

Record Syntax

JSON Format

A single record may be encoded as a JSON object:

Property	Value
<field name / label>	field type appropriate value / <code>null</code>
<code>"expressions"</code>	JSON object mapping field name/label to expression
<code>"file"</code>	binary object (if database has <code>file</code> enabled)
<code>"tags"</code>	JSON array of string(s) (if database has <code>tag</code> enabled)

The `"expressions"` property allows field values to be specified by expression, rather than explicit value. Between the base object and `"expressions"` object, field may only have a single value provided, or an error will be thrown.

Multiple records may be encoded as a JSON array of JSON objects in this format.

DSV Format

Record data may be provided in a delimiter separated values format. In this case the record data itself is contained in a [binary object](#).

Property	Value
<code>"type"</code>	<code>"dsv"</code> , <code>"csv"</code> , or <code>"tsv"</code>
<code>"file"</code>	binary object
<code>"delimiter"</code>	string (required for <code>"dsv"</code>)
<code>"quote"</code>	string (optional)

The `"csv"` and `"tsv"` types specify default delimiters of comma (,) and tab (\t), respectively.

Example

```
{
  "records": {
    "type": "dsv",
    "file" : "<object ID>",
    "delimit": ";"
  }
}
```

```
}
```

The format of the separated values file is largely based on the [RFC 4180 standard](#). The specific requirements are:

- lines must end with `LF (\n)` or `CR LF (\r\n)`
- line breaks cannot be used in values
- the default quote character is `"` (double quotes)
- any field *may* be quoted by the quote character
- any field containing the delimiter must be quoted
- a quote character in a quoted value must be represented by two quote characters
- the first row must contain the names of each field
- blank lines with no data are ignored

Revision #20

Created 9 June 2022 16:21:12 by Nick Dobson

Updated 15 January 2026 20:43:45 by Nick Dobson