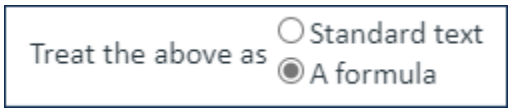
Formulas

Formulas are used throughout the system to perform calculations on selected fields. Formulas for updating fields are used in the following scenarios:

- Mass editing field values.
- Update Fields Actions, used in rules and action buttons.
- Setting default field values.
- Variable Formula Fields, which are evaluated by actions.

The Formula Wizard helps you create formulas by providing a list of relevant field variables, available formulas, and syntax help. Click Formula Help to open the wizard or, if you see a choice between Standard Text and A Formula, select A Formula and then click Formula Help.



Treat the above as a formula

Formula Help

The Formula Help wizard differs based on the context from which it is launched. It generally contains these tabs:

- **Home:** Provides general guidance on formulas and syntax instructions, including which fields and functions can be used. You can insert the functions and fields you need, then edit the formula to arrange them as necessary.
- **Fields:** Contains a list of field variables and field descriptions in the current table. Click on a field name to insert the variable into the formula.
- **Parent Fields:** Contains a list of field variables for the parent record, and only appears when editing an Update Fields action from within a Linked Record action.
- **Global Variables:** Lists global variables, including linked fields from the user table such as $global. my_teams.
- **New Variable:** Allows you to create a new summary variable based on the data set in a table and optionally filtered by a saved search.
- **Functions:** Lists available functions, syntax, and sample usage. Functions can perform calculations, retrieve and adapt values, and perform other simple transformations on data.
- **Values, Teams:** Lists available units of measurement and team names, respectively, which can be inserted into formulas by clicking on the hyperlinked value.

Formula Syntax Variations

Formula syntax differs slightly between contexts. For example, the `dateformat()` function requires a `$formula()` wrapper in a print template, but not in an Update Fields action or a mass edit. This section provides a quick reference for the syntax you need to use in each context.

> In any context, whenever you insert a formula or function inside another, you don't need to use the `$formula()` wrapper. For example, in a context where you generally need a wrapper, you would use it only on the outermost formula: `$formula(concat($company_name,"_",$contact_name)`

Print Templates

Standard variables don't require a `$formula()` wrapper, and the system parses all variables without special syntax as long as they correspond to an actual field variable. If a field variable isn't recognized, the variable is stripped from the resulting document entirely.

Functions always require a `$formula()` wrapper.

If you need to use the $ character in the template text, outside of a formula or variable, you must precede it with another $. Otherwise, it is parsed as an invalid field variable and stripped from the document.

If a field variable needs to be immediately followed by a character that isn't a space, which normally would be parsed as part of the variable, use the `concat()` function. For example, if you use `$start_date:` the colon isn't printed after the start date value. Instead, use `$formula(concat($start_date,":"))` to print a colon after the start date value.

Email Templates

Like print templates, standard variables don't require a `$formula()` wrapper, and the system parses all variables without special syntax as long as they correspond to an actual field variable. However, if a field variable isn't recognized, it is printed as written instead of being stripped out.

Functions always require a `$formula()` wrapper.

If you need to use the $ character in the template text outside of a formula or variable, you can enter it as-is and it is printed as shown, as long as it isn't followed by text that is recognized as a variable name.

Update Field Actions, Mass Edits, and Default Values

This refers to running Update Fields actions, performing a mass edit using the formula option, and setting default values with a formula in field data types that allow it.

The `$formula()` wrapper is not required for standard variables or for functions.

You can use + to combine text strings with field variables, as long as the variables don't contain numeric values: `"Role: " + $update_role + ", Team: " + $role_team_ + ", and Group: " + $role_group + " were added."` However, if any of the field variables store numeric values, you need to use the `concat()` function to combine them with text strings. Otherwise, you can use + to print the sum of the values.

Note that default values in Calculated Result fields also follow these rules, but the formula output must be a number, not a string.

Variable Formula fields

This refers to the text you enter in a Variable Formula field in a record, such as the Condition field in an Approval Template.

The `$formula()` wrapper is not required for standard variables or for functions.

To combine multiple conditions, use && to require both conditions or || to require one condition. For example, this formula in a Variable Formula field would evaluate whether the contract's jurisdiction is the USA and the amount is over $250,000: `$latest_contract_id.jurisdiction==”USA” && $latest_contract_id.contract_amount>”250000”`

Print and Merge Document Actions

This refers to the option to use a formula to generate the file name for Print actions and Merge Document actions.

Like print templates, standard variables don't require a `$formula()` wrapper, and the system parses all variables without special syntax as long as they correspond to an actual field variable. However, if a field variable isn't recognized, it is printed inside a new `$formula()` wrapper instead of being stripped out. For example, if you enter `$start_date` and that variable isn't recognized, the file name shows `$formula($start_date)` instead.

You can include strings directly, without the use of any additional formulas. For example, to title the document "Document for Contract 42" where 42 is the Contract ID, you can use `Document for Contract $id`.

If a field variable needs to be immediately followed by a character that isn't a space, which normally would be parsed as part of the variable, use the `concat()` function. For example, if you use `$start_date:` the colon isn't printed after the start date value. Instead, use `$formula(concat($start_date,":"))` to print a colon after the start date value.

Formula Functions for Updating Field Values

The sections below detail commonly used functions and provide syntax help. Remember that there are variations in how formulas are used in different contexts. For instance, formulas in an Update Fields action do not require the `$formula()` wrapper, while print template formulas do.

Simple Operations

Standard mathematical operations can be used in formulas, such as * for multiplication, / for division, + for addition, and - and subtraction. For example, if $amount is a field variable, the formula $amount*10 multiplies the value in the amount field by ten.

For a list of available operators, see Operators. You can also view the list of operators and definitions from the Formula Help wizard.

Common Functions

Here, find more information about the most commonly used functions. You can find a complete list of functions on the Functions tab of the Formula Help wizard.

Concatenate Strings

concatenatestringsexcerpt

The concatenate strings command allows you to combine field values with text strings, field variables, or other formulas that use variables. For example, if you use `$formula(concat("x",$field_name,"z",...))`, the "x" and "z" placeholders are where you insert text strings, field variables, or formulas.

It can also be used when inserting a Choice field value, if the value is rendered as code such as 1@2 rather than the text of the value. To do so, enter the value as the only parameter.

Syntax

```
$formula(concat("x",$field,"z",...))
```

The variables may be text strings, field variables, or other formulas and variables. Text strings must be surrounded in double quotation marks (""), while variables and formulas do not.

If the first piece is a text string, you can use shorthand to combine variables and strings, but $formula() is still required in email templates:

```
$formula("x"+$variable+"z"+...)
```

Examples

Example formula	Output
`$formula(concat("Your account representative is ",$account_rep,"."))`	Your account representative is Hector Gomez.
`$formula(concat($company_name," Support Contract"))`	Agiloft Support Contract.
`$formula(""+$field1*$field2+" Total")`	2,142 Total
`$formula(concat($contract_start_date," to ",$contract_end_date))`	09/01/19 to 08/31/2020

Currency Conversion

convertcurrencyexcerpt

The convertCurrency formula converts an amount from one currency to another, based on the exchange rates published by the European Central Bank. An optional date parameter allows conversion using historic exchange rates.

Syntax

```
convertCurrency("From_Currency","To_Currency",amount,[date])
```

Examples

`convertCurrency("USD", "EUR", $total_payment, $payment_date)`	Convert the Total Payment from USD to EUR using the historic exchange rate on the Payment Date.
`convertCurrency("JPY", "USD", $contract_amount)`	Convert the Contract Amount from Japanese Yen to US Dollars using the current exchange rate.

Information

Two currency conversion formulas are provided using an open-source API to convert an amount between currencies or return the exchange rate. Currency conversion formulas use the open-source JSON API fixer.io to obtain exchange rates published by the European Central Bank. These rates are updated daily at approximately 4pm Central European Time (CET). If the date parameter is not specified, the current exchange rate is used.

The CLM Template uses a free API key from fixer.io to make requests to their API. You must replace the current access key with a new, free one you can easily download from the fixer.io website. If you intend to use the currency conversion for production environments, then we recommend purchasing a paid version from fixer.io instead of using the free version.

After obtaining a new API key from fixer.io, log in to the KB and update the Fixer.io Service Access Key global variable through **Setup** > **System** > **Manage Global variables** to make use of the new key.

Date Format

dateformatexcerpt

`Dateformat()` is used to display a date/time field in a particular format, often for localization. `Dateformat()` takes two arguments: the desired pattern format, and the field variable. Date time patterns are indicated with a series of letters that represent elements such as month, day in month, day of week, year, hour, and minute. For a full list of possible formats see this page from Java about SimpleDateFormat().

Syntax

```
dateformat("output pattern","$field")
```

Date/time Pattern	Example output
"MM/dd/yy"	05/31/16
"yyyy/dd/MM"	2016/22/09
"MMMMM"	July (name of month)

"d"	10 (day of month)

Examples

The following results are expected when `$contract_start_date` evaluates to February 10, 2016 at 01:00:00.

Formula	Sample output
$formula(dateformat ("d",$contract_start_date))	10. *Inserts which day of the month the contract starts.*
$formula(dateformat ("MMMMM",$contract_start_date))	February. *Inserts the full text name of the month.*
$formula(dateformat ("yyyy",$contract_start_date))	2016. *Inserts the year of the contract start date.*
$formula(dateformat("MM/dd/yyyy hh:mm", NOW()))	*Inserts the current date and timestamp when the document template runs.*

Deep Link

deeplinkexcerpt

The deeplink command returns the encrypted link to view the current record. This is the same link you can access by opening a record and going to ... > Copy Link.

It can also be used to retrieve the encrypted link for a different record, by specifying the table and ID of that record.

Syntax

```
$formula(deeplink("tableName", recordId))
```

Use the logical table name when utilizing the tableName parameter.

Find

findexcerpt

The Find formula finds a sub-sequence of the input sequence that matches the pattern and returns true if, and only if, a sub-sequence of the input string matches the specified pattern.

findexcerpt

Syntax

```
find(pattern, text[,flags])
```

Examples

`find("flour", $ingredients)`	This formula returns true if the text "flour" is found in the text field $ingredients.
`find("flour", $ingredients," im")? "Contains Gluten": "Gluten Free"`	Returns "Contains Gluten" if "flour" or "Flour" is found in the $ingredients field, else it returns "Gluten Free". Multi-line mode is enabled.

Flags

Case-insensitive matching and multi-line mode are enabled by the optional flags "i" and "m" respectively, or "im" to enable both.
For example, the expression `find("flour", $ingredients, "im")` will return "true" if the sequence "flour" or "Flour" is found, and "false" if it is not.

Notes

The find function accepts only strings as arguments. You can force the system to evaluate some non-string data types as strings by adding an empty string with +"". For instance, a link to single fields from other table with multiple values enabled (MVE) is not stored as a string. If the "find" function is used to look for some value such as "anonymous" in a multi-value linked field (MLF) called External CCs, the formula might look like the following:

```
find("anonymous", $external_cc+"")
```

By adding an empty string to the MLF value - using +"" - the MLF value is converted into a string for search purposes.

Example

`find("apples",""+$selected_fruit,"i")`	Returns true if the multi-value linked field `$selected_fruit` contains "apples" or "Apples" or "aPPles".

Matches

matchesexcerpt

The matches formula matches character sequences against patterns specified by regular expressions and returns true if and only if the entire text matches the pattern.

Syntax

```
matches(pattern, text[,flags])
```

Example

`matches("Sales Department",$linked_department)`	Returns true if the $linked_department field contains 'Sales Department'.

Flags

Case-insensitive matching and multi-line mode are enabled by the optional flags "i" and "m" respectively, or "im" to enable both. Multi-line mode affects searches on text fields or other data types with multiple lines. If multi-line mode is not enabled in these circumstances, line breaks are treated as spaces.

Information

The matches function accepts only strings as arguments. You can force the system to evaluate some non-string data types as strings by adding an empty string with +"".

Example

`matches ("3",$id+"")`	Returns true if the $id, converted to a string, is 3.

New Line or Line Break

newlineexcerpt

To insert a new line or a line break, use `\n` within a document template or field value formula. This can help start new paragraphs in certain cases.

Numbers to Words

num2wordsexcerpt

The num2words formula converts numerals into words. For example, you can use the function to render "12" as "twelve," "twelfth," or "12th," or you can format it as currency. If you want to output words in another language, you can specify an alternate language code as the last parameter. The currency option requires a language to be specified in order to identify the appropriate currency.

The format options are cardinal; ordinal; ordinal_num, which renders the ordinal format with numerals; and currency, which requires a language code.

The first input parameter must be a numerical value. For ordinal formatting, the number must be an integer. The default format is cardinal, and the default language is U.S. English.

Syntax

```
num2words(value, "format", "language")
```

Examples

`num2words(12.76, "cardinal")`	twelve point seven six
`num2words(39, "ordinal")`	thirty-ninth
`num2words(39, "ordinal_num")`	39th

`num2words(39, "currency", "es")`	treinta y nueve euros
`num2words(39, "currency", "en")`	thirty-nine U.S. dollars

Rate of Exchange

rateofexchangeexcerpt

The RateOfExchange formula outputs the exchange rate between two currencies, either now or using a historic exchange rate based on a date field. The exchange rates are published by the European Central Bank.

Syntax

```
RateOfExchange("From_Currency","To_Currency",[date])
```

Examples

`RateOfExchange("USD", "EUR")`	Result is the current exchange rate between US dollars and Euros.
`RateOfExchange("JPY", "USD", $payment_date)`	Results in the exchange rate between Japanese yen and US dollars on the date of the transaction, $payment_date.

Information

Currency conversion formulas use the open-source JSON API fixer.io to obtain exchange rates published by the European Central Bank. These rates are updated daily at approximately 4pm Central European Time (CET).

If the date parameter is not specified, the current exchange rate is used.

The fixer.io API has the following limitations on the free license:

- 100 requests per month.
- Only EUR can be used as a base currency.

For more information on the available licenses, see Fixer.io.

Fixer.io Variable

Once you have obtained a new API access key, the value must be set in the Fixer.io Service Access Key global variable.

String Replacement

Text string replacement can replace, or append, an existing string.

To replace part of a text string, delimit the text to be replaced, also known as a search string, and the replacement text with a slash, "/".

> **Example**
>
> /dog/cat/
> changes **'Good dog! Your dog food is in the dog dish'**
> to **'Good cat! Your cat food is in the cat dish'**

Regular expressions can be used to specify the search string. Some examples of regular expressions are given below:

Expression	Matches...
sep [ae] r [ae] te	separate, seperate, seperete, *or* separete
\bcat\b	cat *as an entire word, rather than part of another word, as in* catalog
Dec (ember) ?	Dec *or* December
cat|dog|mouse|fish	cat, dog, mouse, *or* fish
variable [0-9] a	variable0a, variable1a, *etc*., ..., variable9a
th (is) *	th, this, thisis, thisisis, ...
this+	this, thiss, thisss, ...

You can find more information on usage of regular expressions at http://www.regular-expressions.info.

Two functions can be used to find and match on substrings using regex. These are detailed below.

Ternary Operators

ternaryoperatorsexcerpt

Short conditional statements can be inserted with the ternary operator "a ? b : c." This works like an if-else statement: if the condition "a" evaluates true, then insert "b;" if "a" is false, insert "c." You must provide all parameters for the ternary operator to function, even if one parameter simply inserts a blank space. If you want to output a string in the b or c parameters, include quotation marks around the string.

The ternary operator is useful in formulas that divide by a variable, where that variable might sometimes equal zero. Otherwise, the attempt to divide by zero results in an error. Use the ternary operator to check whether that variable is zero and then provide formulas for each situation. For example, if you want to calculate a percentage using two fields, but $field2 might sometimes equal zero, you could use "$field2 == 0 ? 0 : 100*($field1/$field2)" to output zero if $field2 is zero and otherwise output the calculated percentage.

In some cases, the ternary operator might unintentionally render a Choice field value as code, such as 1@2, instead of the text of the value. To show the text instead, use the concat() function inside the ternary syntax. For example: $formula($choice_field=="Choice A" ? concat($preselected_value_in_choice_field2) : concat($preselected_value_in_choice_field3))

Syntax

```
$formula($condition ? "True Output" : "False output")
```

Example

In this example the statement compares the variable $n to the value 1. If it is 1, "$n is 1" is shown. If it's not 1, "$n is not 1" is shown.

```
($n==1) ? "$n is 1" : "$n is not 1"
```

In this example, the statement uses functions inside the ternary formula. This outputs information about the delegation timeline, if the delegation is time-based, and otherwise states that the delegation isn't time-based. Notice that inside the ternary function, $formula is not used a second time.

```
$formula(($delegation_type=="Time-Based") ? "Delegation from " + dateformat("MMM
dd, yyyy", $delegation_start_date) + " to " dateformat("MMM dd, yyyy",
$delegation_end_date) : "Delegation is not time-based.")
```