# Using Fields

# Document Usage Guidelines

- Should be used only for enrolled students
- Not meant to be a self-paced document, an instructor is needed
- Lab Exercise slides reference the hands-on lab exercise guide
- Do not distribute

# Course Goals

- Define fields in Splunk
- Explain index-time versus search-time
- Create a selected field
- Use fields in searches
- Explain temporary versus persistent fields
- Describe data enrichment methods

# Course Outline

- What are Fields?
- What is Field Discovery?
- Use Fields in Searches
- Compare Temporary versus Persistent Fields
- Enrich Data



# What are Fields?

# Topic Objectives

- Define fields and field auto-extraction
- Explore the Fields sidebar
- Add fields to the Selected Fields list
- Explore and generate reports from the Fields window

# What Are Fields?

- Fields are knowledge objects that represent searchable key/value pairs in your event data, e.g. `host=www1` and `status=503`
- Fields can be assigned to or extracted from events at various times in the data pipeline and search process



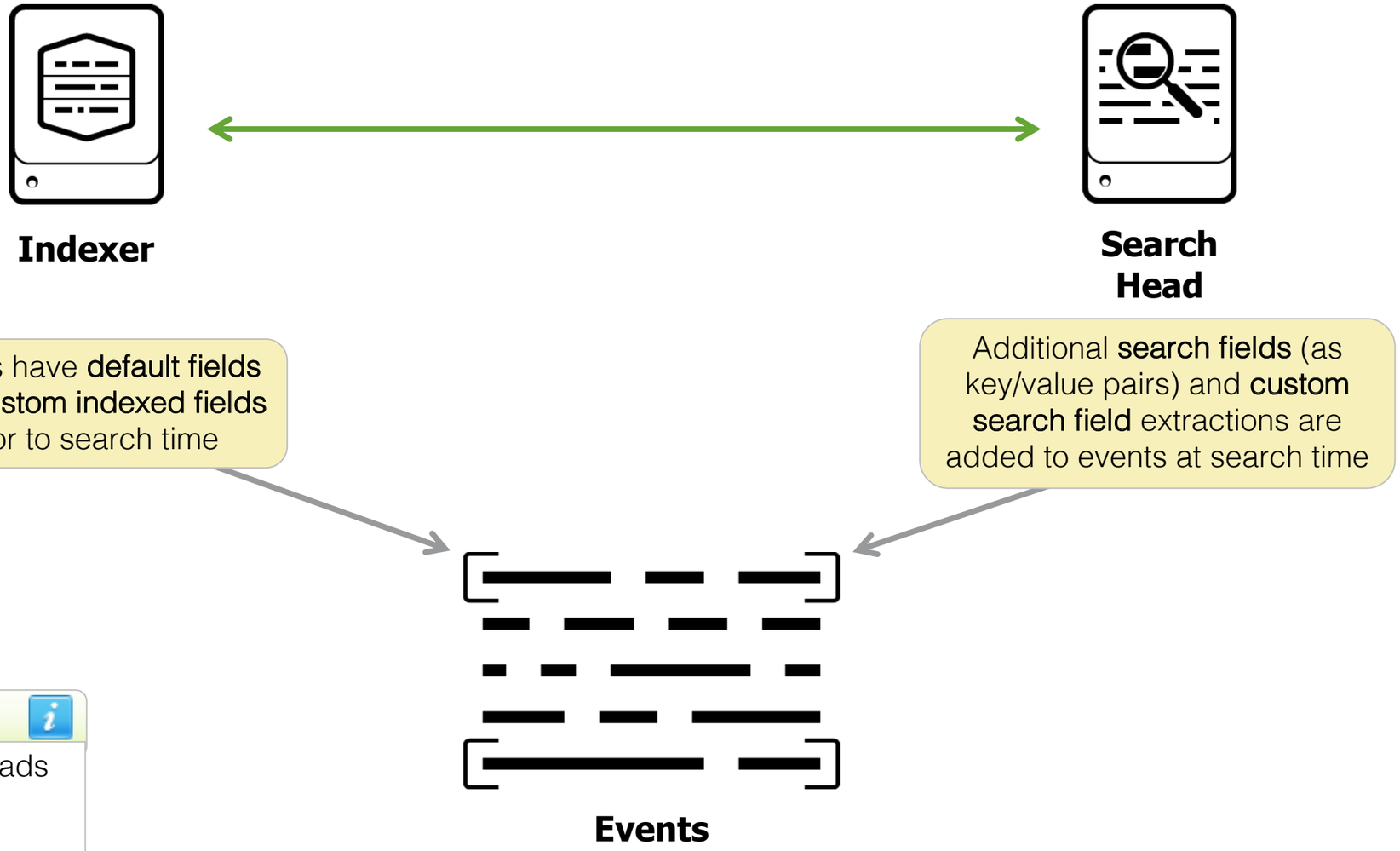
# Field Auto-Extraction: Before Search

- Prior to search time, some fields are already stored with the event in the index
  - Important basic default fields: `host`, `source`, `sourcetype`
  - Other basic default fields: `index, linecount, punct, splunk_server, timestamp`
- Custom indexed fields
- Internal fields

# Field Auto-Extraction: At Search

- At search time, Splunk automatically looks for **`key=value`** patterns and extracts them into field-value pairs for events associated with a specific host, source, or sourcetype
- The search mode determines the fields returned at search time (discussed in next topic)

# Field Auto-Extraction




**Note**

Indexers and Search Heads are components of the Splunk data pipeline.

# Identify Data-Specific Fields

Data-specific fields come from the specific attributes of your data

| i | Time | Event |
|---|---|---|
| > | 5/19/22 4:13:08.000 PM | 198.228.212.52 - - [19/May/2022:16:13:08] "POST /cart/error.do?msg=FormError&JSESSIONID=SD0SL6FF4ADFF4953 HTTP 1.1" 200 2426 "http://www.buttercupgames.com/cart.do?action=purchase&itemId=EST-7" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/534.55.3 (KHTML, like Gecko) Version/5.1.5 Safari/534.55.3" 802 |

200

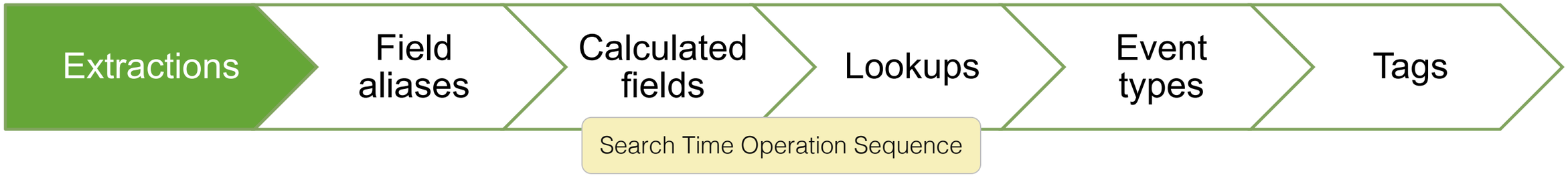
action=purchase

Sometimes it is based on a sequence of characters that are recognized by the sourcetype; in this case, **access_combined** interprets 200 as a status (**status=200**)

Sometimes this is indicated by obvious **key=value** pairs

# Extracted Fields are Knowledge Objects

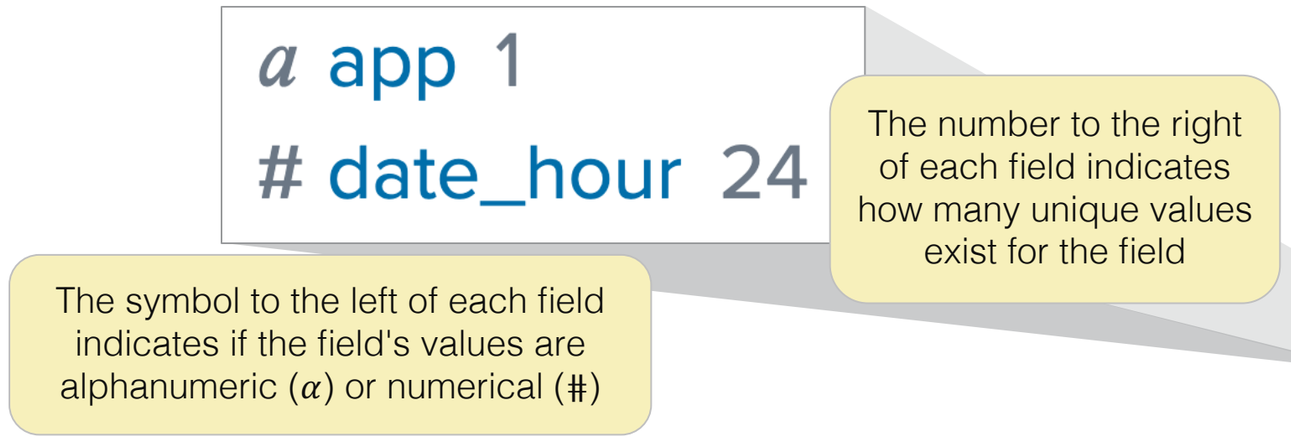
- Knowledge objects provide specific information about your data
- Extracted fields are executed first in search time operations
- Field aliases and calculated fields are custom fields that can add additional context to and build on extracted fields

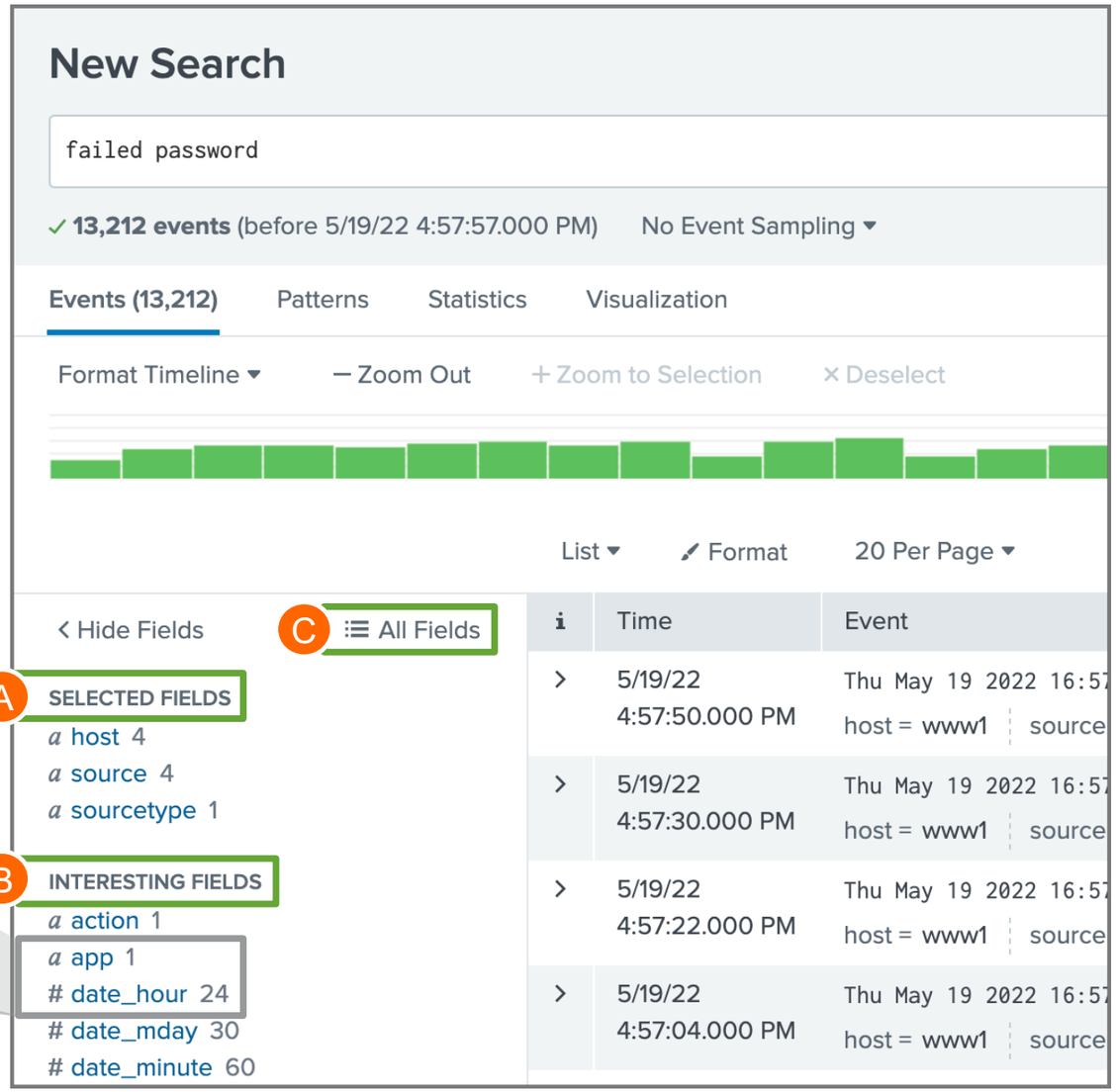


# Fields Sidebar

For the current search:

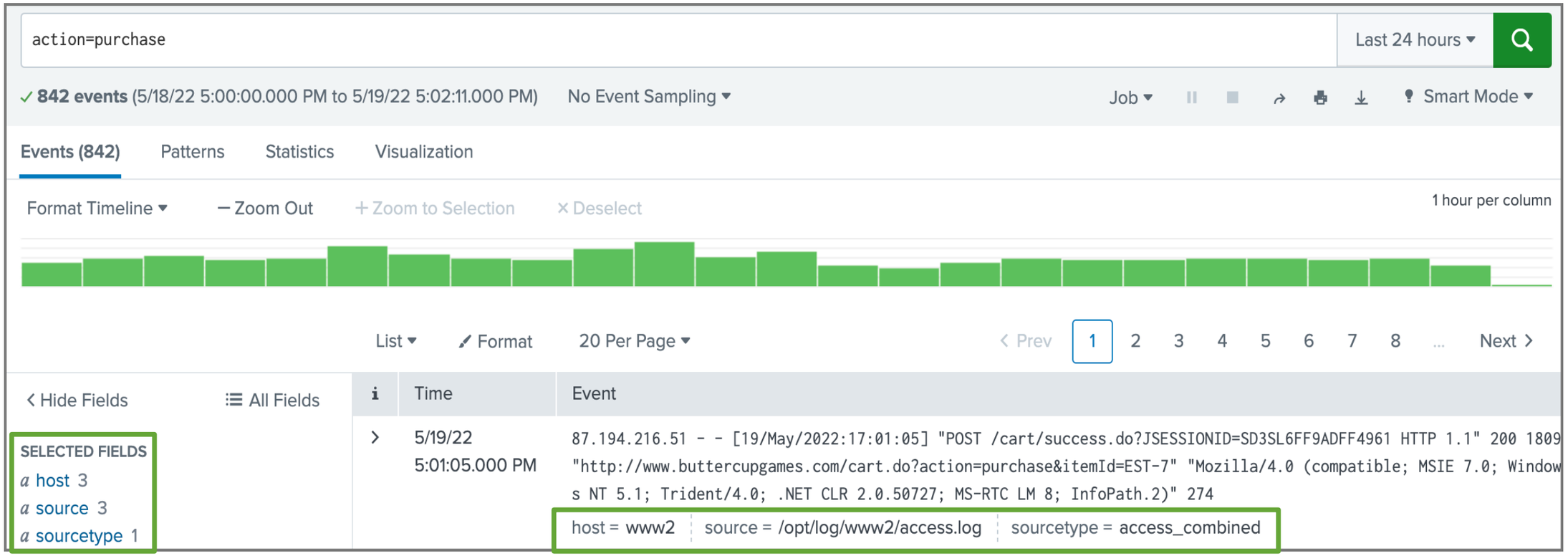
- **A Selected Fields:** a set of fields displayed for each event
- **B Interesting Fields:** occur in at least 20% of resulting events
- **C All Fields:** link to view all fields (including non-interesting fields)



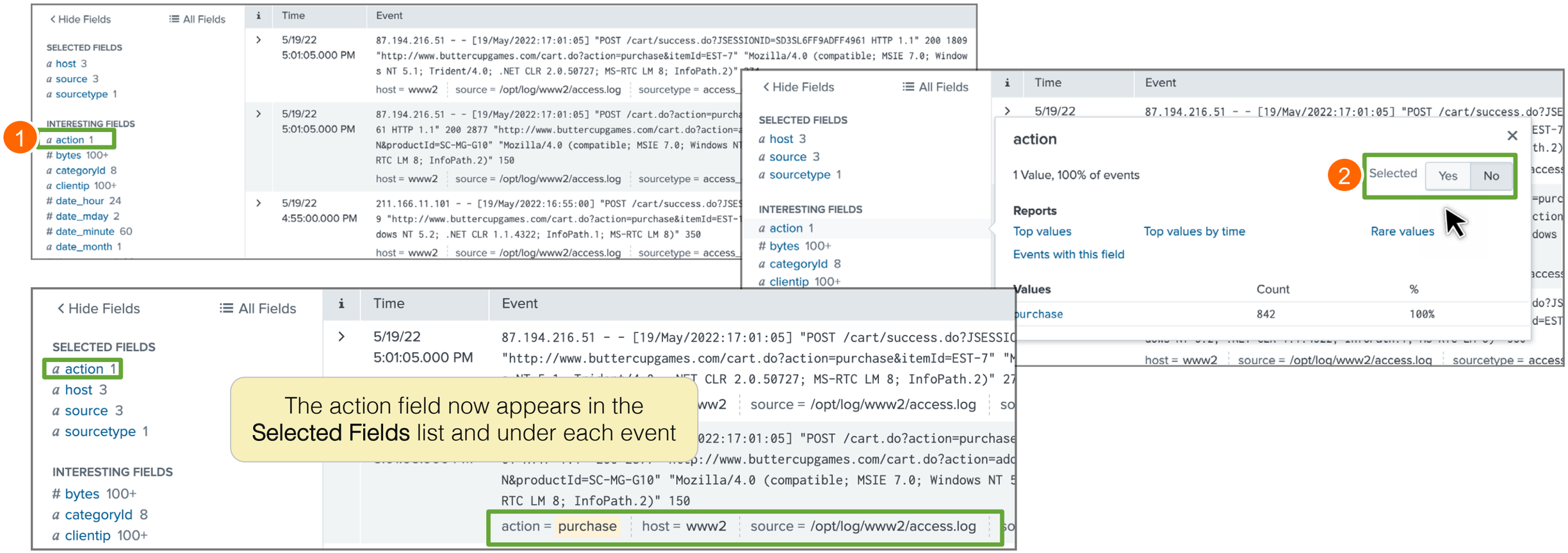
# Selected Fields

- Listed under every event that includes those fields
- Default selected fields: **`host`**, **`source`**, **`sourcetype`**



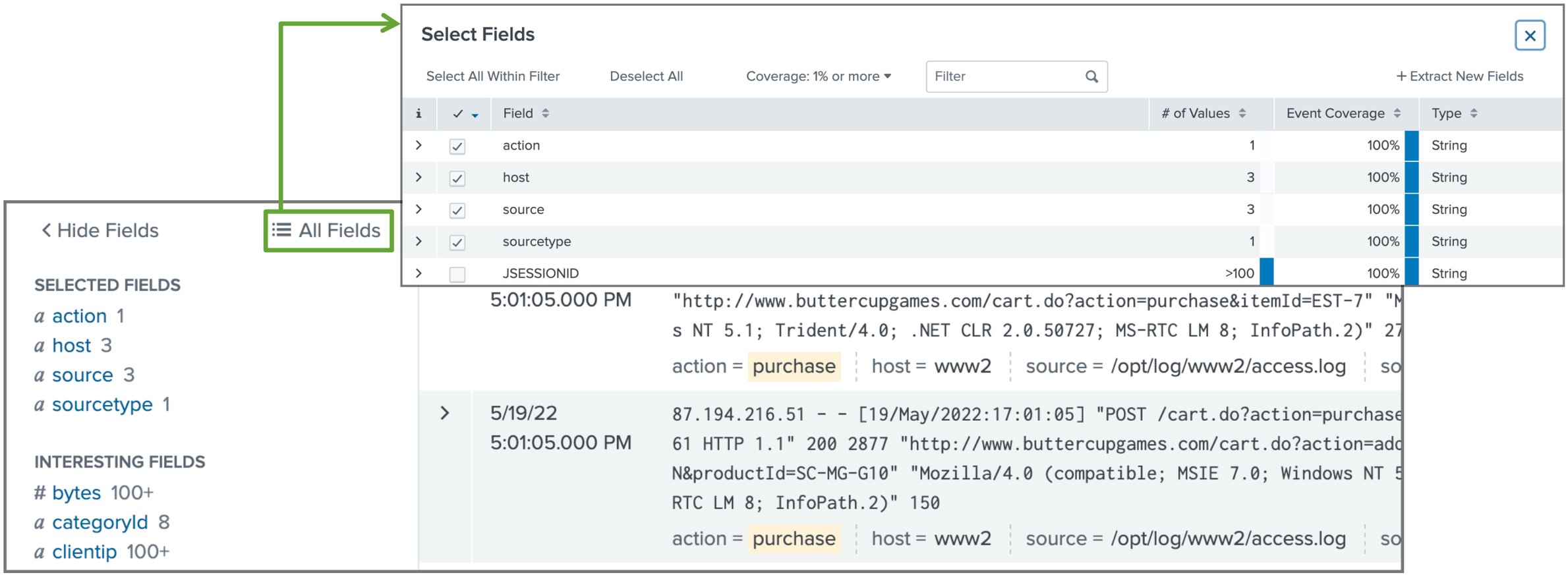
# Add Fields to Selected Field List

## **Interesting Fields** can be added to the **Selected Fields** list



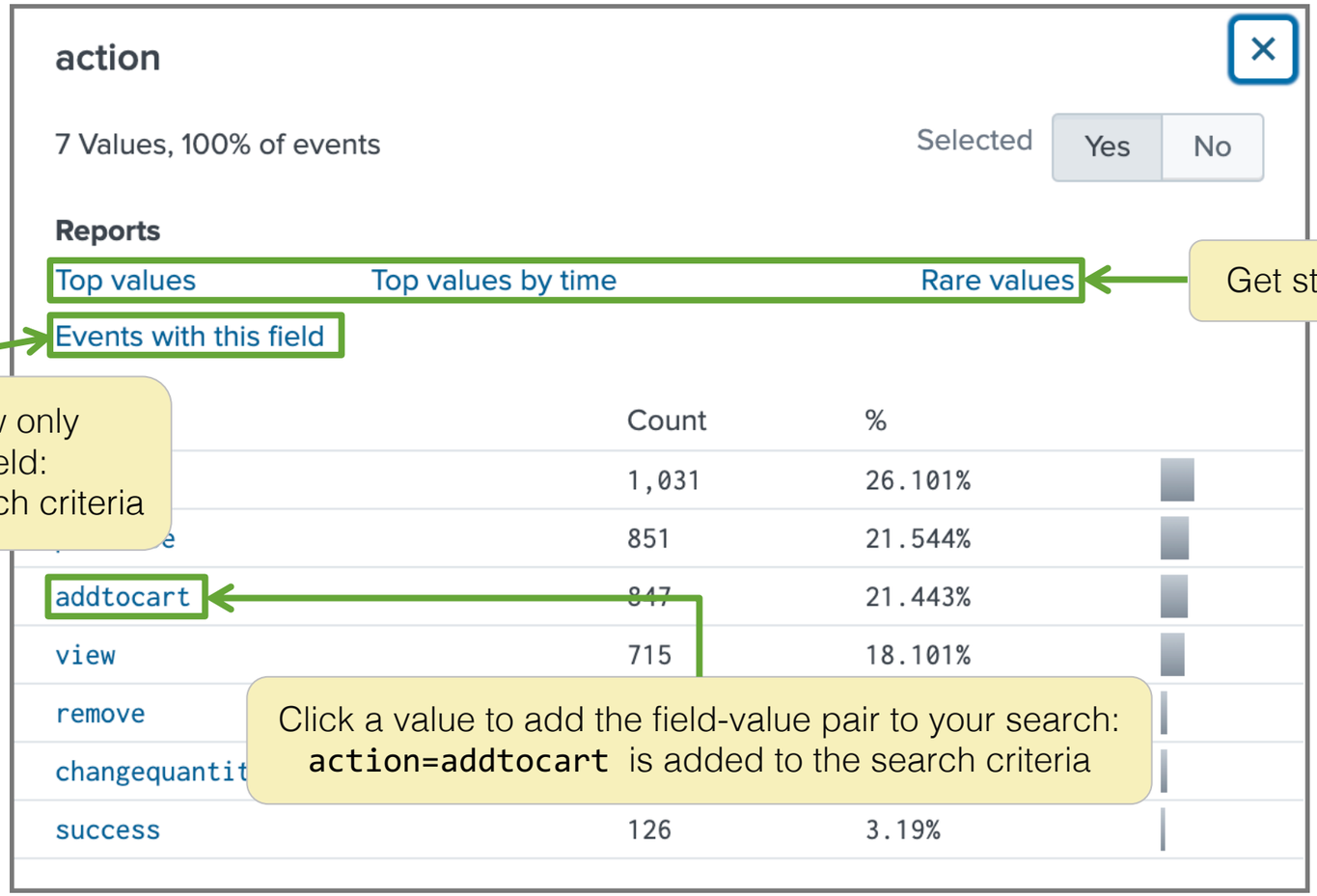
# Make Any Field Selected

From **All Fields**, you can identify other fields as selected fields or change the event coverage amount



# The Field Window

## Appears after clicking on a field in the Fields sidebar



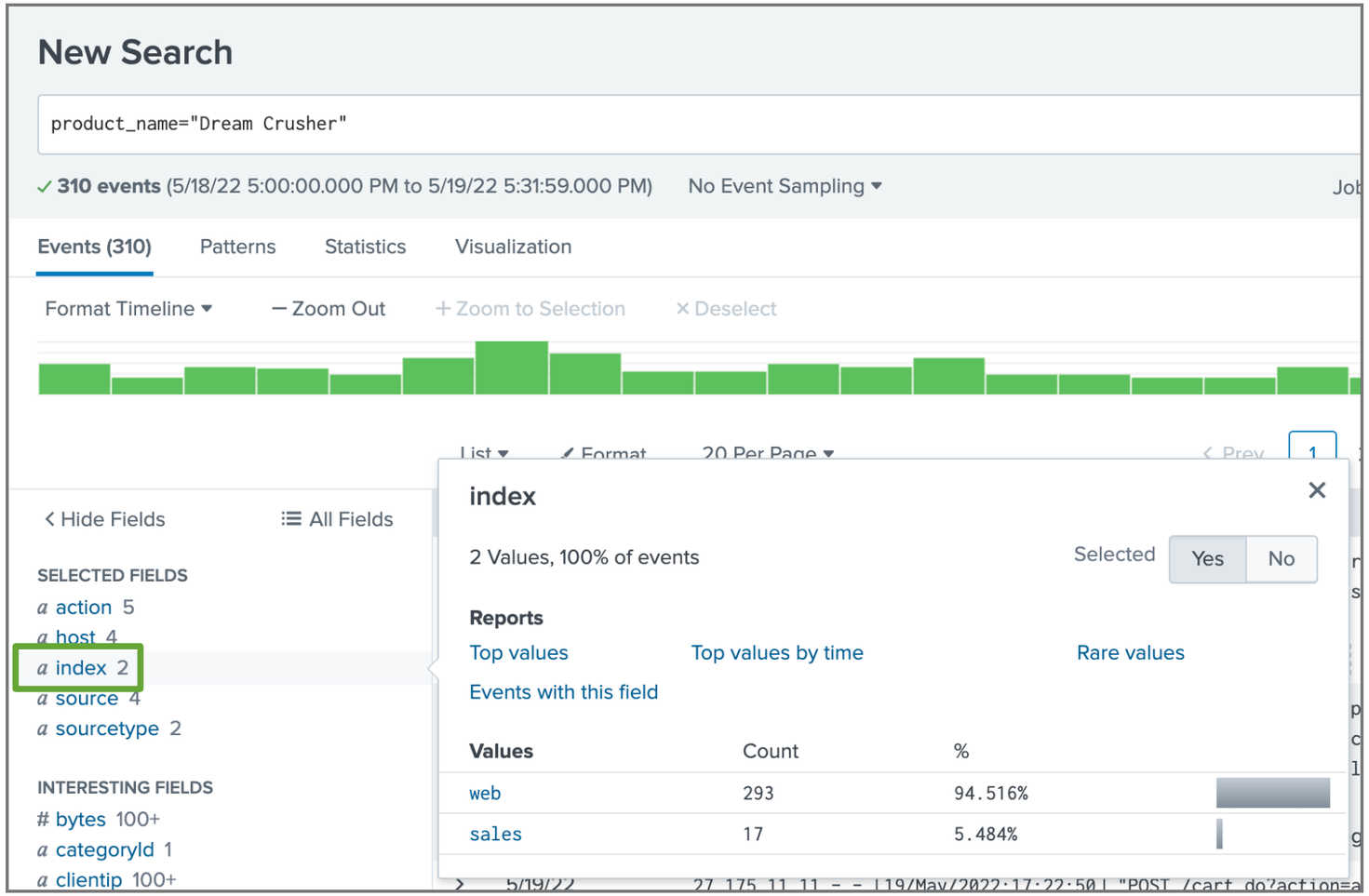

Get statistical results

Narrows the search to show only results that contain this field:
`action=*` is added to the search criteria

Click a value to add the field-value pair to your search:
`action=addtocart` is added to the search criteria

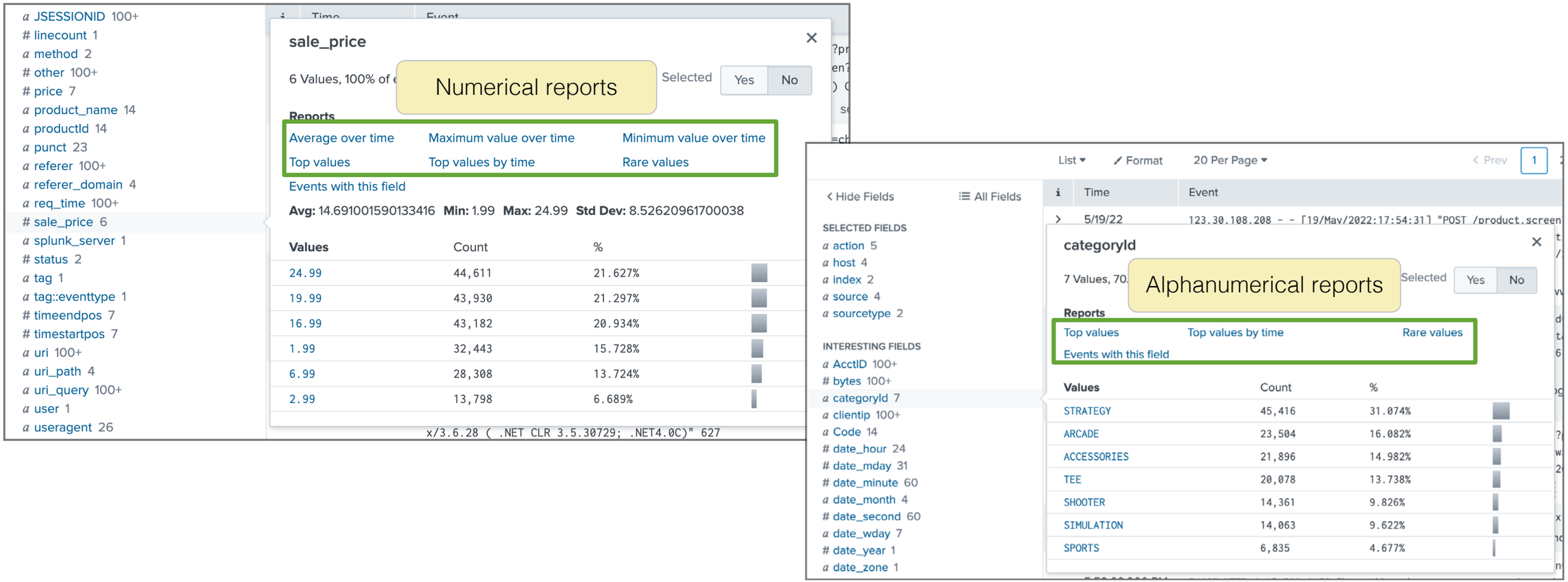
# View the `index` Field

- `index` always appears as a field in search results
- If no index is indicated in the search, data is returned from all default indexes for the role of the user executing the search
  - It is a best practice to always specify indexes when creating a search



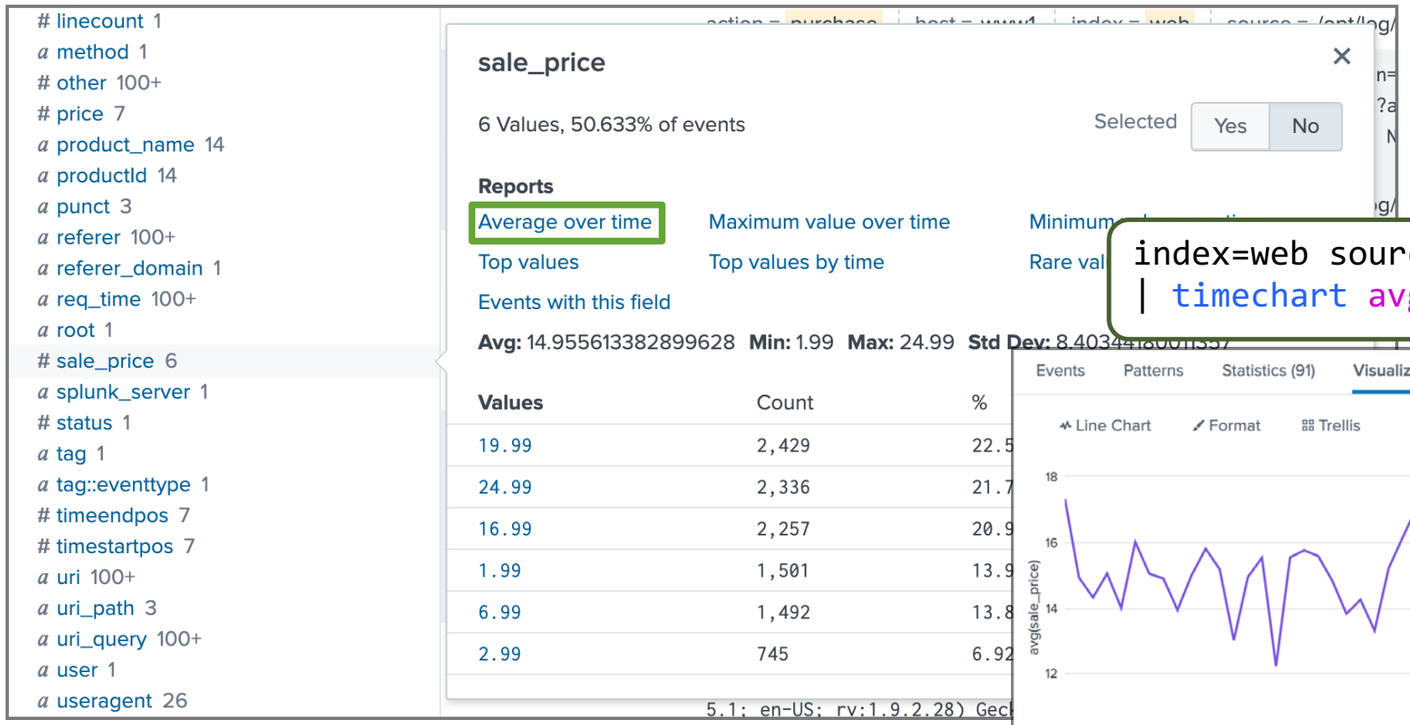
# Field Window: Reports

## Alphanumeric and numeric fields offer different report options

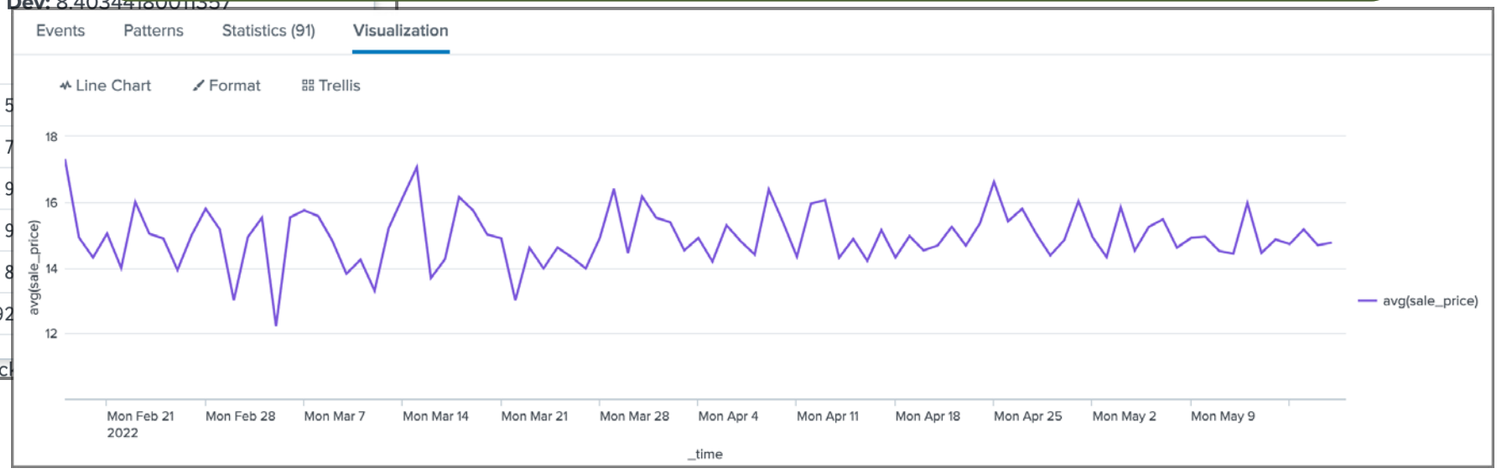


# Field Window: Reports (cont.)

```
index=web sourcetype=access_combined status=200 action=purchase
```

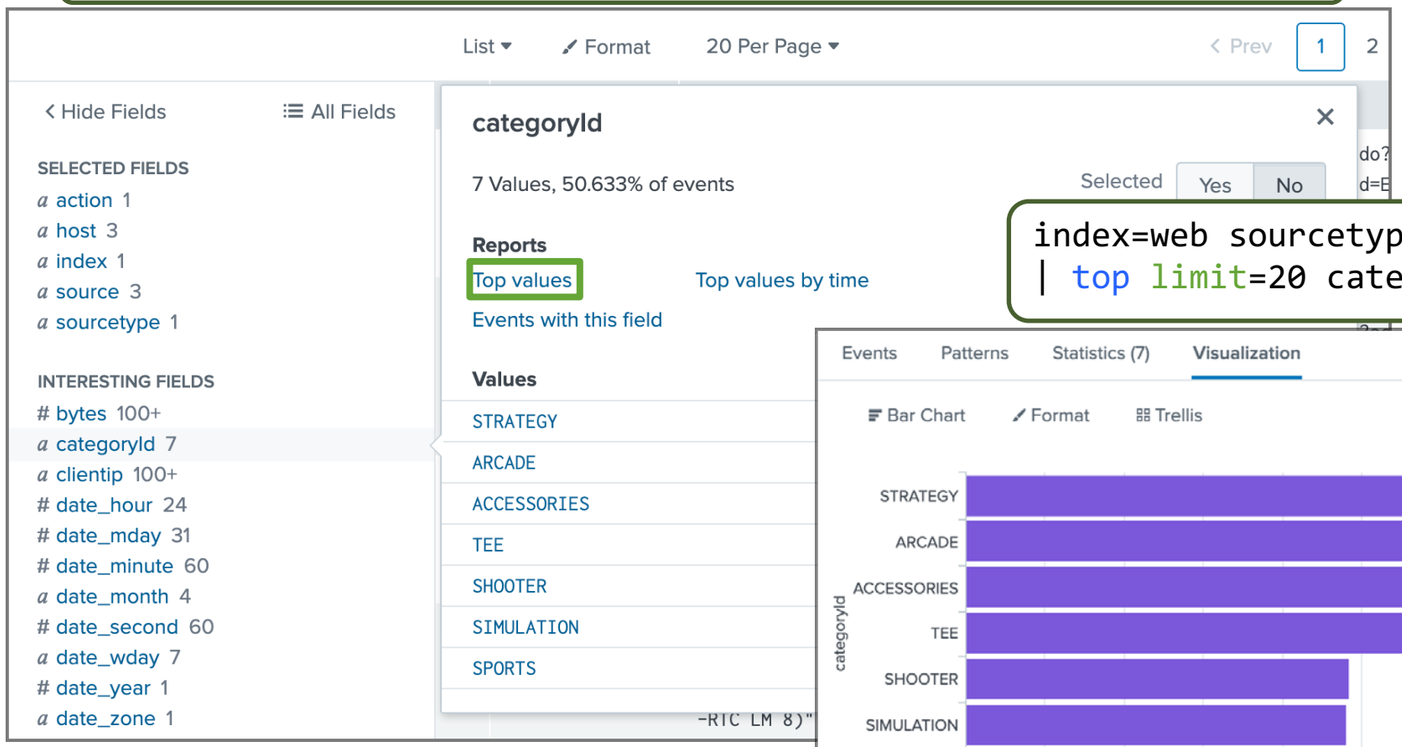



```
index=web sourcetype=access_combined status=200 action=purchase
| timechart avg(sale_price)
```

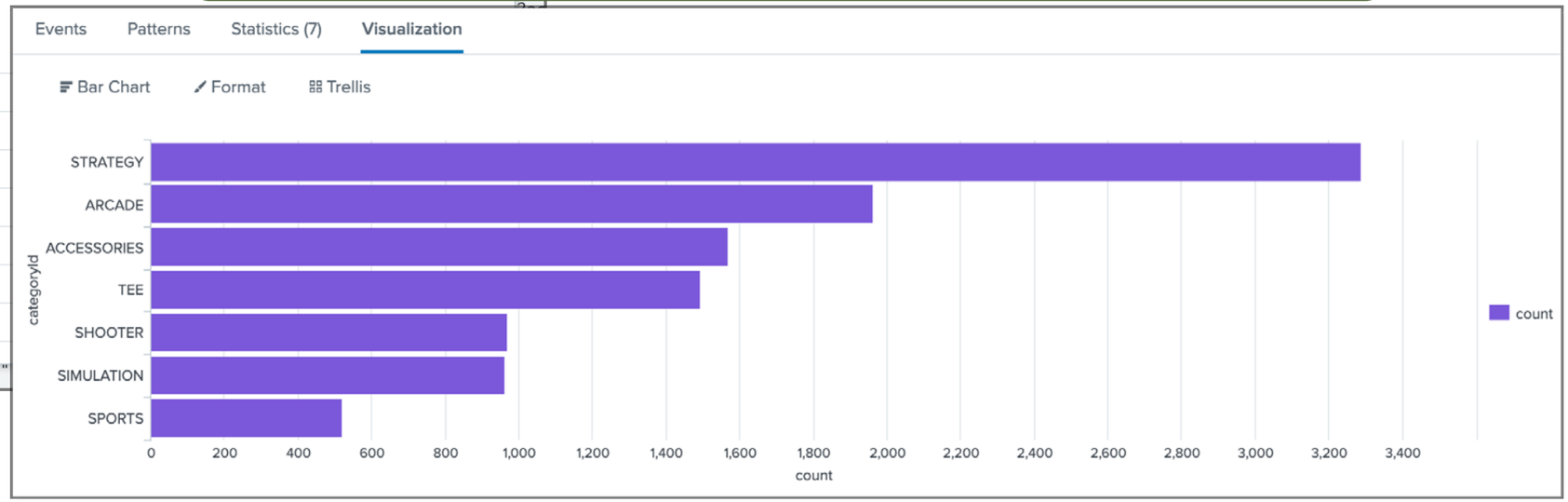


# Field Window: Reports (cont.)

```
index=web sourcetype=access_combined status=200 action=purchase
```




```
index=web sourcetype=access_combined status=200 action=purchase
| top limit=20 categoryId
```



# What is Field Discovery?

# Topic Objectives

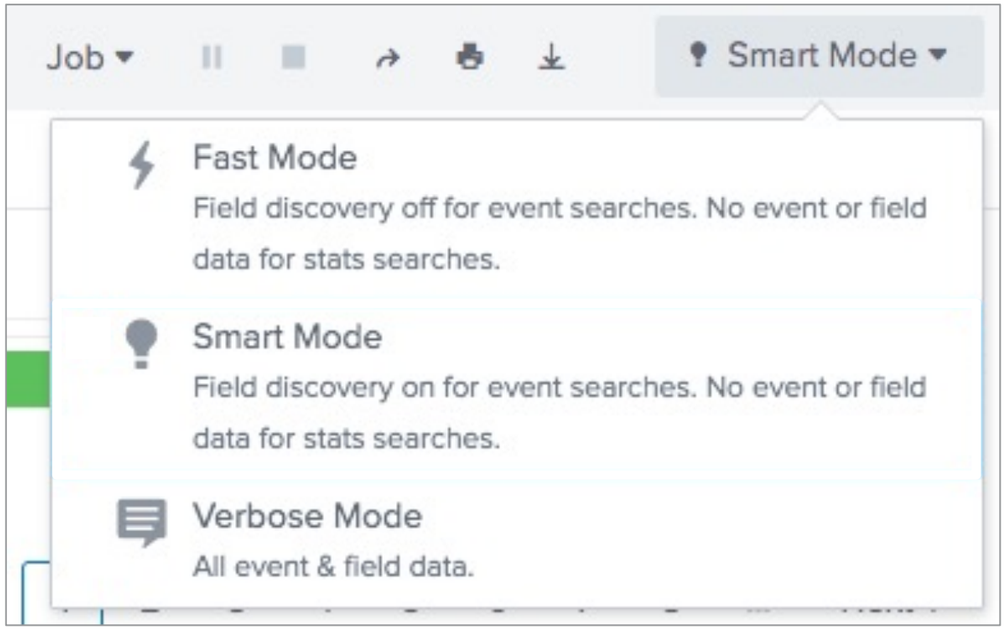
- Understand Field Discovery
- Explore search modes and their effect on search results

# Field Discovery

- At search time, Field Discovery extracts fields from raw event data, including those directly related to the search's results
  - Identifies and extracts the first 100 obvious **`key=values`** pairs in the raw event data
  - Extracts any field explicitly mentioned in the search
  - Performs Custom field extractions defined by the user
- Field Discovery is based on the searched sourcetype, as well as **`key=value`** pairs found in the data

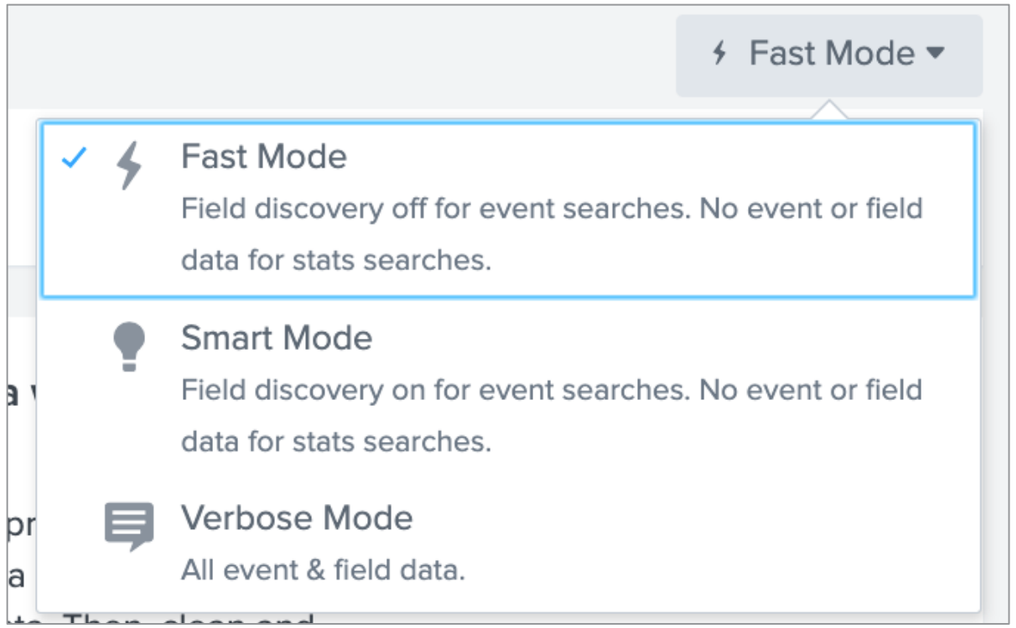
# Search Modes and Field Discovery

- The Splunk user interface provides 3 search modes
- Search modes determine how much field data is returned as search results, and affects how fast the search completes



# Fast Mode

- Prioritizes speed over completeness
- Disables Field Discovery and:
  - Returns default fields and indexed field extractions
  - Extracts and returns specific fields if those fields are specified in the basic search
- If a transforming search is run, the results display on the **Statistics** or **Visualizations** tabs only

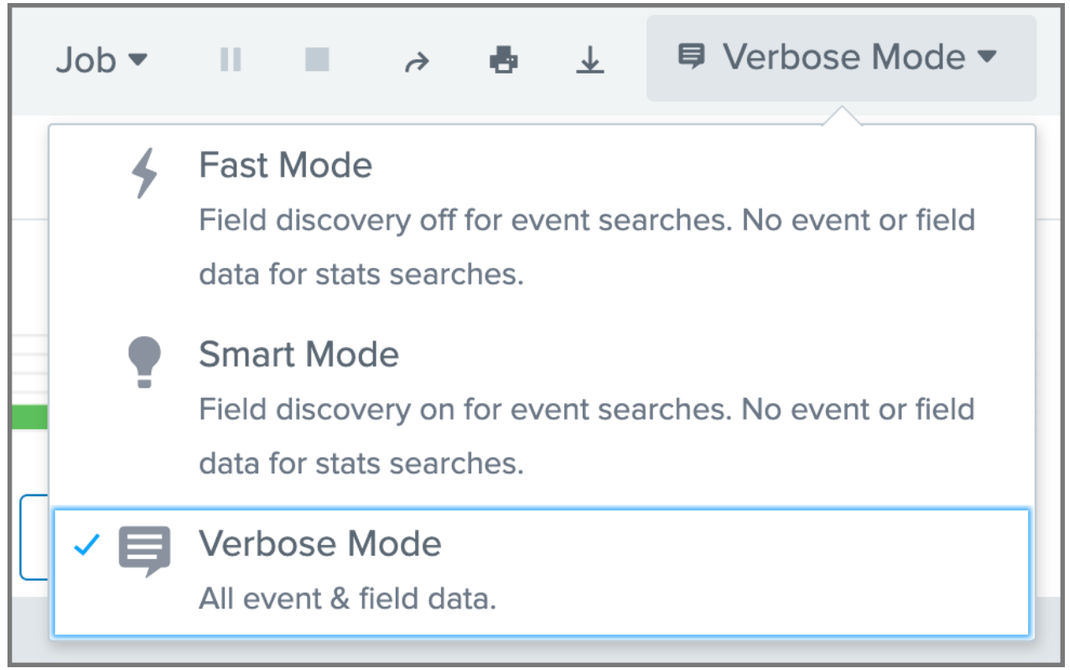



**Note**

A transforming search, also called a reporting search, is a search that uses a transforming command to return statistics tables and visualizations.

# Verbose Mode

- Prioritizes completeness over speed
- Returns all extracted fields
- Returns full event list and event timeline for every search
- Slowest search mode due to increased size of search payload

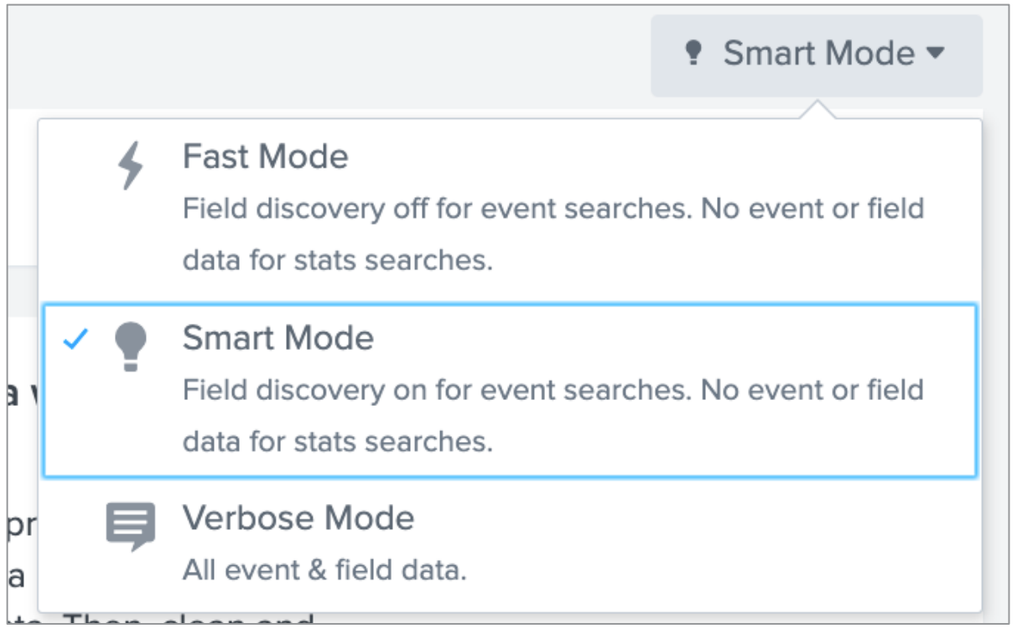



**Note**

A transforming search, also called a reporting search, is a search that uses a transforming command to return statistics tables and visualizations.

# Smart Mode

- Default search mode
- Balances speed and completeness
- If a transforming search is run, user is taken straight to report result table or visualization
  - No event list or timeline is generated
  - Behaves like **Fast Mode**
- If a non-transforming search is run:
  - Event list and timeline is generated
  - Behaves like **Verbose Mode**



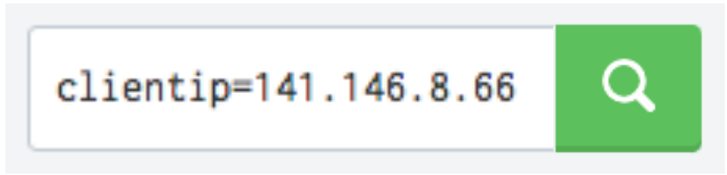
# Use Fields in Searches

# Topic Objectives

- Use fields correctly in basic searches
- Use fields with operators
- Use `rename` command
- Use `fields` command to improve search performance

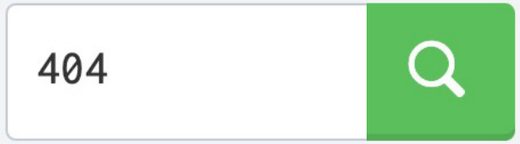
# Fields as Filters

- Fields provide an efficient way to filter events and refine search results when included in the basic search






- Field values can be searched without their field name but will likely return additional results




Returns all events containing "404" including events where `status=404` and `area_code=404`

**Note**

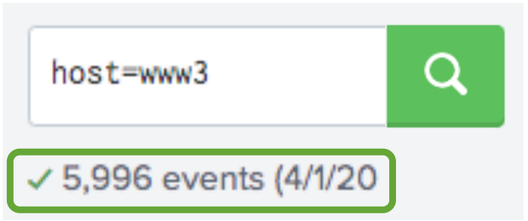
The basic search defines the search criteria for your search and appears as the first line in your search. It is a best practice to make your basic search as specific as possible.

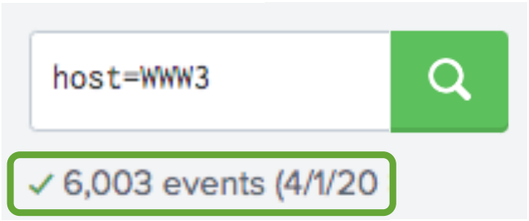
# Use Fields in Searches: Spaces

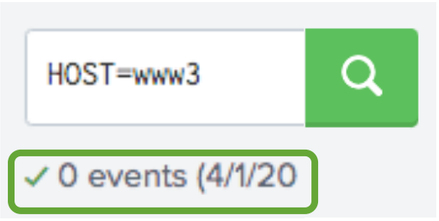
If a value contains a space, it must be enclosed in double quotes:

# Use Fields in Searches: Case

Field names are case sensitive; field values are not

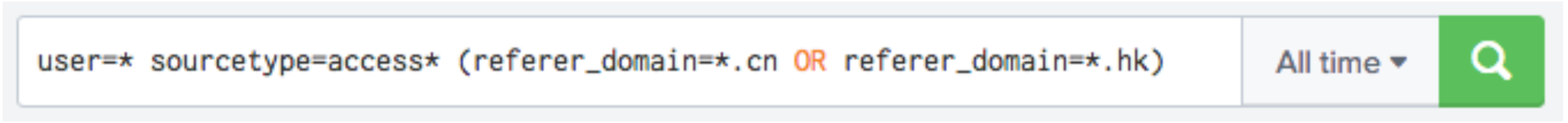


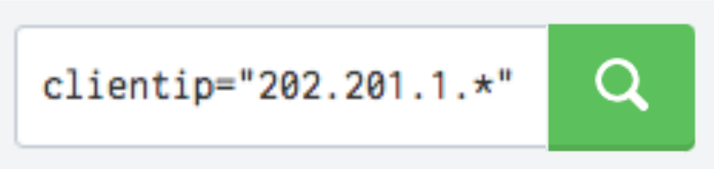
# Use Fields in Searches: Wildcards

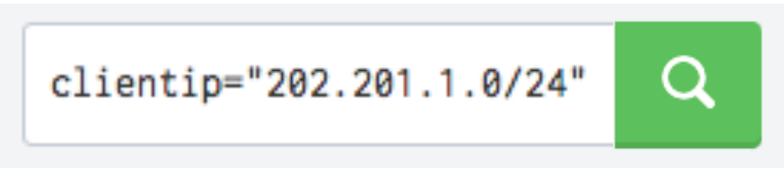
Use wildcards to match a range of field values



# Use Fields in Searches: IP Fields

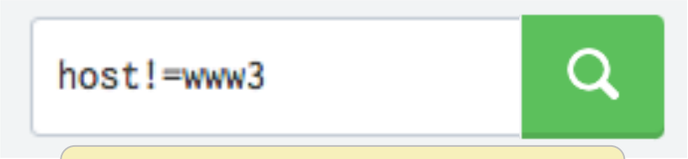
For IP fields, Splunk is subnet/CIDR aware



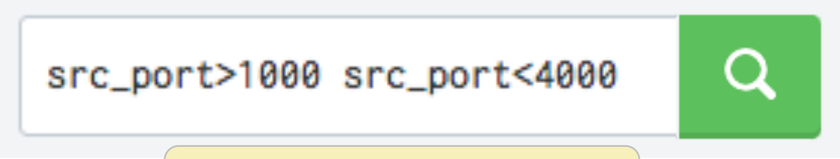



# Field Expressions

- Comparison operators can be used to match a specific value or a range of values
- Available operators: **=  !=  <  <=  >  >=**
- Use relevant operators based on field type




Example for alphanumeric fields



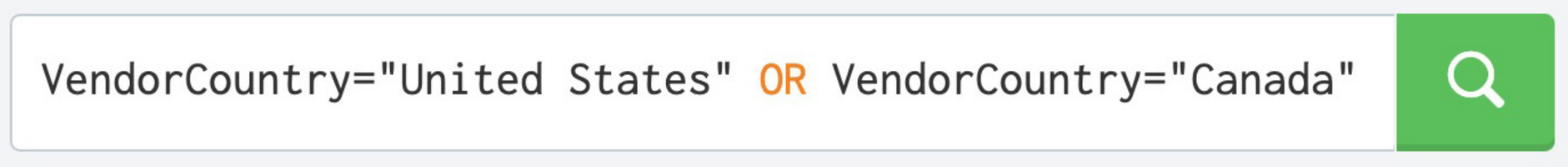

Example for numeric fields



 | 

# Operators: **OR** and **IN**

- Search for multiple values for a field by using the **OR** operator




- Alternatively, you can use the **IN** operator

# Operators: **AND**

Between search terms, **AND** is implied unless otherwise specified

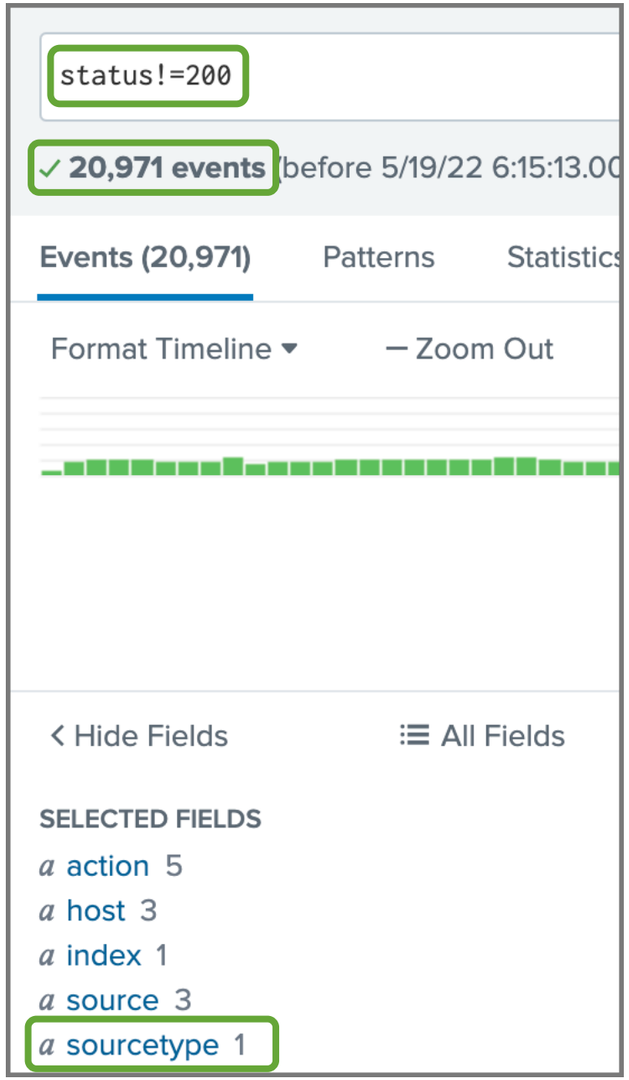

# Operators: != and NOT

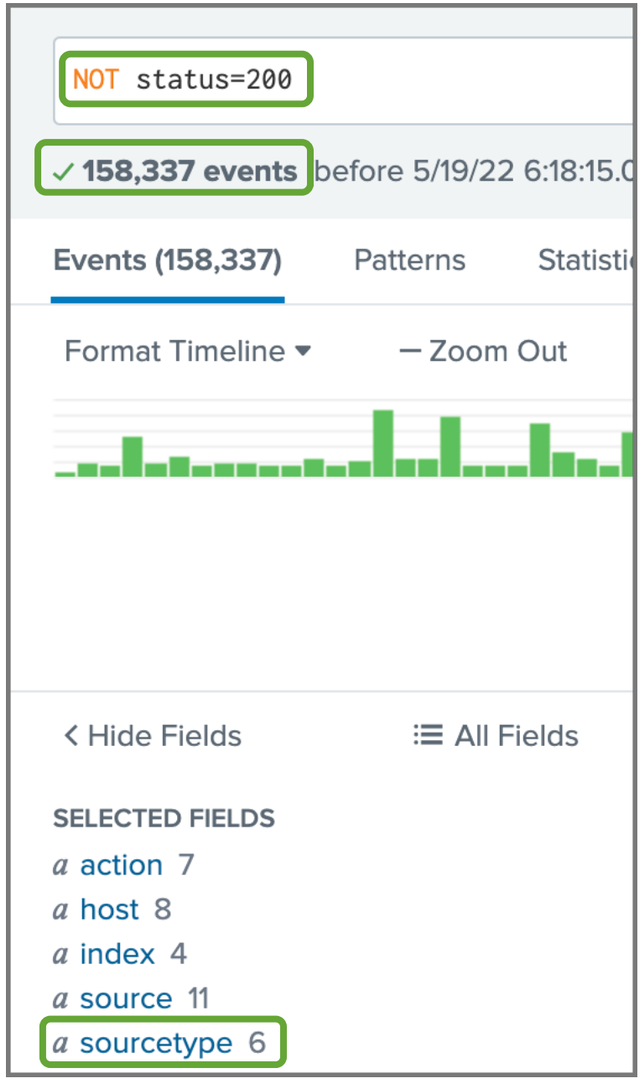
The != field expression and NOT operator exclude events from your search, but may produce different results


Returns events where the `status` field exists and value in field doesn't equal `200`


Returns events where the `status` field exists and value in field doesn't equal `200` and all events where `status` field doesn't exist

# Operators: != and NOT (cont.)



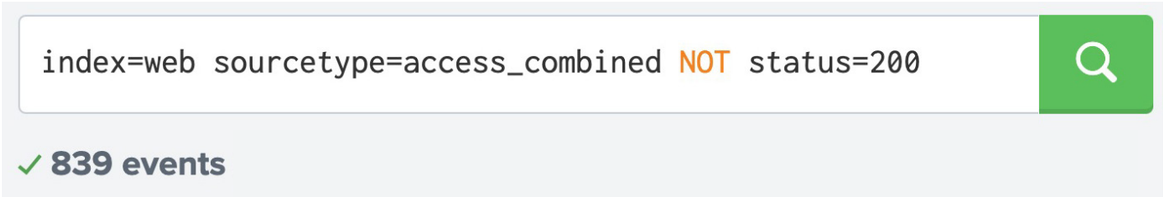




**Note**

The results from a search using != are a **subset** of the results from a similar search using NOT.

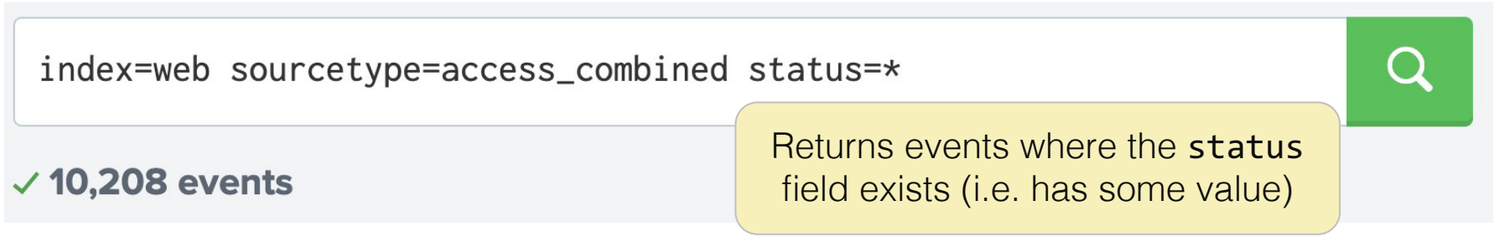
# Compare != and NOT

- If the field you're evaluating always exists in the data you're searching, then != and NOT yield the same results
- These searches yield the same results because the status field exists for each event in the access_combined sourcetype

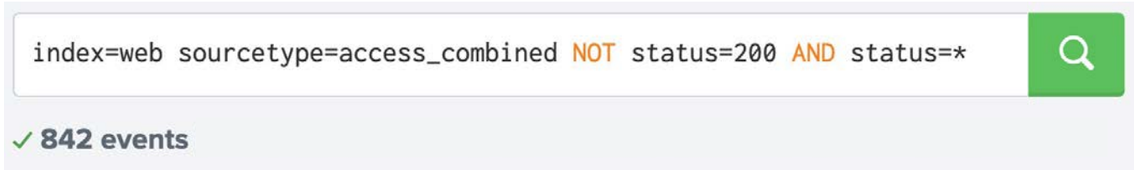
# Compare !=, NOT, and =*

Another way to return only events containing a specific field is to use = with the wildcard *

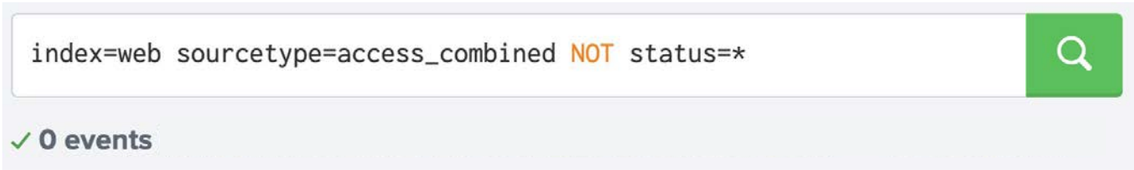


# Compare !=, NOT, and =* (cont.)

- Therefore, these 2 searches always yield same results


- Conversely, `NOT status=*` returns only events where status field doesn't exist:



# `rename` Command

```
...| rename <field> AS <newfield>
```

- Useful for giving fields more meaningful and user-friendly names
- When including spaces or special characters in field names, use double straight quotes, " ”
- Only exists for the lifetime of the search. No permanent change is being made.



# rename Command Example

**Scenario**

Display the **clientip**, **action**, **productId**, and **status** of customer interactions in the online store for the last 4 hours.

```
index=web sourcetype=access_combined
| table clientip, action, productId, status
| rename productId AS ProductID,
  action AS "Customer Action",
  status AS "HTTP Status"
```

| clientip | Customer Action | ProductID | HTTP Status |
|---|---|---|---|
| 12.130.60.4 | | | 200 |
| 27.102.11.11 | | SC-MG-G10 | 200 |
| 27.102.11.11 | | DC-SG-G02 | 200 |
| 27.102.11.11 | | | 200 |
| 27.102.11.11 | view | FI-AG-G08 | 200 |
| 99.61.68.230 | purchase | | 200 |
| 99.61.68.230 | purchase | WC-SH-A01 | 200 |
| 99.61.68.230 | addtocart | WC-SH-A01 | 200 |

# `rename` Command Example (cont.)

Once you rename a field, the new field name must be used in the rest of the search string

```
index=web sourcetype=access_combined
| table clientip, action, productId, status
| rename productId AS ProductID,
  action AS "Customer Action",
  status AS "HTTP Status"
| table action, status
```

No results found.

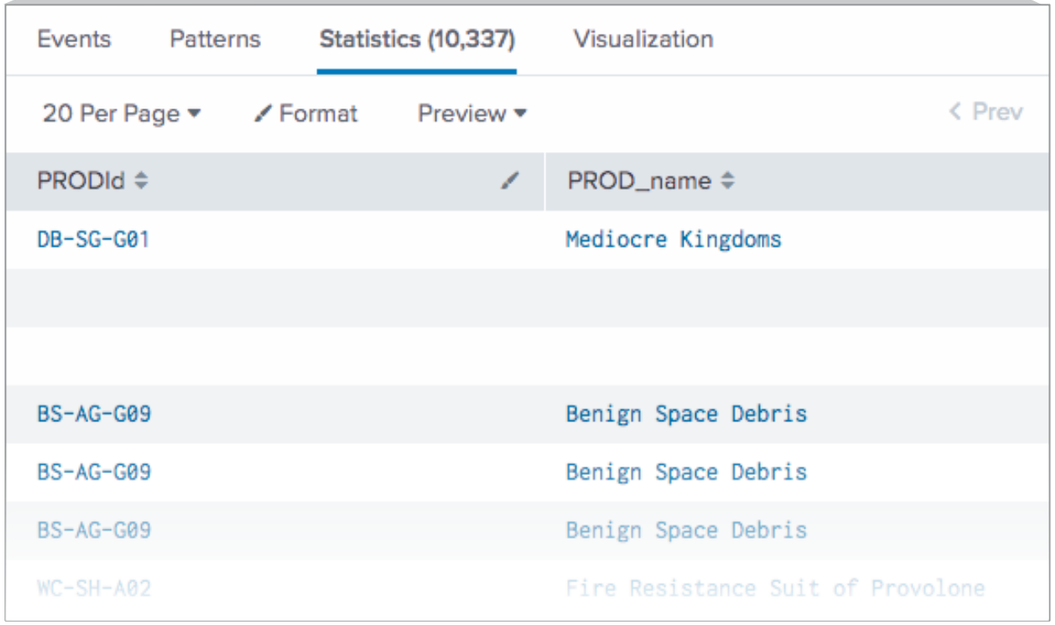
```
index=web sourcetype=access_combined
| table clientip, action, productId, status
| rename productId AS ProductID,
  action AS "Customer Action",
  status AS "HTTP Status"
| sort "Customer Action", "HTTP Status"
```

| clientip | Customer Action | ProductID | HTTP Status |
|---|---|---|---|
| 70.38.1.235 | addtocart | WC-SH-A02 | 200 |
| 203.45.206.135 | addtocart | DC-SG-G02 | 200 |
| 91.205.189.15 | addtocart | MB-AG-G07 | 200 |

# `rename` Command Example (cont.)

Use wildcards (*) to rename multiple fields that match a pattern

```
index=web sourcetype=access_combined
| table productId, product_name
| rename product* AS PROD*
| table PROD*
```



# `fields` Command

```
...| fields [+|-] [<field>|<wc-field-list>]
```

- Include or exclude specified fields in search results
- **`fields`** or **`fields+`** (default) includes **`<wc-field-list>`**
- **`fields-`** excludes specified fields
- Supports wildcarded field lists

**Note**

The **`wc`** in **`<wc-field-list>`** means that fields provided to the **`fields`** command can have wildcards.

# `fields` Command: Search Benefits

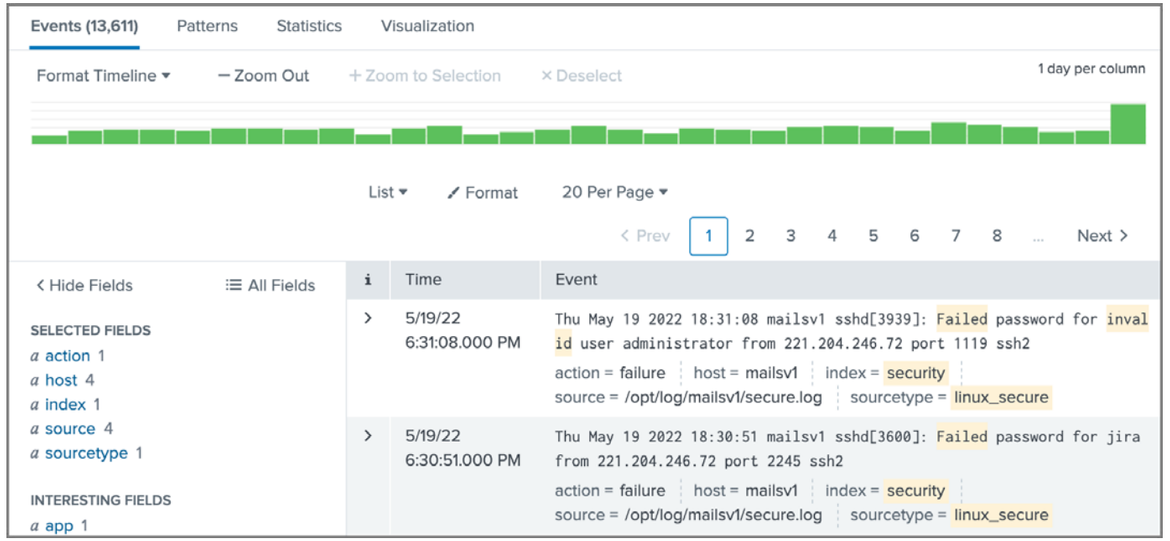
- Field extraction is one of the costliest parts of a search
- Use **`fields`** or **`fields+`** right after the basic search for improved search performance
- Use **`fields-`** at the end of the search to remove fields from view on Statistics and Visualization tabs

# `fields` Command Example

**Scenario**

Display network failures during the previous 30 days.

```
index=security sourcetype=linux_secure (fail* OR invalid)
```

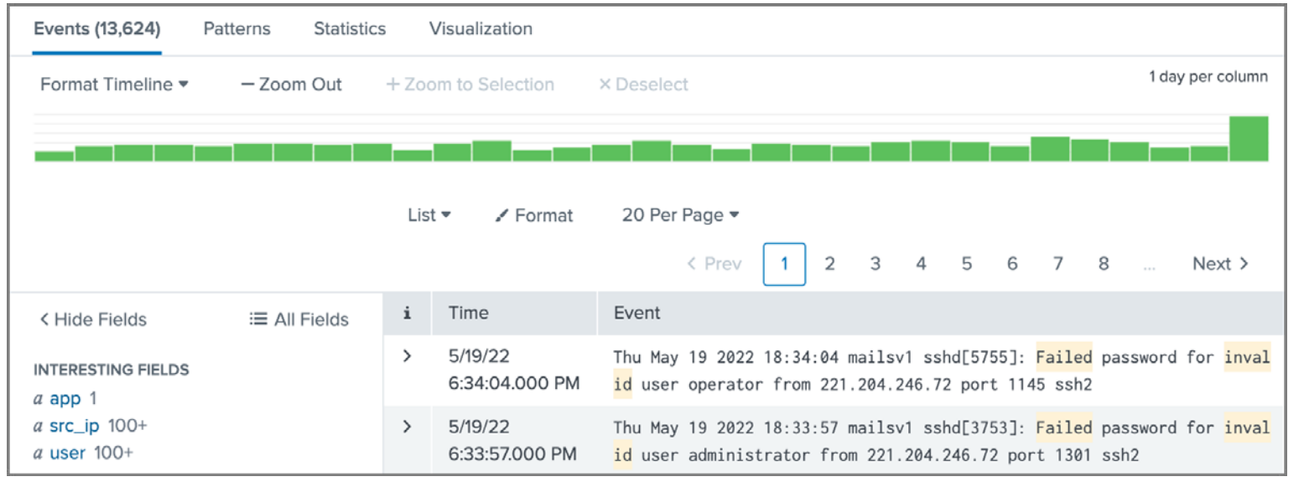



Returned **13,611**results by scanning **13,638** events in **1.22** seconds:

**Scenario**

Display network failures during the previous 30 days. Retrieve only **`user`**, **`app`**, and **`src_ip`**.

```
index=security sourcetype=linux_secure (fail* OR invalid)
| fields user, app, src_ip
```




Returned **13,624** results by scanning **13,638** events in **0.809** seconds:

# Use Fields in Searches Lab Exercise

**Time**: 15 minutes

**Tasks**:

- Use the Fields sidebar to examine search results
- Use keywords, field expressions, and the **`fields`** command to filter for specific events
- Complete the missing portion of a search with the **`rename`** command

# Compare Temporary versus Persistent Fields

# Topic Objectives

- Differentiate between temporary and persistent fields
- Create temporary fields with **`eval`**
- Extract temporary fields with **`erex`** and **`rex`**

# Temporary Fields versus Persistent Fields

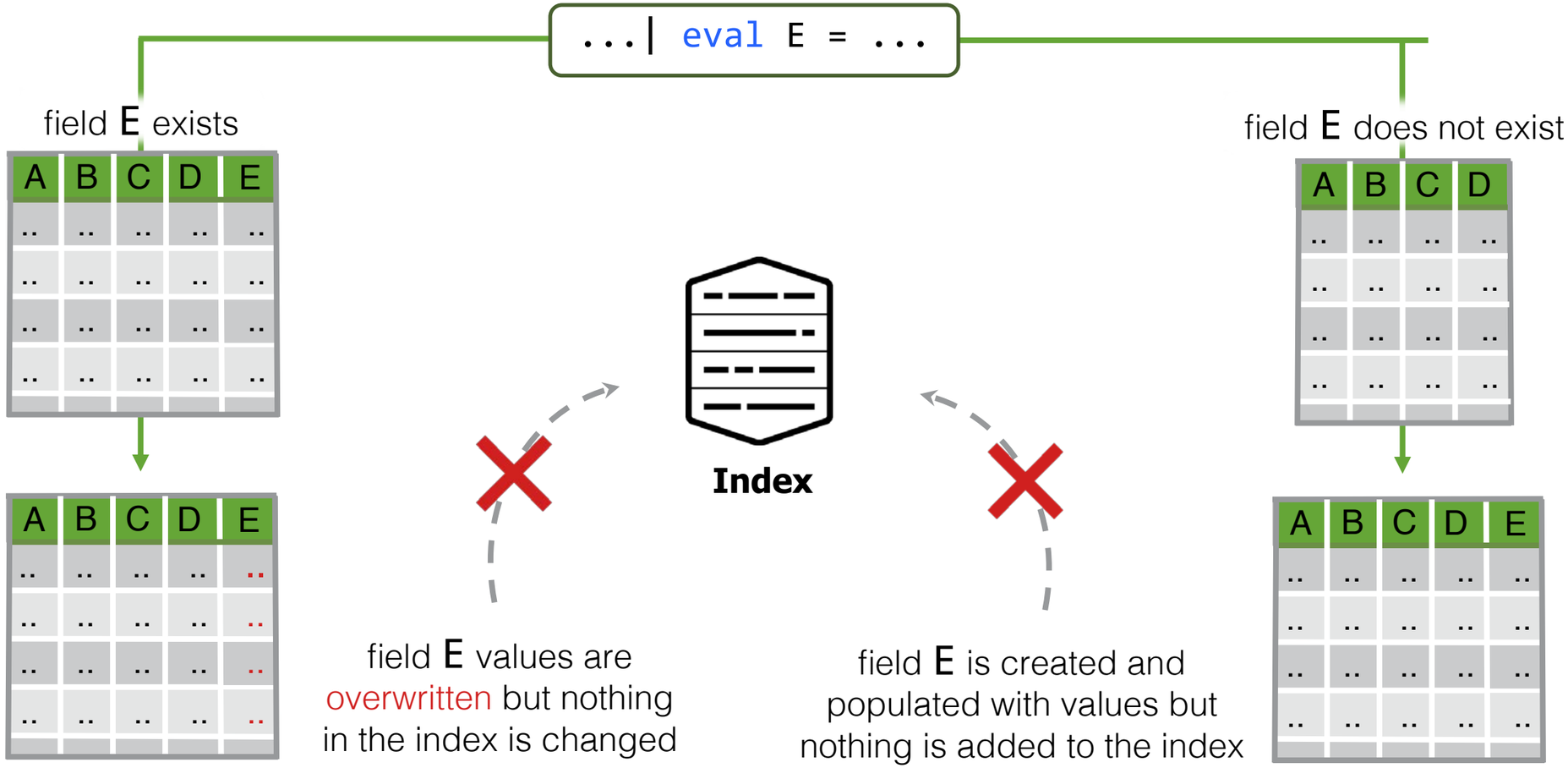
- Temporary fields:
  - Exist for the duration of the search
  - Can be created from search results with the **`eval`** command
  - Can be extracted from search results with the **`erex`** or **`rex`** command
- Persistent fields:
  - Are knowledge objects that can be shared
  - Are created by users, knowledge managers, and admins
  - Outside the scope of this course

# `eval` Command

```
...| eval <field1>=<expression1>[, <field2>=<expression2>]
```

- Calculates an expression and puts the resulting value into a new or existing field which can be reused in the search pipeline
- Extremely powerful and useful command that supports a vast assortment of functions
- Can exist as an expression

# `eval` Command (cont.)



# `eval` Command (cont.)

The `eval` command supports various operators

| Type | Operators |
|---|---|
| arithmetic | `+ - * / %` |
| concatenation | `+ .` |
| boolean | `AND OR NOT XOR` |
| comparison | `< > <= >= != = LIKE` |



 | 

# eval Command (cont.)

```
index=sales sourcetype=vendor_sales VendorCountry IN("United States", Canada)
| stats sum(price) as "USA+Canada Sales" count as "Total Products Sold"
  count(eval(VendorCountry = "United States")) as "Products Sold in US",
  count(eval(VendorCountry = "Canada")) as "Products Sold in Canada" by product_name
| eval "USA+Canada Sales" = "$".'USA+Canada Sales'
```

- Field values are treated in a case sensitive manner
- String values must be "double-quoted"
- Field names must be unquoted or single quoted when they include a special character like a space
- Use a period (.) instead of (+) when concatenating strings and numbers to avoid conflicts

# Ways to Write Multiple **evals**

Expressions can be separate, nested, or linked with a comma

Separate **eval** pipeline segments

```
index=network sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as bytes by usage
| eval bandwidth = bytes/(1024*1024)
| eval bandwidth = round(bandwidth, 2)
```

Nested **eval** commands targeting the same field

```
index=network sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as bytes by usage
| eval bandwidth = round(bytes/(1024*1024), 2)
```

Combining **eval** commands with commas

```
index=network sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as bytes by usage
| eval bandwidth = bytes/(1024*1024),
  bandwidth = round(bandwidth, 2)
```

| usage | bytes | bandwidth |
|---|---|---|
| Borderline | 1298542 | 1.24 |
| Business | 2909449 | 2.77 |
| Personal | 9771346 | 9.32 |
| Unknown | 997092 | 0.95 |
| Violation | 495606 | 0.47 |

# Reference `eval` Fields

Temporary fields created using eval can be referenced throughout the search pipeline

```
index=network sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) AS Bytes by usage
| eval bandwidth = round(Bytes/pow(1024,2), 2)
| sort -bandwidth
| rename bandwidth AS "Bandwidth (MB)"
```

| usage | Bytes | Bandwidth (MB) |
|---|---|---|
| Personal | 299584913 | 285.71 |
| Unknown | 77187989 | 73.61 |
| Business | 66844576 | 63.75 |
| Borderline | 54011022 | 51.51 |
| Violation | 3203231 | 3.05 |

# Temporary Field Extraction

- Temporary field extraction is also known as "extracting fields at search time"
- Extraction only exists for duration of search, doesn't persist as knowledge object
- Good for rarely used fields
- Splunk offers 2 search time extraction commands
  - **`erex`**: don’t have to know regex, just provide example values
  - **`rex`**: must write regex, finds data that matches pattern

# `erex` Command

```
...| erex <field> examples="<example1>, <example2> [,...]"
```

- Extracts a field based on **`<example>`** values you provide
- The examples used must be in the returned results
- **`<field>`** is the name of the new field created for this search

# erex Command Example

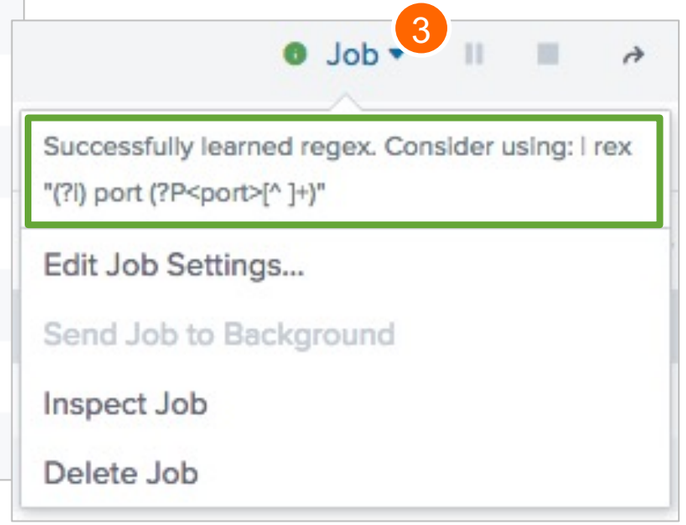
**Scenario**

Sec Ops wants to display the IP address and port of potential attackers. The field port does not currently exist and would need to be created.

```
index=security sourcetype=linux_secure port "failed password"
| erex port examples="3572,2471"
| table src_ip, port
```

1. Creates a temporary new field, **port**
2. Extracts values using examples provided (**3572** and **2471**)
3. To view the regex generated by your search, click the **Job** drop-down menu

| src_ip | port |
|---|---|
| 27.127.239.122 | 3572 |
| 27.127.239.122 | 3572 |
| 27.127.239.122 | 3572 |
| 27.127.239.122 | 3572 |
| 27.127.239.122 | 3572 |
| 67.133.102.54 | 1174 |
| 228.18.132.65 | 2471 |
| 27.175.11.11 | 4898 |



# `rex` Command

```
...| rex [field=<field>] "<regex-expression>"
```

- Matches the value of the field against unanchored ***regex***
- **`<field>`** is any available field you want to extract information from; defaults to **`field=_raw`**
- The **`<regex-expression>`** must include at least one named capture group (which will become the field name)
- You can use **`erex`** to generate an initial regex, then edit it to your specifications for use with the **`rex`** command
- Can perform multiple extractions

# rex Command Example 1

**Scenario**

Display the usernames of potential email attackers.

```
index=network sourcetype=cisco_esa mailfrom=*
1 | rex  "\<(?<potentialAttacker>.+)@"
| table potentialAttacker
```

| potentialAttacker |
|---|
| surveyspot |
| snoopy |
| slickdeals |
| slickdeals |

- The Cisco router server contains the email addresses of those sending email to the company
1. Use **rex** to extract just local-part of email address at search time

| i | Time | Event |
|---|---|---|
| > | 5/2/22 5:54:07.000 PM | Mon May 02 17:54:07 2022 Info: MID 244956 ICID 743920 From: <slickdeals@slickdeals.net><br>host = cisco_router1 potentialAttacker = slickdeals source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |
| > | 5/2/22 2:51:11.000 PM | Mon May 02 14:51:11 2022 Info: MID 244955 ICID 743920 From: <beth.gregory@gmail.com><br>host = cisco_router1 potentialAttacker = beth.gregory source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |
| > | 5/1/22 3:20:24.000 PM | Sun May 01 15:20:24 2022 Info: MID 244953 ICID 743918 From: <ovandenende@dynamac.com><br>host = cisco_router1 potentialAttacker = ovandenende source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |

# rex Command Example 2

**Scenario**

Display the usernames and mail domains from which employees are receiving email.

```
index=network sourcetype=cisco_esa mailfrom=*
| rex  field=mailfrom "(?<potentialAttacker>.+)@(?<domain>.+)"
| table mailfrom, potentialAttacker, domain
```

| mailfrom | potentialAttacker | domain |
|---|---|---|
| surveyspot@surveyspot.com | surveyspot | surveyspot.com |
| snoopy@demo.com | snoopy | demo.com |
| slickdeals@slickdeals.net | slickdeals | slickdeals.net |
| slickdeals@slickdeals.net | slickdeals | slickdeals.net |

The field **mailfrom** is being used to create two new fields, **potentialAttacker** and **domain**

| i | Time | Event |
|---|---|---|
| > | 5/2/22 5:54:07.000 PM | Mon May 02 17:54:07 2022 Info: MID 244956 ICID 743920 From: <slickdeals@slickdeals.net><br>host = cisco_router1 potentialAttacker = slickdeals source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |
| > | 5/2/22 2:51:11.000 PM | Mon May 02 14:51:11 2022 Info: MID 244955 ICID 743920 From: <beth.gregory@gmail.com><br>host = cisco_router1 potentialAttacker = beth.gregory source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |
| > | 5/1/22 3:20:24.000 PM | Sun May 01 15:20:24 2022 Info: MID 244953 ICID 743918 From: <ovandenende@dynamac.com><br>host = cisco_router1 potentialAttacker = ovandenende source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |
| > | 4/30/22 8:43:24.000 PM | Sat Apr 30 20:43:24 2022 Info: MID 244952 ICID 743916 From: <arlenel98@yahoo.com><br>host = cisco_router1 potentialAttacker = arlenel98 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa |

# `rex` Command: `sed` Mode

```
...| rex [field=<field>] mode=sed "<sed-expression>"
```

- Search and replace within a field using a sed (Unix stream editor) expression
- Use the `s` flag to replace strings or the `y` flag to substitute characters
- Example sed expressions:
  - `"s/(\d{4}-){3}/XXXX-XXXX-XXXX-/g"` matches the regex to a series of numbers and replaces them with an anonymized string
  - `"y/string1/string2/"` substitutes characters in “string2” for “string1”; the strings must have the same amount of characters

# rex Command: sed Mode Example

**Scenario**

Customer Success would like to view the number of transactions by account code in the last 4 hours. However, the account code must be masked for the shared dashboard.

```
"s/(\w{4}-)\S+/\1xxxx/g"
```

The backslash / splits the sed expression into four parts:

- The option (s for replace or y for substitute)
- The regex
- The replacement
- The flag (g stands for global so all matching occurrences are replaced)

```
index=sales sourcetype=sales_entries AcctCode
| stats count, values(TransactionID) as TransactionID by AcctCode
| rex field=AcctCode mode=sed "s/(\w{4}-)\S+/\1xxxx/g"
```

| AcctCode | count | TransactionID |
|---|---|---|
| 0012-xxxx | 1 | 213953 |
| 0182-xxxx | 3 | 213782<br>213863<br>213940 |
| 0322-xxxx | 3 | 213750<br>213869<br>213894 |
| 0508-xxxx | 1 | 213831 |

# erex versus rex

```
index=security
sourcetype=linux_secure port
“failed password”
| erex port examples=“4977,1577”
| table src_ip, port
```

## erex

- Easier - regex knowledge is not needed
- Generates regex expression
- Must provide examples from current data
- Should **not** be used in saved reports

**Scenario**

Display IP addresses and ports of potential attackers.

| src_ip | port |
|---|---|
| 192.162.19.179 | 2116 |
| 192.162.19.179 | 2257 |
| 192.162.19.179 | 2463 |
| 192.162.19.179 | 1686 |
| 192.162.19.179 | 4977 |
| 192.162.19.179 | 4288 |

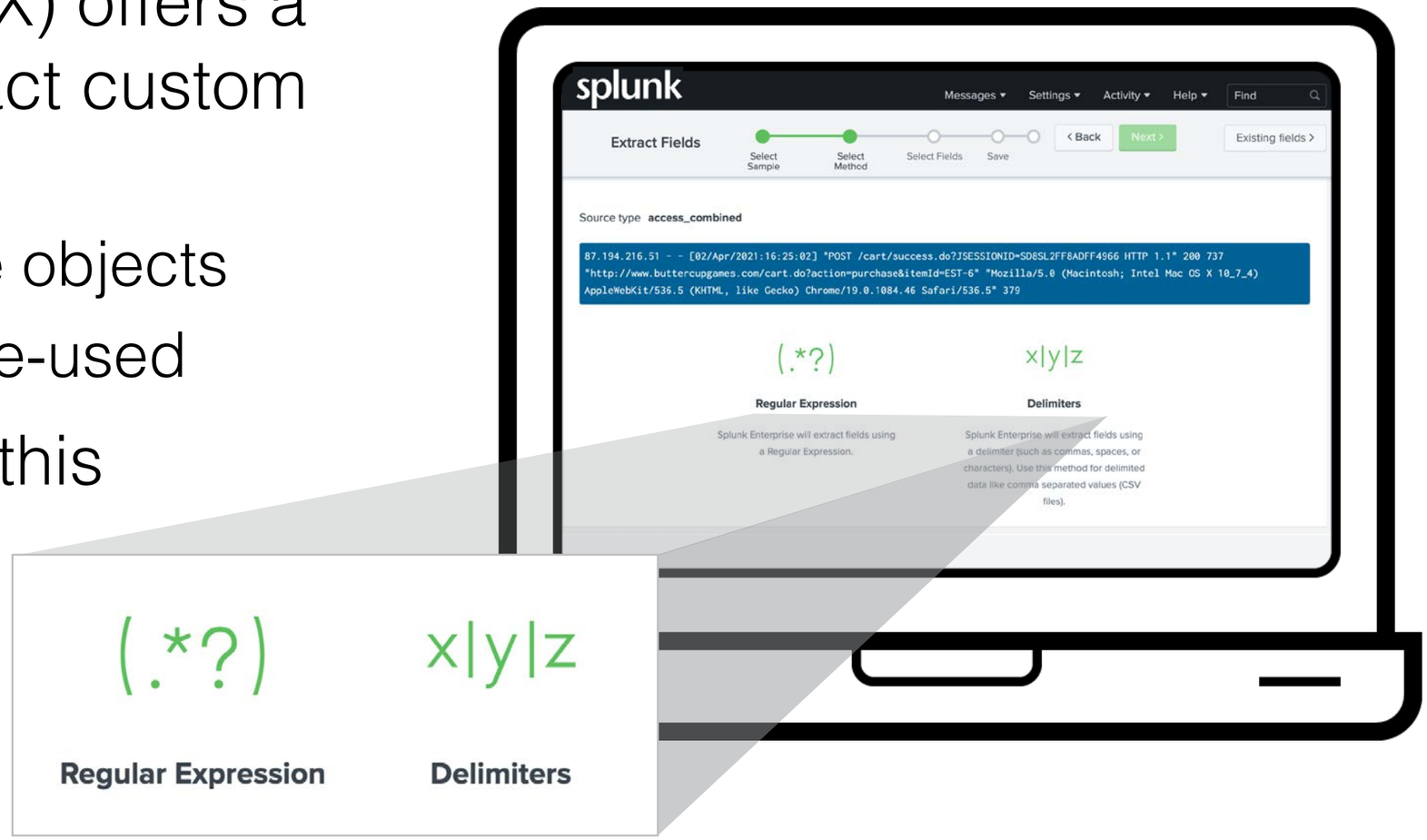
```
index=security
sourcetype=linux_secure port
"failed password"
| rex "port\s(?<port>\d+)"
| table src_ip, port
```

## rex

- Must know regex (difficult)
- Don’t have to provide examples
- Can use regex generated by erex and customize as needed
- Can be used in saved reports

# The Field Extractor

- The Field Extractor (FX) offers a user interface to extract custom fields that:
  - Persist as knowledge objects
  - Can be shared and re-used
- Outside the scope of this course



# Temporary versus Persistent Fields Lab Exercise

**Time**: 15 minutes

**Tasks**:

- Use the **`erex`** command to extract temporary fields and include events based on pattern matching
- Use the **`rex`** command to improve your search results from the previous task



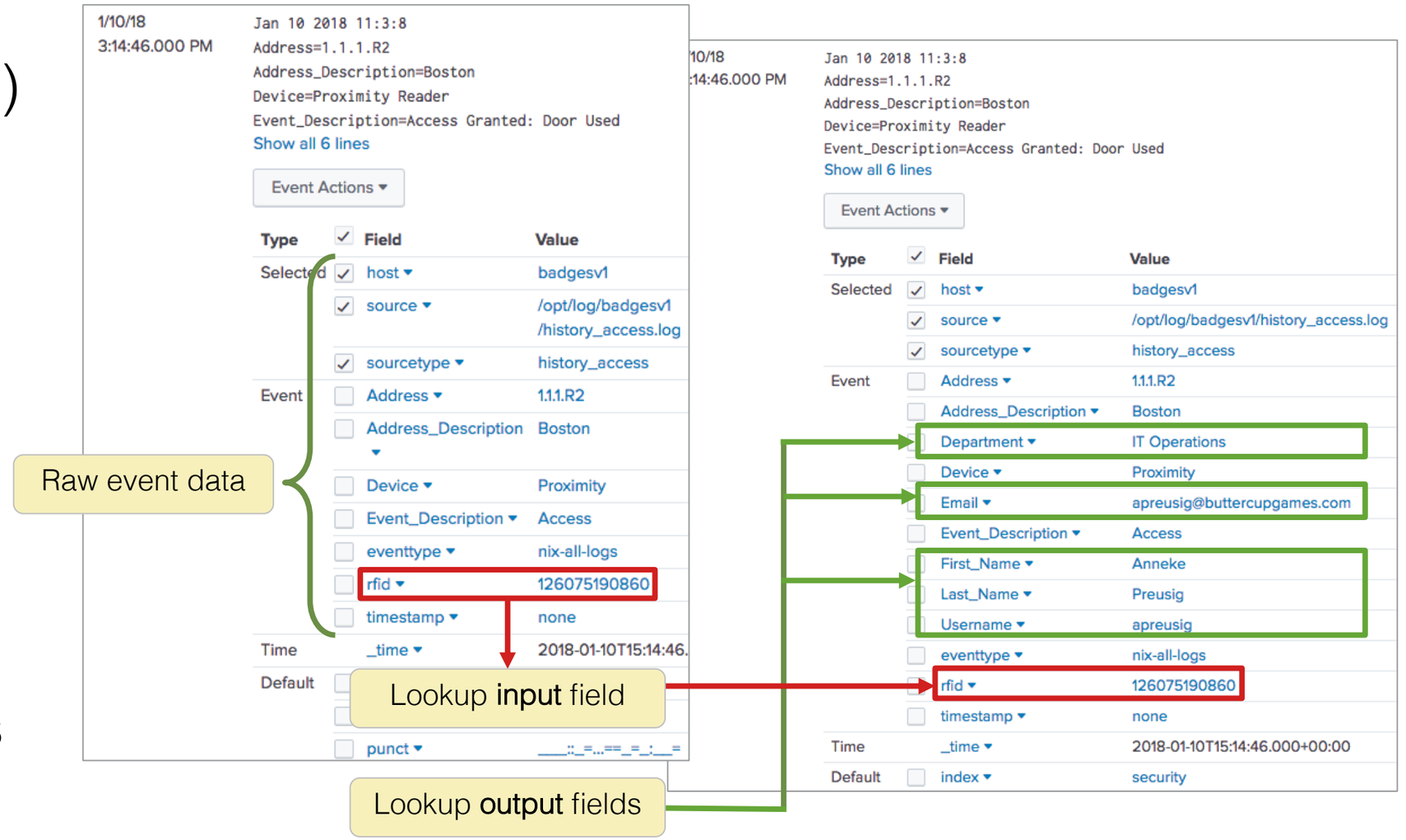
# Enrich Data

# Topic Objectives

- Introduction to how fields from lookups, calculated fields, field aliases, and field extractions enrich data
- Courses dedicated to creating and using lookups, calculated fields, field aliases, and field extractions are available

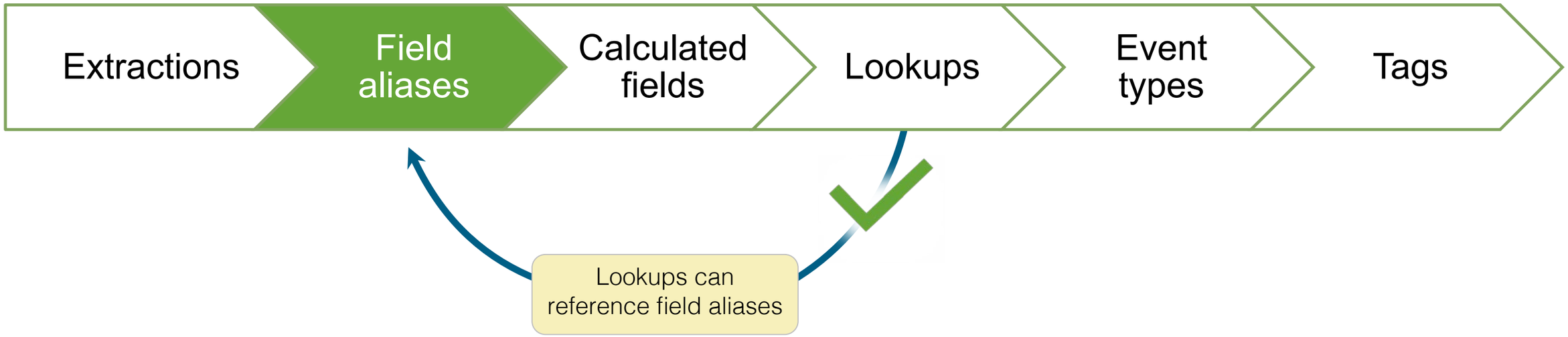
# Lookups at Search Time

- Sometimes static (or relatively unchanging) data is required for searches, but isn't available in the raw event data
- Lookups pull such data from standalone files at search time and add it to search results as field values



# Field Aliases

- A way to associate an additional (new) name with an existing field name, like a nickname
- Field aliases can be used to normalize field names
- Can be referenced by knowledge objects that succeed aliases in the search process pipeline



# Field Alias Example

- Several source types contain some type of a username field
- To make data correlation and searching easier, normalize the Username field with a field alias

`sourcetype=cisco_firewall`

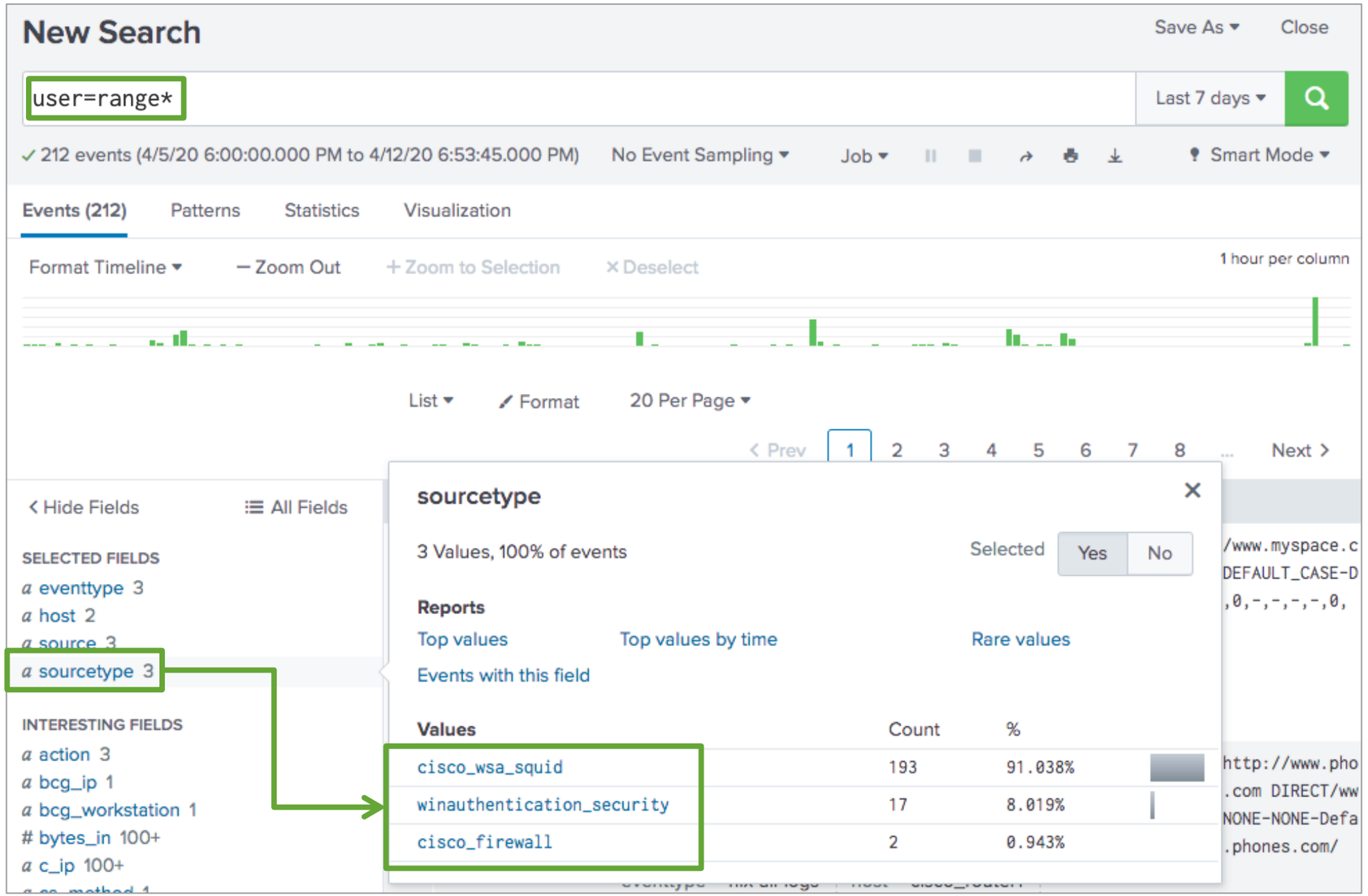
| Type | Field | Value | Actions |
|---|---|---|---|
| Selected | host | cisco_router1 | |
| | index | network | |
| | source | /opt/log/cisco_router1/cisco_fire all.log | |
| | sourcetype | cisco_firewall | |
| Event | Duration | 0h:0m:0s | |
| | Group | buttercupgames | |
| | IP | 10.1.10.15 | |
| | Username | arangel | |
| | bcg_ip | 10.1.10.15 | |

To search for all events involving the user `dhale`, you would have to search for: `Username=arangel OR User=arangel`

`sourcetype=winauthentication_security`

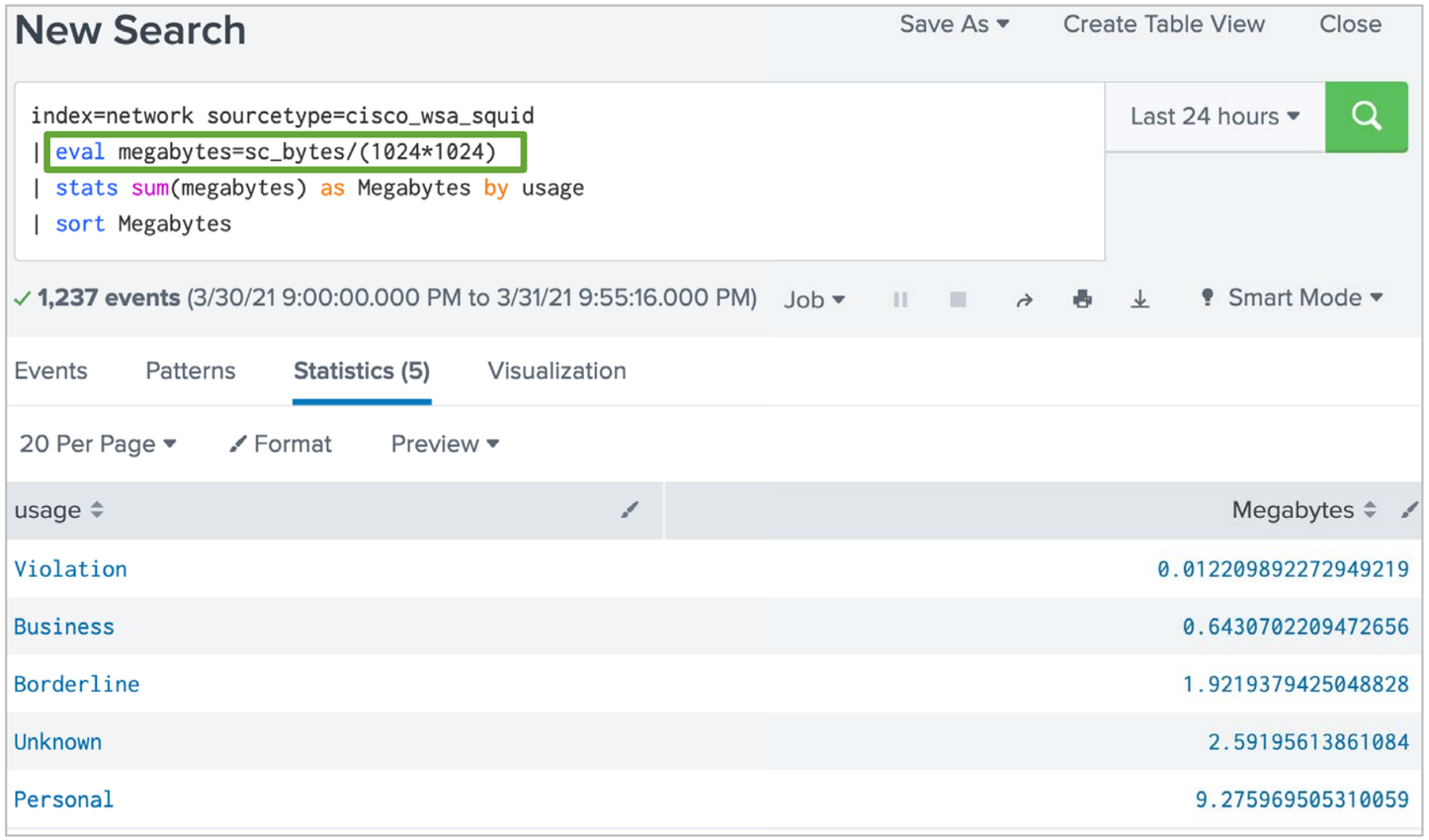
| | |
|---|---|
| EventType | 8 |
| LogName | Security |
| ge | Successful |
| dNumber | 9175 |
| | S-1-5-21-57989841-920026266-725345543-6444 |
| SidType | 1 |
| SourceName | Security |
| Type | Success |
| User | arangel |

# Using a Field Alias



# Calculated Fields

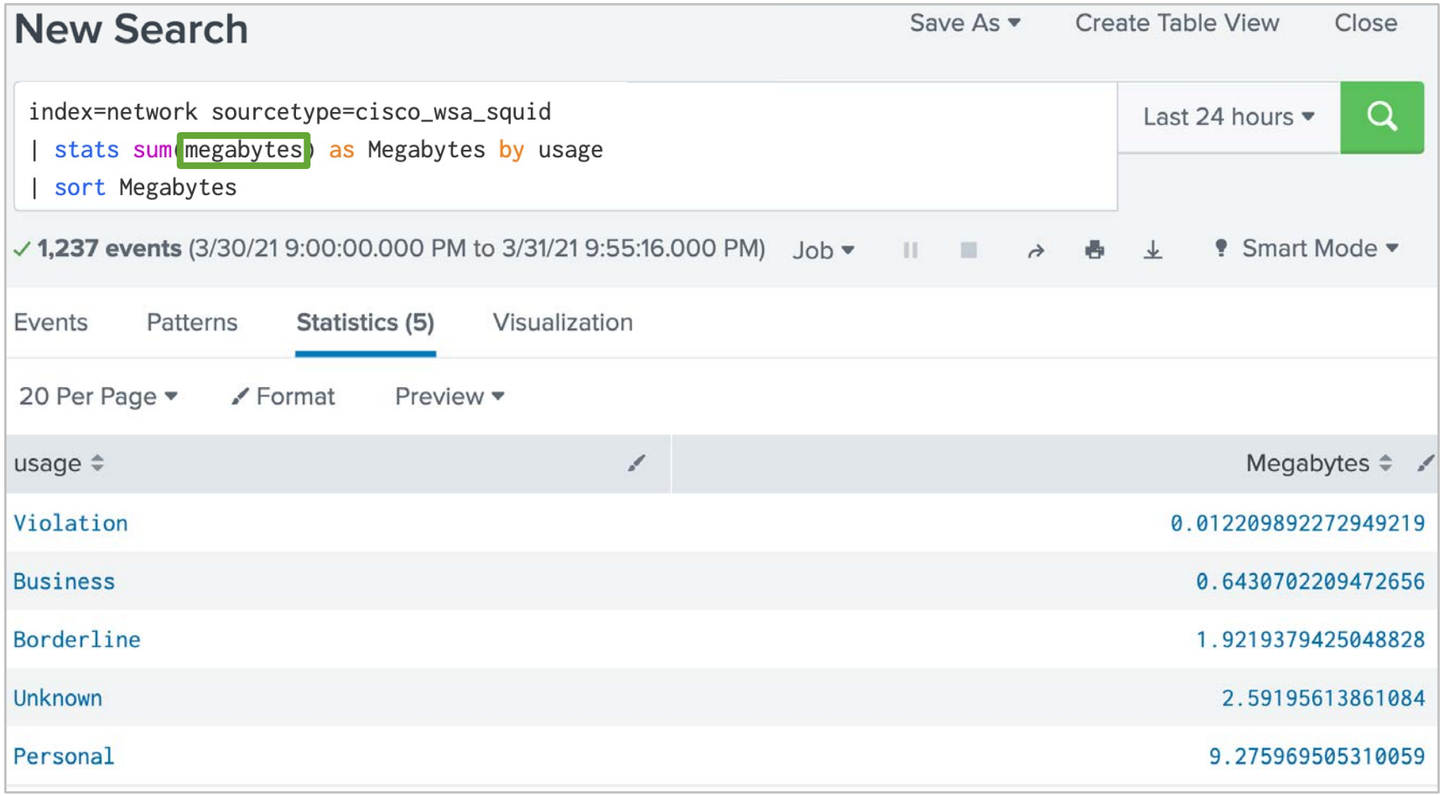
- Shortcut for performing repetitive, long, or complex transformations using the **`eval`** command
- Must be based on an extracted field



# Use a Calculated Field

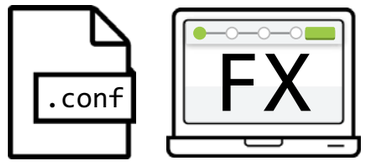
After you have created a calculated field, you can use it in a search like any extracted field

No manual eval is necessary



# Extracted Fields

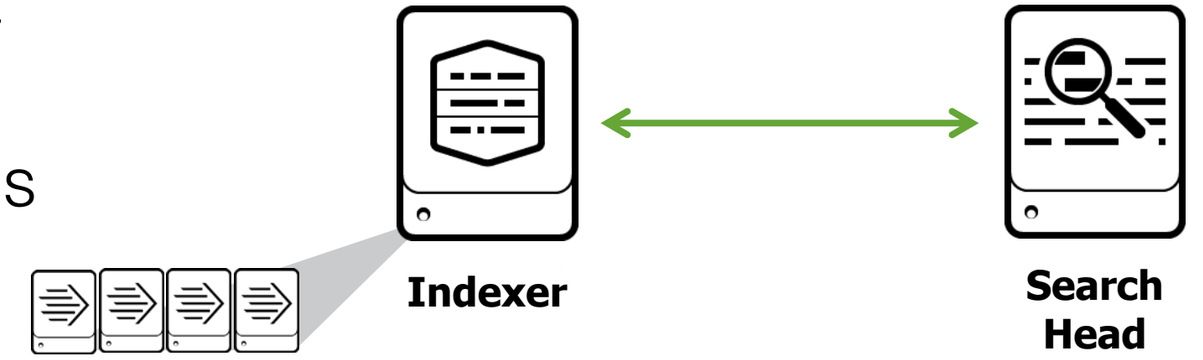
- Fields directly from raw data that provide specific information about events
- Fields can be extracted before or after event indexing
  - Before: indexed fields (default and custom fields)
  - After: search and custom fields




**Custom field extractions** can be defined and retrieved at index-time or search-time using configuration files or the Field Extractor (FX)

**Default fields** are extracted at index-time before event indexing

**Search fields** are extracted at search-time




Note

These elements are all components of the Splunk data pipeline.

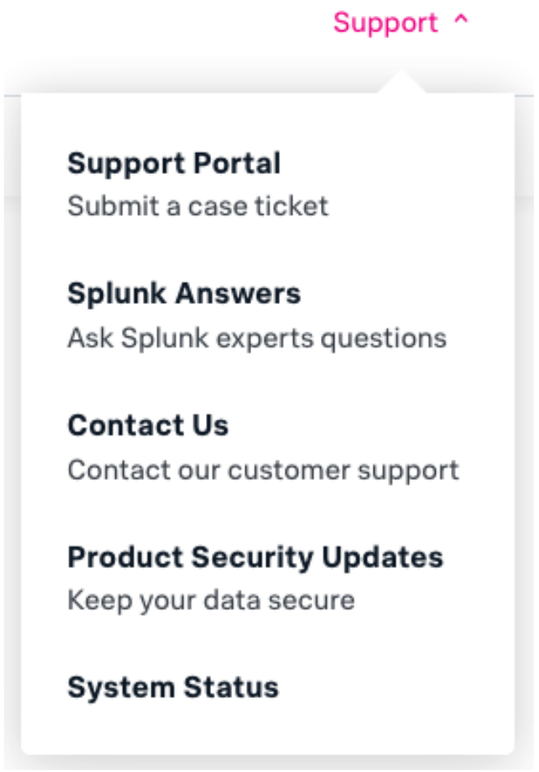
# Wrap-up Slides

# Community

- Splunk Community Portal
  community.splunk.com
  - Answers
  - Discussions
  - Splunk Trust
  - User Groups
  - Ideas
- Splunk Blogs
  splunk.com/blog/
- Splunk Apps
  splunkbase.com
- Splunk Dev Google Group
  groups.google.com/forum/#!forum/splunkdev
- Splunk Docs on Twitter
  twitter.com/splunkdocs
- Splunk Dev on Twitter
  twitter.com/splunkdev
- Splunk Live!
  splunklive.splunk.com
- .conf
  conf.splunk.com

# Support Programs

- **Web**
  - Documentation: dev.splunk.com and docs.splunk.com
  - Wiki: wiki.splunk.com
- **Splunk Lantern**
  Guidance from Splunk experts
  - lantern.splunk.com
- **Global Support**
  Support for critical issues, a dedicated resource to manage your account – 24 x 7 x 365
  - Web: splunk.com/index.php/submit_issue
- **Enterprise, Cloud, ITSI, Security Support**
  - Web: splunk.com/en_us/about-splunk/contact-us.html#tabs/customersupport
  - Phone: (855) SPLUNK-S or (855) 775-8657



# Learning Paths (cont.)

## Knowledge Manager - Recommended Courses

Free eLearning courses are in blue and courses with an * are present in both learning paths.

- What is Splunk *
- Introduction to Splunk *
- Using Fields *
- Introduction to Knowledge Objects
- Creating Knowledge Objects
- Creating Field Extractions
- Enriching Data with Lookups
- Data Models
- Introduction to Dashboards
- Dynamic Dashboards
- Using Choropleth
- Search Optimization *

# Learning Paths

## Search Expert - Recommended Courses

Free eLearning courses are in blue and courses with an * are present in both learning paths.

- What is Splunk *
- Introduction to Splunk *
- Using Fields *
- Scheduling Reports and Alerts
- Visualizations
- Statistical Processing
- Working with Time
- Comparing Values
- Result Modification
- Leveraging Lookups and Subsearches
- Correlation Analysis
- Search Under the Hood
- Multivalue Fields
- Search Optimization *

# Splunk Certification

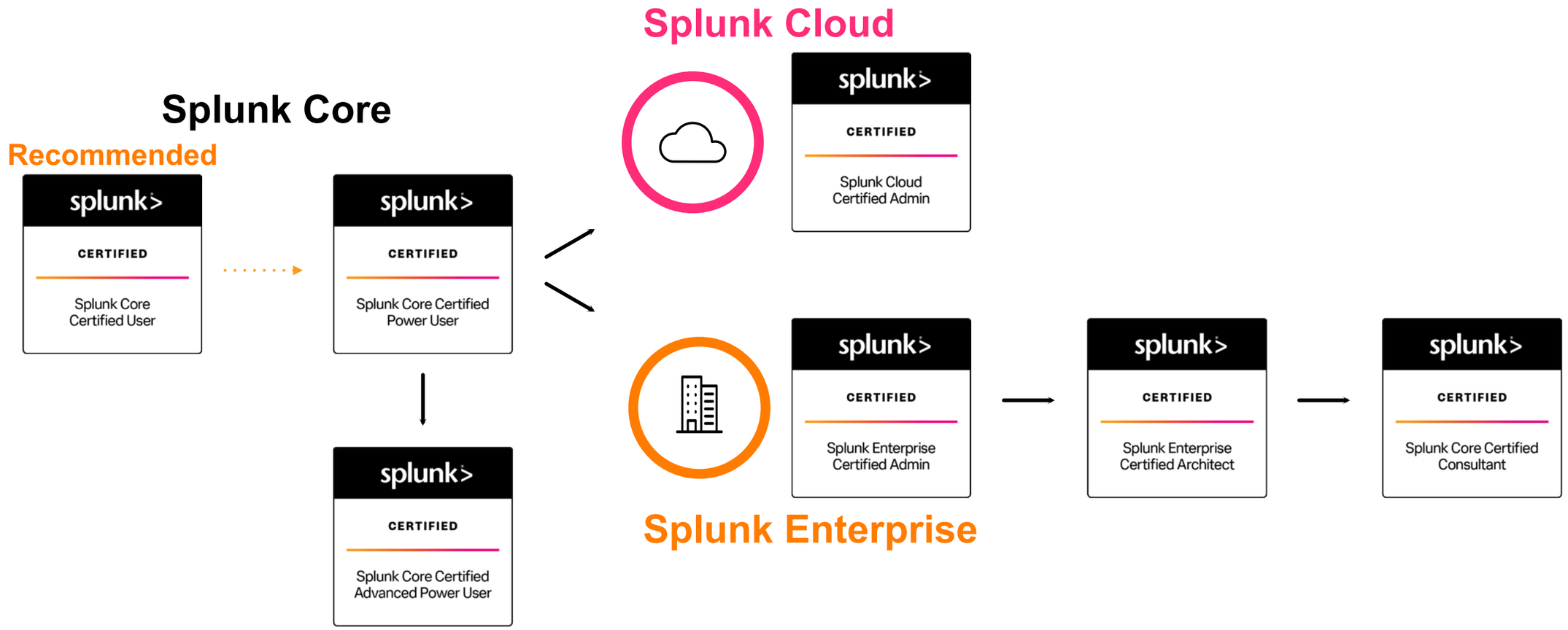
## Offerings & Requirements



# Splunk Core and Beyond

Regardless of which Splunk product you use, it all starts with Splunk Core



# Splunk Core Certified User

## This entry-level certification demonstrates an individual's basic ability to navigate and use Splunk software

**Prerequisite Certification(s):**

- None

**Prerequisite Course(s):**

- None

**Splunk Core Certified User Exam**

Time to study! We suggest candidates looking to prepare for this exam complete Fundamentals 1 **or** the following courses:

- What is Splunk?
- Intro to Splunk
- Using Fields
- Scheduling Reports and Alerts
- Visualizations
- Statistical Processing
- Working with Time
- Leveraging Lookups and Subsearches
- Search Optimization
- Enriching Data with Lookups
- Data Models

See here for registration assistance.

**Congratulations! You are a...**


**Recommended Next Step**

- Splunk Core Certified Power User

# Splunk Core Certified Advanced Power User

This certification demonstrates an individual's ability to generate complex searches, reports, and dashboards with Splunk's core software to get the most out of their data

**Prerequisite Certification(s):**

- Splunk Core Certified Power User

**Prerequisite Course(s):**

- None

**Splunk Core Certified Advanced Power User Exam**

Time to study! We suggest candidates looking to prepare for this exam complete Fundamentals 3, Creating Dashboards, and Advanced Searching & Reporting **or** the following courses:

- Using Fields
- Working with Time
- Comparing Values
- Result Modification
- Leveraging Lookups and Subsearches
- Correlation Analysis
- Search Under the Hood
- Multivalue Fields
- Search Optimization
- Creating Field Extractions
- Enriching Data with Lookups
- Data Models
- Using Choropleth
- Introduction to Dashboards
- Dynamic Dashboards

See here for registration assistance.

**Congratulations! You are a...**


**Recommended Next Steps**

- Splunk Enterprise Certified Admin
- Splunk Cloud Certified Admin

# Thank You



 | 