Use the **where** operator to filter if a Boolean field is true or false. For example, using the boolean field "valid":

- Filter to keep true: | where valid
- Filter to keep false: | where !valid

To filter results in a search query, use "where" as a conditional operator. The where operator must appear as a separate operator distinct from other operators, delimited by the pipe symbol ("|”). In other words, the following construct will not work and will generate a syntax error:

This query will NOT work:

```
...| parse "seconds=*;" as time where > 5
```

Instead, separate the **where** operator from the preceding **parse** operator like this:

```
...| parse "seconds=*;" as time
| where time > 5
```

**Syntax**

- ... | where <boolean expression> | ...

**Rules**

- The pipe delimiter is required to separate the **where** operator as a distinct query operator.
- The **where** operator **cannot** be used inline as a query clause, like "... | extract a where b==something | ..."
- Multiple **where** operators are processed in the order they are specified, with each subsequent **where** operator further filtering results.
- **Keyword expressions** can be used in the boolean expression, such as OR and AND.
- If you are using **in** or **not in** to match integers, **cast "x" to a number first**.
- The **matches** operator can be used in the boolean expression. You can use an **RE2 compliant** regular expression or use asterisks * as wildcards.
- Any operator that returns a boolean value can be used in the boolean expression. Such as **compareCIDRPrefix**, **contains**, **in**, **isBlank**, **isEmpty**, **isNull**, **isNumeric**, **isPrivateIP**, **isPublicIP**, **isValidIP**, and **math expressions**.

Copyright 2018 Sumo Logic, Inc. All rights reserved.
Examples

• ... | where a<b
• ... | where a=x
• ... | where a>=x
• ... | where a<=x
• ... | where a<x
• ... | where x<10
• ... | where x="some string"
• ... | where error="fail*"
• ... | where user<>"root"
• ... | where x matches "some string"
• ... | where x matches "fail*"
• ... | where x matches /regex/
• ... | where !(x matches /regex/)
• ... | num(x) | where x in (4, 3, 5)
• ... | where x in ("error", "fail")
• ... | where x not in ("error", "fail")
• ... | where x matches "Android" or x matches "iPhone" or x matches "iPad"

Using the "not" option

If you need a query using the where operator, where xxx DOES NOT match yyy, use "!" followed by the matches operator enclosed in parenthesis.

For example:

      ...| where !(<field xxx> matches "<value yyy>") | ...

or:

      ...| where !(status matches "200")

Use where to check for null values

For details, see isNull operator.