Browsing examples:

- search 10 minutes
- search 1 hour
- search 1 day
- search 1 week
- search 1 month
- search 1 year
- search 2 years
- search 10 years
- search 100 years

Regular expressions (REGEXES)

A regular expression is a pattern used to match characters or strings of text. Regular expressions are case-sensitive by default, but can be made case-insensitive using the `-i` option. Parentheses `()` are used for capturing groups of text that can be referenced in the output. Backslashes `\` are used to escape special characters.

For example:

```
<table>
<thead>
<tr>
<th>Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\d{1,3}</code></td>
<td>Matches one to three digits</td>
</tr>
<tr>
<td><code>\w{10}</code></td>
<td>Matches any word that is 10 characters long</td>
</tr>
<tr>
<td><code>\s+</code></td>
<td>Matches one or more spaces</td>
</tr>
<tr>
<td><code>\S+</code></td>
<td>Matches one or more non-space characters</td>
</tr>
<tr>
<td><code>[^\d]</code></td>
<td>Matches any character that is not a digit</td>
</tr>
<tr>
<td><code>\b.*\b</code></td>
<td>Matches any word that is surrounded by a boundary</td>
</tr>
<tr>
<td><code>\b\w{2}\b</code></td>
<td>Matches any word that is exactly two characters long</td>
</tr>
<tr>
<td><code>\b\w{2}\b.*\b</code></td>
<td>Matches any word that is exactly two characters long and is surrounded by a boundary</td>
</tr>
<tr>
<td><code>\b\w{2}\b.*\b</code></td>
<td>Matches any word that is exactly two characters long and is surrounded by a boundary</td>
</tr>
<tr>
<td><code>\b\w{2}\b.*\b</code></td>
<td>Matches any word that is exactly two characters long and is surrounded by a boundary</td>
</tr>
</tbody>
</table>

Common Splunk time formats

Splunk provides complex date and time expressions for event time parsing and display. The following table lists some common Splunk time formats and examples of how to use them.

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\d\d\d\d-\d\d-\d\d</code></td>
<td>Matches any four-digit year</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dT\d\d:\d\d:\d\d</code></td>
<td>Matches any four-digit year with time in 24-hour format</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dT\d\d:\d\d:\d\dZ</code></td>
<td>Matches any four-digit year with time in 24-hour format and UTC time zone</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dZ</code></td>
<td>Matches any four-digit year with UTC time zone</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dZ</code></td>
<td>Matches any four-digit year with UTC time zone</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dZ</code></td>
<td>Matches any four-digit year with UTC time zone</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dZ</code></td>
<td>Matches any four-digit year with UTC time zone</td>
</tr>
<tr>
<td><code>\d\d\d\d-\d\d-\d\dZ</code></td>
<td>Matches any four-digit year with UTC time zone</td>
</tr>
</tbody>
</table>

Security practices:

- Use strong passwords
- Enable two-factor authentication
- Keep software up to date
- Regularly back up data
- Use firewalls and intrusion detection systems
- Monitor logs for suspicious activity
- Be cautious of phishing attempts
- Use antivirus software
- Be aware of social engineering tactics
- Regularly update firewalls and other security measures

Conclusion:

By implementing these security practices, organizations can significantly reduce their risk of cyber attacks and protect their sensitive data. Regular security audits and employee training are also crucial in maintaining a secure environment.
### COMMON SEARCH COMMANDS

**COMMAND** | **DESCRIPTION** | **EXAMPLES**
--- | --- | ---
| **base** | Returns the basic field that matches the search arguments. | `base` | **foo**
| **case** | Returns values contained in a field, if any, that match the arguments passed. | `case(field)` | `case(text, number)`
| **count** | Returns the number of occurrences of any field. | `count(field)` | `count(a)`
| **sort** | Sorts events in a table by a particular field. | `sort(field)` | `sort@time` | `sort(time)`
| **range** | Returns the range of a particular field. | `range(field)` | `range(a)`
| **add** | Adds field values from an external source. | `add(field)` | `add(field)`
| **mvjoin** | Joins multiple fields into one. | `mvjoin(field)` | `mvjoin(field)`
| **search** | Searches for particular fields in events. | `search(field)` | `search(a)`
| **remove** | Removes fields from search results. | `remove(field)` | `remove(field)`
| **replace** | Replaces field values. | `replace(field)` | `replace(field)`

### COMMON STAT FUNCTIONS

**FUNCTION** | **DESCRIPTION** | **EXAMPLES**
--- | --- | ---
| **perc<X>(Y)** | Returns the X-th percentile value of the field Y. For example, perc5(total) returns the 5th percentile value of a field "total". | `perc<X>(Y)` | `perc<X>(Y)`
| **mode** | Returns the most frequent value of the field X. | `mode(X)` | `mode(X)`
| **last** | Returns the last value of the field X. | `last(X)` | `last(X)`
| **sum** | Returns the sum of the values of the field X. | `sum(X)` | `sum(X)`
| **count** | Returns the number of occurrences of the field. | `count(X)` | `count(X)`
| **mean** | Returns the mean of the field X. | `mean(X)` | `mean(X)`
| **median** | Returns the median of the field X. | `median(X)` | `median(X)`
| **q<X>(Y)** | Returns the X-th quantile value of the field Y. | `q<X>(Y)` | `q<X>(Y)`

### SEARCH LANGUAGE

A search is a command that uses a query language to request that Splunk do something with data. Searches are specified using the search language, which is a powerful language used to perform various operations on data. Searches are written using the `|` command to chain together a series of commands and arguments.

#### Example:

```
search index=web httpd
```

This search will retrieve all events from the `web` index and `httpd` events.

### Relative Time Modifiers

Relative time modifiers allow you to specify a range of time from a particular time or based on the current time. For example, you can use `-1h` to retrieve events from the last hour.

#### Example:

```
search last=1h
```

This search will retrieve events from the last hour.

### Optimizing Searches

Optimizing searches involves using Splunk's search language to retrieve data efficiently. Here are some tips for optimizing searches:

1. **Limit the amount of data returned** by adding a `| limit` command after the search arguments.
2. **Use the `search` command** to specify the data you want to retrieve.
3. **Specify the exact time range** you need using the `last` modifier.
4. **Use the `|` command to chain together** multiple search arguments.

### Example:

```
split=unique
```

This search will retrieve a subset of the multivalued field from a start position (zero-based) Y to Z (optional).