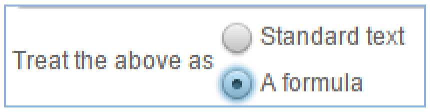
Formulas

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.
- Updating fields actions, used in rules and action buttons.
- Setting default field values.
- Variable formula fields, which are evaluated by actions.

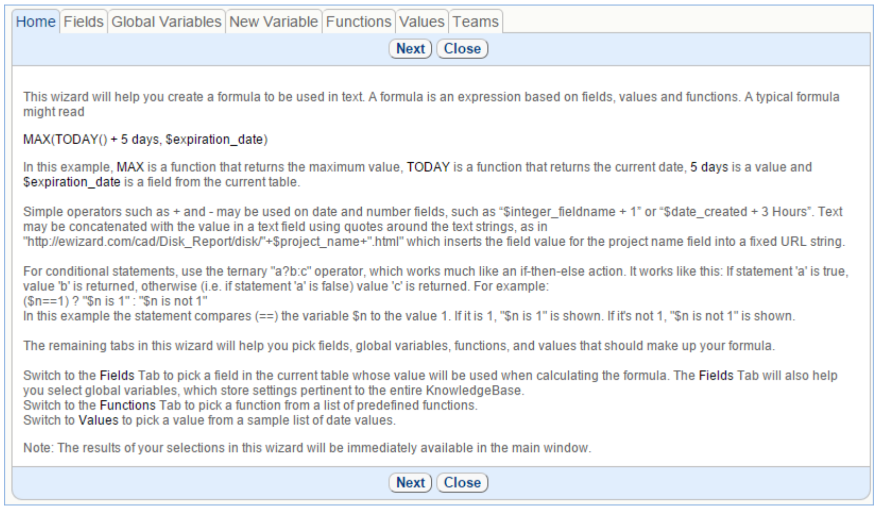
The Formula Wizard helps you create formulas by providing a list of relevant field variables, available formulas, and syntax help. Click Formula Help Formula Help to open the wizard. Sometimes you'll see a choice between Standard text and Formula; select 'A formula' and the help icon will appear.


Treat the above as Standard text [✓ A formula]

Formula Help

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

- **Home tab:** 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.



This wizard will help you create a formula to be used in text. A formula is an expression based on fields, values and functions. A typical formula might read

MAX(TODAY() + 5 days, $expiration_date)

In this example, MAX is a function that returns the maximum value, TODAY is a function that returns the current date, 5 days is a value and $expiration_date is a field from the current table.

Simple operators such as + and - may be used on date and number fields, such as "$integer_fieldname + 1" or "$date_created + 3 Hours". Text may be concatenated with the value in a text field using quotes around the text strings, as in "http://ewizard.com/cad/Disk_Report/disk/"+$project_name+".html" which inserts the field value for the project name field into a fixed URL string.

For conditional statements, use the ternary "a?b:c" operator, which works much like an if-then-else action. It works like this: If statement 'a' is true, value 'b' is returned, otherwise (i.e. if statement 'a' is false) value 'c' is returned. For example:
($n==1) ? "$n is 1" : "$n is not 1"
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.

The remaining tabs in this wizard will help you pick fields, global variables, functions, and values that should make up your formula.

Switch to the Fields Tab to pick a field in the current table whose value will be used when calculating the formula. The Fields Tab will also help you select global variables, which store settings pertinent to the entire KnowledgeBase.
Switch to the Functions Tab to pick a function from a list of predefined functions.
Switch to Values to pick a value from a sample list of date values.

Note: The results of your selections in this wizard will be immediately available in the main window.

- **Fields tab:** 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 tab:** This tab lists field variables for the parent record, and only appears when editing an Update Fields action from within a Linked Record action.
- **Global Variables tab:** Lists global variables, including linked fields from the user table such as $global.my_teams.
- **New Variable tab:** Use this tab to create a new summary variable based on the data set in a table and optionally filtered by a saved search.
- **Functions tab:** Lists available functions, syntax, and sample usage.
- **Values, Teams tabs:** Lists available units of measurement and team names, respectively, which can be inserted into formulas by clicking on the hyperlinked value.

Formulas for Updating Field Values

The sections below detail commonly used formulas and provide syntax help.

Simple Formulas

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.

String Replacement

Text string replacement may replace, or append to, 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.

New line or line break

newlineexcerpt

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

Concatenate Strings

concatenatestringsexcerpt

Use the following syntax to concatenate field values with other text strings. For example, inserting a field value into a sentence.

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 quotes" while variables and formulas do not.

Often, a shorthand can be used to combine variables and strings, provided the first piece is a string.

```
"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.
`""+$field1*$field2+" Total"`	2,142 Total

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.

Dateformat

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("yyyy",NOW()))	*Inserts the current year when the print template runs.*

Ternary operators

ternaryoperatorsexcerpt

Short conditional statements can be inserted with the ternary operator "a ? b : c". This works like an if-else statement, as follows: if the condition "a" evaluates true, then insert "b"; if "a" is false, insert "c".

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"
```

Currency conversion

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 Demo KB uses a free API key from fixer.io to make requests to their API. If you intend to use the currency conversion for production environments, then we recommend purchasing a paid version of the API key. The various plans available are shown at https://fixer.io/product

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

RateOfExchange

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:

- 1000 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 an API access key, the value must be set in the Fixer.io Service Access Key global variable.

convertCurrency

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

Currency conversion formulas use the open-source JSON API at 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.