column with data type set to **Number (whole)** can accommodate values up to this limit; for larger values, because **Number (whole)** does not use floating-points, Tableau displays "Null." When the data type is set to **Number (decimal)**, larger values can be accommodated.

Boolean

A field that contains the values `TRUE` or `FALSE`. An unknown value arises when the result of a comparison is unknown. For example, the expression `7 > Null` yields unknown. Unknown booleans are automatically converted to Null.

See Also

Understanding Calculations in Tableau (calculations_calculatedfields_understand.htm)

Create a Simple Calculated Field (calculations_calculatedfields_formulas.htm)

Get Started with Calculations in Tableau (calculations_calculatedfields_create.htm)