Struct Definitions Reference

This page provides a reference for the purpose and structure of all structs standard groups and databases.

Structs groups and databases are marked with a JSON object using the key `xs_struct`. This contains three standard parameters:

- `"type"` - the name of the type of struct element as a string
- `"v"` - the current version this instance of the specified type as a string
- `"conf"` - optional JSON object, format depends on type

The version allows structs elements to be upgraded as the structs definitions evolve. It uses a basic semantic versioning format, specifically `MAJOR`.`MINOR`.`PATCH`, where:

- `MAJOR` changes indicate breaking structural changes
- `MINOR` changes add functionality in a backward compatible manner
- `PATCH` changes make backward compatible bug fixes

Minor and patch changes will automatically be applied to structs elements in server upgrades, whereas major changes will typically require user interaction. Major changes will be avoided as much as possible to maintain stable workflows. Patch version updates will only be used in cases where a change was needed to correctly adhere to the documented specifications on this page.

Virtual fields are calculated from other fields and cannot be populated manually.

The `uuid` type is a string that must adhere to the standard UUID format e.g. `1372DE9C-D78A-44EC-8136-87D4DB8720F0`.

Groups

Note that group versioning is used to manage databases required within groups.

Project

Top level struct group. All struct groups and databases must be decendants of a project to be recognized. Name and label are customizable.

Created with the `STRUCT CREATE PROJECT` action.

Struct Parameters

Parameter	Value
type	project
v	1.0.0

Group Parameters

Parameter	Value
name	*
label	*

Category

Mid-level struct group for organization. Must be child of project or category.

Created with the `STRUCT CREATE CATEGORY` action.

Struct Parameters

Parameter	Value
type	category
v	1.0.0

Group Parameters

Parameter	Value
name	*
label	*

Model

Group for which all data is locally co-relevant. Must be child of either project or category. Name and label are customizable.

Created with the `STRUCT CREATE MODEL` action.

Struct Parameters

Parameter	Value
type	model
v	1.0.0

Group Parameters

Parameter	Value
name	*
label	*

Origin

Group for all data from a single origin. Must be the child of a model. Name and label are customizable.

Created with the `STRUCT CREATE ORIGIN` action.

Struct Parameters

Parameter	Value
type	origin
v	1.0.0

Group Parameters

Parameter	Value
name	*
label	*

Definitions

Group containing definitions databases.

Group Parameters

Parameter	Value
name	def
label	Definitions

Struct Parameters

Parameter	Value
type	origin
v	1.0.0

Task

Group containing task tracking databases. Must be a child of an origin group.

Group Parameters

Parameter	Value
name	task
label	Task

Struct Parameters

Parameter	Value
type	task
v	1.0.0

Mnemonic

Group containing mnemonic data databases. Must be a child of an origin group.

Group Parameters

Parameter	Value
name	mn
label	Mnemonic

Struct Parameters

Parameter	Value
type	mn
v	1.0.0

Mnemonic Bin

Group containing binned mnemonic data databases. Must be a child of a mnemonic group.

Group Parameters

Parameter	Value
name	mn_bin
label	Bin

Struct Parameters

Parameter	Value
type	mn_bin
v	1.0.0

Databases

Structs databases typically specify a set of required fields, and may permit the inclusion of additional custom fields. Changes to the spec involving fields will usually be treated as minor version changes, though they may require manual user correction if an added field conflicts with a custom field already present in a particular database instance.

Definitions

All definitions databases must be a direct children of a definitions group, and all definitions groups must contain one of each definition database.

Event Definitions

Holds event definitions, specifying how they are displayed, interpretted and processed.

Database Parameters

Parameter	Value
name	event
label	Event
format	`"{name}"`
order	(`name` , asc)
singular	`"event definition"`

Struct Parameters

Parameter	Value
type	def_event
v	1.0.0

Fields

Name	Type	Req	Description
event_id	`int(8)`	?	unique event ID
name	`utf8vstring(128)`	?	unique event name
desc	`utf8text`		plain text event description
meta	`jsonobject`		additional arbitrary metadata
conf	`jsonobject`		configuration for pseudo-mnemonics

Mnemonic Definitions

Holds mnemonic definitions, specifying how they are displayed, interpretted and processed.

Database Parameters

Parameter	Value
name	mn
label	Mnemonic
format	`"{name} ({unit})"`
order	(`name` , asc)
singular	`"mnemonic definition"`

Struct Parameters

Parameter	Value
type	def_mn
v	1.0.0

Fields

Name	Type	Req	Description
mn_id	`int(4)`	?	unique mnemonic ID
name	`utf8vstring(128)`	?	unique mnemonic name
desc	`utf8text`		plain text mnemonic description
unit	`utf8vstring(32)`		measurement unit (for example, `"V"`, `"mA"`)
state	`struct_mn_state`	?	current state of mnemonic
origins	`jsonobject`	?	map of model(s) to associated origin(s)
full	`asciivstring(32)`		the primary database for the mnemonic, default `f8`
bin	`set(asciistring)`		the opt-in bin database(s) to include the mnemonic in
format	`asciivstring(32)`		printf-style format to render values
enums	`jsonobject`		mapping of permitted text values to numeric values
labels	`list(jsonobject)`		mapping of numeric values or ranges to labels
aliases	`set(utf8string)`		set of additional names associated with the mnemonic
meta	`jsonobject`		additional metadata as needed
query	`asciivstring(32)`		query name for pseudo-mnemonics
conf	`jsonobject`		configuration for pseudo-mnemonics

Notes

Changelog

1.0.0

- `enum` changed to `enums` since "enum" is often a reserved keyword
- `meas` field removed (measure now assumed from `unit`)

Mnemonic Tracking

Used for tracking mnemonic selection activity. Though this is not technically a definition database, it is used alongside the mnemonic definitions database, and is thus defined in the def context.

Database Parameters

Parameter	Value
name	mn_track
label	Mnemonic Tracking

Parameter	Value
format	`"{t} {user} {mn_id}"`
order	(`name`, asc)
singular	`"mnemonic definition"`

Struct Parameters

Parameter	Value
type	def_mn_track
v	1.0.0

Fields

Name	Type	Req	Description
t	`instant(us)`	?	time of selection
mn_id	`int(4)`	?	mnemonic ID selected
user	`user_id`	?	user taking action
mn_ids	`set(int(4))`		other mnemonic ID(s) selected
models	`set(asciistring)`		model(s) in current context

Changelog

1.0.1 (planned)

- order changed to (`t`, desc)
- format changed to `"{t} {mn_id} {user}"`

Nominal Definitions

Holds mnemonic nominal range definitions.

Database Parameters

Parameter	Value
name	nominal
label	Nominal
format	`"{mn_id} {color} ({min}, {max}) {label}"`
order	(`mn_id`, asc), (`label`, asc)
singular	`"nominal definition"`

Struct Parameters

Parameter	Value
type	def_nominal

Parameter	Value
v	1.0.0

Fields

Name	Type	Req	Description
nominal_id	`uuid`	?	unique nominal range ID
mn_id	`int(4)`	?	unique mnemonic ID
label	`utf8vstring(128)`	?	nominal range label
desc	`utf8text`		plain text nominal range description
color	`struct_nominal_color`		additional arbitrary metadata
min	`float8`		min value for the range
max	`float8`		max value for the range
models	`set(asciistring)`		models for which this range should apply (all if `null`)

Notes

The `struct_nominal_color` type is an enum of **green** (0), **yellow** (1), and **red** (2).

Changelog

1.1.0 (planned)

- `label` changed to `name` for consistency with other structs databases

Plot Configuration Definitions

Holds mnemonic plot configuration definitions.

Database Parameters

Parameter	Value
name	plot
label	Plot Conf
format	`"{name}"`
order	(`name`, asc)
singular	`"plot configuration"`

Struct Parameters

Parameter	Value
type	def_plot
v	1.0.0

Fields

Name	Type	Req	Description
name	`utf8vstring(128)`	?	unique conf name
desc	`utf8text`		plain text conf description
conf	`jsonobject`	?	configuration
models	`set(asciistring)`		models for which this conf should apply (any if `null`)

Changelog

1.1.0 (planned)

- `conf` changed from `jsonobject` to `struct_plot_conf`

Profile Definitions

Holds mnemonic profile definitions.

Database Parameters

Parameter	Value
name	profile
label	Profile
format	`"{name}"`
order	[`name`, asc]
singular	`"profile"`

Struct Parameters

Parameter	Value
type	def_profile
v	1.0.0

Fields

Name	Type	Req	Description
name	`utf8vstring(128)`	?	unique conf name
desc	`utf8text`		plain text profile description
models	`set(asciistring)`		models for which this conf should apply (all if `null`)
data	`jsonobject`	?	profile data configuration
plot	`jsonobject`		profile plot configuration

Notes

Requires review before use.

Changelog

1.1.0 (planned)

- TBD

Diagram Definitions

Holds diagram definitions. The diagram itself is in an attached SVG file.

Database Parameters

Parameter	Value
name	diagram
label	Diagram
format	`"{name}"`
order	(`name`, desc)
singular	`"diagram"`

Struct Parameters

Parameter	Value
type	def_diagram
v	1.0.0

Fields

Name	Type	Req	Description
name	`utf8vstring(128)`	?	unique conf name
file_name	`utf8filename`	?	file name
conf	`jsonobject`		diagram configuration
meta	`jsonobject`		additional metadata as needed

Changelog

1.1.0 (planned)

- order changed to (`name`, asc)
- added `desc` field

Filter Definitions

Provisional, not yet implemented

Holds mnemonic filter definitions.

Struct Parameters

Parameter	Value
type	def_filter
v	1.0.0

Database Parameters

Parameter	Value
name	filter
label	Filter
format	`"{name}"`
order	(`name` , asc)
singular	`"filter definition"`

Fields

Name	Type	Req	Description
filter_id	int(4)	unique filter ID	
name	`utf8vstring(128)`	?	unique filter name
desc	`utf8text`		plain text profile description
condition	`jsonobject`	?	filter condition definition
t_start_offset	`duration(us)`	?	start time offset (`0` if not provided)
t_end_offset	`duration(us)`	?	end time offset (`0` if not provided)
meta	`jsonobject`		additional metadata as needed

Trend Definitions

Provisional, not yet implemented

Holds mnemonic trend definitions.

Struct Parameters

Parameter	Value
type	def_trend
v	1.0.0

Database Parameters

Parameter	Value
name	trend

Parameter	Value
format	`"{name}"`
order	(`name`, asc)
singular	`"trend definition"`

Fields

Name	Type	Req	Description
name	`utf8vstring(128)`	?	unique trend name
desc	`utf8text`		plain text trend description
profiles	set(utf8string)		profile name(s) to include in trend
t	`list(jsonobject)`		time range(s) to include
intervals	`jsonarray`		intervals to include/omit
plot	`jsonobject`		plot configuration

Event Databases

Event databases come in three forms, simple events, single file per event, and multiple files per event.

Event

Each record is a single event. May be a direct child of either a model or origin.

Database Parameters

Parameter	Value
name	* (default `"event"`)
label	* (default `"Event"`)
format	* (default `"{t_start} {event_id} {label}"`)
order	* (default (`t_start`, desc), (`event_id`, asc))
singular	* (default `"event"`)

Struct Parameters

Parameter	Value
type	event
v	1.0.2

Fields

Name	Type	Req	Virtual	Description

uuid	`uuid`	?		event UUID
e_id	`int(8)`	?		event ID (default to `0` if not provided)
t_start	`instant(us)`	?		start time
t_end	`instant(us)`			end time (if `null`, event is an open interval)
dur	`duration(us)`	?	?	duration in microseconds (`null` if open)
interval	`boolean`	?	?	`t_start` != `t_end`
open	`boolean`	?	?	`t_end` is `null`
type	`struct_event_type`	?		event type (default to `message` if not provided)
level	`struct_event_level`	?		event level (default to `none` if not provided)
name	`utf8vstring(128)`			event name (if associated with event definition)
label	`utf8vstring(128)`	?		plain text label
content	`utf8text`			extended event content
meta	`jsonobject`			additional metadata as needed
conf	`jsonobject`			configuration for specific event types

Notes

Virtual fields are calculated from other fields and cannot be populated manually.

Changelog

1.0.2

- corrected `t_end` and `dur` as not required

1.0.1

- corrected `name` as not required

1.0.0

- `pid` (primary ID) changed to `e_id` (event ID) to avoid confusion
- `sid` removed (additional IDs may be added as needed)
- `int` changed to `interval` (`int` is commonly reserved keyword)
- `dur`, `interval`, and `open` are now derived fields from `t_start` and `t_end`
- added `struct_event_type` and `struct_event_level` data types
- added `name` as event definition association

Event File

Uses same structure as event database, with one additional field.

Database Parameters

Parameter	Value
name	* (default `"eventf"`)
label	* (default `"Event File"`)
singular	* (default `"event file"`)

Name	Type	Req	Description
file_name	`utf8filename`	?	file name

Event Files

Uses same structure as event database, but with a child file database, allowing each event to contain zero or more files.

Database Parameters

Parameter	Value
name	* (default `"eventfs"`)
label	* (default `"Event Files"`)
singular	* (default `"event files"`)

Mnemonic Databases

Contain mnemonic data. Unless otherwise indicated, mnemonic databases can be configured with a paritioning system which subdivides the tables internally into UTC calendar months. This is beneficial for selective mnemonic data repopulation, since each month can be instantly erased before being regenerated, and doesn't require downtime of the entire dataset. At creation time a start and end year must be specified, and partitions will be created for each month in that range. Data with timestamps outside the range will be stored in either a pre-range or post-range partition as applicable.

Mn Full

Each record is a single mnemonic data point. The data type for mnemonic values is configurable, and determines the database name. By default the mnemonic data group will contain a full `float8` database.

Struct Parameters

Parameter	Value
type	mn_full
v	1.0.0
conf.type	numeric data type

Database Parameters

Parameter	Value
name	`"i1"`, `"i2"`, `"i4"`, `"i8"`, `"f4"`, or `"f8"`
label	`"Full {conf.type}"`
format	`"{t} {mn_id} {v}"`
singular	`"mnemonic datapoint"`

Fields

Name	Type	Req	Description
t	`instant(us)`	?	time
mn_id	`int(4)`	?	unique mnemonic ID
v	`{conf.type}`		value

Mn Buffer

Each record is a single mnemonic data point. Holds data imported through the buffer pipeline. Unlike other mnemonic databases, may contain duplicate data points.

This database does not hold data indefinitely. The automated archive pipeline will remove data as it is archived and mined into the primary mnemonic databases.

Struct Parameters

Parameter	Value
type	mn_buffer
v	1.0.0

Database Parameters

Parameter	Value
name	`"buffer"`
label	`"Buffer"`
format	`"{t} {mn_id} {v}"`
singular	`"mnemonic buffer datapoint"`

Fields

Name	Type	Req	Description
t	`instant(us)`	?	time
mn_id	`int(4)`	?	unique mnemonic ID
v	`{conf.type}`		value

Mn Delta

An optimized mnemonic storage solution with each record representing one or more mnemonic data points, by only including points where the mnemonic value actually changes. The value data type is customizable, as with the Mn Full database. By default the mnemonic data group will contain a delta `float8` database.

Struct Parameters

Parameter	Value
type	mn_delta
v	1.0.0
conf.type	numeric data type

Database Parameters

Parameter	Value
name	`"di1"`, `"di2"`, `"di4"`, `"di8"`, `"df4"`, or `"df8"`
label	`"Delta {conf.type}"`
format	`"{t} {mn_id} {v} ({n})"`
singular	`"mnemonic delta datapoint"`

Fields

Name	Type	Req	Description
t	`instant(us)`	?	time
mn_id	`int(4)`	?	unique mnemonic ID
v	`{conf.type}`		value
n	`int(4)`	?	number of datapoints included in this point

Mn Bin Time

Contains mnemonic data binned on fixed time intervals. By default these will be created for 1 minute (`"t60"`) and 10 minute (`"t600"`) bin sizes.

Struct Parameters

Parameter	Value
type	mn_bin_time
v	1.0.0
conf.t	bin size in seconds

Database Parameters

Parameter	Value
name	`"t{conf.t}"`

Parameter	Value
label	`"Time ({conf.t}s)"`
format	`"{t} {mn_id} {avg} ({min}, {max})"`
singular	`"mnemonic bin"`

Fields

Name	Type	Req	Description
t	`instant(us)`	?	start time
mn_id	`int(4)`	?	unique mnemonic ID
t_min	`instant(us)`	?	time of first datapoint
t_max	`instant(us)`	?	time of last datapoint
n	`int(4)`	?	number of datapoints in bin
avg	`float(8)`	?	average
min	`float(8)`	?	min
max	`float(8)`	?	max
med	`float(8)`	?	median
var	`float(8)`	?	sample variance
std	`float(8)`	?	sample standard deviation

Mn Bin Interval

Contains mnemonic data binned by interval events.

Struct Parameters

Parameter	Value
type	mn_bin_interval
v	1.0.0

Database Parameters

Parameter	Value
name	`"interval"`
label	`"Interval"`
format	`"{t} {e_id} {mn_id} {avg} ({min}, {max})"`
singular	`"mnemonic bin"`

Fields

Name	Type	Req	Description

uuid	`uuid`	?	
e_id	`int(8)`	?	
t_start	`instant(us)`	?	start time
t_end	`instant(us)`	?	start time
mn_id	`int(4)`	?	unique mnemonic ID
t_min	`instant(us)`	?	time of first datapoint
t_max	`instant(us)`	?	time of last datapoint
n	`int(4)`	?	number of datapoints in bin
avg	`float(8)`	?	average
min	`float(8)`	?	min
max	`float(8)`	?	max
med	`float(8)`		median
var	`float(8)`		sample variance
std	`float(8)`		sample standard deviation

Mn File Archive

Contains all mnemonic archive files for an origin. Must be a child of an origin group.

Struct Parameters

Parameter	Value
type	mn_file_archive
v	1.0.0

Database Parameters

Parameter	Value
name	`"archive"`
label	`"Archive"`
format	`"{t_start} {t_end}"`
singular	`"archive file"`

Fields

Name	Type	Req	Description
uuid	`uuid`	?	file UUID
t_start	`instant(us)`	?	start time of the time interval that the file covers
t_end	`instant(us)`	?	end time of the time interval that the file covers

Name	Type	Req	Description
dur	`duration(us)`	?	**virtual** duration in microseconds
file_name	`utf8filename`	?	archive file name
t_min	`instant(us)`	?	time of first data in file
t_max	`instant(us)`	?	time of last data in file
format	`asciivstring(32)`		file format (default `"xbin"`)
meta	`jsonobject`		additional metadata as needed
conf	`jsonobject`		configuration for format as needed

Mn File Buffer

Contains all mnemonic buffer files for an origin. Must be a child of an origin group.

Struct Parameters

Parameter	Value
type	mn_file_buffer
v	1.0.0

Database Parameters

Parameter	Value
name	`"buffer"`
label	`"Buffer"`
format	`"{file_name}"`
singular	`"buffer file"`

Fields

Name	Type	Req	Description
uuid	`uuid`	?	file UUID
file_name	`utf8filename`	?	buffer file name
t_min	`instant(us)`	?	time of first data in file
t_max	`instant(us)`	?	time of last data in file
dur	`duration(us)`	?	**virtual** duration in microseconds
state	`struct_buffer_state`	?	buffer file state
flag	`struct_buffer_flag`		buffer file flag
format	`asciivstring(32)`		buffer file format (default `"csv"`)

Name	Type	Req	Description
conf	`jsonobject`		configuration for format as needed

Notes

The state field may be one of four values:

- `PENDING` - the file data is present in the mnemonic buffer database but has not been processed further
- `PROCESSED` - the file has been converted into a standard xbin file format
- `ARCHIVED` - the file contents have been distributed to the appropriate archive file(s)
- `DEPRECATED` - the file is preserved but no longer included in archive files

The flag field may be one of two values:

- `DEPRECATE` - the file is queued for deprecation
- `DELETE` - the file is queued for deletion

Task Databases

Store logs and associated files for data processing tasks.

Archive Task

Each record logs a single execution of an archive task.

Struct Parameters

Parameter	Value
type	task_archive
v	1.0.1

Database Parameters

Parameter	Value
name	`"archive"`
label	`"Archive"`
format	`"{task_id} {t}"`
singular	`"archive task"`

Fields

Name	Type	Req	Description
task_id	`task_id`	?	unique task ID
t	`instant(us)`	?	time when task submitted

Name	Type	Req	Description
meta	`jsonobject`		additional metadata as needed
conf	`jsonobject`		task configuration
archives	`list(jsonobject)`		archives updated
archived	`list(jsonobject)`		buffer file(s) archived
processed	`list(jsonobject)`		buffer file(s) processed
restored	`list(jsonobject)`		buffer file(s) restored
deprecated	`list(jsonobject)`		buffer file(s) deprecated
deleted	`list(jsonobject)`		buffer file(s) deleted

Mine Task

Each record logs a single execution of a mine task.

Struct Parameters

Parameter	Value
type	task_mine
v	1.0.0

Database Parameters

Parameter	Value
name	`"mine"`
label	`"Mine"`
format	`"{task_id} {t_start}"`
singular	`"mine task"`

Fields

Name	Type	Req	Description
task_id	`task_id`	?	unique task ID
t	`instant(us)`	?	time when task submitted
uuid	`uuid`	?	source archive file UUID
t_start	`instant(us)`	?	source archive file start time
t_end	`instant(us)`	?	source archive file end time
meta	`jsonobject`		additional metadata as needed
conf	`jsonobject`		task configuration

Spectra

The spectra definition is a property for event databases.

Property	Value	Req	Description
tabs	array of tab conf(s)		custom tabs for UI
presearch	array of presearch confs		custom pre-search components for UI
filters	array of filter confs		
grouping	array of field name(s)		
charts	charts conf	?	
tables	array of table conf		
query	query conf		
labels	labels conf		

Spectra Tab Conf

Configuration for a spectra search tab. This may be a `string`, referencing the name of a custom tab implementation, or an object with a `"type"` property specifying a tab type and additional properties applicable for that type. Currently there are no custom tab types, but they may be added in the future.

Spectra Database Tab

Under Construction

The database tab employs a record search for a separate target database of any type, and a solution for converting a selection from the target database to the spectra database.

Property	Value	Req	Description
type	`"database"`	?	tab type name
database	database specifier	?	target database specifier
map	see below	?	solution to map target selection to spectra selection

The `"map"` property may be a `string`, `array` of `strings`, or `object`.

If a `string`, the value must be the name of a custom selection function (none currently exist, they may be added in the future).

Spectra Presearch Conf

Specifies a set of components to display before the main spectra search component.

Spectra Field Presearch

Specifies a standalone component to search a particular field.

Property	Value	Req	Description
type	`"field"`	?	presearch type name

Property	Value	Req	Description
field	field specifier	?	
options	see below		options for search dropdown

Spectra Filters Conf

Specifies filters / badges for spectra search.

Property	Value	Req	Description
name	`string`	?	system name for filter
label	`string`		display label (uses name if absent)
badge	`string`		badge label (uses name if absent)
desc	`string`		description for badge / filter tooltip
color	`string`		color code or CSS class
e	`expression`	?	expression to apply for filter

Spectra Charts Conf

Specifies options for each spectra chart.

Property	Value	Req	Description
summary	spectra chart conf	?	summary chart conf
spectra	spectra chart conf	?	spectra chart conf

Spectra Chart Conf

Specifies options for a single spectra chart.

Property	Value	Req	Description
x	`string[]`	?	x axis options
y	`string[]`	?	y axis options
tooltip	`string`		record format string

Spectra Tables Conf

Under Construction

Spectra Query Conf

Under Construction

Spectra Labels Conf

Labels are specified as an `object` mapping standard label values to custom values. These will be defined as needed.

Revision #65
Created 13 July 2023 17:27:53 by Nick Dobson
Updated 16 May 2024 20:16:38 by Bradley Tse