Splunk® Enterprise
Search Reference 8.0.4

Understanding SPL syntax

Generated: 6/18/2020 8:19 am

Copyright (c) 2020 Splunk Inc. All Rights Reserved
Understanding SPL syntax

The following sections describe the syntax used for the Splunk SPL commands. For additional information about using keywords, phrases, wildcards, and regular expressions, see Search command primer.

Required and optional arguments

SPL commands consist of required and optional arguments.

- Required arguments are shown in angle brackets < >.
- Optional arguments are enclosed in square brackets [ ].

Consider this command syntax:

```
bin [<bins-options>...] <field> [AS <newfield>]
```

The required argument is `<field>`. To use this command, at a minimum you must specify `bin <field>`.

The optional arguments are `[<bins-options>...]` and `[AS <newfield>]`.

User input arguments

Consider this command syntax:

```
replace (<wc-string> WITH <wc-string>)... [IN <field-list>]
```

The user input arguments are: `<wc-string>` and `<field-list>`.

Repeating arguments

Some arguments can be specified multiple times. The syntax displays ellipsis ... to specify which part of an argument can be repeated. The ellipsis always appear immediately after the part of the syntax that you can repeat.

Consider this command:

```
convert [timeformat=string] (<convert-function> [AS <field>])...
```

The required argument is `<convert-function>`, with an option to specify a field with the `[AS <field>]` clause.

Notice the ellipsis at the end of the syntax, just after the close parenthesis. In this example, the syntax that is inside the parenthesis can be repeated `<convert-function> [AS <field>]`.

In the following syntax, you can repeat the `<bins-options>`.

```
bin [<bins-options>...] <field> [AS <newfield>]
```

Grouped arguments

Sometimes the syntax must display arguments as a group to show that the set of arguments are used together. Parenthesis ( ) are used to group arguments.
For example in this syntax:

```
replace (<wc-string> WITH <wc-string>)... [IN <field-list>]
```

The grouped argument is  

```
(<wc-string> WITH <wc-string>).... This is a required set of arguments that you can repeat multiple times.
```

**Keywords**

Many commands use keywords with some of the arguments or options. Examples of keywords include:

- AS
- BY
- OVER
- WHERE

You can specify these keywords in uppercase or lowercase in your search. However, for readability, the syntax in the Splunk documentation uses uppercase on all keywords.

**Quoted elements**

If an element is in quotation marks, you must include that element in your search. The most common quoted elements are parenthesis.

Consider the syntax for the chart command:

```
chart [chart-options] [agg=<stats-agg-term>]
{ <stats-agg-term> | <sparkline-agg-term> | "("<eval-expression">")" }...
[ BY <row-split> <column-split> ] | [ OVER <row-split> ] [BY <column-split>] ]
```

There are quotation marks on the parenthesis surrounding the `<eval-expression>`. This means that you must enclose the `<eval-expression>` in parenthesis in your search.

In the following search example, the `<eval-expression>` is `avg(size)/max(delay)` and is enclosed in parenthesis.

```
... | chart eval(avg(size)/max(delay)) AS ratio BY host user
```

**Argument order**

In the command syntax, the command arguments are presented in the order in which the arguments are meant to be used.

In the descriptions of the arguments, the **Required arguments** and **Optional argument** sections, the arguments are listed alphabetically. For each argument, there is a **Syntax** and **Description**. Additionally, for Optional arguments, there might be a **Default**.

**Data types**

The nomenclature used for the data types in SPL syntax are described in the following table.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>chart</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;bool&gt;</td>
<td>boolean</td>
<td>Use <code>true</code> or <code>false</code>. Other variations are accepted. For example, for <code>true</code> you can also use 't', 'T', 'TRUE', 'yes', or the number one (1). For <code>false</code> you can also specify 'no', the number zero (0), and variations of the word <code>false</code>, similar to the variations of the word <code>true</code>.</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
| <field> | A field name. You cannot specify a wildcard for the field name. | See `<wc-field>`.
| <int> or <integer> | An integer that can be a positive or negative value. | Sometimes referred to as a "signed" integer. See `<unsigned int>`.
| <string> | string | See `<wc-string>`.
| <unsigned int> | unsigned integer | An unsigned integer must be positive value. Unsigned integers can be larger numbers than signed integers.
| <wc-field> | A field name or a partial name with a wildcard character to specify multiple, similarly named fields. | Use the asterisk ( * ) character as the wildcard character.
| <wc-string> | A string value or partial string value with a wildcard character. | Use the asterisk ( * ) character as the wildcard character.

### Boolean operators

When a boolean operator is included in the syntax of a command, you must always specify the operator in uppercase. Boolean operators include:

- **AND**
- **OR**
- **NOT**

To learn more about the order in which boolean expressions are evaluated, along with some examples, see Boolean expressions in the *Search Manual*.

To learn more about the the NOT operator, see Difference between NOT and != in the *Search Manual*.

### BY clauses

A `<by-clause>` and a `<split-by-clause>` are not the same argument.

When you use a `<by-clause>`, one row is returned for each distinct value `<by-clause>` field. A `<by-clause>` displays each unique item in a separate row. Think of the `<by-clause>` as a grouping.

The `<split-by-clause>` displays each unique item in a separate column. Think of the `<split-by-clause>` as a splitting or dividing.

Wildcard characters ( * ) are not accepted in BY clauses.

### Fields and wildcard fields

When the syntax contains `<field>` you specify a field name from your events.

Consider this syntax:
bin [<bins-options>...] <field> [AS <newfield>]

The <field> argument is required. You can specify that the field displays a different name in the search results by using the [AS <newfield>] argument. This argument is optional.

For example, if the field is `categoryId` and you want the field to be named `CategoryID` in the output, you would specify:

```
categoryId AS CategoryID
```

The <wc-field> argument indicates that you can use wild card characters when specifying field names. For example, if you have a set of fields that end with "log" you can specify `*log` to return all of those fields.

If you use a wild card character in the middle of a value, especially as a wild card for punctuation, the results might be unpredictable.

**See also**

In the *Search Manual*:

- Anatomy of a search
- Wildcards
- Field expressions
- Quotes and escaping characters