Aggregate Function Reference

All standard aggregate functions and available modifiers are listed on this page.

An aggregate function call contains an expression in curly braces ("{}"), which is calculated for the item and all sub-items (or, in some cases, for another subset of related items in the structure), and then the resulting values are aggregated according to the meaning of the aggregate function.

> ⚠️ It is not possible to include both upward-looking and downward-looking aggregate functions within the same formula. When using one of the two upward-looking aggregate functions, PARENT and JOIN (when used with an upward-looking modifier), you cannot include any of the other aggregate functions listed above.
>
> For example, the formula for calculating the percentage of Story Points of an issue compared to the aggregate Story Points of its parent (**story_points / PARENT {SUM {story_points}}**) would fail, because PARENT looks one level up in your hierarchy, while SUM aggregates the levels below.
>
> We are working to fix this limitation in a future version.

Aggregation Functions

SUM

Sum calculates a numerical total for the values calculated for the item and/or its sub-items.

Summary	X	SUM{X}
▾ ☑ T1	3	6
☑ T1.1	2	2
▾ ☑ T1.2		1
☑ T1.2.1	1	1

Note that when the value of the expression under aggregation is not numeric (and cannot be converted to number), it is ignored.

> ✅ If a certain issue (or another kind of item) is included multiple times in the sub-tree, the sum will include the value for that issue only *once*. This behavior can be overridden by using the #all modifier.

Accepts modifiers: #all, #children, #leaves, #strict.

COUNT

Count calculates a count of defined values (or truthy values, if the #truthy modifier is specified) for the item and/or its sub-items.

Summary	X	COUNT{X}
▾ ☑ T1	3	3
☑ T1.1	2	1
▾ ☑ T1.2		1
☑ T1.2.1	1	1

> ⊘ If a certain issue (or another kind of item) is included multiple times in the sub-tree, it will be counted only *once*. This behavior can be overridden by using the #all modifier.

Accepts modifiers: #all, #children, #leaves, #strict, #truthy.

AVG

Avg calculates an average of defined values for the item and/or its sub-items. The result for avg is generally the same as sum/count. Returns nothing if there are no defined values for {x}.

Summary	X	AVG{X}
▾ ☑ T1	3	2
☑ T1.1	2	2
▾ ☑ T1.2		1
☑ T1.2.1	1	1

> ⊘ If a certain issue (or another kind of item) is included multiple times in the sub-tree, the average value will include the value for that issue only *once*. This behavior can be overridden by using the #all modifier.

Accepts modifiers: #all, #children, #leaves, #strict.

MAX

Max returns the maximum defined value for the item and/or its sub-items. Numeric, date, duration and text fields can be compared. Text fields are compared lexicographically.

Summary	X	MAX{X}
▾ ☑ T1	3	3
☑ T1.1	2	2
▾ ☑ T1.2		1
☑ T1.2.1	1	1

Accepts modifiers: #children, #leaves, #strict.

MIN

Min returns the minimum defined value for the item and/or its sub-items. Numeric, date, duration and text fields can be compared. Text fields are compared lexicographically.

Summary	X	MIN{X}
▾ ☑ T1	3	1
☑ T1.1	2	2
▾ ☑ T1.2		1
☑ T1.2.1	1	1

Accepts modifiers: #children, #leaves, #strict.

JOIN

Join concatenates (joins) strings from the item and its parents (or other items, if modifiers are used).

- By default it joins all parent string values from root to the self value.
- If the current row has children and #subtree modifier is set, join appends the values for children, wrapping them into characters (braces by default).
- Wrapping characters can be set by #beforeChildren and #afterChildren (see example for #subtree to see how it works).

Summary	X	JOIN{X}
▾ ☑ T1	3	3
☑ T1.1	2	3, 2
▾ ☑ T1.2		3, ?
☑ T1.2.1	1	3, ?, 1

Accepts modifiers: #ancestors, #subtree, #children, #leaves, #strict, #reverse, #separator, #beforeChildren, #afterChildren, #fromDepth, #toDepth, #distinct.

PARENT

Parent extracts the value from the parent row or from an ancestor row by specified depth.

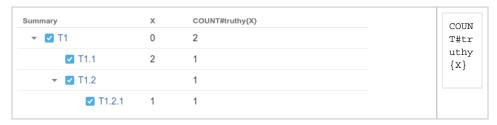| Summary	X	PARENT{X}
▾ ☑ T1	3	
☑ T1.1	2	3
▾ ☑ T1.2		3
☑ T1.2.1	1	

Accepts modifier: #depth.

Aggregation Modifiers

#all

Tells the aggregate function to include duplicate items. By defaults, aggregate functions ignore duplicate items.

Summary	X	SUM{X}	SUM#all{X}	COUNT{X}	COUNT#all{X}
▾ ☑ T1		2	6	1	3
☑ T1.1	2	2	2	1	1
☑ T1.1	2	2	2	1	1
☑ T1.1	2	2	2	1	1

```
SUM#all{X}
COUNT#all{X}
```

Can be used with: sum, count, avg.

#truthy

Only count row if the subexpression produces a truthy value .

Summary	X	COUNT#truthy{X}
▾ ☑ T1	0	2
☑ T1.1	2	1
▾ ☑ T1.2		1
☑ T1.2.1	1	1

```
COUNT#truthy{X}
```

Can be used with: count.
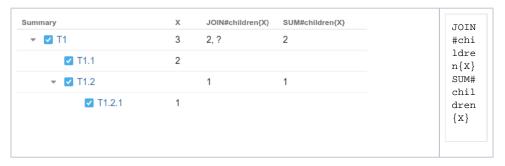
#strict

Do not process the current row item as part of the aggregation.

Cannot be used together with #children, #ancestors or #leaves, since these already exclude the current row.

Summary	X	JOIN#strict{X}	SUM#strict{X}
▾ ☑ T1	3	2, ?(1)	3
☑ T1.1	2		
▾ ☑ T1.2		1	1
☑ T1.2.1	1		

```
JOIN
#str
ict
{X}
SUM#
stri
ct
{X}
```

Can be used with: sum, count, avg, join, min, max.

#children

Only process direct children of the current row.

Summary	X	JOIN#children{X}	SUM#children{X}
▾ ☑ T1	3	2, ?	2
☑ T1.1	2		
▾ ☑ T1.2		1	1
☑ T1.2.1	1		

```
JOIN
#chi
ldre
n{X}
SUM#
chil
dren
{X}
```

Can be used with: sum, count, avg, join, min, max.

#leaves

Only process leaves (items without children) in the subtree of the current row.

Summary	X	JOIN#leaves{X}	SUM#leaves{X}
▾ ☑ T1	3	2, 1	3
☑ T1.1	2	2	2
▾ ☑ T1.2		1	1
☑ T1.2.1	1	1	1

```
JOIN
#lea
ves
{X}
SUM#
leav
es
{X}
```

Can be used with: sum, count, avg, join, min, max.

#subtree

Process the whole subtree of the current row. This is default behavior for sum, count, avg, min, max.

Summary	X	JOIN#subtree{X}	SUM{X}
▾ ☑ T1	3	3(2, ?(1))	6
☑ T1.1	2	2	2
▾ ☑ T1.2		?(1)	1
☑ T1.2.1	1	1	1

```
JOIN
#sub
tree
{X}
```
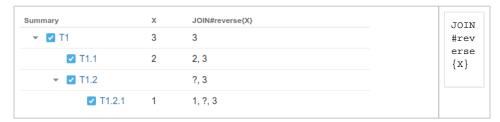
Can be used with: join.

#ancestors

Only process ancestors of the current row. This is default behavior for join, parent.

Can be used with: join.

#reverse

Reverses the order of row processing.

Summary	X	JOIN#reverse{X}	
▾ ☑ T1	3	3	JOIN#reverse{X}
☑ T1.1	2	2, 3	
▾ ☑ T1.2		?, 3	
☑ T1.2.1	1	1, ?, 3	

Can be used with: join.

#separator

Defines the separator for string joining. This modifier has a string parameter. The default is ", ".

Summary	X	JOIN#separator="->"{X}	
▾ ☑ T1	3	3	JOIN#separator="->"{X}
☑ T1.1	2	3->2	
▾ ☑ T1.2		3->?	
☑ T1.2.1	1	3->?->1	

Can be used with: join.

#beforeChildren

See #afterChildren.

#afterChildren

Defines the exit separator between children and parent rows. This modifier has a string parameter. The default exit separator is:

- "(" - for #beforeChildren
- ")" - for #afterChildren

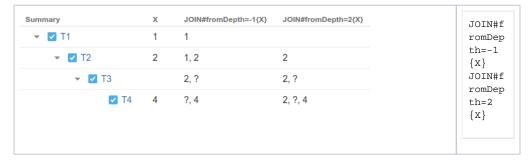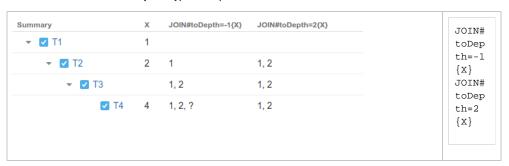| Summary	X	Formula	
▾ ☑ T1	3	3<{2, ?<{1}>}>	JOIN#subtree#beforeChildren="<{"#afterChildren="}>"{X}
☑ T1.1	2	2	
▾ ☑ T1.2		?<{1}>	
☑ T1.2.1	1	1	

Can be used with: join.

#fromDepth

Specifies the position of the first row the aggregate function should take as input for a sequence.

Position is specified by an integer parameter denoted as n below:

- Positive values mean the absolute depth of the row in the structure, e.g. n=1 means root.
- Negative values mean the depth relative to current row, e.g. n=-1 is the current item's direct parent.
- Default is 1.
- n should not be 0.

This modifier does not work with any tree types except #ancestors.

Summary	X	JOIN#fromDepth=-1{X}	JOIN#fromDepth=2{X}
▾ ☑ T1	1	1	
▾ ☑ T2	2	1, 2	2
▾ ☑ T3		2, ?	2, ?
☑ T4	4	?, 4	2, ?, 4

```
JOIN#f
romDep
th=-1
{X}
JOIN#f
romDep
th=2
{X}
```

Can be used with: join.

#toDepth

Specifies the position of the last row the aggregate function should take as input for a sequence.

Position is specified by an integer parameter denoted as n below:

- Positive values mean the absolute depth of row in the structure, e.g. n=1 means root.
- Negative values mean the depth relative to current row, e.g. n=-1 is the current item's direct parent.
- 0 means current row.
- Default is 0.

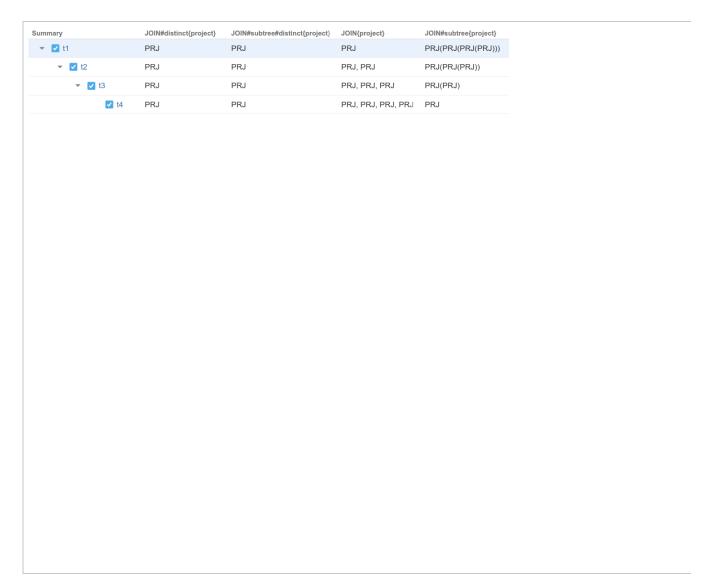This modifier does not work with any tree types except #ancestors.

Summary	X	JOIN#toDepth=-1{X}	JOIN#toDepth=2{X}
▾ ☑ T1	1		
▾ ☑ T2	2	1	1, 2
▾ ☑ T3		1, 2	1, 2
☑ T4	4	1, 2, ?	1, 2

```
JOIN#
toDep
th=-1
{X}
JOIN#
toDep
th=2
{X}
```

Can be used with: join.

#distinct

Makes join only concatenate distinct values. A duplicate value won't be added more than once if this modifier is on.

Modifiers #beforeChildren and #afterChildren don't work when this option is on.

Summary	JOIN#distinct{project}	JOIN#subtree#distinct{project}	JOIN{project}	JOIN#subtree{project}
▾ ☑ t1	PRJ	PRJ	PRJ	PRJ(PRJ(PRJ(PRJ)))
▾ ☑ t2	PRJ	PRJ	PRJ, PRJ	PRJ(PRJ(PRJ))
▾ ☑ t3	PRJ	PRJ	PRJ, PRJ, PRJ	PRJ(PRJ)
☑ t4	PRJ	PRJ	PRJ, PRJ, PRJ, PRJ	PRJ

Can be used with: join.

#depth

Specifies the position of the parent that possesses value.

Position is specified by an integer parameter denoted as n below:

- Positive values mean the absolute depth of the row in the structure, e.g. n=1 means root.
- Negative values mean the depth relative to the current row, e.g. n=-1 is the current item's direct parent.
- Default is 1.
- n should not be 0.

Summary	X	-1	-2	1	2
▾ ☑ T1	3			3	
☑ T1.1	2	3		3	2
▾ ☑ T1.2		3		3	
☑ T1.2.1	1		3	3	

```
PARENT#depth=-1{X}    // default one
PARENT#depth=-2{X}    // "grandparent"
PARENT#depth=1 {X}    // root row
PARENT#depth=2 {X}
```

Can be used with: parent.