

Time series

Generated on October 14, 2026

Time series

Distribution

```
tsDistribution · tsDistribution(serie: time series)
```

How long a series spent at each distinct value - the run-time-per-state breakdown behind a machine-state or alarm report. Each interval between two samples counts for the value in force during it, the earlier sample's; intervals where the value is null are left out.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to summarise.

Returns `object` : A record with one field per distinct value, named by the value's text (true, OFF, 2.0), holding the time spent at it in days.

Examples

```
tsDistribution(history(PLANT_LOG, "PUMP_ON", START, addHours(START, 2))) # {false: 0.020833333333333332, true: 0.07916666666666667}
tsDistribution([[START, "ON"], [addHours(START, 1), "OFF"], [addHours(START, 3), "OFF"]]) # {OFF: 0.08333333333333333, ON: 0.9166666666666667}
```

Data Orchestrator: available in plant formulas.

Filter series

```
tsFilter · tsFilter(serie: time series, paramName: name, predicate: expression)
```

Keeps only the samples a condition accepts, breaking the series where a run of rejected samples begins. Each time and value pair is bound to paramName in turn.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to filter.
<code>paramName</code>	<code>name</code>	The name each sample is bound to, written bare rather than quoted.
<code>predicate</code>	<code>expression</code>	The condition, evaluated once per sample. Refer to the sample by paramName.

Returns `time series` : The accepted samples, with breaks where samples were dropped.

Examples

```
tsFilter(history(PLANT_LOG, "LIT_101", START, addHours(START, 2)), s, s[1] > 42) # [[2026-09-29T08:30, 44.0], [
```

Data Orchestrator: available in plant formulas.

First value

`tsFirst` · `tsFirst(serie: time series)`

The first sample of a series, as its [time, value] pair: `tsFirst(s)[1]` is the value, `tsFirst(s)[0]` its time.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to read.

Returns `[]`: The earliest sample, as a [time, value] pair, or null when the series is empty.

Examples

```
tsFirst(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # [2026-09-29T08:00, 41.5]
tsFirst(history(PLANT_LOG, "LIT_101", START, addHours(START, 2)))[1] # 41.5
```

Data Orchestrator: available in plant formulas.

Last value

`tsLast` · `tsLast(serie: time series)`

The last sample of a series, as its [time, value] pair: `tsLast(s)[1]` is the value, `tsLast(s)[0]` its time.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to read.

Returns `[]`: The latest sample, as a [time, value] pair, or null when the series is empty.

Examples

```
tsLast(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # [2026-09-29T10:00, 43.0]
tsLast(history(PLANT_LOG, "LIT_101", START, addHours(START, 2)))[1] # 43.0
```

Data Orchestrator: available in plant formulas.

Merge

`tsMerge` · `tsMerge(series: object)`

Interleaves several named series onto one timeline, so every instant carries the last known value of each - what a multi-variable chart or export needs.

Parameters

Name	Type	Description
<code>series</code>	<code>object</code>	A record whose fields name the series to merge.

Returns `[]` : A list of time and record pairs, the record keyed by the input names.

Examples

```
tsMerge({level: history(PLANT_LOG, "LIT_101", START, addHours(START, 1)), pump: history(PLANT_LOG, "PUMP_ON", S
```

Data Orchestrator: available in plant formulas.

Middle value

```
tsCenter · tsCenter(serie: time series)
```

The value in force halfway through a series' time span. For numeric series only: every value is read as a number, and text values are an error.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to read.

Returns `double` : The value in force at the midpoint, as a decimal number, or null when the series is empty.

Examples

```
tsCenter(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # 44.0
```

Data Orchestrator: available in plant formulas.

Series maximum

```
tsMax · tsMax(serie: time series)
```

The largest value in a series. Over a compacted period it is exact on windows that hold the segments' spikes, as windows aligned to the segments do.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to scan.

Returns `double` : The largest value.

Examples

```
tsMax(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # 44.0
```

Data Orchestrator: available in plant formulas.

Series minimum

`tsMin` · `tsMin(serie: time series)`

The smallest value in a series. Over a compacted period it is exact on windows that hold the segments' spikes, as windows aligned to the segments do.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to scan.

Returns `double` : The smallest value.

Examples

```
tsMin(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # 41.5
```

Data Orchestrator: available in plant formulas.

Split

`tsSplit` · `tsSplit(serie: time series, segments: [])`

Cuts a series into one sub-series per time segment - pair it with `psplit` to bucket a window, then aggregate each bucket.

Parameters

Name	Type	Description
<code>serie</code>	<code>time series</code>	The series to cut.
<code>segments</code>	<code>[]</code>	The start and end pairs to cut it at, as <code>psplit</code> produces.

Returns `[]` : A list of series, one per segment.

Examples

```
tsSplit(history(PLANT_LOG, "LIT_101", START, addHours(START, 2)), psplit(START, addHours(START, 2), 2)) # [[20
tsSplit(history(PLANT_LOG, "LIT_101", START, addHours(START, 2)), psplit(START, addHours(START, 2), 2)).forEach
```

Data Orchestrator: available in plant formulas.

Time average

`tsAverage` · `tsAverage(serie: time series)`

The time-weighted mean of a series - a sample held for an hour counts for an hour, not as one reading, which is what makes this the correct average for logged data. Over a compacted period, on windows aligned to the compacted segments, it is the logged average to within the segment's range times 1 ms over the segment's length: three ten-millionths of the range for one-hour segments.

Parameters

Name	Type	Description
serie	time series	The series to average.

Returns double : The time-weighted mean.

Examples

```
tsAverage(history(PLANT_LOG, "LIT_101", START, addHours(START, 2))) # 43.0
tsAverage(history(PLANT_LOG, "PUMP_ON", START, addHours(START, 2))) # 0.75
```

Data Orchester: available in plant formulas.

This page describes Amtiri Script 5.5.0 as shipped with Data Orchester engine 6.22.0 and Site 1.27.0.