Kinabase Computed Fields Guide

This guide outlines how to use the Computed Fields in Kinabase.

Build powerful, reliable formulas and dynamic text

Computed fields let you derive values from other fields, perform unit-aware mathematics, and generate dynamic text. This document is your quick reference for syntax, formulas, supported units, conditions, and common patterns.

Table of Contents

1. Accessing Field Values

You can display the value of any field by using the $ symbol followed by the field name in ALL CAPS. Spaces in field names are replaced by underscores.

E.g. if your field is called "Client Name", the field accessor would be $CLIENT_NAME.

1.1. Subfields

Some field types have extra data available in subfields.

These are accessible via $FIELD_NAME.SUBFIELD_NAME.

The field type & subfield include:

➔ **Date range:** START, END, ACTUAL_START, ACTUAL_END

E.g. When you select a Date Range field type and name it "Event Date" for example, you will be able to fill several subfields within it such as start, end, actual start, and actual end.

Here is how to use it in the computed text field: $EVENT_DATE.ACTUAL_START. This will display the actual event start date.

➔ **Risk Score:** IMPACT, LIKELIHOOD

E.g. When you select a Risk Score field type and name it "Risk Level", you will be able to access subfield within it such as impact and likelihood.

Here is how to use it in the computed text field: $RISK_LEVEL.IMPACT. This will display the impact score.

➔ **Address:** TITLE, LINE_1, LINE_2, CITY, COUNTY, COUNTRY, POSTCODE, LAT, LNG

E.g. When you select an Address field type and name it "Location", you will be able to access subfield within it such as title, line 1, line 2, city, county, country, postcode, latitude, and longitude.

Here is how to use it in the computed text field: $LOCATION.COUNTRY. This will display the address country.

2. Mathematical Functions

2.1. Operator:

All basic mathematical operators (+ , -, *, /, ^, and brackets) are supported.

Operators must have spaces surrounding them.

E.g. In a computed field: ((2 + 4) - 5) / (2 ^ 2) = 0.25

2.2. Math Functions

All number handling functions are supported. Assume you have a Number field called “Number”, and the number in that field is 4.7:

- ➔ *E.g. ROUND($NUMBER) will round to the nearest integer, hence it will result in 5.*
- ➔ *E.g. FLOOR($NUMBER) will round down, hence it will result in 4.*
- ➔ *E.g. CEIL($NUMBER) / CEILING($NUMBER) will round up, hence it will result in 5.*
- ➔ *E.g. ABS($NUMBER) will display the absolute value, hence it will result in 4.7 if it was originally negative.*

2.2.1. Trigonometry

Trigonometry functions are supported. Assume you have a Number field called “Number”, and the number in that field is 4.7 which is in radians:

- ➔ *E.g. SIN($NUMBER) -> SIN(4.7) = -1*
- ➔ *E.g. COS($NUMBER) -> COS(4.7) = -0.0124*
- ➔ *E.g. TAN($NUMBER) -> TAN(4.7) = 80.7*

2.2.2. Logarithms

All logarithms are supported. Assume you have a two Number fields called “Base” and the other one called “Value”. Assume the number in the Base is 2 and in the Value is 8:

- ➔ *E.g. LOG($VALUE, $BASE) -> 3 (logarithm with custom base)*
- ➔ *E.g. LN($VALUE) -> 2.079 (Natural Logarithm (base e))*
- ➔ *E.g. LOG2($VALUE) -> 3 (Base-2 logarithm)*
- ➔ *E.g. LOG10($VALUE) -> 0.9030 (Base-10 logarithm)*

3. Numbers with units

Units can be added to constant numbers used in expressions.

They can also be combined with * and / (without spaces).

*E.g. $MILEAGE * £3.50/mi would result in an output in pounds. (Where Mileage is in mi "miles")*

Further examples:

- 45m3 = 45 cubic metres
- 70mi/h = 70 miles per hour
- 30deg = 30 degrees (angle)

3.1. Full List of Units

3.1.1. SI Base Units

Unit	Description	Symbol to Use
Metre	Length	m
Kilogram	Mass	kg
Second	Time	s
Ampere	Electric Current	A
Kelvin	Temperature measured from absolute zero	K
Mole	Amount of substance	mol
Byte	Amount of data (note: 1kB= 1000B)	B

All of the above support adding SI prefixes next to the record (e.g. kg for kilogram, s for second)

3.1.2. Additional Time Units

Unit	Description	Symbol to Use
Minute	60 seconds	min
Hour	60 minutes, 3600 seconds	h
Day	24 hours	d

3.1.3. Additional Mass Units

Unit	Description	Symbol to Use
Gram	0.001 kg	g
Tonne	1000 kg	t

3.1.4. Dimensionless Units

Unit	Description	Symbol to Use
Radians	Angle (2π in full revolution)	rad
Degrees	Angle (360 in a full revolution)	deg
Percentage	Proportion, 1 / 100	%

3.1.5. Temperature

Unit	Description	Symbol to Use
Celsius	Temperature measured from freezing point of water at 0degC.	degC
Kelvin	Scientific temperature measured from absolute zero.	K
Fahrenheit	Temperature measured using freezing point of water at 32degF.	degF

3.1.6. Common UK Units

Unit	Description	Symbol to Use
Miles	Distance, = 1609.344 m	mi
Pints	Volume, = 568.26 ml	pt
Pounds Sterling	UK Currency	£

3.1.7. Area

Unit	Description	Symbol to Use
Hectare	10,000 square metres (100m x 100m)	ha
Square Kilometres	1,000,000 square metres (1km x 1km)	km2
Square Centimetres	1cm x 1cm	cm2

3.1.8. Volume

Unit	Description	Symbol to Use
Litre	Volume 0.001 cubic metres	l
Millilitre	0.001 litres	ml

4. Logic Functions

4.1. If Functions

Conditional logic is possible through the IF function, which has the following syntax:

➔ IF ([condition], [result if true], [result if false])

*E.g. IF($COST > £4, $COST * 1.5, $COST * 2.5) would return the record's cost multiplied by 1.5 if it costs more than £4.00, and the cost multiplied by 2.5 otherwise (and could be simplified to $COST * IF($COST > £4, 1.5, 2.5))*

4.2. Conditions

4.2.1. Numerical

➔ = (Equals)
➔ != (Does not equal)
➔ >, < (Greater then, less than)
➔ >=, <= (Greater than or equal to, less than or equal to)

*E.g. IF($AGE >= 50, $SALARY * 1.2, 0) would return the record's salary multiplied by 1.2 if the age is greater than or equal to 50, and 0 otherwise.*

4.2.2. Text

➔ is blank (is empty) e.g. $NAME is blank
➔ matches (equality) e.g, $NAME matches "John"
➔ contains, starts with, ends with e.g. $NAME contains "Y"

E.g. IF($NAME matches "Anna" , 1, 0) would return 1 if the name matches Anna, and 0 otherwise.

4.2.3. Date + Time

➔ is in (equality, taking into account precision) e.g. $START_DATE is in 2024
➔ is before, is after e.g. $START_DATE is before 2025

*E.g. IF($WORK_START_DATE is before 2018, $SALARY * 1.1, 0) would return the record's salary multiplied by 1.1 if the work start date is before 2018, and 0 otherwise.*

4.2.4. Combining Conditions

You can combine multiple conditions using '**not'**, '**and'**, '**or'** and '**xor'** operators.

➔ **… not …** (inverts result) e.g. not $LENGTH > 1m is equivalent to $LENGTH <= 1m

*E.g. IF(not $AGE > 50, $SALARY * 1.2, 0) is equivalent to IF($AGE <= 50, $SALARY * 1.2, 0)*

➔ **… and …** (true if both sides are true, can be chained)
e.g. $LENGTH > 1m and $WIDTH > 1m and $HEIGHT >1m

*E.g. IF($AGE = 50 and $NAME contains "y", $SALARY * 1.2, 0)*
would return the record's salary multiplied by 1.2 if the age is equal to 50 and the name contains letter "y", and 0 otherwise.

➔ … **or** … (true if either side is true, can also be chained)
e.g. $LENGTH > 1m or $WIDTH > 1m or $HEIGHT > 1m is true if any dimension exceeds 1 metre.

*E.g. IF($AGE >= 40 or $WORK_START_DATE is before 2018, $SALARY * 1.2, 0)*
would return the record's salary multiplied by 1.2 if either the age is above or equal to 40, or if the work start date is before 2018, and 0 otherwise.

➔ **… xor …** (exclusive or, true if either side is true but not if both sides are true)

*E.g. IF($AGE = 40 xor $WORK_START_DATE is before 2018, $SALARY * 1.2, 0)*
wouldn't return anything if the age is 40 and the start date is before 2018 since both sides are true. One side only must be true.

5. Empty Value Handling

If a field is left empty (null), operations involving that field will also return null.

To fall back to a default value, you can use the coalescing operator ?? which has syntax:

➔ [value to check] ?? [fallback value]

E.g. Assume you have an "Income" and "Profit" fields:

➔ $PROFIT ?? ($INCOME *10%) would return the record's profit if that field is filled in, and 10% of the record's income otherwise.

6. Duration Difference Functions:

Use these functions to measure how much time separates two dates, in different units like duration, years, months, or days.

Assume you have two “Date + Time” fields named “Date 1” and “Date 2”:

➔ E.g. DIFF($DATE1, $DATE2) -> Duration difference (with lower precisions)

➔ E.g. DIFF_YEARS($DATE1, $DATE2) -> Difference in years

➔ E.g. DIFF_MONTHS($DATE1, $DATE2) -> Difference in months

➔ E.g. DIFF_DAYS($DATE1, $DATE2) -> Difference in days

7. Computed Text Fields

Computed text fields use the same method of accessing field values, but instead insert the field value into the rest of template.

*E.g. Assume you have an apple **Type** and **Review** fields, here is how to use the computed text:*

➔ This apple is a **$TYPE** and tastes **$REVIEW** could display as:
"This apple is a **Granny Smith** and tastes **Amazing**.

There are a few special cases for this formatting:

- Adding a second $ ends the accessor early:
  E.g. This fruit is a $TYPE$FRUIT could display as "This fruit is a GRAPEFRUIT"

- Actual dollar symbols can be added using $$:
  E.g. This $TYPE apple costs $$COST could display as "This Braeburn apple costs $3.00"

Revision History

Revision	Date	Author	Description
A	25-11-2025	M.Lavelle	Release