Searching and Reporting with Splunk 5.0

Document Usage Guidelines

- Should be used only for enrolled students
- Not meant to be a self-paced document
- Not for distribution

Class goals

- Gain a deeper understanding of search and reporting concepts
- Create efficient, well-formed searches
- Perform calculations and evaluations on search results
- Generate reports and charts
- Correlate events with transactions
- Create and use lookups
- Create and use report acceleration
- Create and use macros

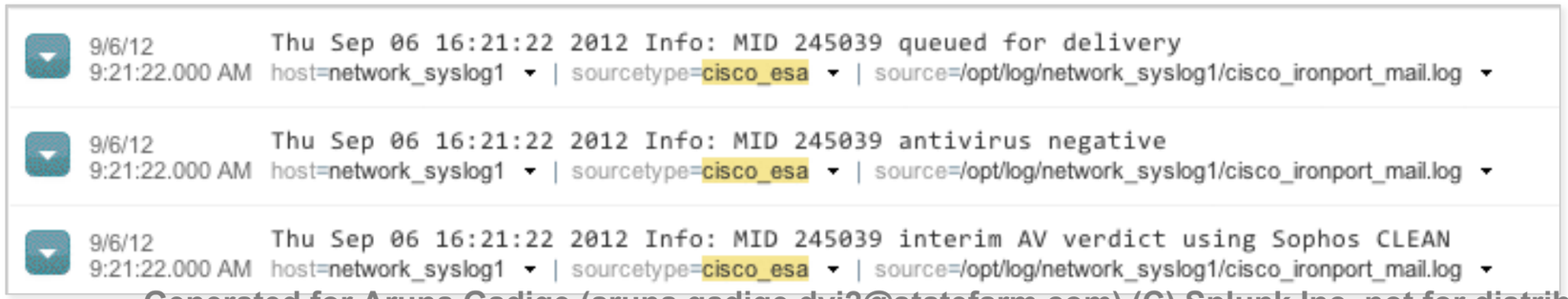
Course scenario

- Examples used in this course are based on an online retail business
- Searches and reports are based on:
  - Business analytics from the web access logs and lookups
  - Internal operations information from mail and internal network data

Data used in this course

- Examples and labs in this course are based on a variety of data:
  - Web server access logs (`sourcetype=access_*`)
    - Events related to the online store web activity and sales

  - Cisco mail logs (`sourcetype=cisco_e*`)
    - Events related to the internal email system



Data used in this course (cont'd)

- Cisco network logs (`sourcetype=cisco_w*`)
  - Events related to employee internet access

- You'll become familiar with the fields and characteristics of these data sources as the course progresses

Fields review

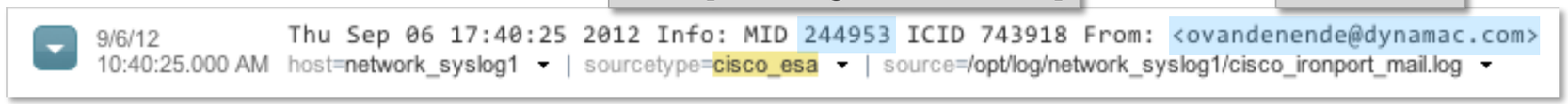
- Splunk extracts important fields
- A few are highlighted below (and there are many more!)

webserver log (access_*)

clientip | Referrer_domain | method | action

mail log (cisco_e*)

MID [Message Internal ID] | mailfrom



network log (cisco_w*)

cs_username | s_hostname

The Lab Environment

Linux Web Servers
[Apache]

Hosts: `www1` `www2` `www3`

Sourcetypes: `linux_secure` `access_combined`

Internal Mail Server

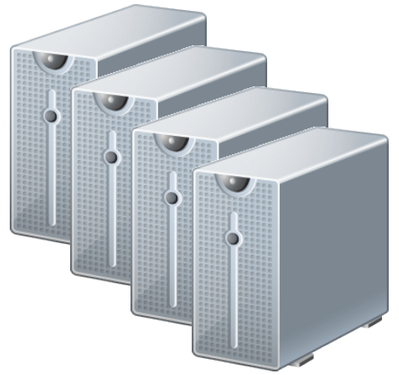
Host: `mailsv1`

Sourcetypes: `linux_secure` `sendmail_syslog`

Cisco Appliances

Host: `network_syslog1`

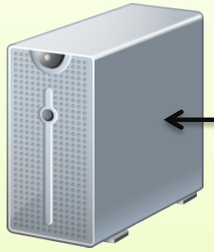
Sourcetypes: `cisco_esa` `cisco_wsa_squid` `linux_secure`



Windows Mail Servers
[IIS Voicemail]

Hosts: `vmail1` `vmail2` `vmail3` `vmail4`

Sourcetype: `iis-2`



Splunk Indexer
indexer.splunk.com

/opt/log

Trading App Server

Host: `trade_server1`

Sourcetype: `trade_entries`

Scenario callouts

- Many of the examples in this course relate to a specific scenario
- For each example, a question is given that might be asked by a colleague or manager in the company
  - The answers live in the Splunk data!
  - Example:

Who are the top site visitors?

Course outline

1. Search Fundamentals
2. Getting Statistics
3. Analyzing, Calculating, and Formatting
4. Creating Charts
5. Correlating Events
6. Enriching Data with Lookups and Workflow Actions
7. Report Acceleration
8. Creating and Using Macros

Section 1:
Search Fundamentals

Section objectives

- Review basic search commands
- Review general search practices
- Examine the anatomy of a search
- Describe search language syntax concepts
- Review fields and use the fields command
- Create a table

Basic search review

- **Keywords**
  search for "error"
- **Phrases**
  "web error" (different than web AND error)
- **Fields**
  search for status=404
- **Booleans**
  OR, AND, NOT; AND is implied; Can use ()'s
- **Wildcards**
  status="40*" matches 401, etc.
- **Comparisons**
  =,!=,<,<=,>=,> (delay > 10)

General search practices

- Time is the most efficient filter in Splunk
  - The most effective thing you can do is to narrow by time
- The more you tell the search engine, the better chance for good result
  - When applicable, searching for "access denied" is always better than searching for “denied”
  - To make searches more efficient, include as many terms as possible
    - You want to find events with "error" and "sshd"
    - 90% of the events include "error", but only 5% "sshd", include both values in the search

General search practices (cont'd)

- Inclusion is generally better than exclusion
  - Searching for `"access denied"` is faster than `NOT "access granted"`
- Apply powerful filtering commands as early in your search as possible
  - Filtering to one thousand events and then ten events is faster than filtering to one million events and then narrowing to ten

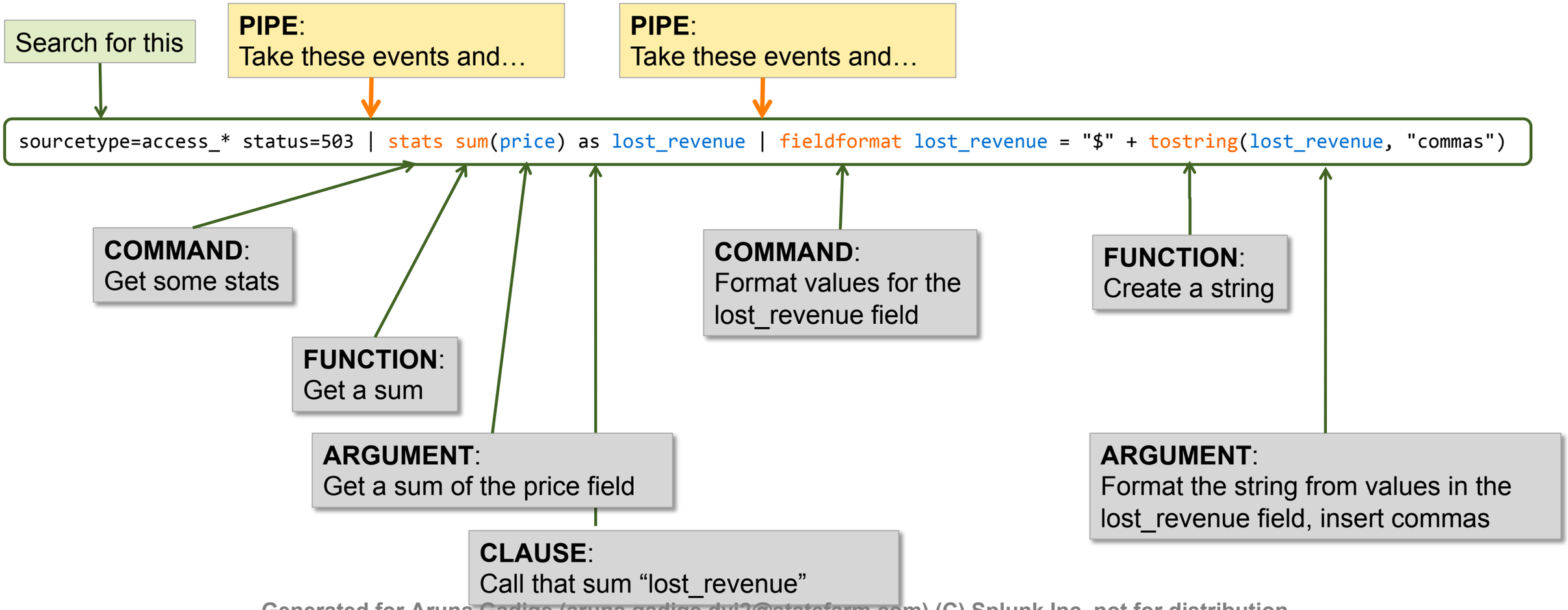
Search language syntax concepts

Searches are made up of 5 basic components

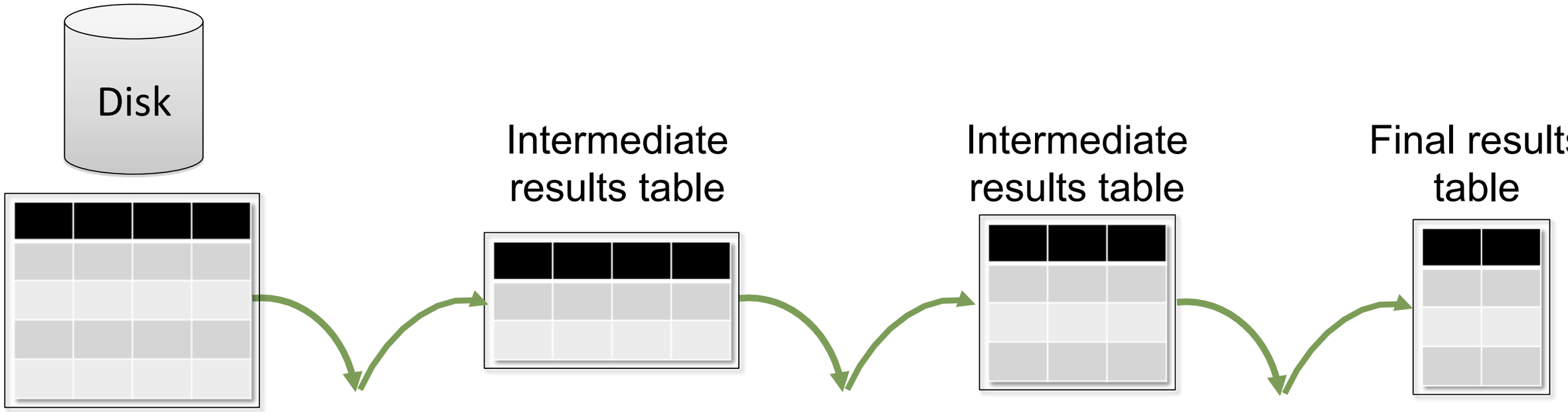
- **Search terms** – what are we looking for?
  - Keywords, phrases, Booleans, etc.
- **Commands** – what should we do with the results?
  - Create a chart, compute statistics, evaluate and format, etc.
- **Functions** – how should we chart, compute, or evaluate?
  - Get a sum, get an average, transform the values, etc.
- **Arguments** – are there variables we should apply to this function?
  - Calculate average value for a specific field, convert milliseconds to seconds, etc.
- **Clauses –** how should we group the results?
  - Get the average of values for the price field grouped by product, etc.

Search language example

This diagram represents a search, broken into its syntax components



Anatomy of a search




```
sourcetype=syslog ERROR | top user | fields - percent
```

Fetch events from disk that match

Summarize into table of top 10 users

Remove column showing percentage

Fields command overview

- Field extraction is one of the most costly parts of a search
- `fields` command allows you to include or exclude specified fields in your search or report
  - `fields` + to include (default, + is implied)
    - Occurs before field extraction
    - Improved performance
  - `fields` - to exclude
    - Occurs after field extraction
    - No performance benefit
    - Exclude fields used in search to make the table-display easier to read

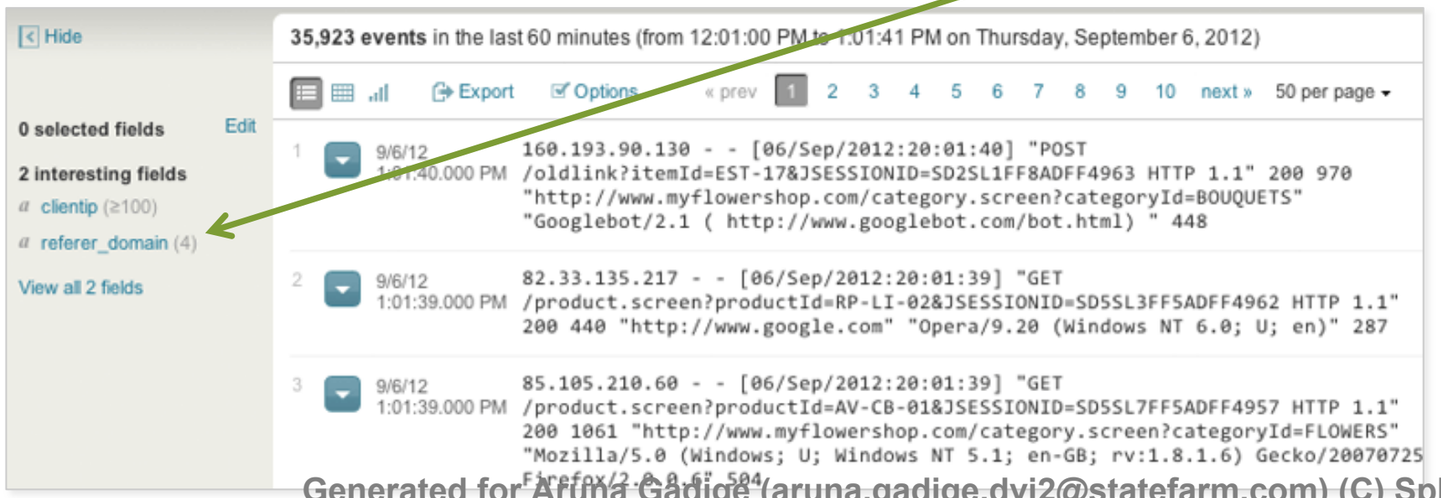
Include specific fields (field +)

- Improves performance – only the fields you specify are extracted
- `fields` does not exclude internal fields `_raw` and `_time` unless specified with remove fields command `fields - <field name>`

  `_raw` – original data of an event

  `_time` – timestamp in UNIX time

```
sourcetype=access_combined
| fields clientip, referer_domain
```



Remove specific fields (fields -)

- Use the remove `fields` command to exclude specific fields
- Removing fields can be useful for display with statistics and reporting commands

```
sourcetype=access_combined | top clientip
```

	clientip	count	percent
1	208.20.41.19	16	3.088803
2	17.247.28.82	15	2.895753
3	127.154.239.235	15	2.895753
4	55.42.187.44	14	2.702703
5	37.191.221.48	13	2.509653
6	182.103.110.180	13	2.509653
7	157.182.123.214	13	2.509653
8	79.36.9.202	12	2.316602
9	72.36.113.98	12	2.316602
10	7.64.242.253	12	2.316602

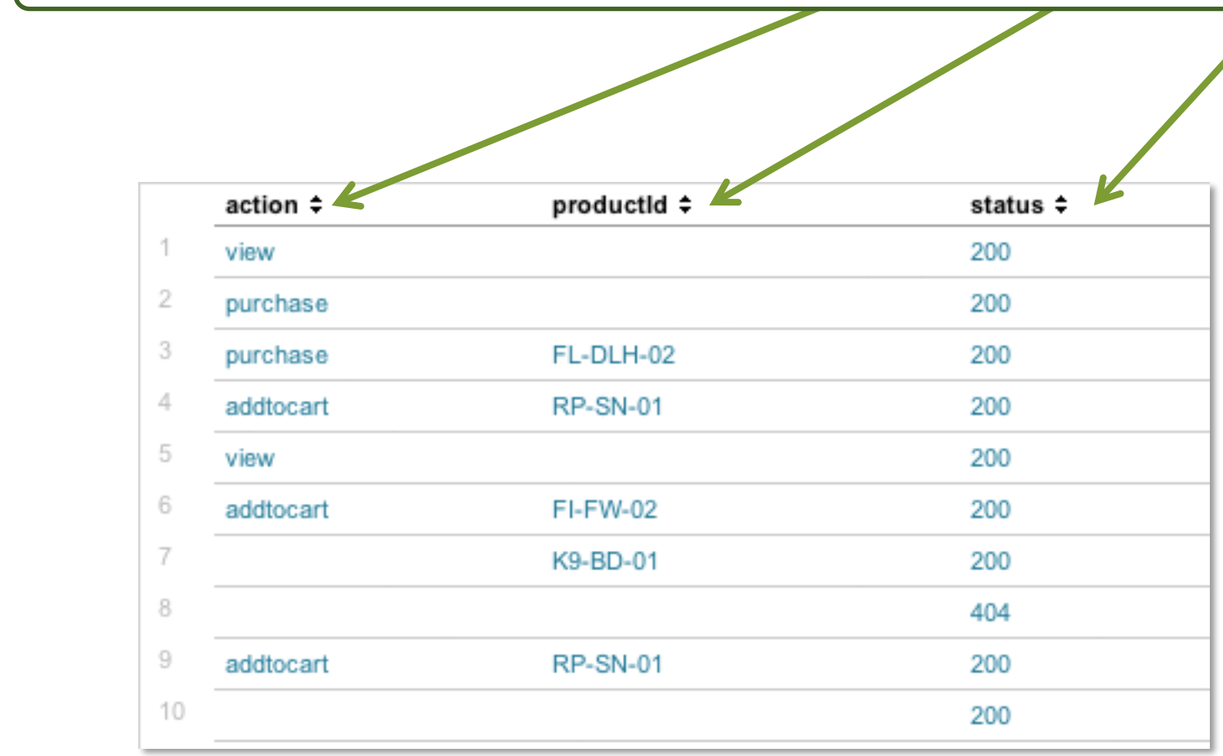
```
sourcetype=access_combined | top clientip | fields - percent
```

	clientip	count
1	208.20.41.19	16
2	7.64.242.253	15
3	17.247.28.82	15
4	127.154.239.235	15
5	55.42.187.44	14
6	37.191.221.48	13
7	182.103.110.180	13
8	157.182.123.214	13
9	79.36.9.202	12
10	72.36.113.98	12

Create a table

- `table` command returns a table formed by only fields in the argument list
- Columns are displayed in the order
  - Column headers are field names
  - Rows are field values
  - Each row represents an event

```
sourcetype=access_combined | table action, productId, status
```



	action	productId	status
1	view		200
2	purchase		200
3	purchase	FL-DLH-02	200
4	addtocart	RP-SN-01	200
5	view		200
6	addtocart	FI-FW-02	200
7		K9-BD-01	200
8			404
9	addtocart	RP-SN-01	200
10			200

Rename fields

- Use the `rename` command to rename fields in your display
- Useful for giving fields more meaningful names
- Use quotes to rename to a phrase
  - `rename productId as "Product ID"`

```
sourcetype=access_combined | table action, productId, status
| rename productId as "Product ID"
```

	action	Product ID	status
1	addtocart	K9-CW-01	200
2		K9-BD-01	404
3			200
4		K9-BD-01	200
5			200
6		RP-LI-02	200
7		FL-DLH-02	200
8		RP-SN-01	200
9	purchase	K9-BD-01	503
10		K9-CW-01	200

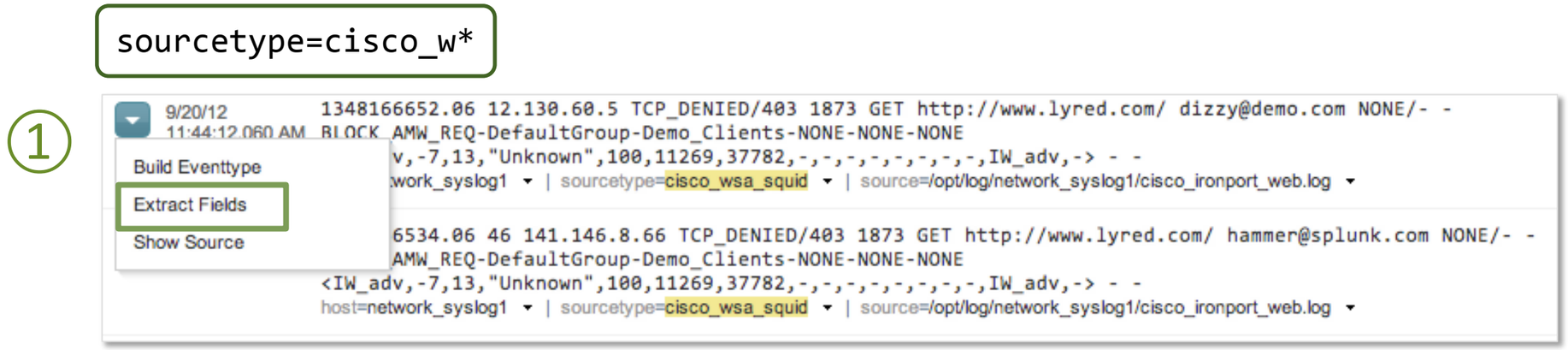
Extract fields methods

1. IFX (Interactive Field Extractor)
   - Graphic UI
   - Generates regex for you
   - Persist as knowledge objects
   - Re-usable in multiple searches
2. `rex`
   - NO UI; You must write regex
   - Only persists for the duration of the search
   - Do not persist as knowledge objects
   - Good for rarely used fields
3. Erex
   - Combination

For more details, see docs.splunk.com/Documentation/Splunk/5.0/SearchReference

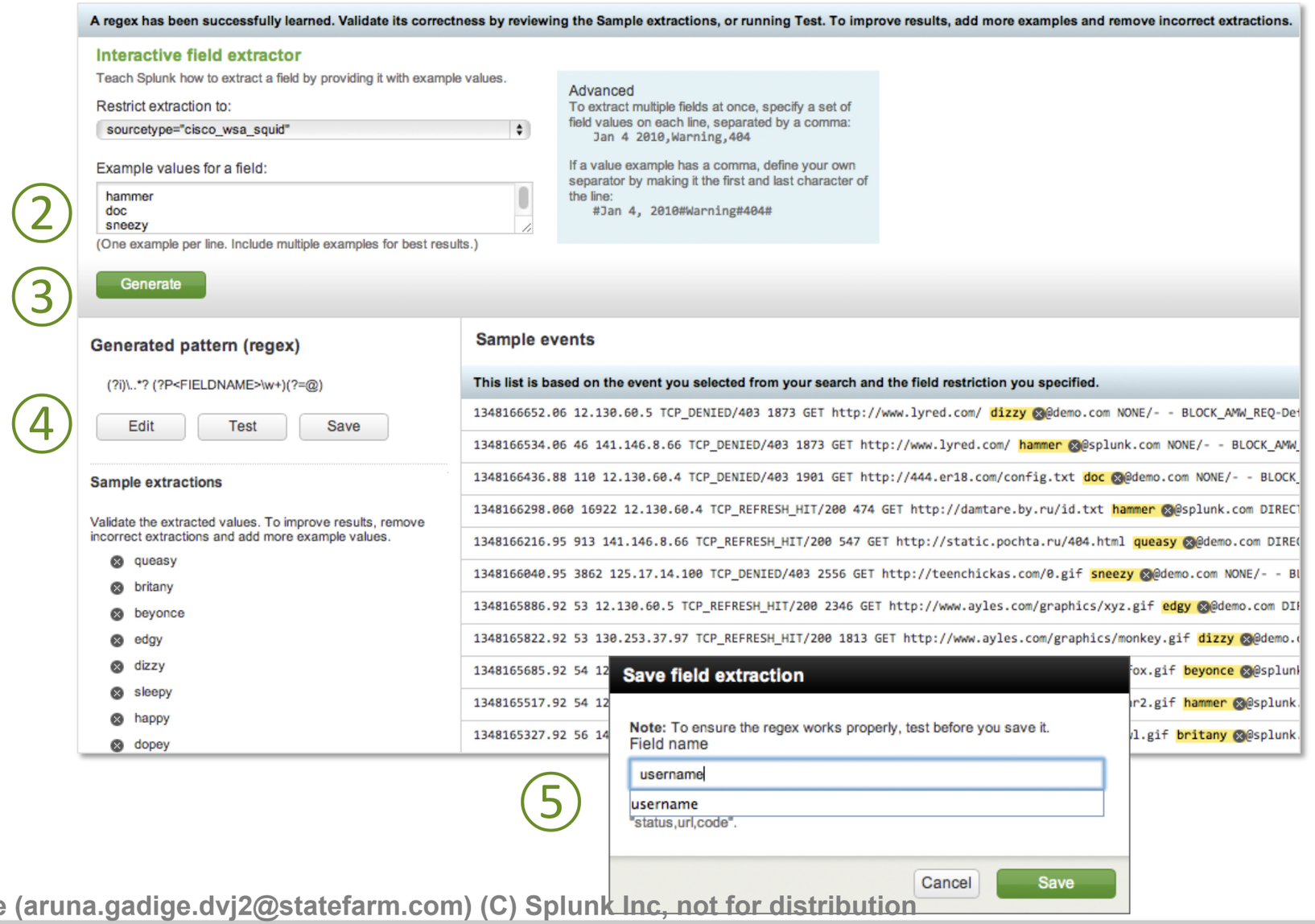
Extract fields with IFX

- You can use IFX [**I**nteractive **F**ield e**X**tractor]
  1. Select **Extract fields** from the event menu



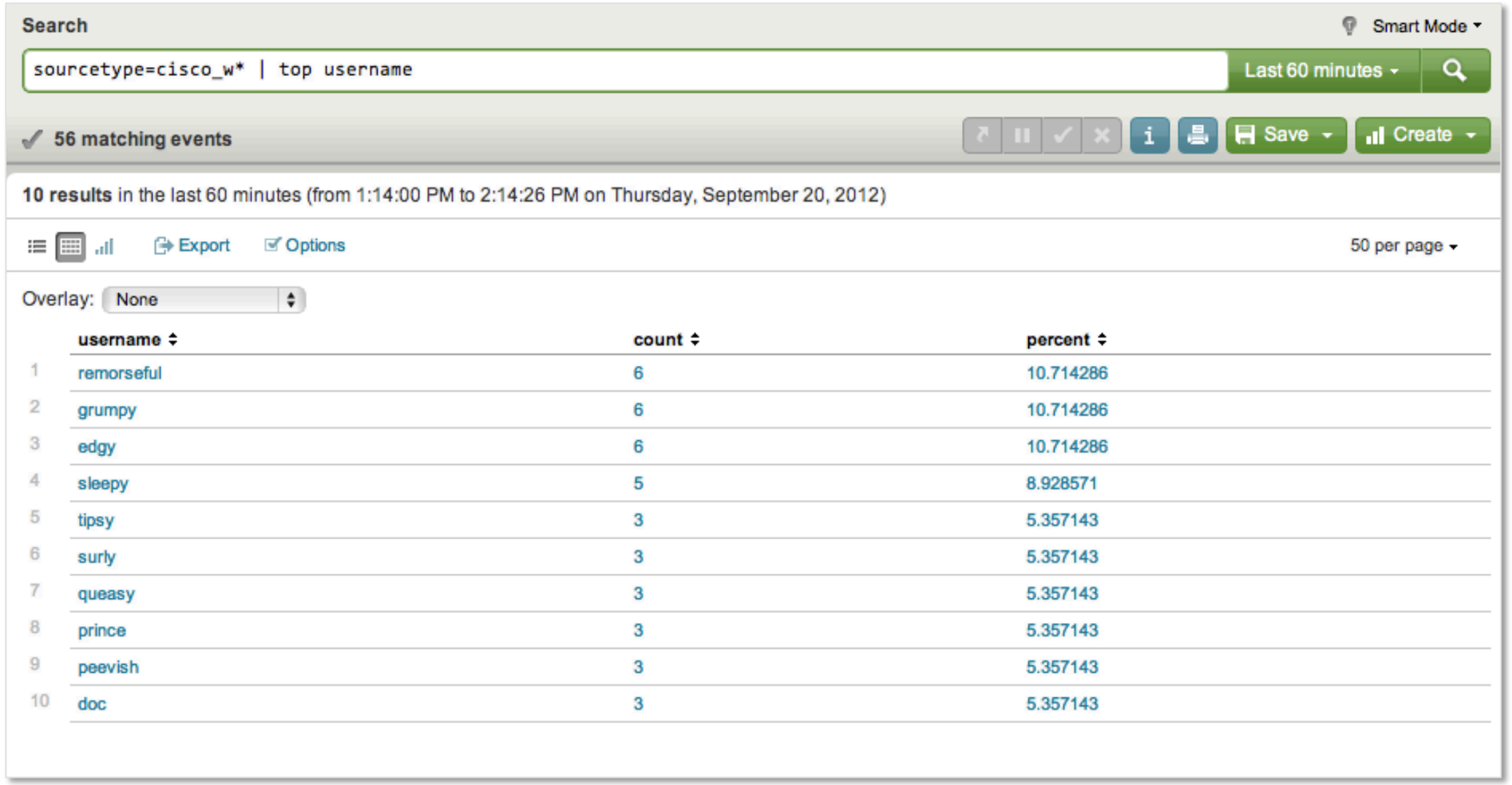
Extract fields with IFX (cont'd)

2. Provide example values of the field you want to extract
3. Generate the `regex`
4. Edit or test the `regex`
5. Save
   - For this example we named this field `username`



Using fields extracted from IFX

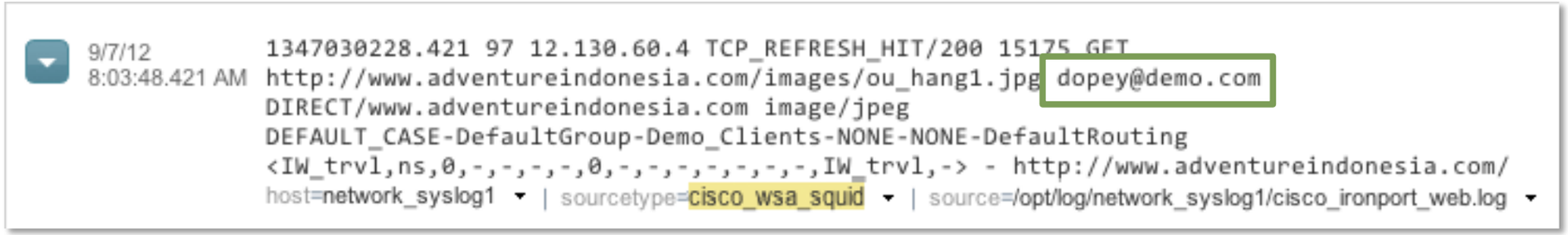
```
sourcetype=cisco_w* | top username
```



	username	count	percent
1	remorseful	6	10.714286
2	grumpy	6	10.714286
3	edgy	6	10.714286
4	sleepy	5	8.928571
5	tipsy	3	5.357143
6	surly	3	5.357143
7	queasy	3	5.357143
8	prince	3	5.357143
9	peevish	3	5.357143
10	doc	3	5.357143

Extracting fields with rex

- The internal network data includes a user name for each event, but has not been defined as a field
- The `rex` command allows you to extract the field at search time




```
sourcetype=cisco_w* | rex "\s+(?<username>\w+)@" | top username
```

	username	count	percent
1	grumpy	6	17.647059
2	doc	4	11.764706
3	bashful	4	11.764706
4	dopey	3	8.823529
5	sneezy	2	5.882353
6	sleepy	2	5.882353
7	peevish	2	5.882353
8	dizzy	2	5.882353
9	britany	2	5.882353
10	surly	1	2.941176

For example purposes for this course, the user name field has been extracted as cs_username

Extract fields from a table-formatted event

- Many data types are formatted as large single events in a table
- Each event contains fields with multiple values
  - Here, the first row represents the fields, all other rows represent values

sourcetype=ps

9/7/12
8:14:17.000 AM

USER	PID	PSR	pctCPU	CPUTIME	pctMEM	RSZ_KB	VSZ_KB	TTY	S	ELAPSED	COMMAND	ARGS
root	1	4	0.0	00:00:03	0.0	1488	19252	?	S	42-21:22:18	init	<noArgs>
root	2	4	0.0	00:00:00	0.0	0	0	?	S	42-21:22:18	[kthreadd]	<noArgs>
root	3	0	0.0	00:01:10	0.0	0	0	?	S	42-21:22:18	[ksoftirqd/0]	<noArgs>
root	4	0	0.0	00:00:39	0.0	0	0	?	S	42-21:22:18	[kworker/0:0]	<noArgs>
root	5	6	0.0	00:00:00	0.0	0	0	?	S	42-21:22:18	[kworker/u:0]	<noArgs>
root	6	0	0.0	00:00:00	0.0	0	0	?	S	42-21:22:18	[migration/0]	<noArgs>
root	7	1	0.0	00:00:00	0.0	0	0	?	S	42-21:22:18	[migration/1]	<noArgs>
root	8	1	0.0	00:00:00	0.0	0	0	?	S	42-21:22:18	[kworker/1:0]	<noArgs>
root	9	1	0.0	00:01:18	0.0	0	0	?	S	42-21:22:18	[ksoftirqd/1]	<noArgs>

Show most relevant lines (Exceeds 500 limit)
host=splunk.mycompany.com mail ▾ | sourcetype=ps ▾ | source=ps ▾

Extract fields from a table-formatted event (cont'd)

- `multikv` command extracts fields you specify
  - Field names are from the first row of each event as displayed on previous slide
- Command creates a separate event for each row
  - For better display, this example also pipes to the `table` command

```
sourcetype=ps | multikv fields USER pctCPU COMMAND
| table USER, pctCPU, COMMAND
```

	USER	pctCPU	COMMAND
1	root	0.0	init
2	root	0.0	[kthreadd]
3	root	0.0	[ksoftirqd/0]
4	root	0.0	[kworker/0:0]
5	root	0.0	[kworker/u:0]
6	root	0.0	[migration/0]
7	root	0.0	[migration/1]
8	root	0.0	[kworker/1:0]
9	root	0.0	[ksoftirqd/1]
10	root	0.0	[migration/2]
11	root	0.0	[kworker/2:0]
12	root	0.0	[ksoftirqd/2]
13	root	0.0	[migration/3]

Lab 1

Time: 10-15 minutes

Tasks:

- Log into Splunk on classroom server
- To become familiar with the data used in this course, examine all three sources of data
- Perform basic searches on **'webserver log'*** over the last 24 hours
- Perform basic searches on firewall data
- Create tables that include specific fields

Challenge:

- In the mail logs, use the rex command to extract a new field

* Note: **'store data'** means the **access_combined** data from flowershop.com

Section 2:
Getting Statistics

Section objectives

- Display `top` and `rare` values for given fields
- Describe the `stats` command
- Use the `stats` command

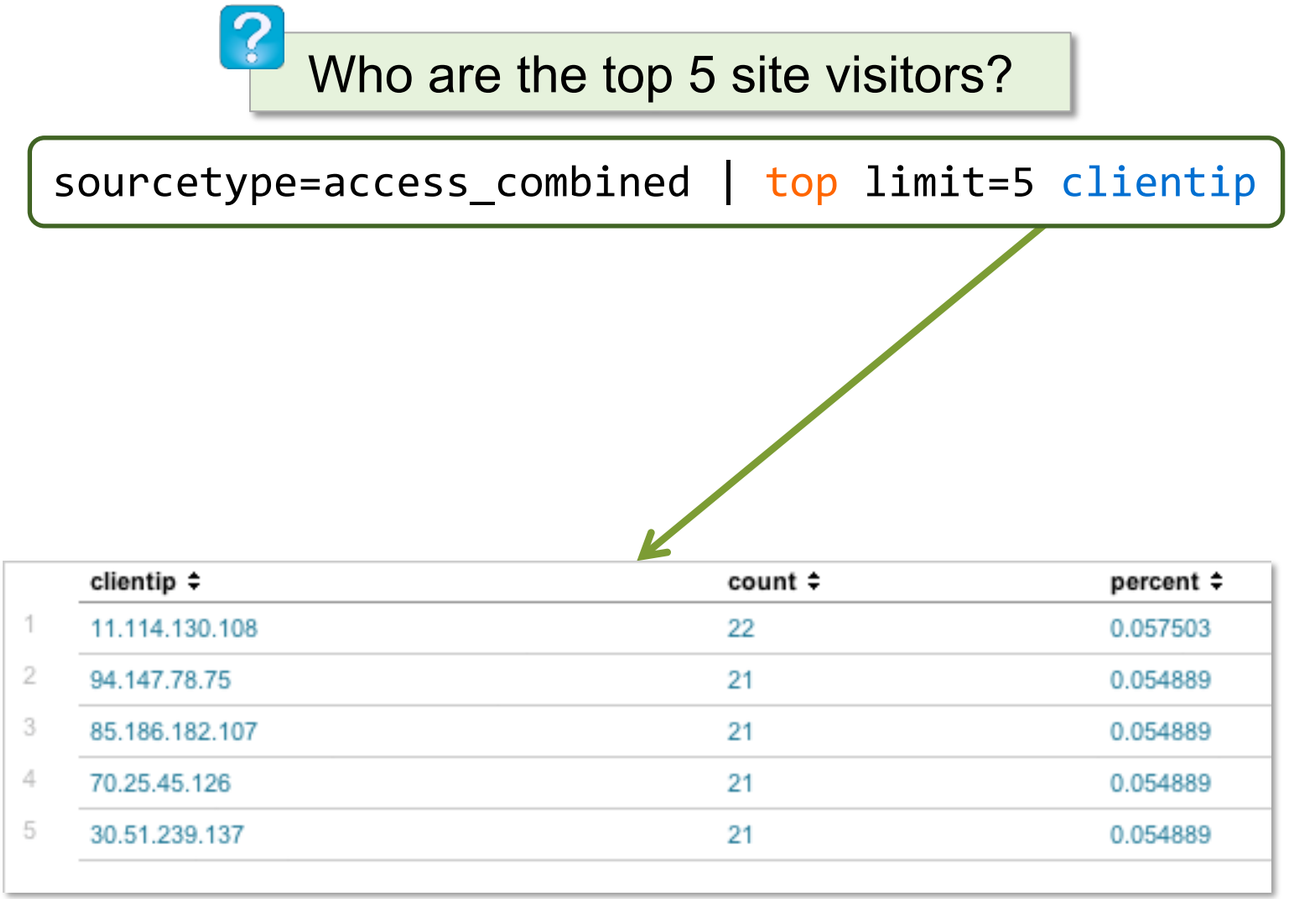
Reporting commands overview

The following reporting commands are discussed in this section:

- `top` – displays the most common values of a field
- `rare` – displays the least common values of a field
- `stats` – calculates statistics on the events that match your search criteria

Getting top values

- `top` command finds the most common values of a given field in the result set
  - Returns top 10 results by default
- Output is in table format
- Automatically returns a **count** and **percent**
- Adding `limit=#` after the `top` command returns the specified number of results



Getting top values (cont'd)

Using the `fields` command, you can remove the percent field from the results

Who are the top 5 site visitors, by number of visits?

```
sourcetype=access_combined | top limit=5 clientip | fields - percent
```

	clientip	count
1	11.114.130.108	22
2	85.186.182.107	21
3	70.25.45.126	21
4	30.51.239.137	21
5	251.131.58.237	21

Getting top values (cont'd)

- Adding the by clause to the `top` command, we can view the top sites and their associated "acceptable use" category
- `sort -count` ensures the table is sorted by count in descending order

What are the usage categories of the top sites?

```
sourcetype=cisco_w* | top s_hostname by usage | sort -count
```

	usage	s_hostname	count	percent
1	Personal	www.healthscout.com	17	45.945946
2	Personal	www.vpl.ca	7	18.918919
3	Unknown	www.ayles.com	6	42.857143
4	Unknown	-	6	42.857143
5	Personal	www.alyandaj.com	5	13.513514
6	Personal	damtare.by.ru	3	8.108108
7	Personal	www.espn.go.com	2	5.405405
8	Personal	www.areavoices.com	2	5.405405
9	Unknown	www.goppo.com	2	14.285714
10	Borderline	static.pochta.ru	1	100.000000
11	Personal	www.hybridarcade.com	1	2.702703
12	Violation	-	1	100.000000

Note: the usage field comes from an automatic lookup, which is covered later in this course.

Getting rare values

- The `rare` command returns the least common field values of a given field in the results
- Options are identical to the `top` command

What are the least frequently visited sites?

```
sourcetype=cisco_w* | rare s_hostname
```

	s_hostname	count	percent
1	static.pochta.ru	1	1.960784
2	www.hybridarcade.com	1	1.960784
3	www.areavoices.com	2	3.921569
4	www.espn.go.com	2	3.921569
5	www.goppo.com	2	3.921569
6	damtare.by.ru	3	5.882353
7	www.alyandaj.com	5	9.803922
8	www.vpl.ca	5	9.803922
9	www.ayles.com	6	11.764706
10	-	7	13.725490

Getting statistics

- `stats` command allows you to get statistics on the data that matches your search criteria
- You can apply different functions to the `stats` command
- Common functions include:
  - `count` – returns the number of events that match the search criteria
  - `distinct_count, dc` – returns a count of unique values for a given field
  - `sum` – returns a sum of numeric values
  - `avg` – returns an average of numeric values
  - `list` – lists all values of a given field
  - `values` – lists unique values of a given field

stats – count

- `count` returns the number of matching events based on the current search criteria

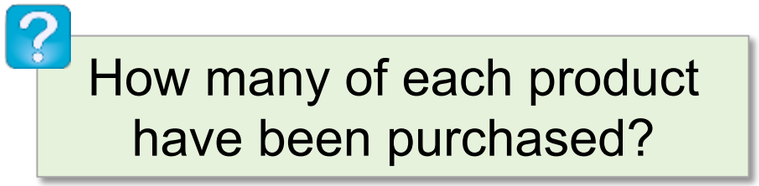
How many products were purchased?

```
sourcetype=access_* action=purchase | stats count
```

	count ⇕
1	7316

stats – count by

- by clause returns a count for each field value of a named field or set of fields
- This example counts the number of events when `action=purchase` for each `productId`

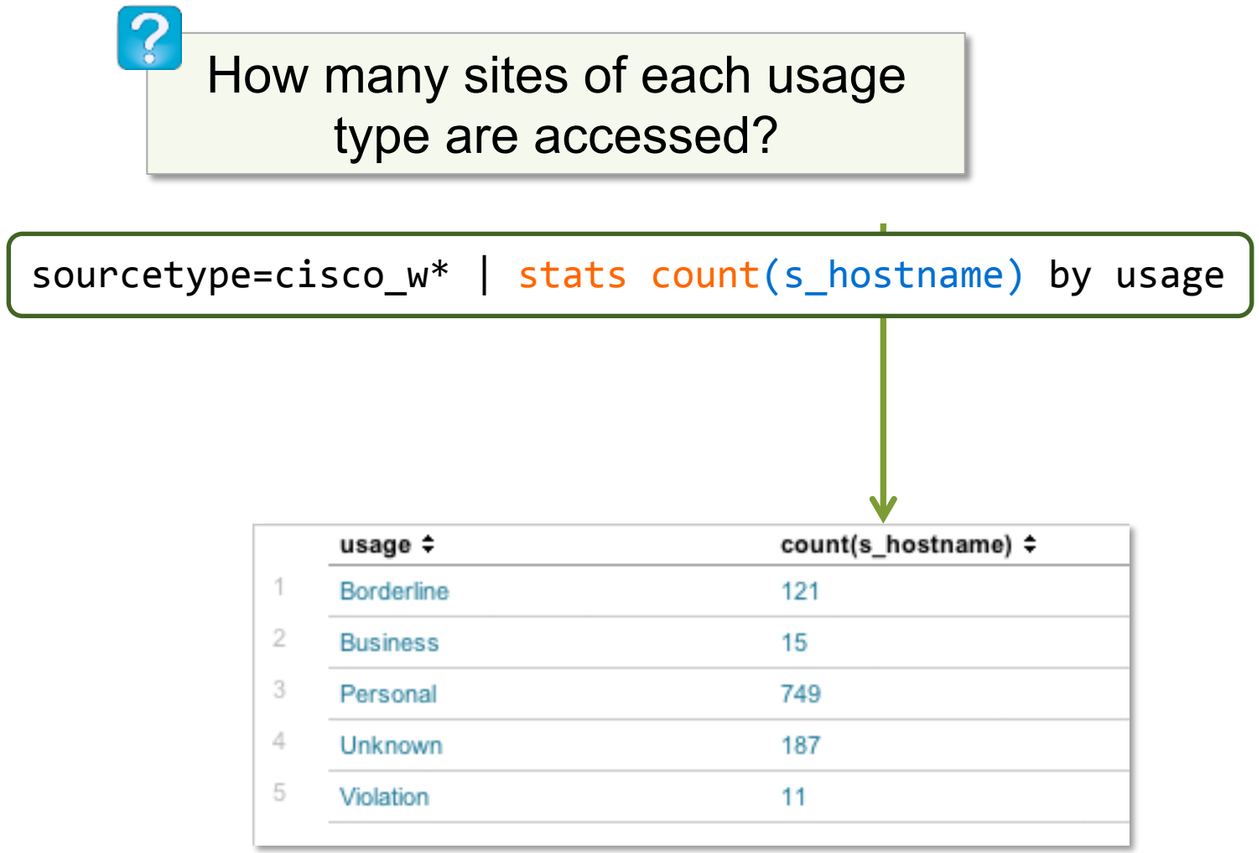



```
sourcetype=access_* action=purchase | stats count by productId
```

	productId	count
1	AV-CB-01	803
2	AV-SB-02	857
3	FI-FW-02	847
4	FI-SW-01	887
5	FL-DLH-02	552
6	FL-DSH-01	578
7	K9-BD-01	558
8	K9-CW-01	541
9	RP-LI-02	541
10	RP-SN-01	569

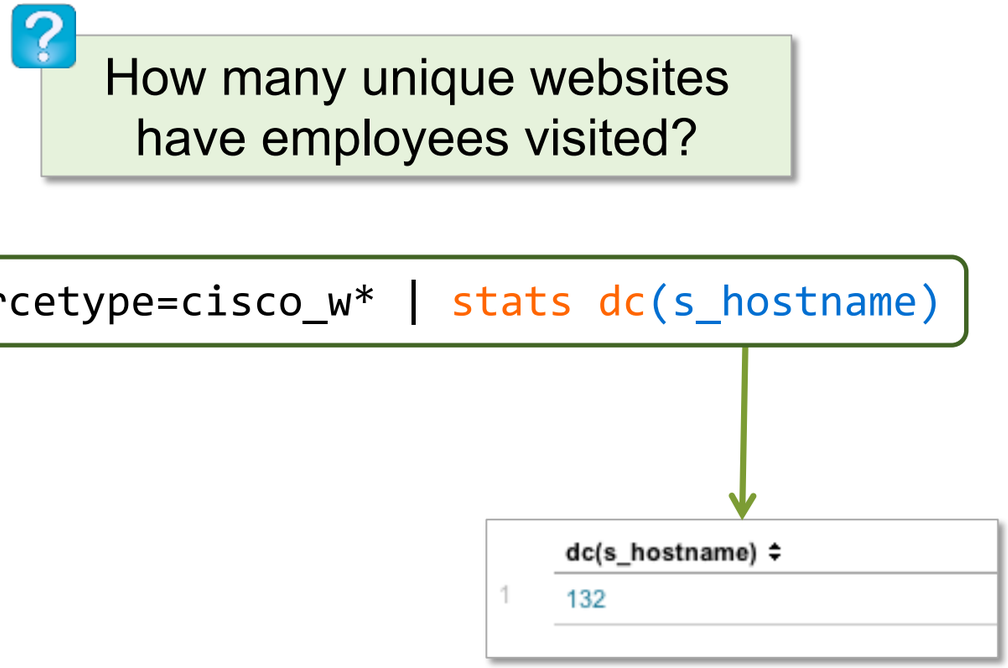
stats – count(field)

- Adding a field as an argument to the count function returns the number of occurrences for that field or set of fields



stats – distinct count

- `distinct_count()` or `dc()` provides a count of how many unique values there are for a given field in the result set
- This example counts how many unique values exist for `s_hostname`



stats – sum(field)

- For fields with a numeric value, you can sum the actual values of that field
- This example gets a sum of the values of the `sc_bytes` field for each website

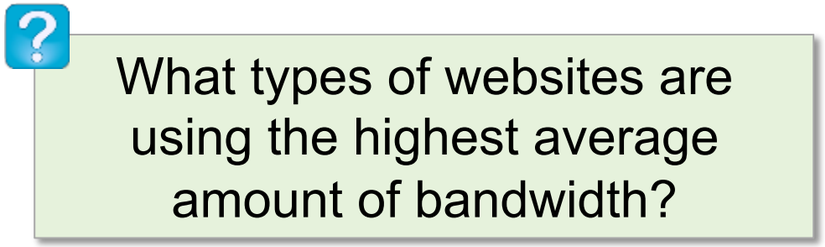
How much bandwidth is each website using?

```
sourcetype=cisco_w* | stats sum(sc_bytes) by s_hostname
```

	s_hostname	sum(sc_bytes)
1	adorepoem.com	3738
2	damtare.by.ru	12798
3	i-am-lost.net	17724
4	immensevids.com	3750
5	sportsillustrated.cnn.com	207458
6	static.pochta.ru	7111
7	txjsrf.com	571
8	viptraff.ru	5589
9	www.adventureindonesia.com	329322

stats – avg(field)

- The avg function averages numeric values of a given field
- This example averages all the values of the `sc_bytes` field, grouped by the `usage` field



What types of websites are using the highest average amount of bandwidth?

```
sourcetype=cisco_w* | stats avg(sc_bytes) by usage
```

	usage	avg(sc_bytes)
1	Borderline	3383.145833
2	Business	1738.900000
3	Personal	15294.034591
4	Unknown	1972.843284
5	Violation	2556.000000

addcoltotals

- `addcoltotals` command computes the sum of all numeric values for a given field in the result set
  - Adds the total to the bottom of the column
  - Use `label` to identify total

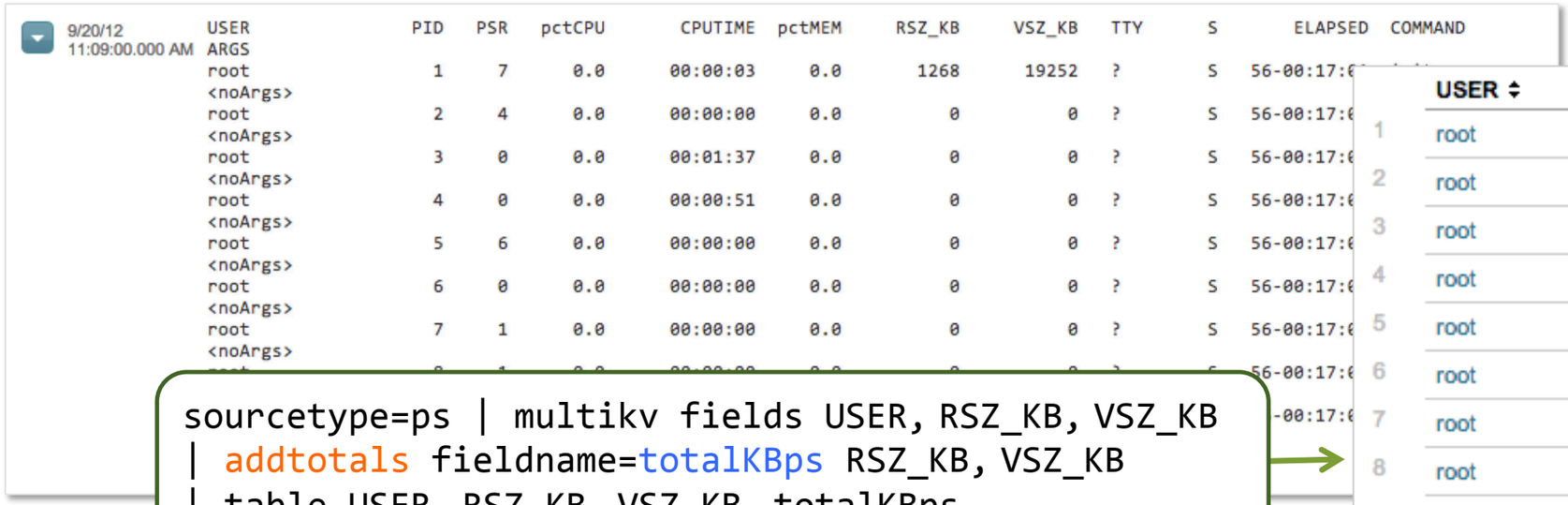
How many items are being removed from carts before purchase?

	productId	count
1	AV-CB-01	1
2	AV-SB-02	4
3	FI-FW-02	3
4	FI-SW-01	2
5	K9-BD-01	1
6	K9-CW-01	3
7	RP-LI-02	1
8	RP-SN-01	2
9	Total	17

```
sourcetype=access_* action=remove
| stats count by productId
| addcoltotals label=Total labelfield=productId
```

addtotals

- `Addtotals` computes the sum of numeric fields for each event
  - Define a field name in which to place the total value
  - Optionally, specify only certain fields to include in the sum
  - In this example, **addtotals** is applied to the `RSZ_KB` and `VSZ_KB` fields
  - By default, the command computes a sum for **all** numeric fields in the event



	USER	RSZ_KB	VSZ_KB	totalKBps
1	root	200040	346548	546588
2	root	147256	412436	559692
3	root	122800	1626784	1749584
4	root	122248	1626784	1749032
5	root	122244	1626784	1749028
6	root	122168	1626784	1748952
7	root	122164	1626784	1748948
8	root	121668	1626784	1748452
9	root	121656	1626784	1748440
10	root	120180	1626784	1746964

stats – list(field)

- `list` function lists all field values for a given field
- This example lists the websites each user visits
  - Since the security logs generate an event for each network request, the same hostname appears multiple times
  - If you want a list of "unique" field values, use the `values` function

What websites are accessed by each user?

```
sourcetype=cisco_w* | stats list(s_hostname) by cs_username
```

	cs_username	list(s_hostname)
1	bashful@demo.com	www.adventureindonesia.com www.adventureindonesia.com i-am-lost.net www.ayles.com www.fftoday.com www.jcpenny.com www.exploratorium.edu www.exploratorium.edu www.exploratorium.edu www.exploratorium.edu www.exploratorium.edu www.exploratorium.edu www.alyandaj.com www.alyandaj.com www.healthscout.com www.healthscout.com www.fftoday.com www.jcpenny.com www.jcpenny.com
2	beyonce@splunk.com	www.ayles.com www.fftoday.com static.pochta.ru

stats – values(field)

- `values` function creates a list of **unique** field values for a given field

```
sourcetype=cisco_w*
| stats values(s_hostname) by cs_username
```

cs_username	values(s_hostname)
bashful@demo.com	343.boolans.com ad.doubleclick.net archis.org asiamo.net cl.kazaa.com excalibur.websiteactive.com global.nytimes.com i-hack.cn kaarlemculloch.com nopalevo.com panthose-city.titleprisonpraise.cn static.pochta.ru www-cdn.dell.com www.aaa-livedoor.net www.aaanativearts.com www.actresspictures.co.uk www.ayles.com www.azcentral.com www.bigbowl.com www.blackhorsefarmmaine.com www.bloggerforum.com www.blossomfloristla.com www.boatloco.com www.boingboing.net www.bradblog.com www.brooklynpubliclibrary.org www.cancuncare.com www.caraibes-webdo.net www.carstickers.com www.cbssports.com www.cbssportsstore.com www.celebrityweasel.com www.checkbook.org ⋮

What is the unique list of websites accessed by each user?

as

- The `as` clause can be used to rename calculated field names, which can then be used in subsequent commands
- In this example, the value of `count(JSESSIONID)` is placed in a new field named `sessions`
  - The `sessions` field is then used with the `addcoltotals` command

```
sourcetype=access_*
| stats count(JSESSIONID) as sessions by host
| addcoltotals sessions
```

	host	sessions
1	www1	369
2	www2	304
3	www3	331
4		1004

stats – sparkline

- Used in conjunction with the `stats` and `chart` commands
- Creates a mini-timeline in a report
  - Represents the same time span as the search – in this case `"last 7 days"`
  - Not to be confused with `timechart`, which creates a standalone visualization

Note: chart and timechart are covered later in this course

What is the purchase trend for each product ID over the last 7 days?

```
sourcetype=access* action=purchase
| stats sparkline count by productId
| sort -count
```

	productId	sparkline	count
1	FL-DLH-02		249
2	FI-SW-01		247
3	K9-BD-01		237
4	AV-CB-01		236
5	AV-SB-02		231
6	FI-FW-02		230
7	RP-SN-01		227
8	FL-DSH-01		226
9	RP-LI-02		222
10	K9-CW-01		217

Lab 2

- **Time:**
  - 30-35 minutes
- **Tasks:**
  - Report on top and rare values
  - Remember, there are two methods to eliminate or filter:
    1. NOT referer_domain="*myflowershop*”
    2. referer_domain!="*myflowershop*”
  - Use the stats command and associated functions
  - Create a new dashboard and add panel

Section 3:
Analyzing, Calculating, and Formatting

Section objectives

- Describe the `eval` command
- Perform calculations on values with `eval`
  - Convert values
  - Round values
  - Format values
  - Use conditional statements
- Further filter calculated results

Eval command overview

- `eval` allows you to calculate and manipulate field values in your display
  - Useful for calculations such as add, subtract, multiply, divide
  - Does not re-write event data in the index
- Supports a variety of functions
- Results of `eval` are written to a specified field
  - Can be a new or existing field
  - If the destination field exists, the values of the field are replaced by the results of `eval`

Convert values with eval

- In this example, the report displays the sum of bytes for each usage category
  - It's hard to determine how much bandwidth is being used by looking at bytes

What types of websites are using the most bandwidth in bytes?

```
sourcetype=cisco_w* | stats sum(sc_bytes) as bytes by usage
```

	usage	bytes
1	Borderline	172969
2	Business	352
3	Personal	78546
4	Unknown	17767
5	Violation	2556

- First, we'll use `eval` to convert the `bytes` value into megabytes…

Convert values with eval (cont'd)

- The results of `eval` must always be set to a new or existing field
- In this example:
  - Indicate which field `eval` results should populate (here, we create a new field)
  - Divide `bytes` field value by `1024` and again by `1024` to convert bytes to MB

What types of websites are using the most bandwidth in megabytes?

```
sourcetype=cisco_w* | stats sum(sc_bytes) as bytes by usage | eval MB = bytes/1024/1024
```

	usage	bytes	MB
1	Borderline	222519515	212.211146
2	Business	143625504	136.971954
3	Personal	1410063410	1344.741259
4	Unknown	299804514	285.915865
5	Violation	5188585	4.948220

Define a field to set the eval results to

Divide the field value by 1024/1024

Round values

- But there's still these pesky decimal points!
- The round(field or number, decimals) function sets the value of a field to the number of decimals you specify
  - In this example, we divide the value of the bytes field by 1024/1024 then round to 2 decimal points

What types of websites are using the most bandwidth in megabytes, rounded to 2 decimal points?

```
sourcetype=cisco_w* | stats sum(sc_bytes) as bytes by usage | eval MB = round(bytes/1024/1024, 2)
```

	usage	bytes	MB
1	Borderline	222549045	212.24
2	Business	143625504	136.97
3	Personal	1410063410	1344.74
4	Unknown	299804514	285.92
5	Violation	5188585	4.95

Divide the value of bytes by 1024/1024, then round to 2 decimal points

Remove fields

- Now that we've calculated and formatted our results in the new MB field, we can remove the bytes field from the report as a final command
  - It's safe to remove fields after their values have been used in previous parts of the search string

How many MBs are consumed for each usage type?

```
sourcetype=cisco_w*
| stats sum(sc_bytes) as bytes by usage
| eval MB = round(bytes/1024/1024, 2) | fields - bytes
```

	usage	MB
1	Borderline	213.73
2	Business	136.97
3	Personal	1344.74
4	Unknown	285.92
5	Violation	4.95

Compare values with eval

- Can perform mathematical functions against fields with numeric field values
- This example compares the flowershop price against the competitor's
  - Subtract the value of `flowersrus_price` from `price`
  - `flowersrus_price` is another field available via a lookup!

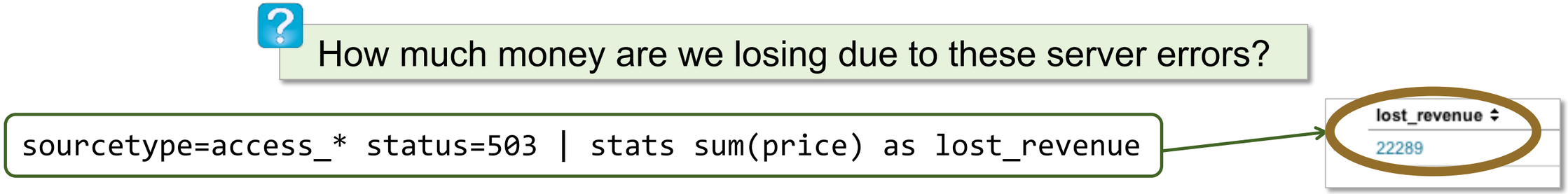
How do our prices compare to the competition?

```
sourcetype=access_combined product_name=*
| eval difference = price - flowersrus_price
| table product_name, price, flowersrus_price, difference
```

	product_name	price	flowersrus_price	difference
1	Greetings Fruit Basket	12	6	6
2	Dozen Red Roses	99	149	-50
3	Sweet Dreams Bouquet	89	92	-3
4	Birthday Bouquet	299	339	-40
5	Cake Serving Set	89	85	4
6	Day Spa Certificate	35	40	-5
7	Mixed Rose Bouquet	15	21	-6
8	Tulip Bouquet	250	279	-29
9	Cake Serving Set	89	85	4
10	Dozen Red Roses	99	149	-50

Format values with fieldformat

- `fieldformat` is used to format values for a more meaningful display
- Consider this search that determines lost revenue in the last 24 hours, based on events that have a `503` status and sum of the `price` field values



- The result is somewhat ambiguous, especially if it will ultimately display on a dashboard

Format values with fieldformat (cont'd)

- `fieldformat` allows you to format a field value without changing the underlying value
- The `tostring` function can convert a numeric field value to a string
  - In this example, a `"$"` is prepended to the value
- The `"commas"` argument automatically applies commas appropriately

```
sourcetype=access_* status=503 | stats sum(price) as lost_revenue
| fieldformat lost_revenue = "$" + tostring(lost_revenue, "commas")
```

lost_revenue
$22,289

Apply to lost_revenue field

Prepend with a dollar sign

Convert the value of lost_revenue to a string

Auto-apply commas appropriately

tostring considerations

- If you use `tostring` with the `eval` command, the resulting value is no longer considered numeric
  - Sorting columns of numbers and timestamps may not work as expected
  - Automatic drilldown will not work with post-reporting conversions on charts
- However, using `tostring` with the `fieldformat` command retains the numeric properties of field values

Using multiple eval commands

- Remember each subsequent command references the results of the previous commands
- In this example, we:
  - Set the `sales` field with total revenue for each product `ID`
  - Set a new field `USD` with the value of `sales` and prepend with `"$"`
  - Set a new field `GBP` with the exchange rate (value of `sales` field * exchange rate)
  - Prepend the `GBP` field with `"£"`
  - Remove the `sales` field from the final output

```
sourcetype=access_* action=purchase
| stats sum(price) as sales by productId
| eval USD = "$" + sales
| eval GBP = (sales*.722909)
| eval GBP = "£" + GBP
| fields - sales
```

	productId	sales	GBP	USD
1	AV-CB-01	200250	£14	
2	AV-SB-02	12810	£92	
3	FI-FW-02	10164	£73	
4	FI-SW-01	78765	£56	
5	FL-DLH-02	54648	£39	
6	FL-DSH-01	28322	£20	
7	K9-BD-01	166543	£12	
8	K9-CW-01	48149	£34	
9	RP-LI-02	205039	£14	
10	RP-SN-01	19845	£14	

	productId	GBP	USD
1	AV-CB-01	£144763	$200250
2	AV-SB-02	£9260.46	$12810
3	FI-FW-02	£7347.65	$10164
4	FI-SW-01	£56939.9	$78765
5	FL-DLH-02	£39505.5	$54648
6	FL-DSH-01	£20438.8	$28273
7	K9-BD-01	£120395	$166543
8	K9-CW-01	£34807.3	$48149
9	RP-LI-02	£148225	$205039
10	RP-SN-01	£14346.1	$19845

Conditional statements

- The `if` function takes three arguments
- The first argument is a Boolean expression
  - If it evaluates to TRUE, the result evaluates to the second argument
  - If it evaluates to FALSE, the result evaluates to the third argument
- Arguments must be enclosed in `"quotes"`

How much network bandwidth is going toward non-business activity?

Test the value of the usage field equal to "Business"

If true, set the value of usage to "Business Usage"

If false, set the value of usage to "Other"

```
sourcetype=cisco_w* | eval usage = if(usage == "Business","Business Usage","Other")
| stats sum(sc_bytes) as bandwidth by usage
| eval bandwidth_used = tostring(round(bandwidth/1024/1024, 2)) + " MB"
| fields - bandwidth
```

	usage	bandwidth_used
1	Business Usage	0.02 MB
2	Other	11.42 MB

Where command

- Runs an `eval` expression to filter the results
  - The result of the expression must be Boolean
  - Keeps only the results for which the evaluation is true, >20 or =100
- Useful in further filtering a search
- This example counts the occurrences of each hostname in the result set, then only returns a result when the count exceeds 20

Which sites were visited more than 20 times?

	s_hostname	count
1	-	136
2	damtare.by.ru	23
3	www.adventureindonesia.com	38
4	www.areavoices.com	30
5	www.ayles.com	60
6	www.collectiblestoday.com	40
7	www.exploratorium.edu	179
8	www.fftoday.com	48
9	www.healthscout.com	57
10	www.starteasy.com	69

```
sourcetype=cisco_w*
| stats count by s_hostname | where count > 20
```


Lab 3

- **Time:**
  - 25-30 minutes
- **Tasks:**
  - Use the `eval` command to convert field values `round` field values
  - Use `fieldformat` to display values differently, without changing original properties of a field
  - Create a new Sales Dashboard and add a panel
  - Use conditional statements (`if`)
  - Filter results with the `where` command

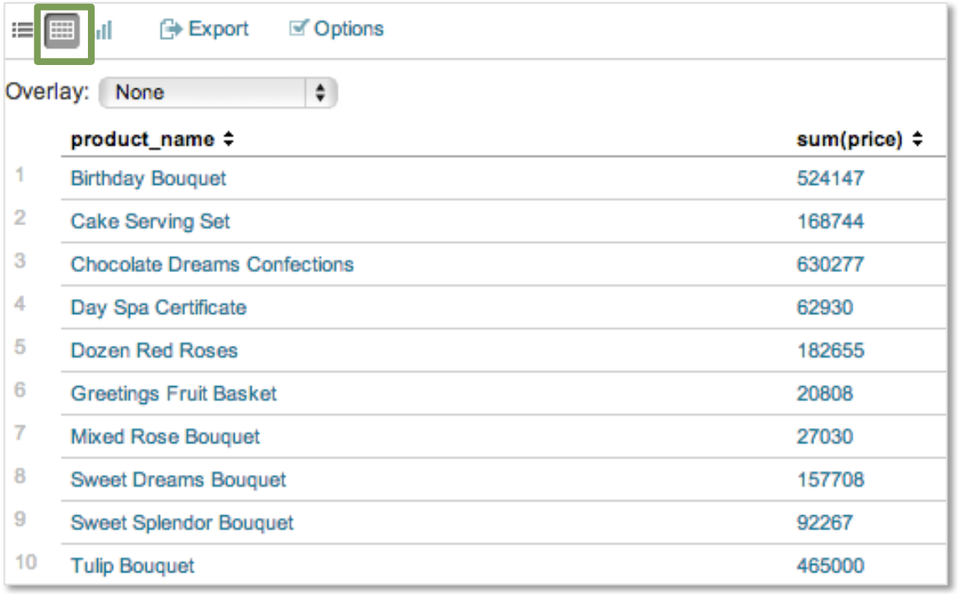
Section 4:
Creating Charts

Section objectives

- Describe chart types
- Describe the `chart` command
- Create a basic chart
- Split values into multiple series
- Define stacked mode
- Omit null and other values from charts
- Create a `timechart`
- Chart multiple values on the same timeline
- Format charts
- Apply statistical functions

View results as a chart

- When a search returns statistical values, you can easily view results as a chart
- Can also create charts in Report Builder



	product_name	sum(price)
1	Birthday Bouquet	524147
2	Cake Serving Set	168744
3	Chocolate Dreams Confections	630277
4	Day Spa Certificate	62930
5	Dozen Red Roses	182655
6	Greetings Fruit Basket	20808
7	Mixed Rose Bouquet	27030
8	Sweet Dreams Bouquet	157708
9	Sweet Splendor Bouquet	92267
10	Tulip Bouquet	465000

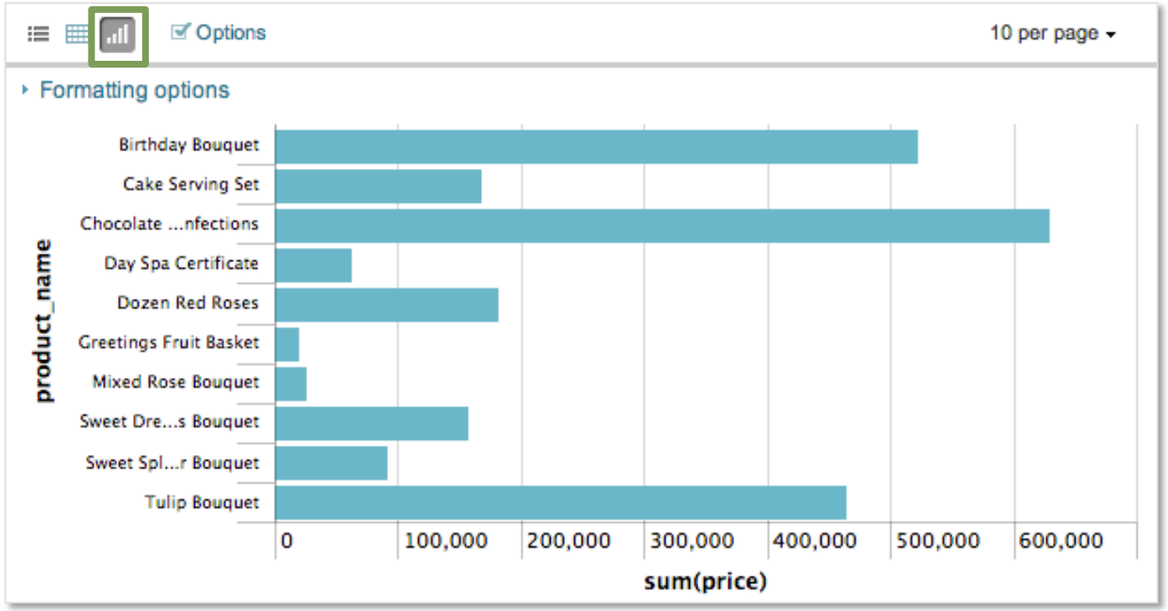
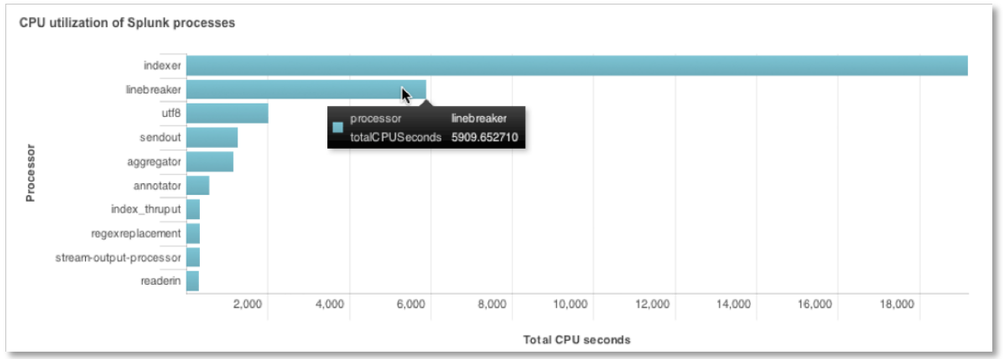
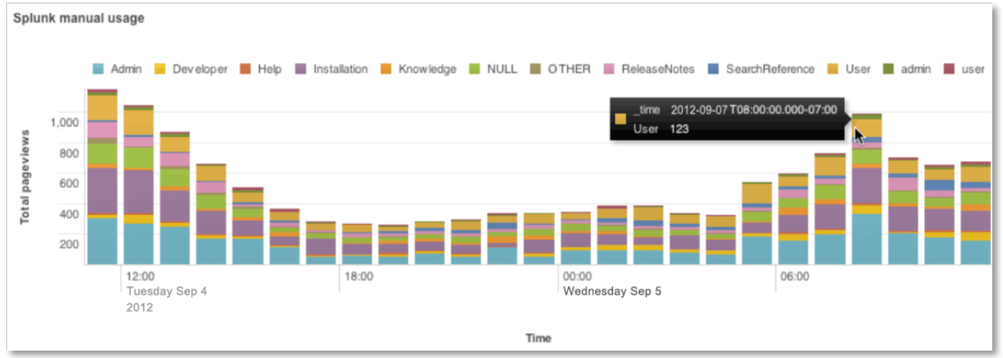


Chart types

- Column and bar charts
  - Compare the frequency of field values
- Stacked column and bar charts
  - All columns are segments of a single column
- Line chart
  - Show trends either over time or in comparison to another field value





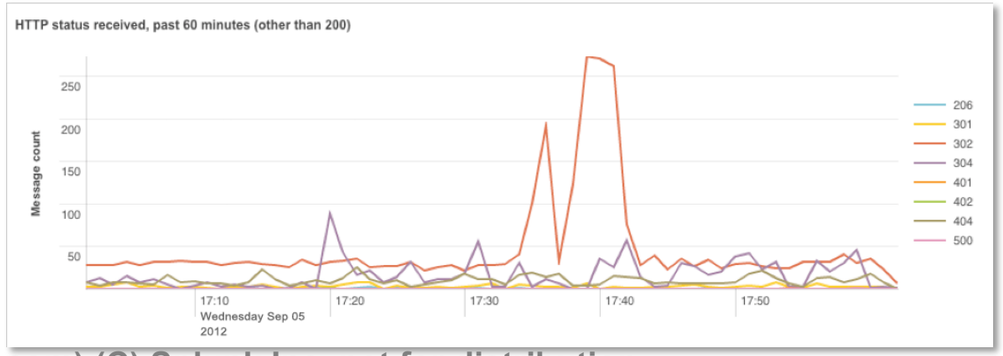
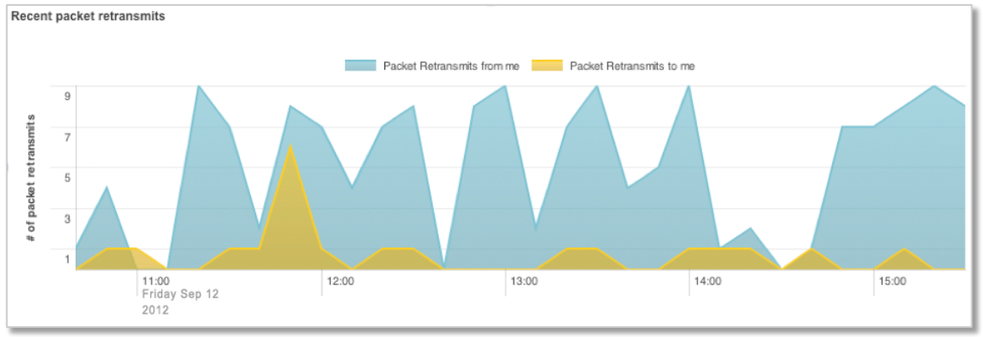
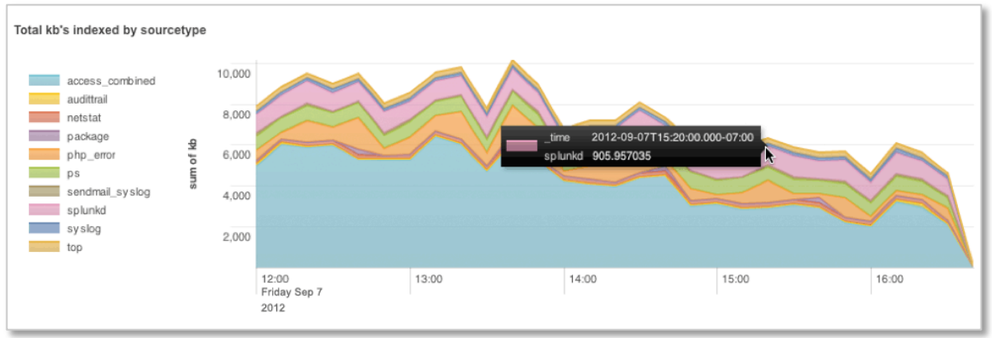


Chart types (cont'd)

- Area chart
  - Show trends either over time or in comparison to another field value
- Stacked area chart
  - Show multiple series among the trends in your data
- Pie chart
  - Show the relationship of parts of your data to the entire set of data as a whole





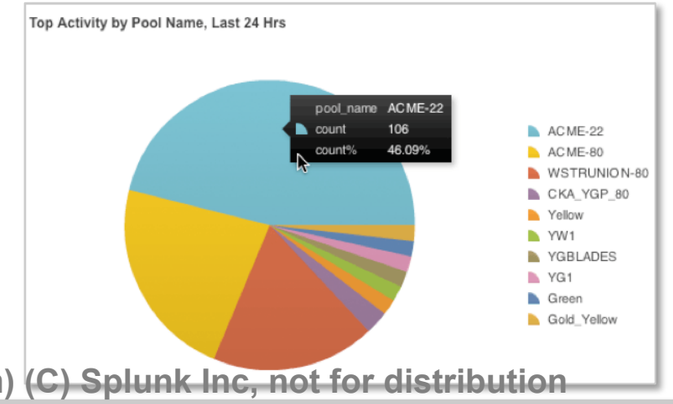


Chart command overview

- `chart` command can display any series of data that you want to plot
- You decide what field is tracked on the x-axis
  - Where `stats` uses the `by` clause to group data, `chart` uses the `by` or `over` clauses to determine which field takes the x-axis

    `chart avg(bytes) by host` – the host field is the x-axis since there's no split for the series

    `chart avg(bytes) over host by date_wday` – the host field is the x-axis and the series is split by day of week

  - Because the chart command is designed to return chartable results, the value of the y-axis should always be numeric

Basic chart example – column

- This example shows a basic chart
- count function counts the number of events for each http status in the result set

Are any hosts throwing a lot of errors?

```
sourcetype=access_* | chart count by status
```

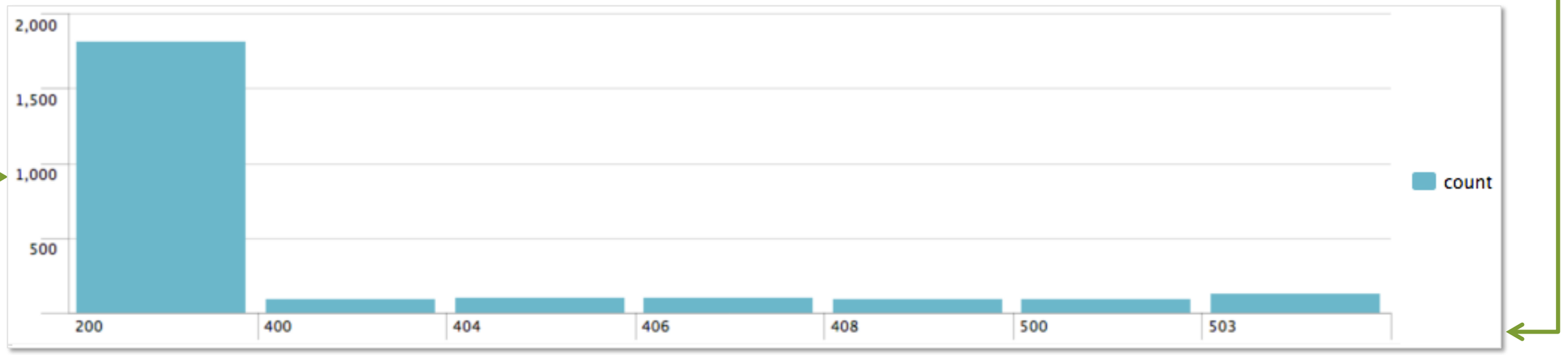


Chart – split by

- If you want to split the series, identify the x-axis field with over, then use by to split the series by additional fields
- In this example, we split the series by host

Are any hosts throwing a lot of errors?

```
sourcetype=access_* | chart count over status by host
```

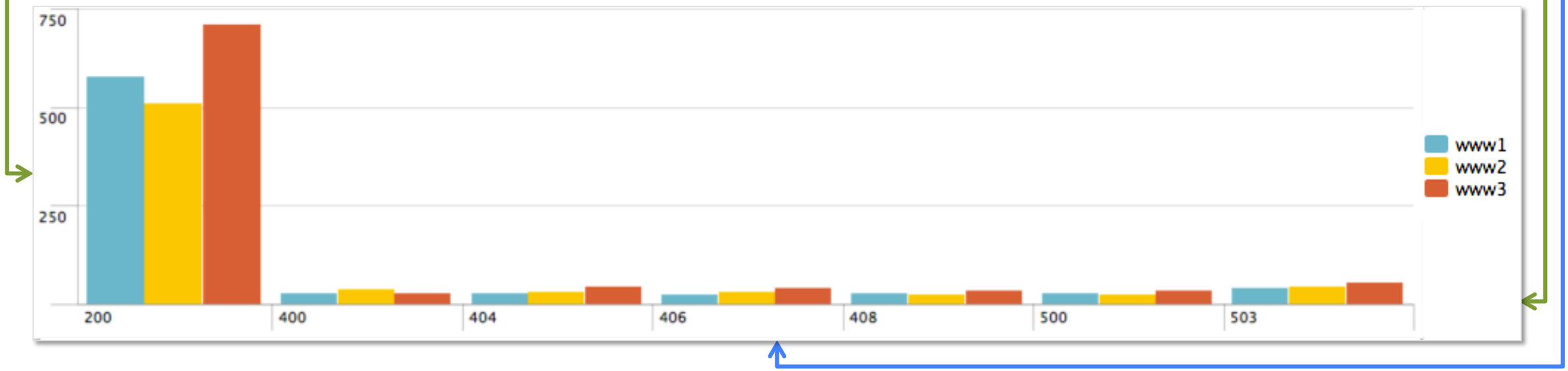
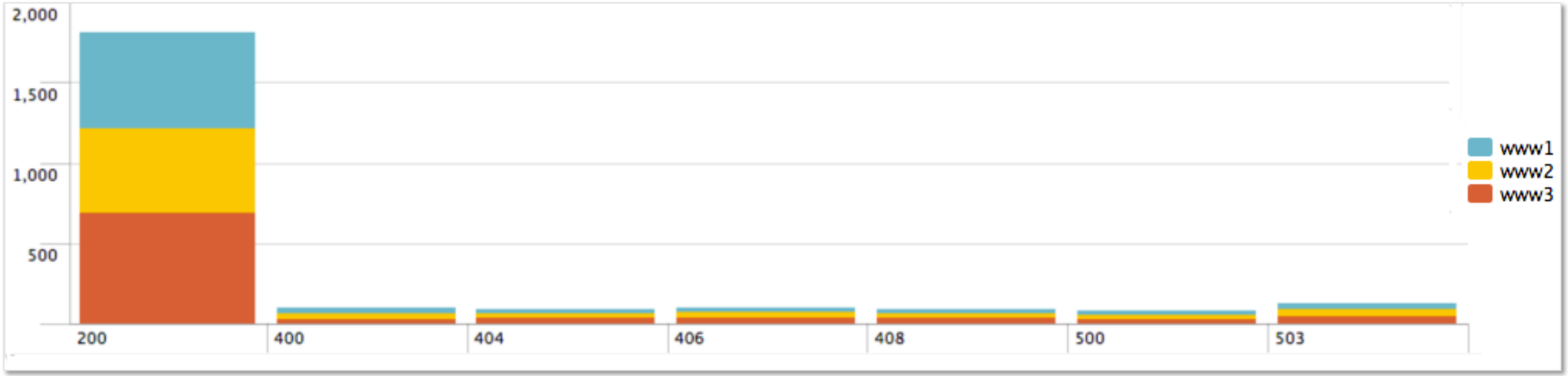


Chart – stacked mode

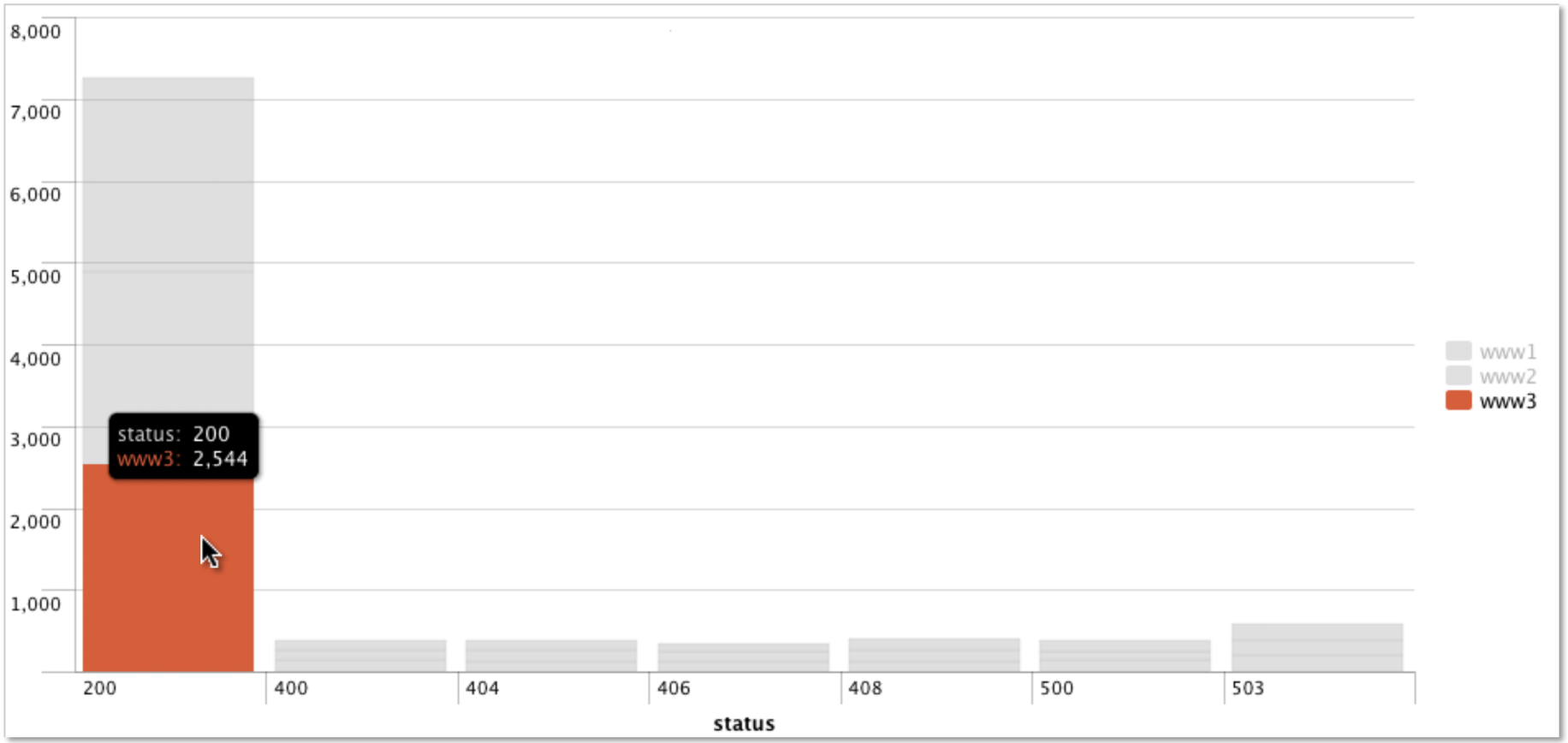
In Stacked mode, all split-by values are displayed in a single column


```
sourcetype=access_* | chart count over status by host
```



Hovering on segments

Hovering over a segment in the column displays the statistic for that segment

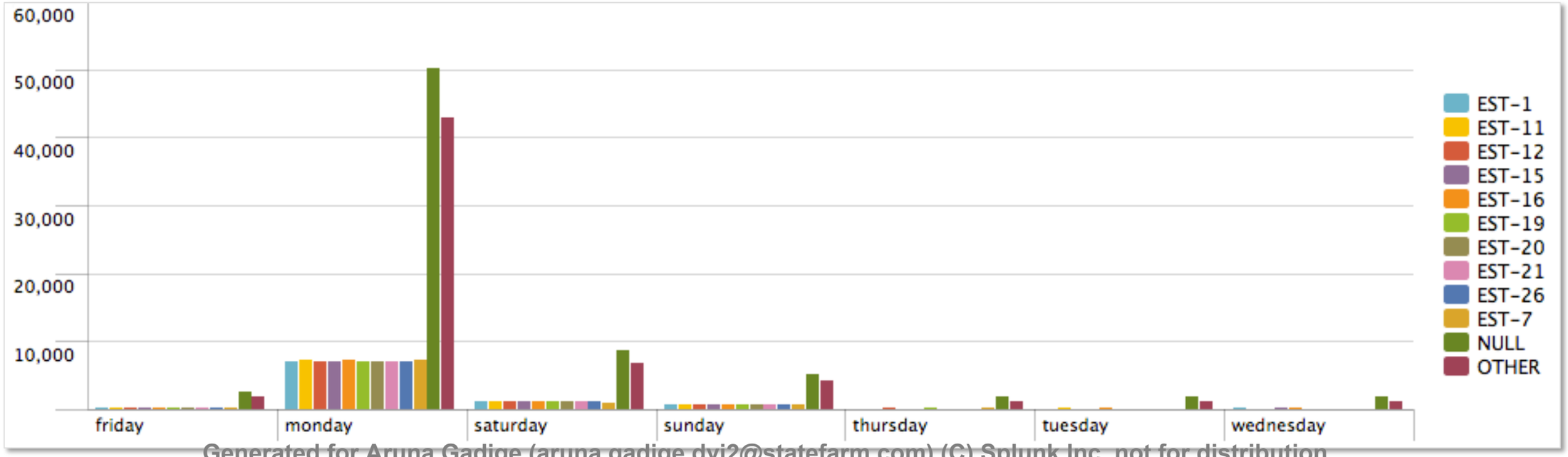


Omitting null

- This example charts the file types that use a lot of bandwidth
- Notice there are NULL and OTHER values that we do not want to show

How many items were active over the past 7 days?

```
sourcetype=access_combined | chart count over date_wday by itemId
```

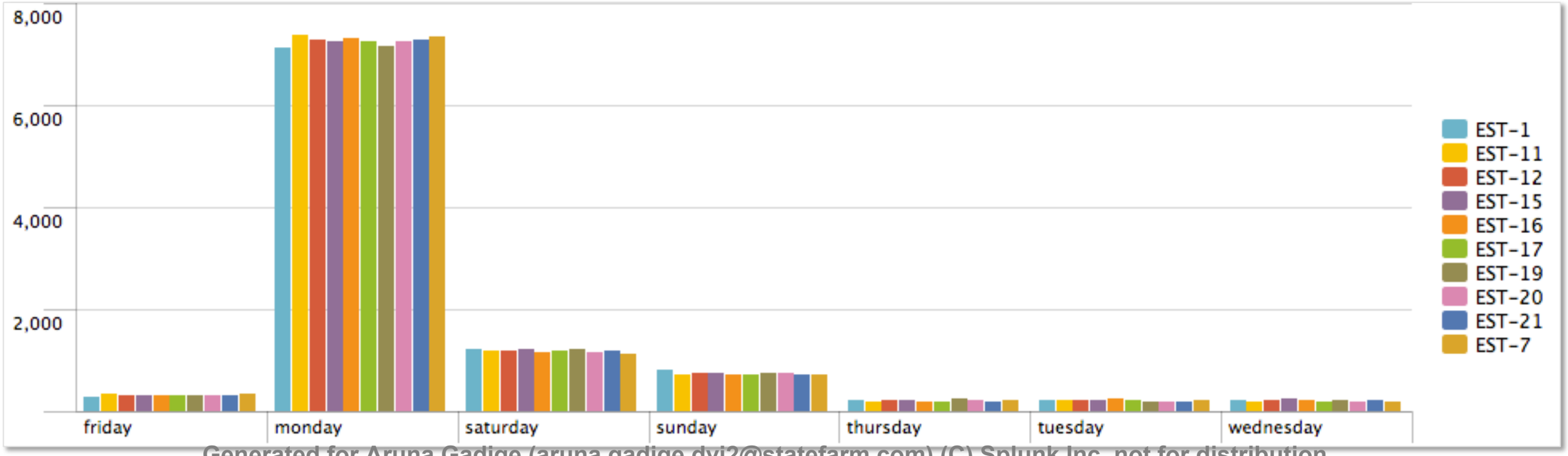


Omitting null and other (cont'd)

Adding the options `useother=f` and `usenull=f` removes the empty and **other** field values from the display

How many items were active over the past 7 days?

```
sourcetype=access_combined
| chart count over date_wday by itemId useother=f usenull=f
```



Timechart command overview

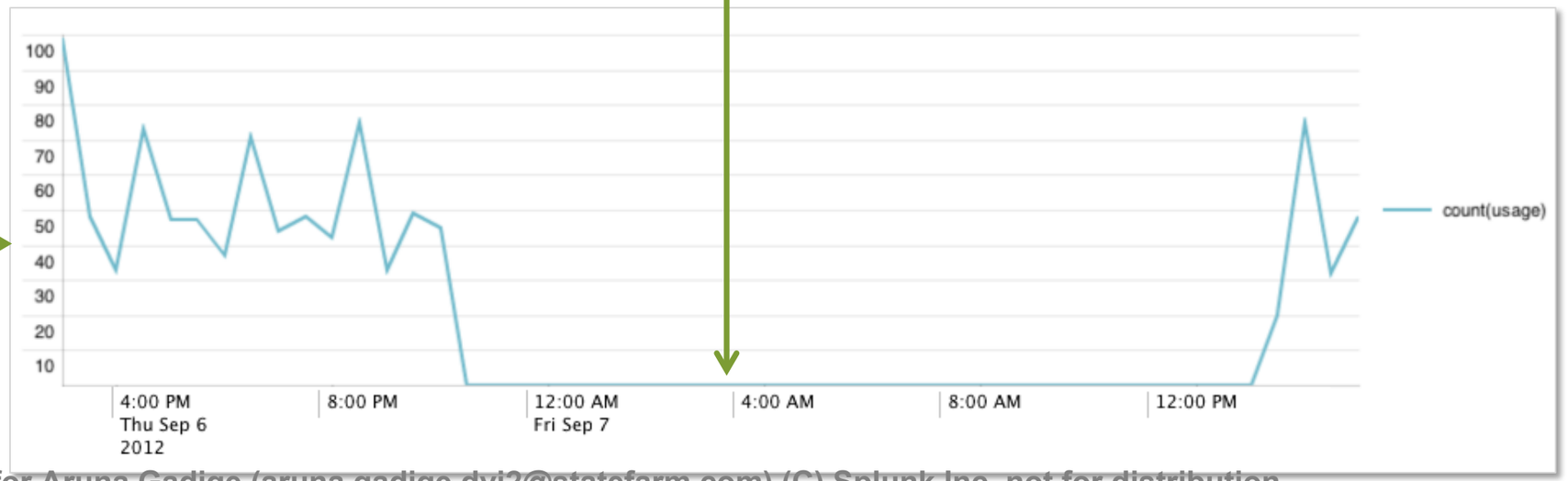
- Timecharts perform statistical aggregations against time
- Plot trends and find anomalies over time
- `_time` is always the x-axis
- You can optionally split data by another field
  - Each distinct value of the `"split by"` field is a separate series in the chart
- Timecharts are best represented as line or area charts

Basic timechart

- This basic timechart displays the number of usage violations over the last 24 hours
- Note: you used these functions and arguments with stats and chart

How many usage violations in the last 24 hours?

```
sourcetype=cisco_w* usage=Violation | timechart count(usage)
```

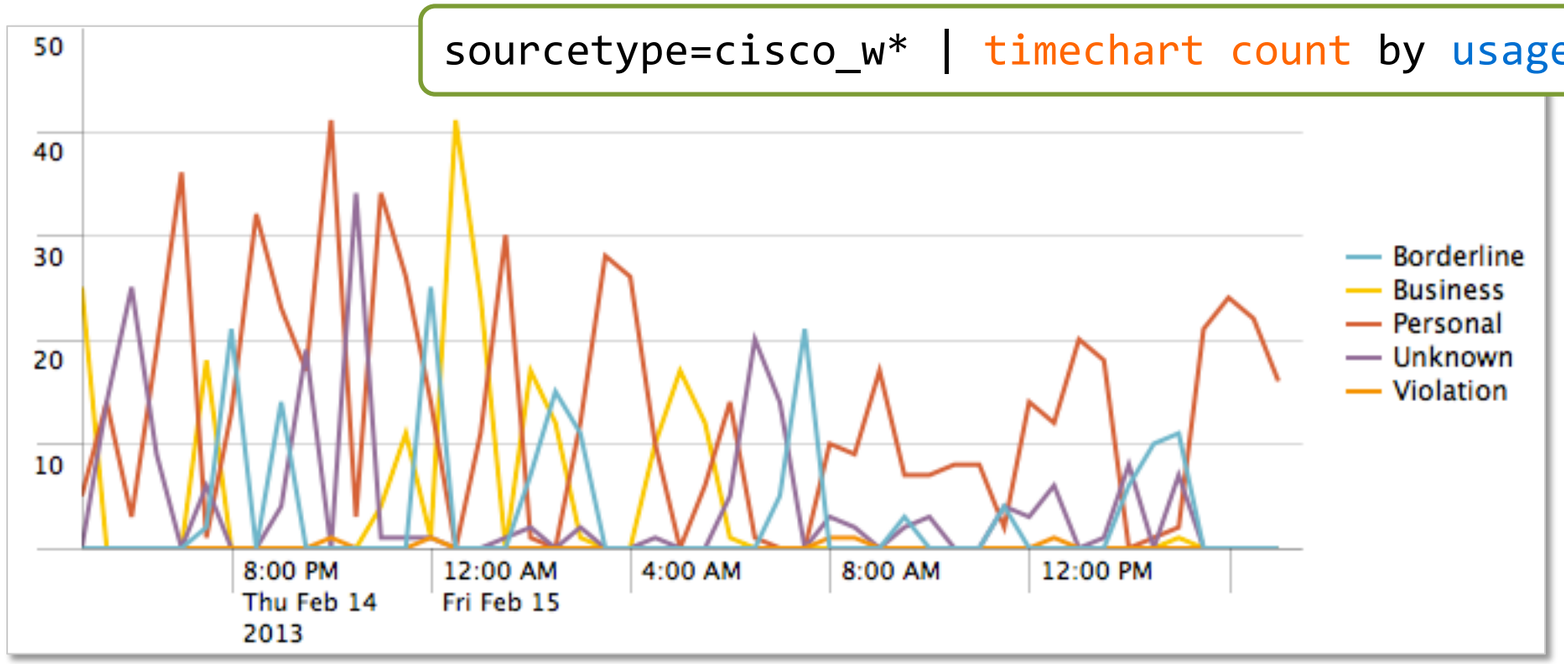


Charting multiple values

- This example displays the usage categories over a 24 hour period
- Splitting by the `usage` field, each line represents a unique field value
- y-axis represents the count for each field value

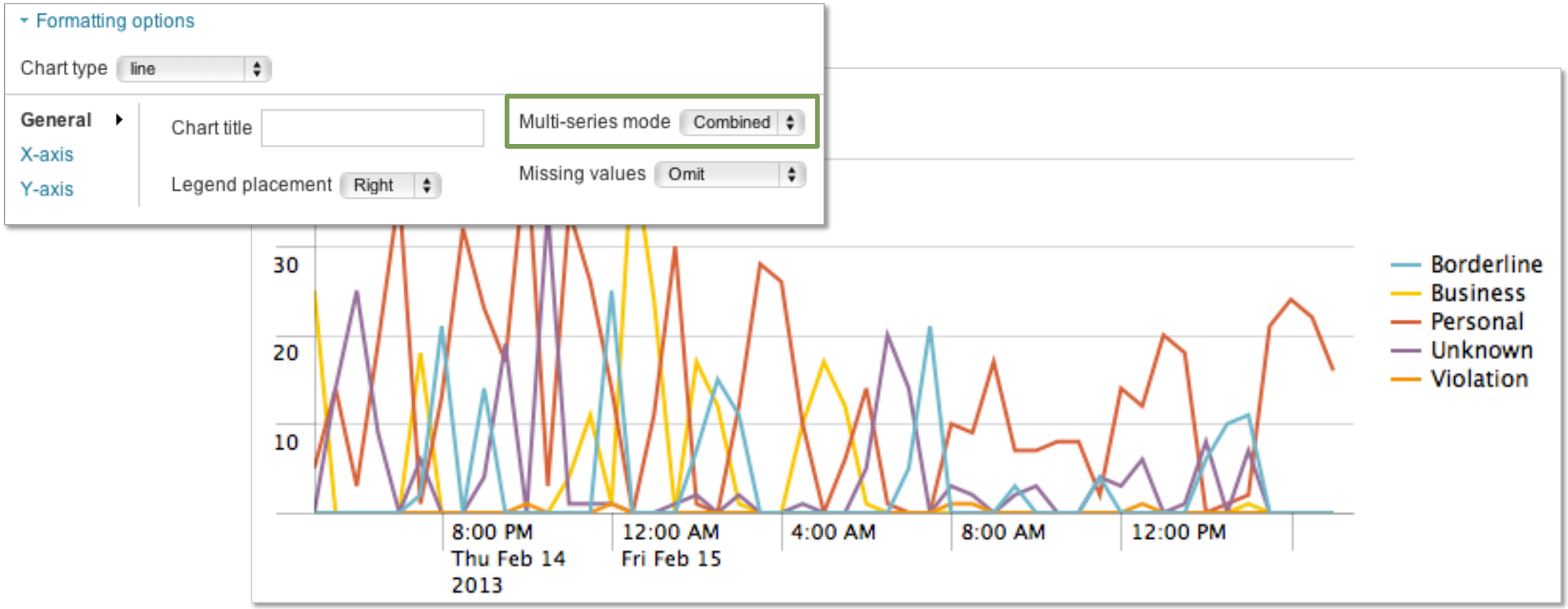
What's the overall usage trend for the last 24 hours?

```
sourcetype=cisco_w* | timechart count by usage
```



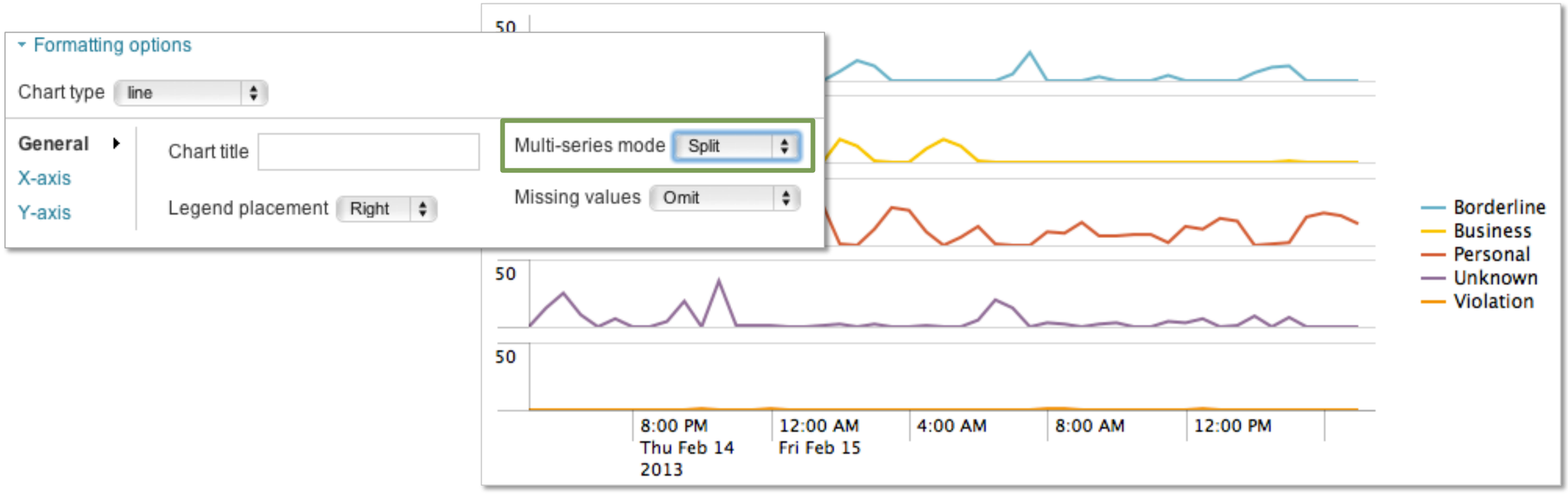
Formatting options

When the Multi-series mode set to Combined, all fields share the y-axis



Formatting options (cont'd)

- Setting the Multi-series mode to Split causes the y-axis to split for each field value
- y-axis is divided into sections, each spanning the max and min count

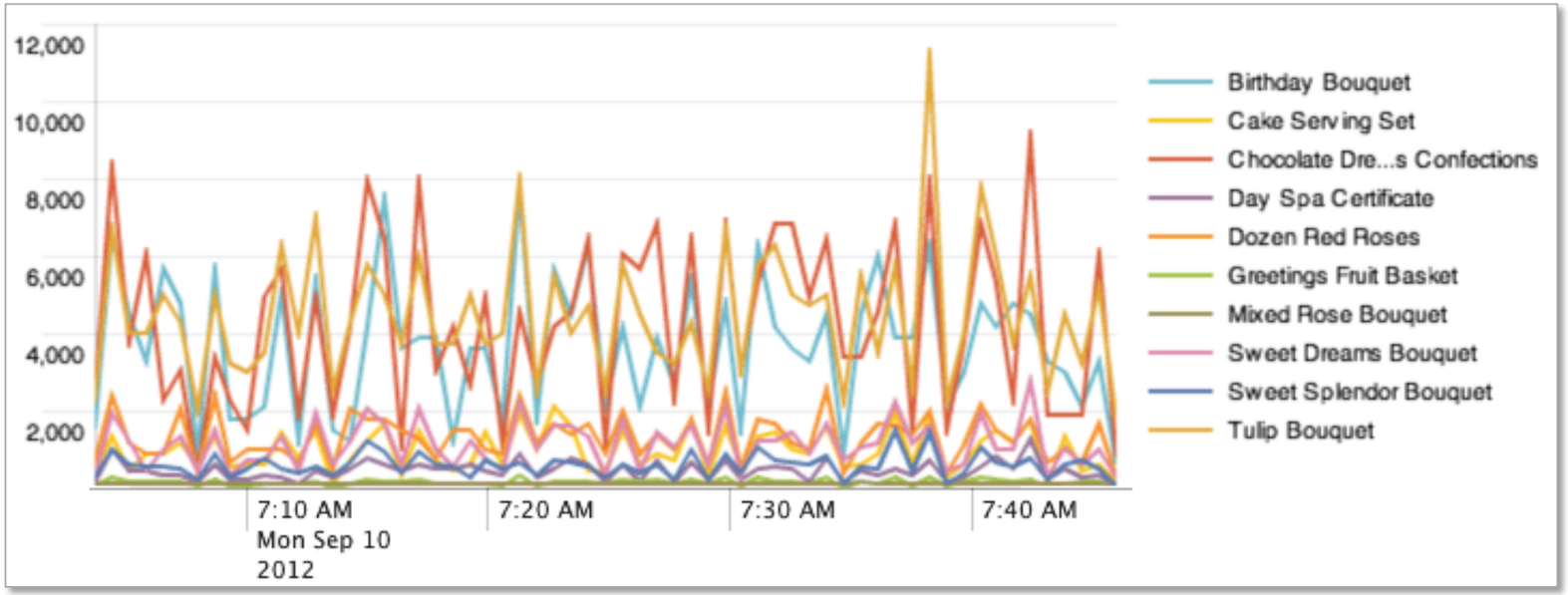


Applying statistical functions

- As with the `stats` and `chart` commands, you can apply statistical functions to the `timechart` command

During the last hour, how much revenue did we receive for each product?

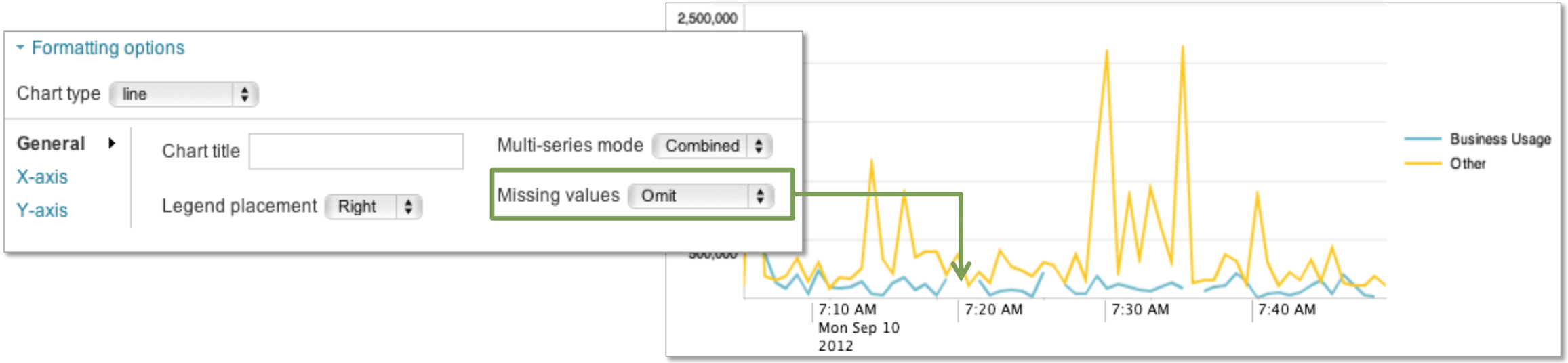
```
sourcetype=access_* action=purchase | timechart sum(price) by product_name
```



Handling missing values – Omit

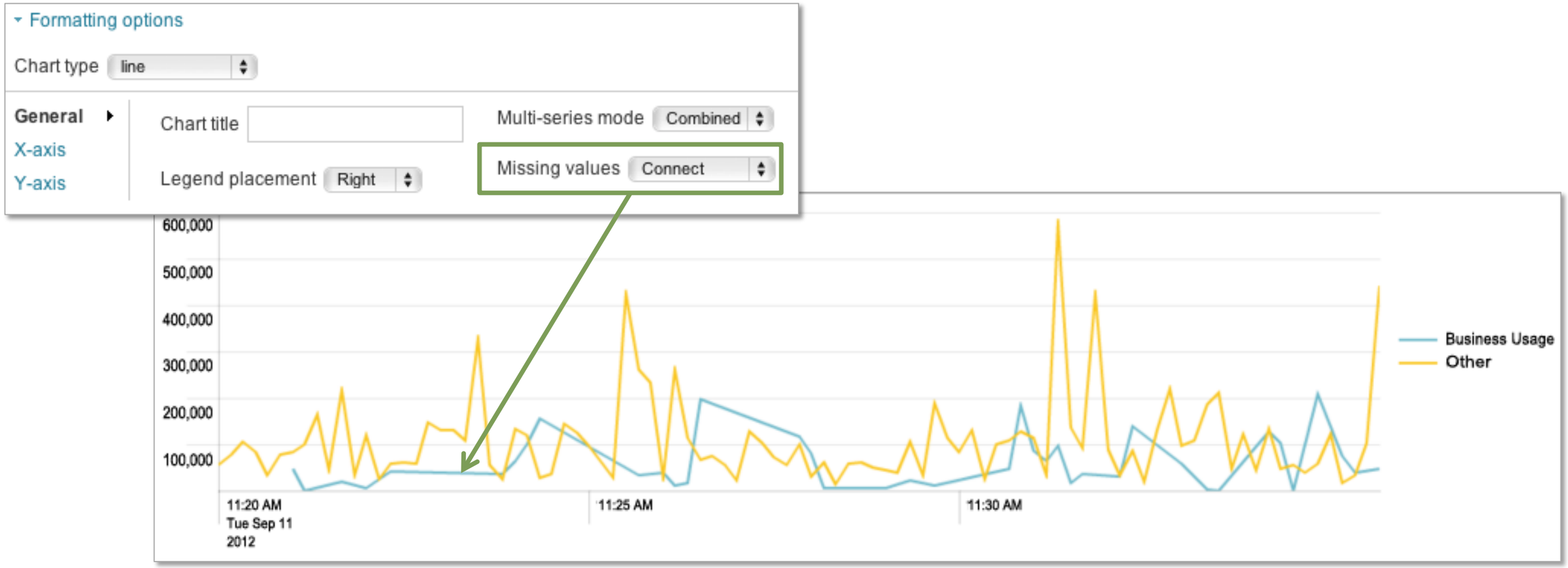
- Three options for handling missing values in a timechart – Omit, Connect, Treat as zero
- The Omit option displays gaps in the series

```
sourcetype=cisco_w* | eval usage = if(usage == "Business", "Business Usage", "Other")
| timechart max(sc_bytes) by usage
```



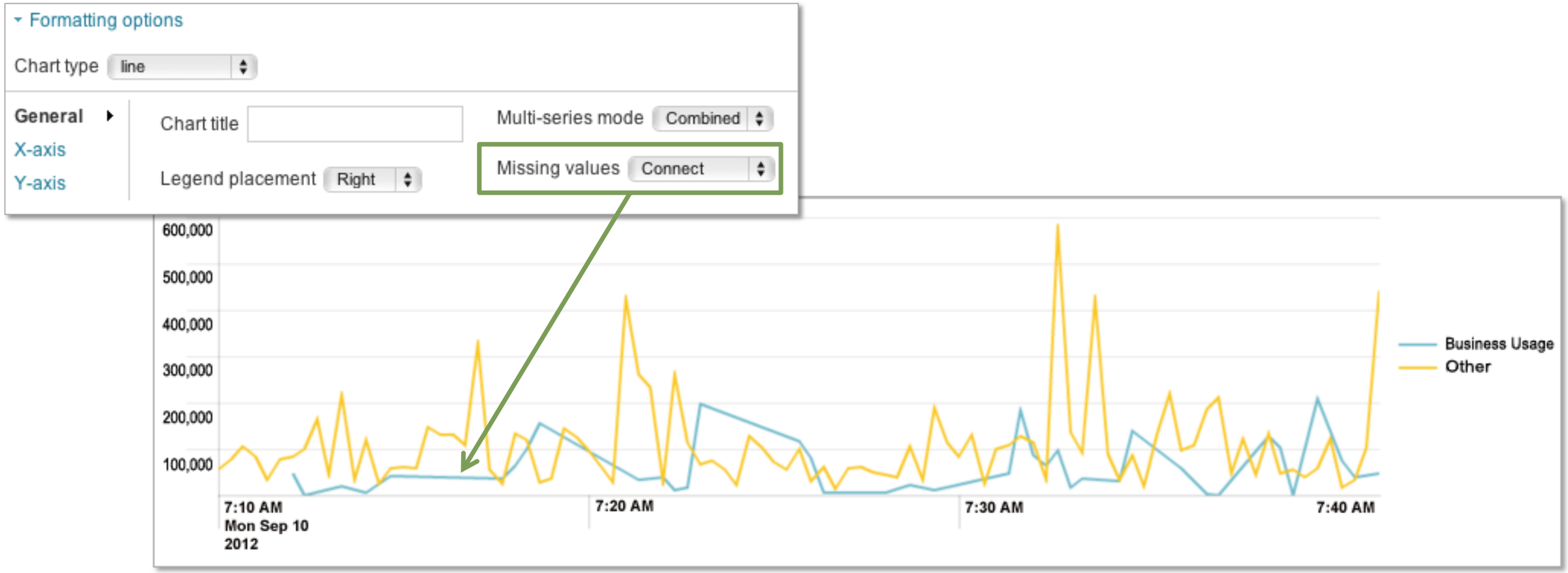
Handling missing values – Connect

- The Connect option fills the gaps with a trendline



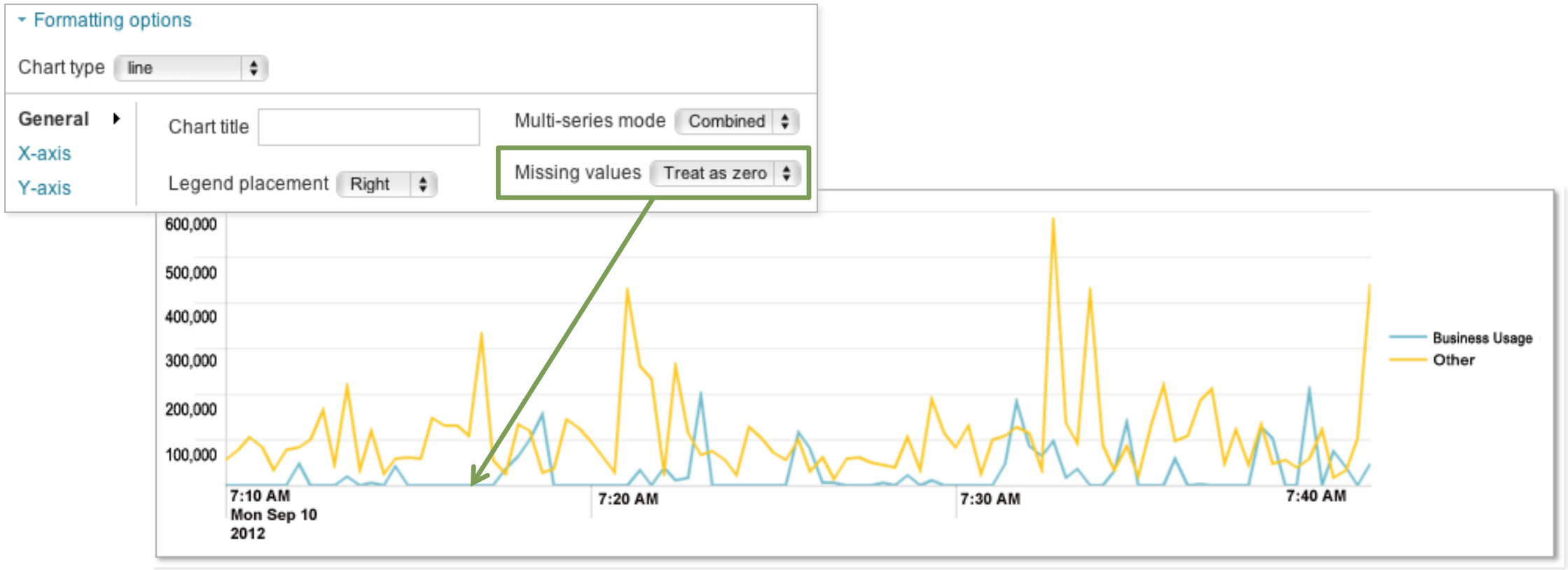
Handling missing values – Connect

- The Connect option fills the gaps with a trendline



Handling missing values – Treat as zero

- The Treat as zero option flattens the trendline to the zero value



Lab 4

- **Time:**
  - 20-25 minutes
- **Tasks:**
  - Create a basic column chart -
  - Create a multi-series chart, change its layout and save the search
  - Work with formatting options
  - Create a basic `timechart`
  - Add charts to a Dashboard

Section 5:
Correlating Events

Section objectives

- Identify transactions
- Group events using fields
- Group events using fields and time
- Search with transactions
- Report on transactions
- Determine when to use `transactions` vs. `stats`

Transaction overview

- A transaction is any group of conceptually related events that span time
- The events can come from multiple applications or hosts
  - Events related to a single purchase from an online store can span across an application server, database, e-commerce engine
  - A single email message can create multiple events as it travels through various queues

Basic transaction example

- Each event in the network web traffic logs represents a single user generating a single http request
- Visiting a single website normally generates multiple http requests
  - HTML, JavaScript, CSS files
  - Images
  - …

```
sourcetype=access_combined
```

1 11/2/12 2:33:08.000 PM
27.112.101.10 - - [02/Nov/2012:21:33:08] "GET /product.screen?productId=FI-FW-02&JSESSIONID=SD1SL4FF6ADFF4956 HTTP 1.1" 200 1848 "http://www.myflowershop.com/oldlink?itemId=EST-7" "Googlebot/2.1 (http://www.googlebot.com/bot.html) " 838
host=www2 | sourcetype=access_combined | source=/opt/log/www2/access.log

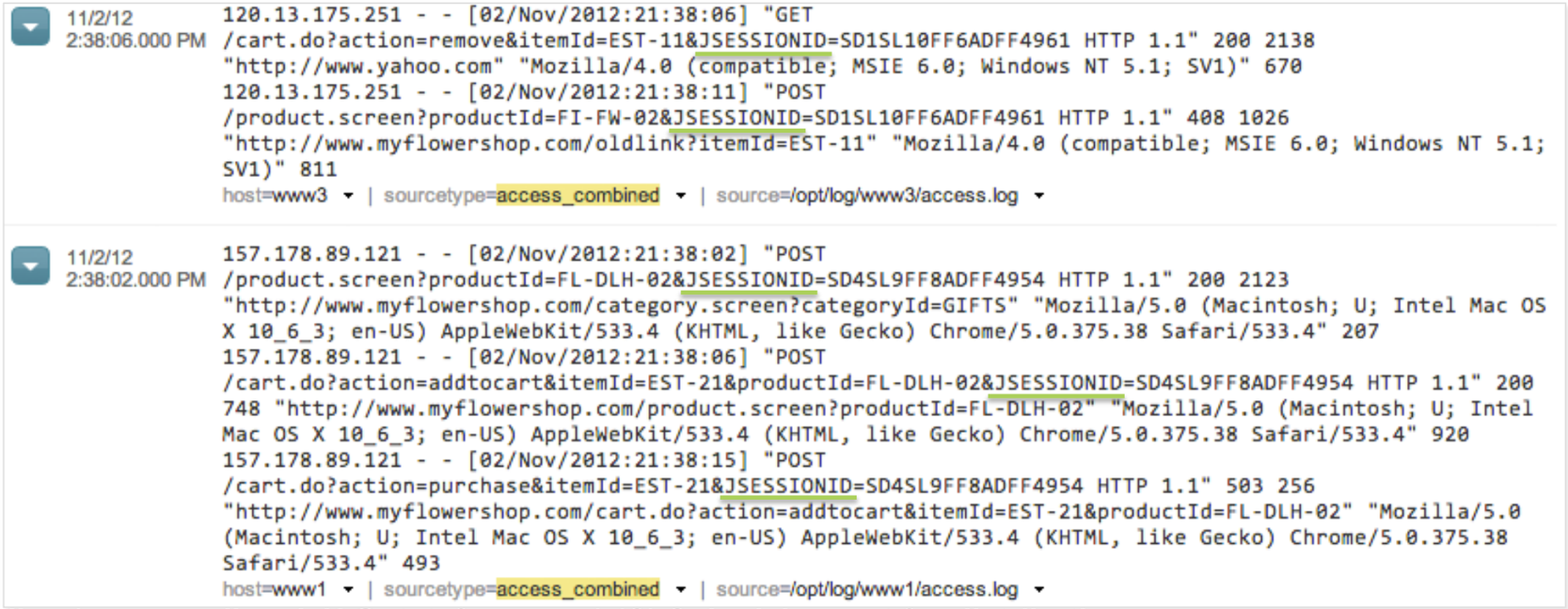
2 11/2/12 2:33:06.000 PM
131.93.29.242 - - [02/Nov/2012:21:33:06] "GET /oldlink?itemId=EST-11&JSESSIONID=SD1SL6FF7ADFF4961 HTTP 1.1" 200 3962 "http://www.myflowershop.com/product.screen?productId=RP-SN-01" "Opera/9.01 (Windows NT 5.1; U; en)" 252
host=www2 | sourcetype=access_combined | source=/opt/log/www2/access.log

3 11/2/12 2:33:06.000 PM
203.27.185.47 - - [02/Nov/2012:21:33:06] "GET /category.screen?categoryId=GIFTS&JSESSIONID=SD6SL9FF3ADFF4953 HTTP 1.1" 200 3384 "http://www.myflowershop.com/oldlink?itemId=EST-15" "Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_6_3; en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.38 Safari/533.4" 750
host=www2 | sourcetype=access_combined | source=/opt/log/www2/access.log

Group by fields

- Using the `transaction` command, you can group and create a single event when certain fields have the same value
- You can create transactions across multiple tiers (i.e., web server, application server) provided a **common field** is shared (`JSESSIONID`)

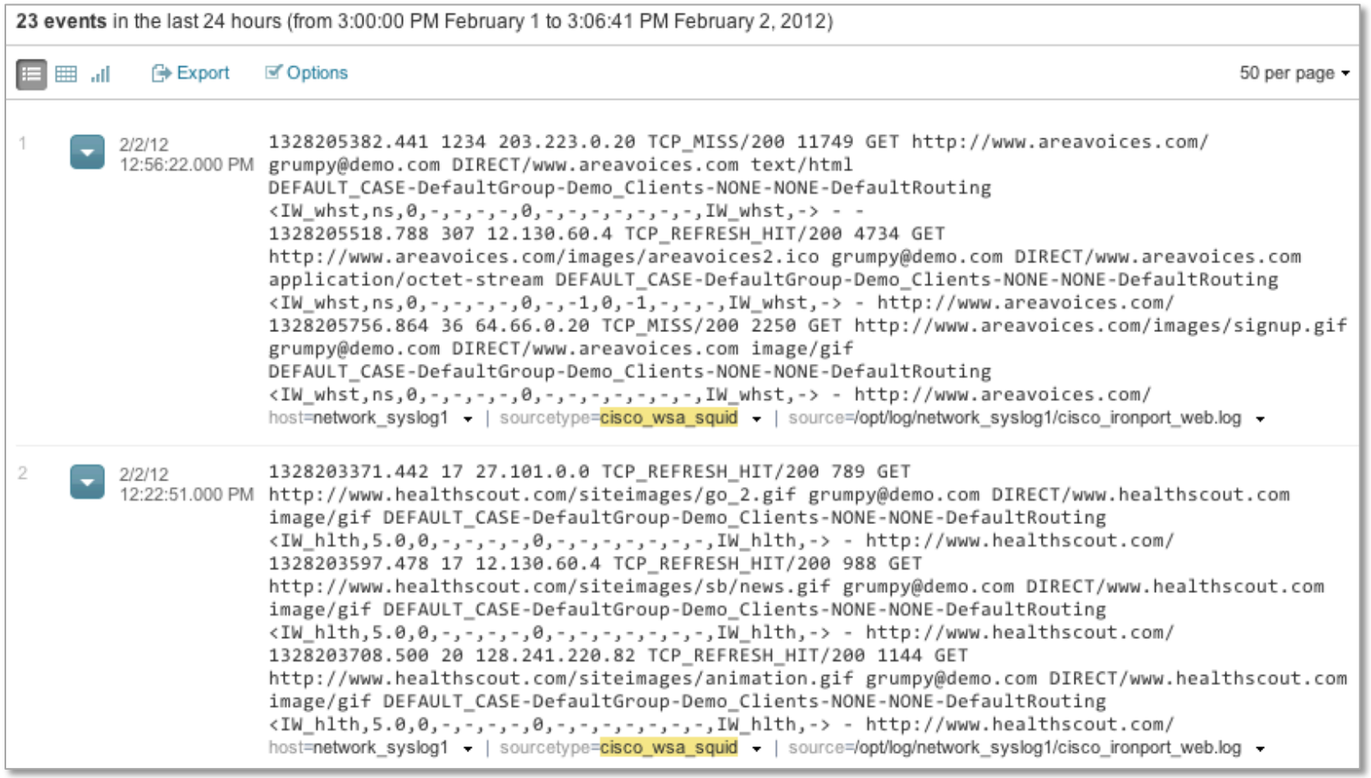
```
sourcetype=access_combined
| transaction JSESSIONID
```



Group by fields and time

- You can also define a max overall time span and max gap between events
  - `maxspan=20m`
    - Maximum total time between the *earliest* and *latest* events
    - If not specified, default is -1 (or no limit)
  - `maxpause=5m`
    - Maximum total time *between* events
    - If not specified, default is -1 (or no limit)

```
sourcetype=cisco_w*
| transaction s_hostname cs_username maxspan=20m maxpause=5m
```



Group by startswith / endswith

- You can use the `startswith` and `endswith` options to form transactions based on terms, field values, or evaluations
- In this example, the first event in the transaction includes `addtocart` and the last event includes `purchase`

```
sourcetype=access_* | transaction clientip startswith=eval(action="addtocart") endswith=eval(action="purchase")
```

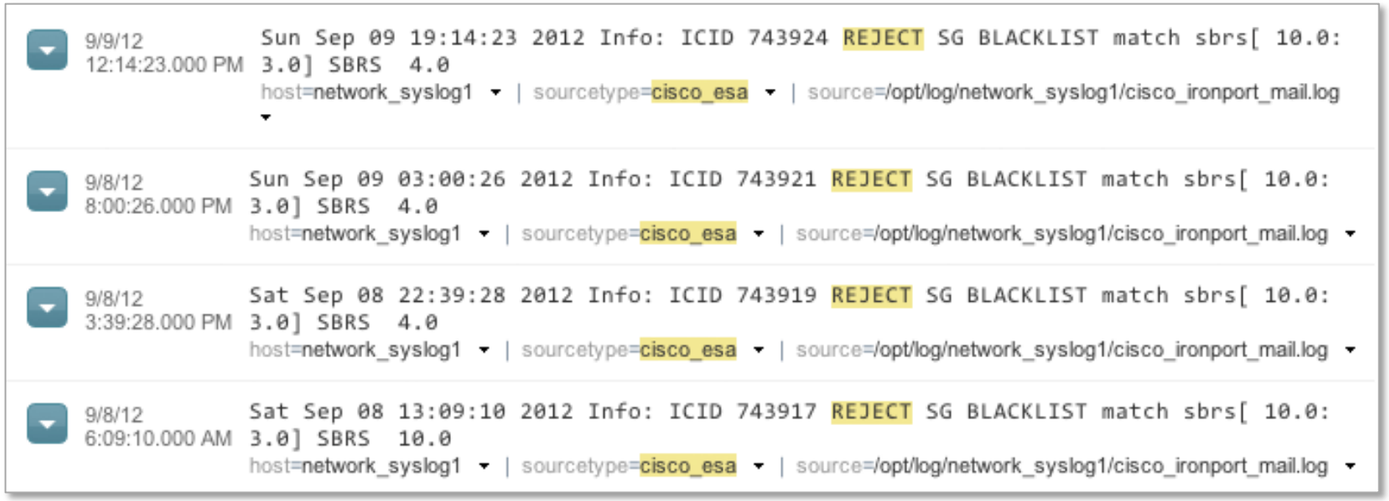
Transaction-specific fields

- The transaction command produces some additional fields, *duration* and *eventcount*.
  - The *duration* value is the difference between the timestamps for the first and last event in the transaction.
  - The *eventcount* value is the number of events in the transaction.

Investigating with transactions

- Transactions can be useful when a single event may not provide enough information
- This example searches email logs for the term “REJECT”
- Events that include the term don’t provide much information about the rejection

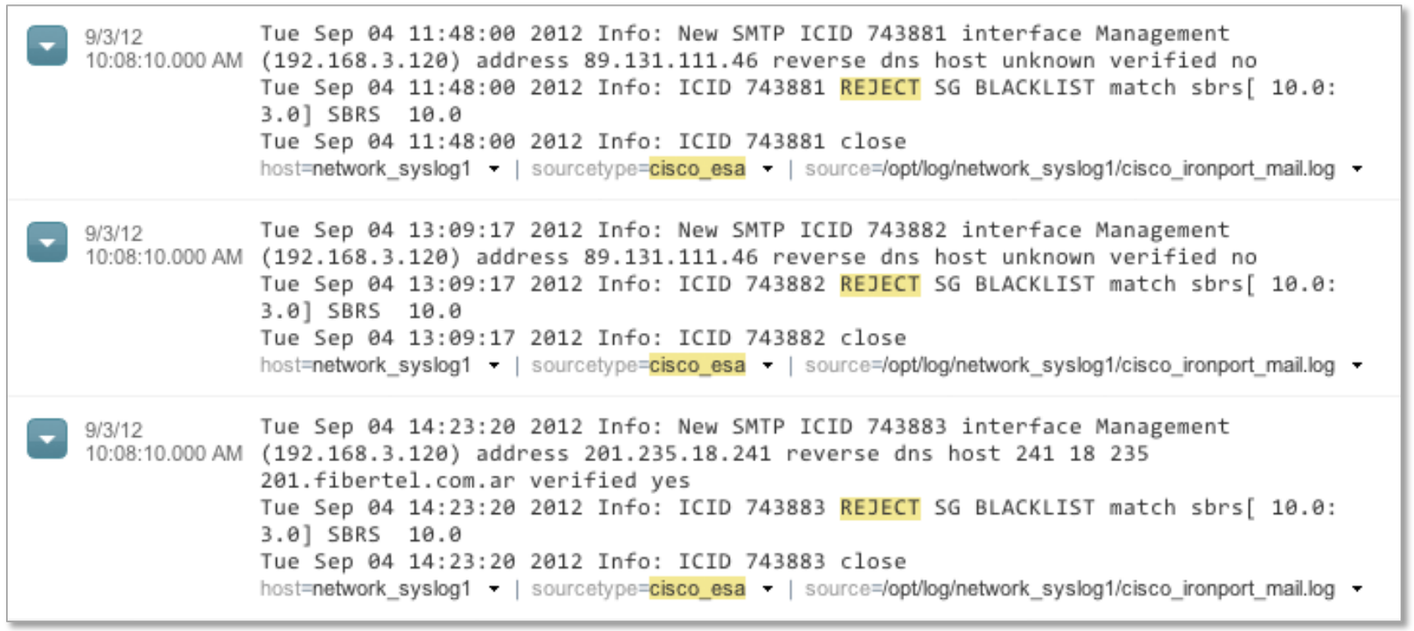
```
sourcetype=cisco_esa REJECT
```



Investigating with transactions (cont'd)

- By creating a transaction, we can then search and see additional events related to the rejection
  - IP address of sender
  - Reverse DNS lookup results
  - Action taken by the mail system following the rejection

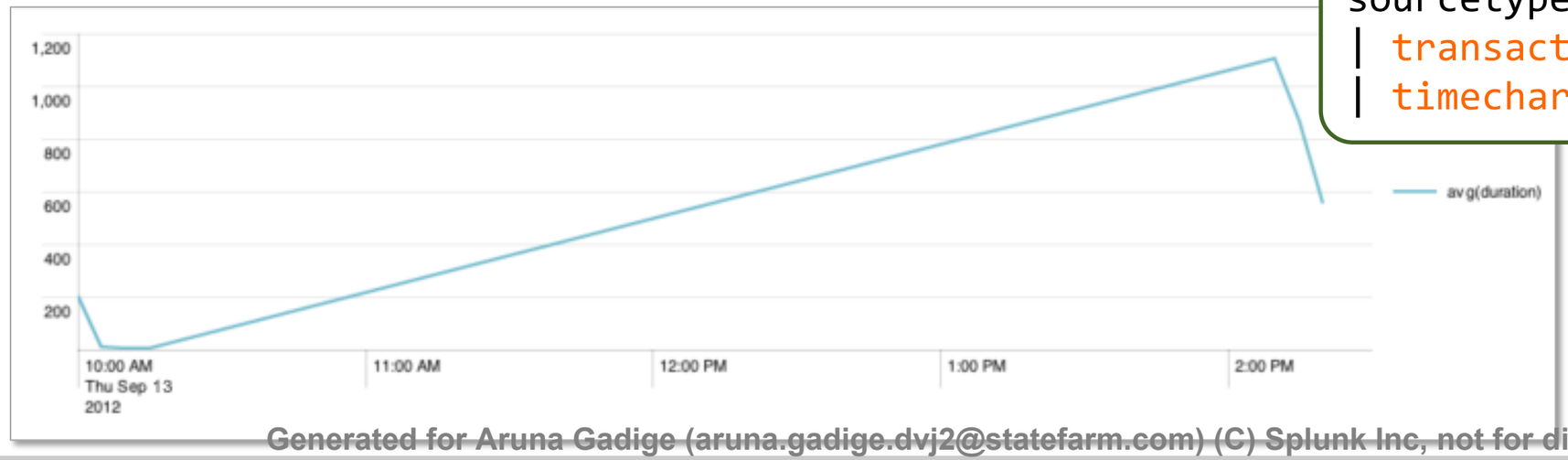
```
sourcetype=cisco_e* | transaction mid dcid icid | search REJECT
```



Reporting on transactions

- You can use the same statistics and reporting commands with transactions
- This example takes advantage of the `duration` field
  - It shows a trend of the mail queue slowing over 4 hours, then correcting
  - Adding events to the transaction from additional hosts or sources would uncover the cause of the slowdown

```
sourcetype=cisco_e*
| transaction mid dcid icid
| timechart avg(duration)
```



Transaction vs. Stats

- Use `transaction` when you need to see the events correlated together
- Use `stats` when you just want to see the results of a calculation
- Use `transaction` when you must define event grouping based on `start / end` values
- Use `stats` when you can group events based on a field value (e.g. "by `src_ip`")
- When you can do it either way, choose `stats` because it is more efficient

```
sourcetype=trade_entries | transaction TradeID | table TradeID, eventcount
```

vs.

```
sourcetype=trade_entries | stats count by TradeID
```


Lab 5

- **Time:**
  - 15 minutes
- **Tasks:**
  - Create a transaction using common fields
  - Create a transaction using common field values, maxspan, and maxpause

* Note: You may need to expand time on `sourcetype=cisco_esa` to the **Last 24 hours**.

Section 6:

Enriching Data with Lookups and Workflow Actions

Section objectives

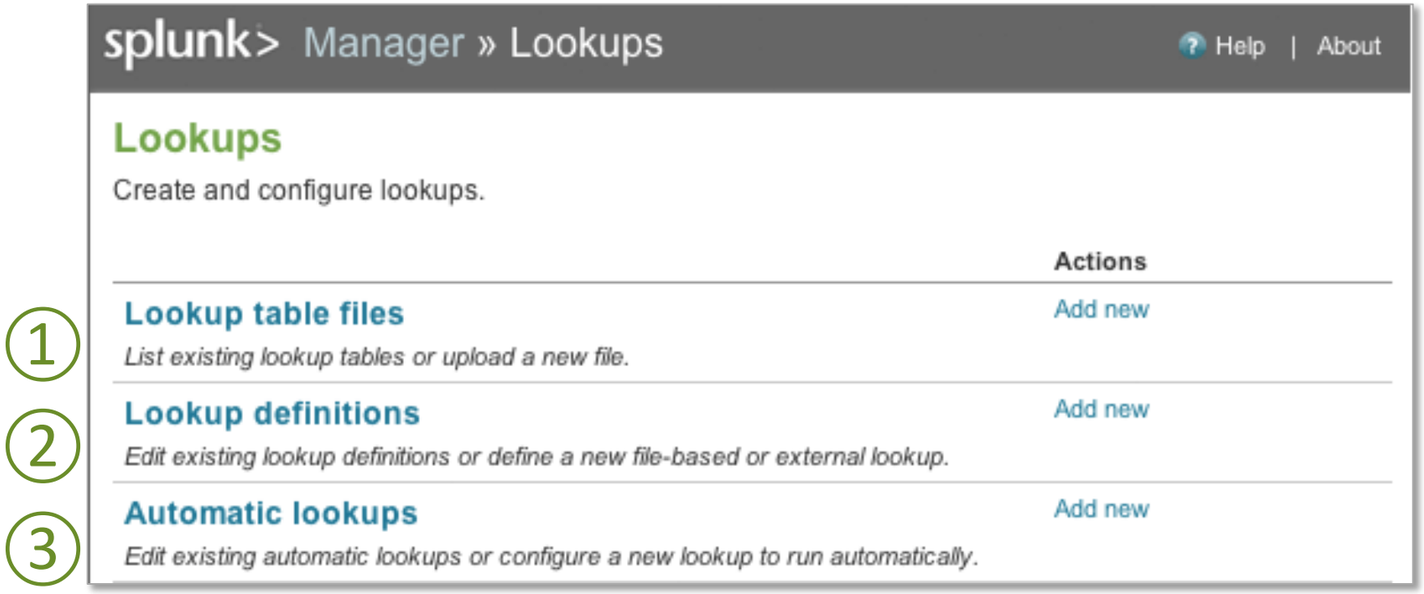
- Describe lookups
- Examine a lookup file example
- Create a lookup table
- Define a lookup
- Configure an automatic lookup
- Configure a time-based lookup
- Use the lookup in searches and reports
- Add a workflow action

Lookups

- Some data should not be indexed within Splunk
  - Static or relatively unchanging data
- Lookups allow you to add more fields to your events:
  - http status code descriptions (“file not found”, “service unavailable”)
  - Descriptive text descriptions for errors or item IDs
  - User names based on asset tags or static IP addresses
- Two types of lookups
  - File lookup
  - Scripted lookup

Define a file lookup

1. Create the lookup table
2. Define the lookup
3. Optionally, configure the lookup to run automatically



Lookup file example

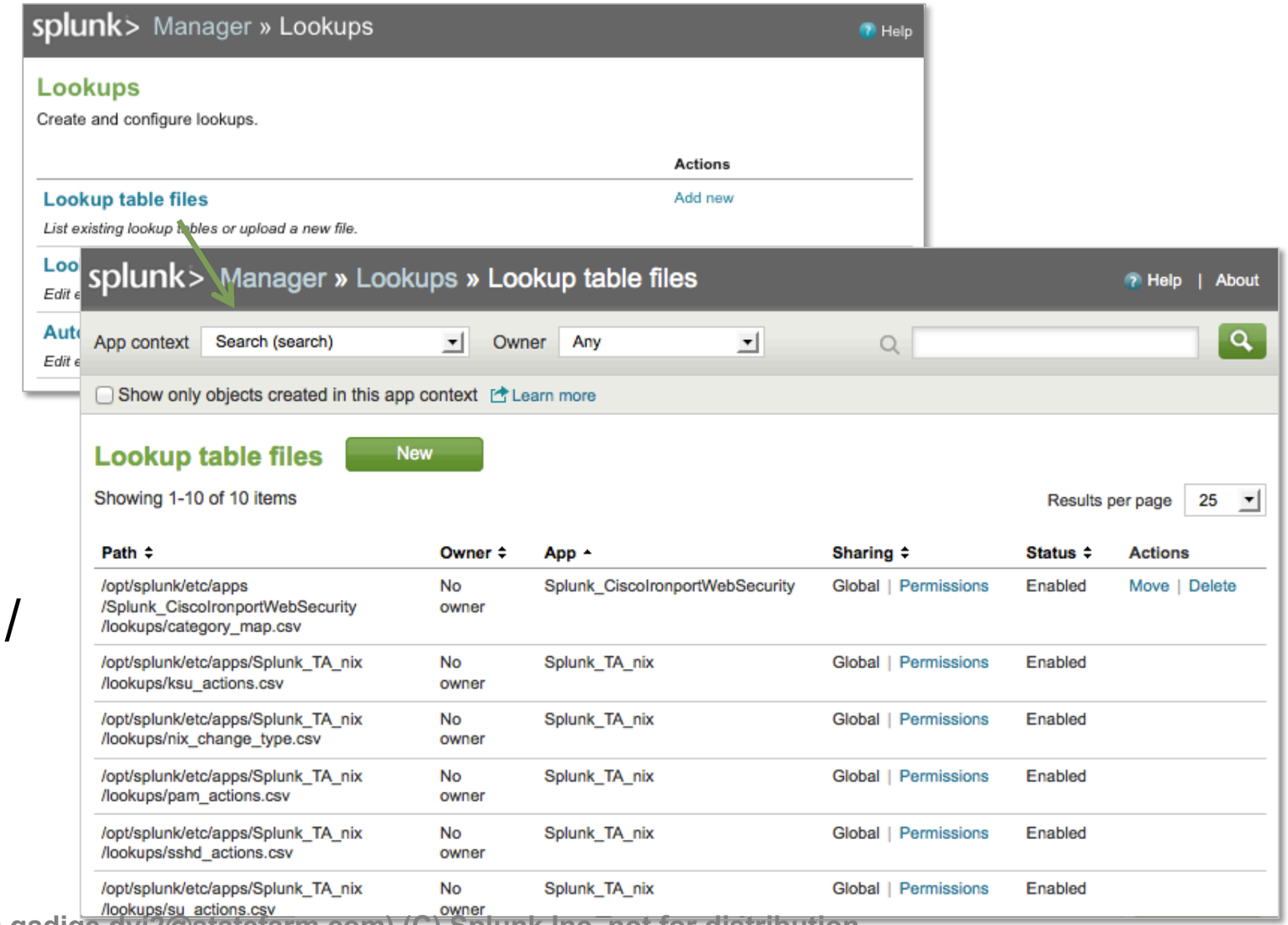
- This example displays a lookup `.csv` file used to associate a product name, price, and competitor's prices with a product ID
- First row represents field names (header)
- The `productId` field exists in the `access_combined` events
  - This is the **input** field
- `product_name`, `price`, `flowersdotcom_price`, and `flowersrus_price` fields will be available to search after the lookup is defined
  - These are the **output** fields

prod_lookup.csv

```
productId,product_name,price,flowersdotcom_price,flowersrus_price
AV-CB-01,Tulip Bouquet,250,269,279
AV-SB-02,Mixed Rose Bouquet,15,19,21
FL-DSH-01,Sweet Splendor Bouquet,49,55,55
FL-DLH-02,Dozen Red Roses,99,129,149
K9-BD-01,Birthday Bouquet,299,355,339
K9-CW-01,Sweet Dreams Bouquet, 89,99,92
FI-SW-01,Cake Serving Set,89,79,85
FI-FW-02,Greetings Fruit Basket,12,8,6
RP-LI-02,Chocolate Dreams Confections,379,299,319
RP-SN-01,Day Spa Certificate,35,39,40
```

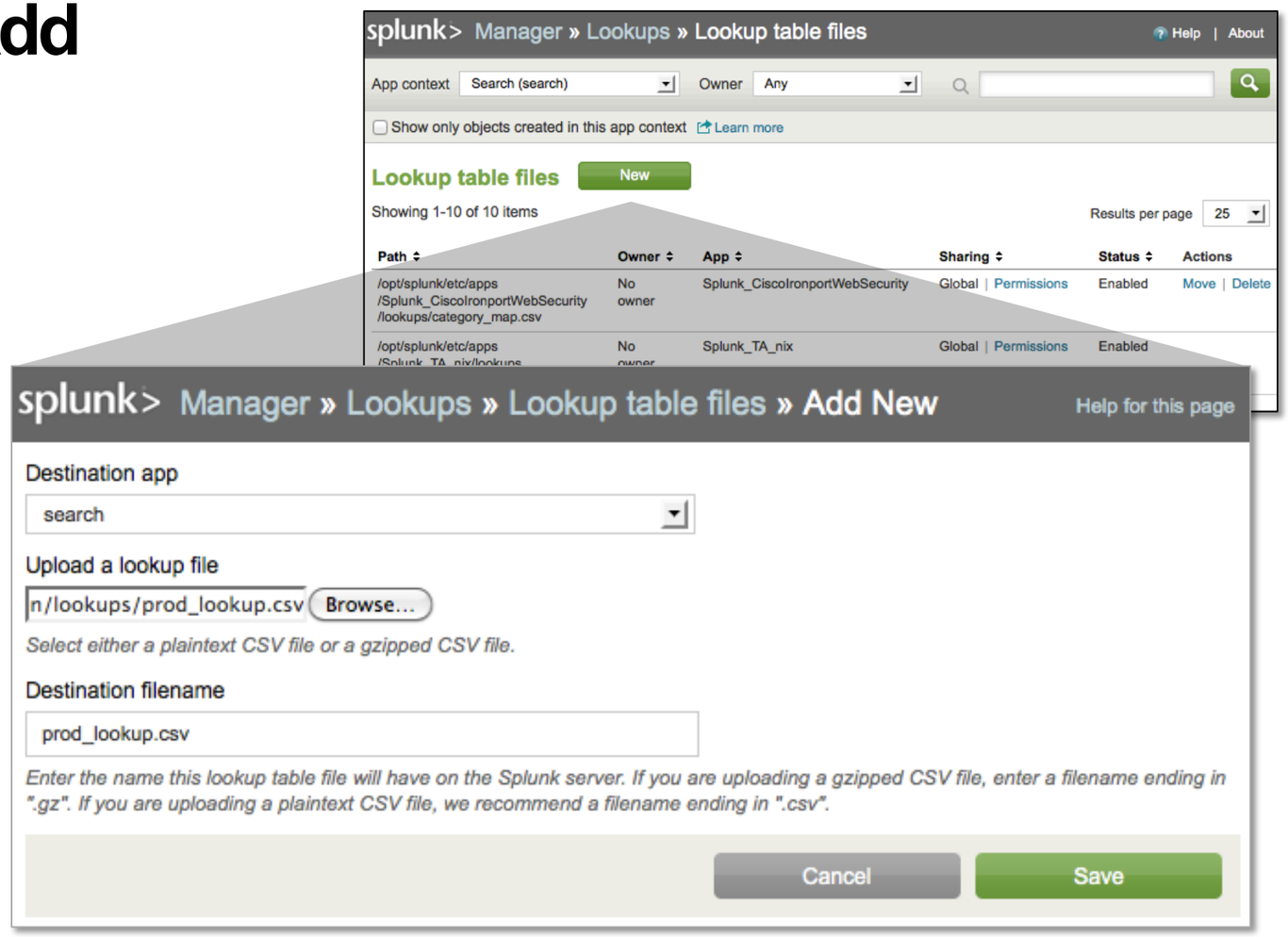
1. Create a lookup table

- The first step in creating a lookup is to add (upload) the lookup table
- **Manager >Lookups> Lookup table files** lists all existing lookup tables
  - From this list you can edit, change permissions, enable / disable, and delete



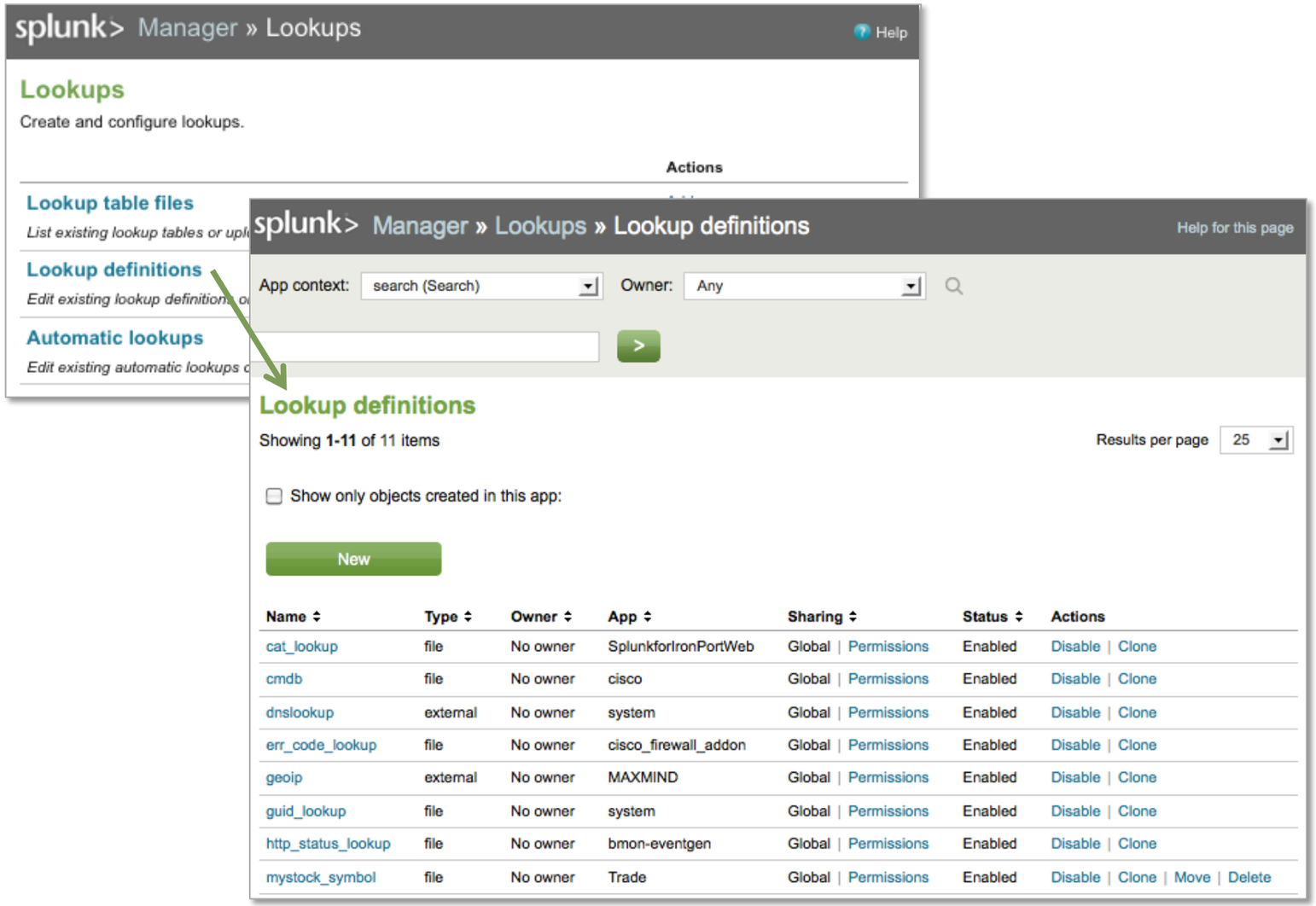
1. Create a lookup table (cont'd)

1. Click **New** to display the **Add New** form
2. Select a Destination app
3. Browse and select the `.csv` file to use for the lookup table
4. Enter a name for the lookup file
5. Save



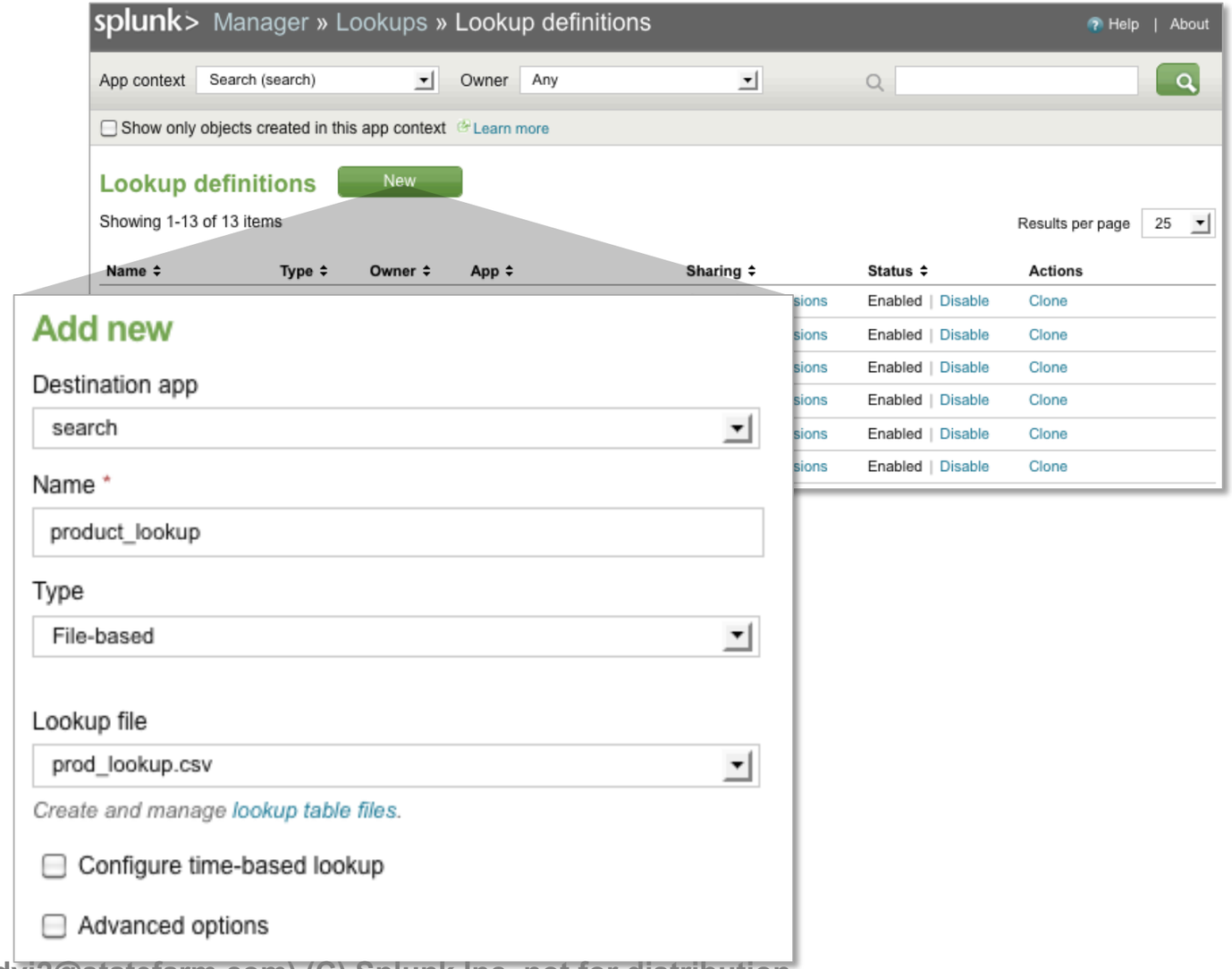
2. Define the lookup

- **Manager > Lookups > Lookup definitions** lists all existing lookup definitions
  - From this list you can edit, change permissions, enable / disable, delete, and clone



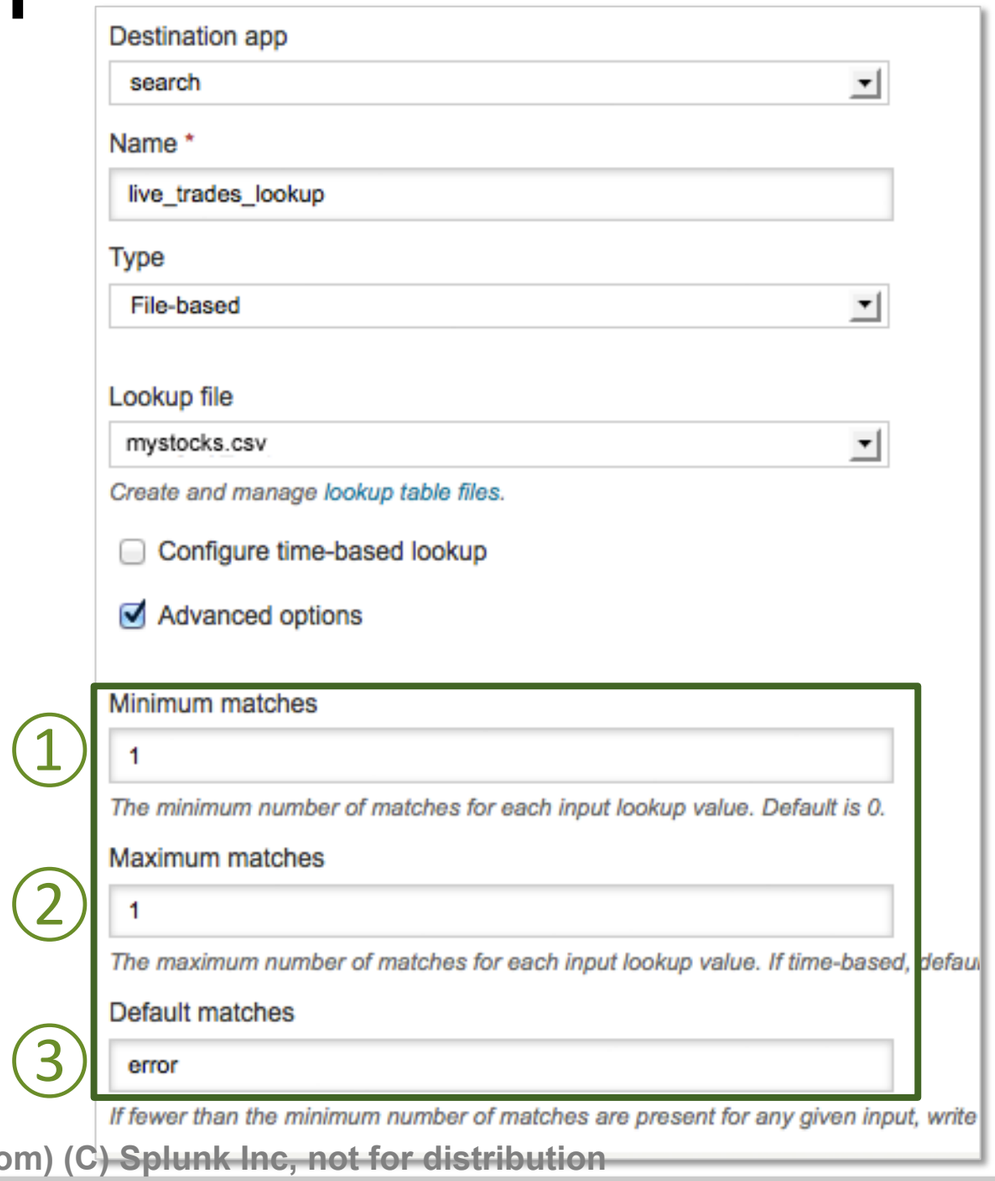
2. Define the lookup (cont'd)

1. Click **new** to display the **Add New** form
2. Select a Destination app
3. Enter a name for the lookup definition
4. Select the `.csv` file to use with the definition



Advanced Options

- Under **Advanced options**, you can specify:
  1. Minimum number of matches for each input lookup value
  2. Maximum number of matches for each input lookup value
  3. Default value to output, if fewer than the minimum number of matches are present for a given input



Verify the lookup

- After you've created the lookup definition, you can verify the data using the `inputlookup` command

```
| inputlookup product_lookup
```

	flowersdotcom_price	flowersrus_price	price	productId	product_name
1	269	279	250	AV-CB-01	Tulip Bouquet
2	19	21	15	AV-SB-02	Mixed Rose Bouquet
3	55	55	49	FL-DSH-01	Sweet Splendor Bouquet
4	129	149	99	FL-DLH-02	Dozen Red Roses
5	355	339	299	K9-BD-01	Birthday Bouquet
6	99	92	89	K9-CW-01	Sweet Dreams Bouquet
7	79	85	89	FI-SW-01	Cake Serving Set
8	8	6	12	FI-FW-02	Greetings Fruit Basket
9	299	319	379	RP-LI-02	Chocolate Dreams Confections
10	39	40	35	RP-SN-01	Day Spa Certificate

Using lookup manually

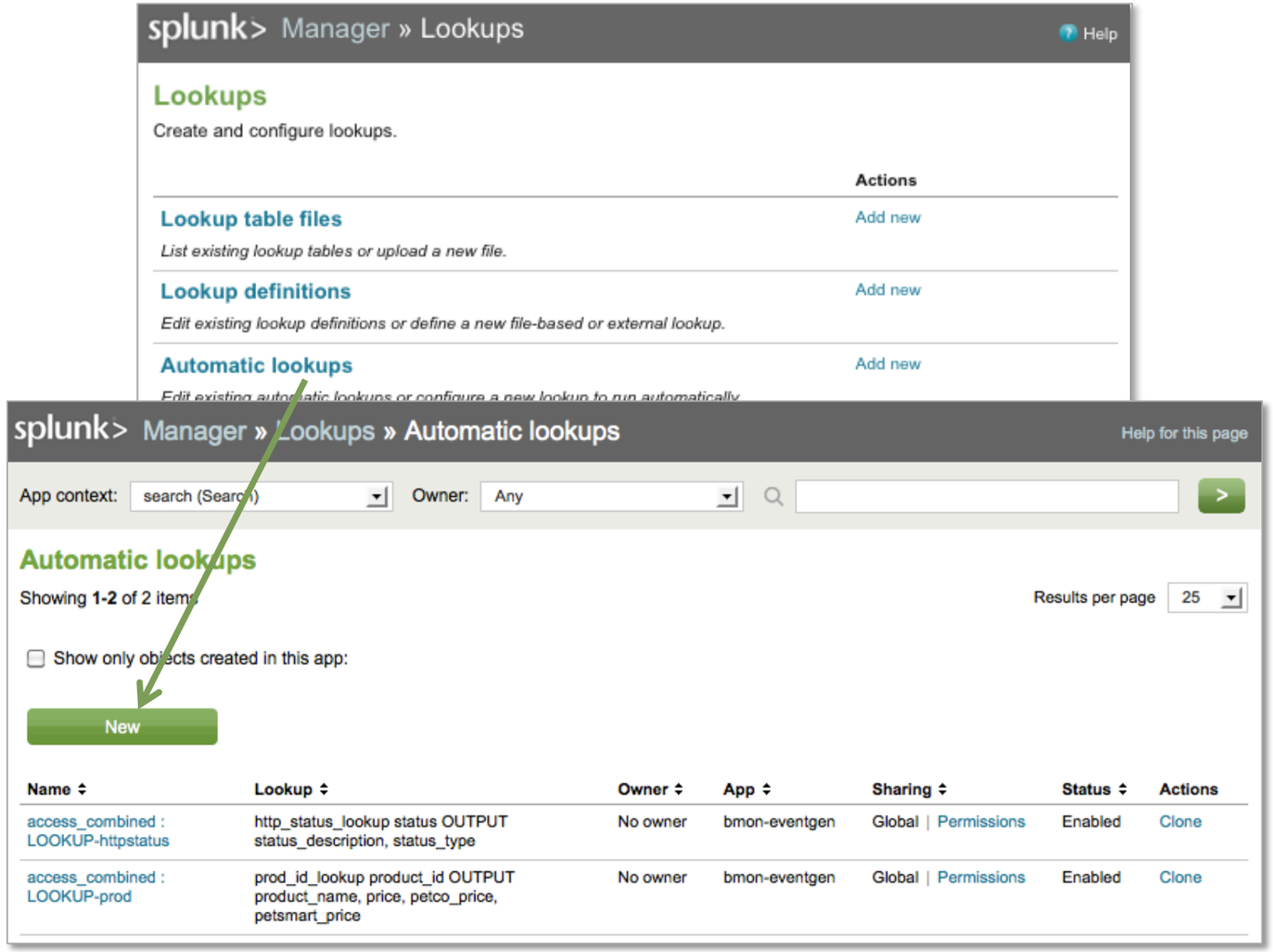
- If a lookup is not configured to run automatically, use the `lookup` command in your search to use the lookup fields
- Format is: `| lookup <lookup_name> <input field> OUTPUT <output field>, <output field>`

```
sourcetype=access_*
| lookup product_lookup productId OUTPUT product_name, price
| stats sum(price) by product_name
```

	product_name	sum(price)
1	Birthday Bouquet	11960
2	Cake Serving Set	3560
3	Chocolate Dreams Confections	17813
4	Day Spa Certificate	1155
5	Dozen Red Roses	4059
6	Greetings Fruit Basket	456
7	Mixed Rose Bouquet	510
8	Sweet Dreams Bouquet	2492
9	Sweet Splendor Bouquet	1323
10	Tulip Bouquet	8250

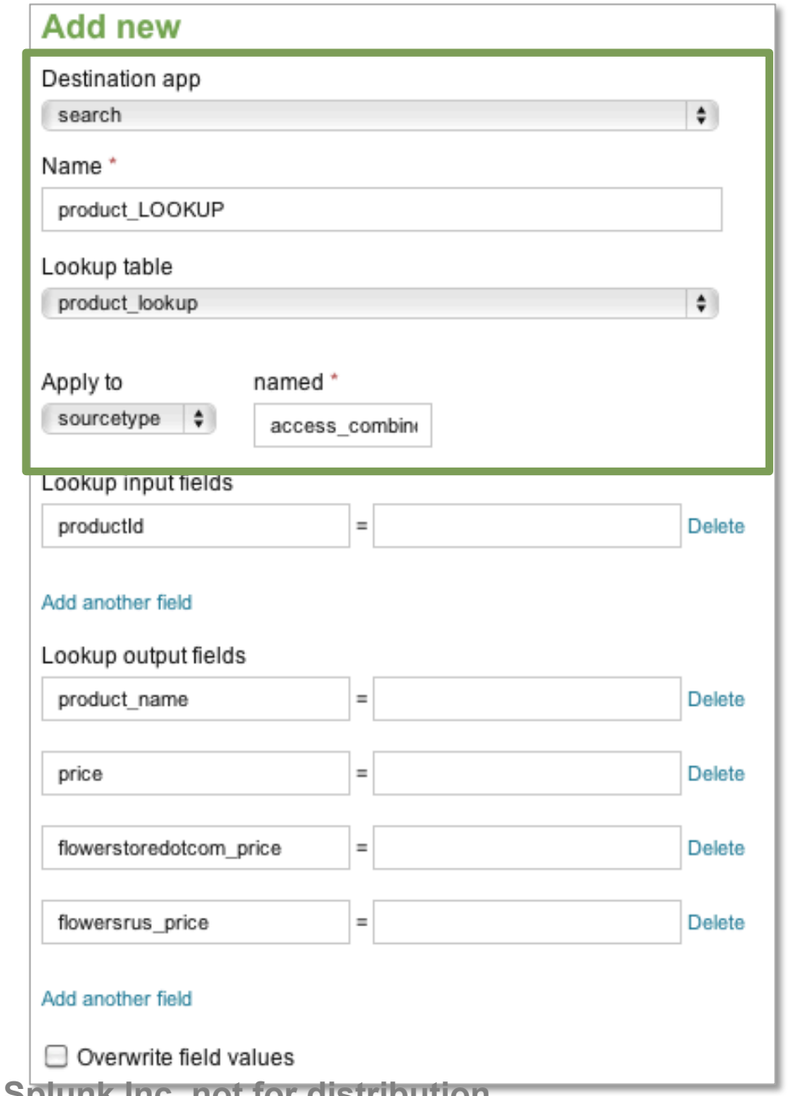
3. Configure an automatic lookup

- **Manager > Lookups > Automatic lookups** lists all existing automatic lookup configurations
  - From this list, you can edit, change permissions, enable/disable, and clone



3. Configure an automatic lookup (cont'd)

1. Select the Destination app
2. Enter a name
3. Select the lookup table
4. Choose a sourcetype, source, or host and identify by name



3. Configure an automatic lookup (cont'd)

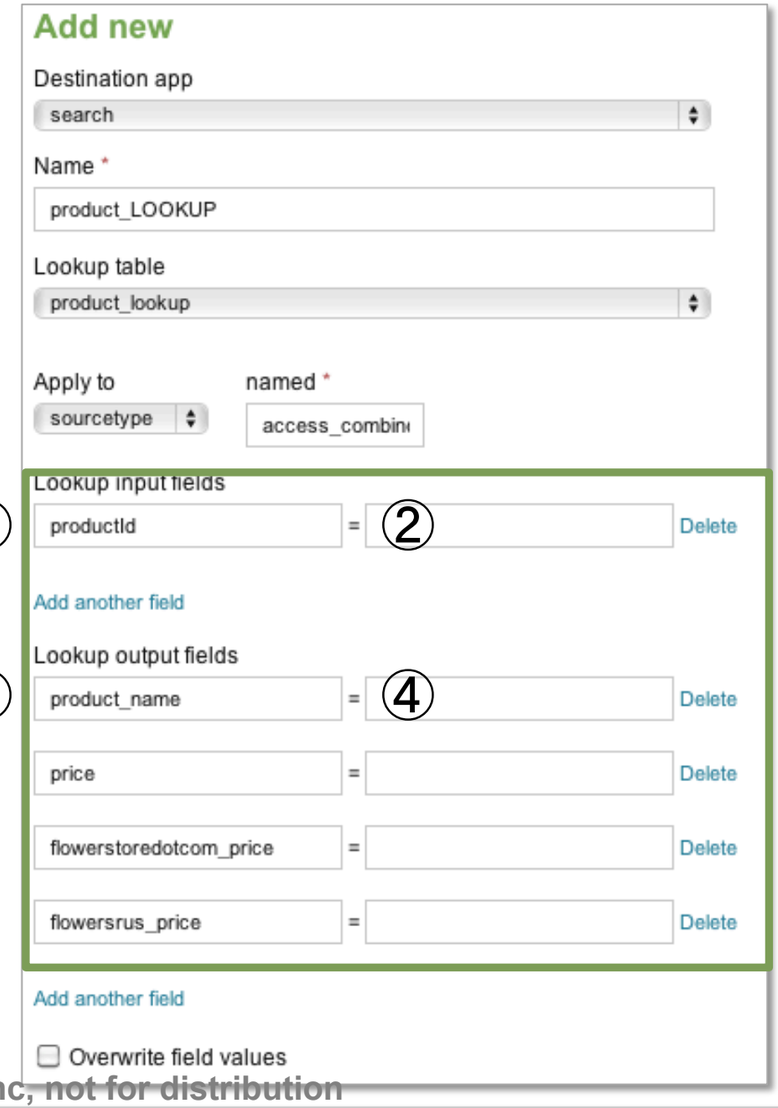
5. Define the Lookup input field(s)
   - Field(s) that exists in your events that you are relating to the lookup table
6. Define the Lookup output field(s)
   - Field(s) from your lookup table that are added to the events

① Column name in CSV

② Field name in Splunk

③ Field name in lookup table

④ Name you want displayed in Splunk



Using the automatic lookup

- To use an automatic lookup, include the output fields in your search

prod_lookup.csv

```
productId,product_name,price,flowersdotcom_price,flowersrus_price
AV-CB-01,Tulip Bouquet,250,269,279
AV-SB-02,Mixed Rose Bouquet,15,19,21
FL-DSH-01,Sweet Splendor Bouquet,49,55,55
FL-DLH-02,Dozen Red Roses,99,129,149
K9-BD-01,Birthday Bouquet,299,355,339
K9-CW-01,Sweet Dreams Bouquet, 89,99,92
FI-SW-01,Cake Serving Set,89,79,85
FI-FW-02,Greetings Fruit Basket,12,8,6
RP-LI-02,Chocolate Dreams Confections,379,299,319
RP-SN-01,Day Spa Certificate,35,39,40
```

	product_name	sales
1	Chocolate Dreams Confections	$584,039
2	Birthday Bouquet	$459,264
3	Tulip Bouquet	$395,250
4	Dozen Red Roses	$158,301
5	Sweet Dreams Bouquet	$144,180
6	Cake Serving Set	$136,081
7	Sweet Splendor Bouquet	$77,665
8	Day Spa Certificate	$55,895
9	Mixed Rose Bouquet	$23,760
10	Greetings Fruit Basket	$18,828

```
sourcetype=access_* action=purchase
| stats sum(price) as sales by product_name
| fieldformat sales = "$" + tostring(sales, "commas")
| sort -sales
```

Time-based lookups

- If a field in the lookup table represents a timestamp, you can create a time-based lookup
- Example: use DHCP logs to identify users on your network based on their IP address and the timestamp
- A script might copy DHCP log events to a .csv file when an ACK event occurs

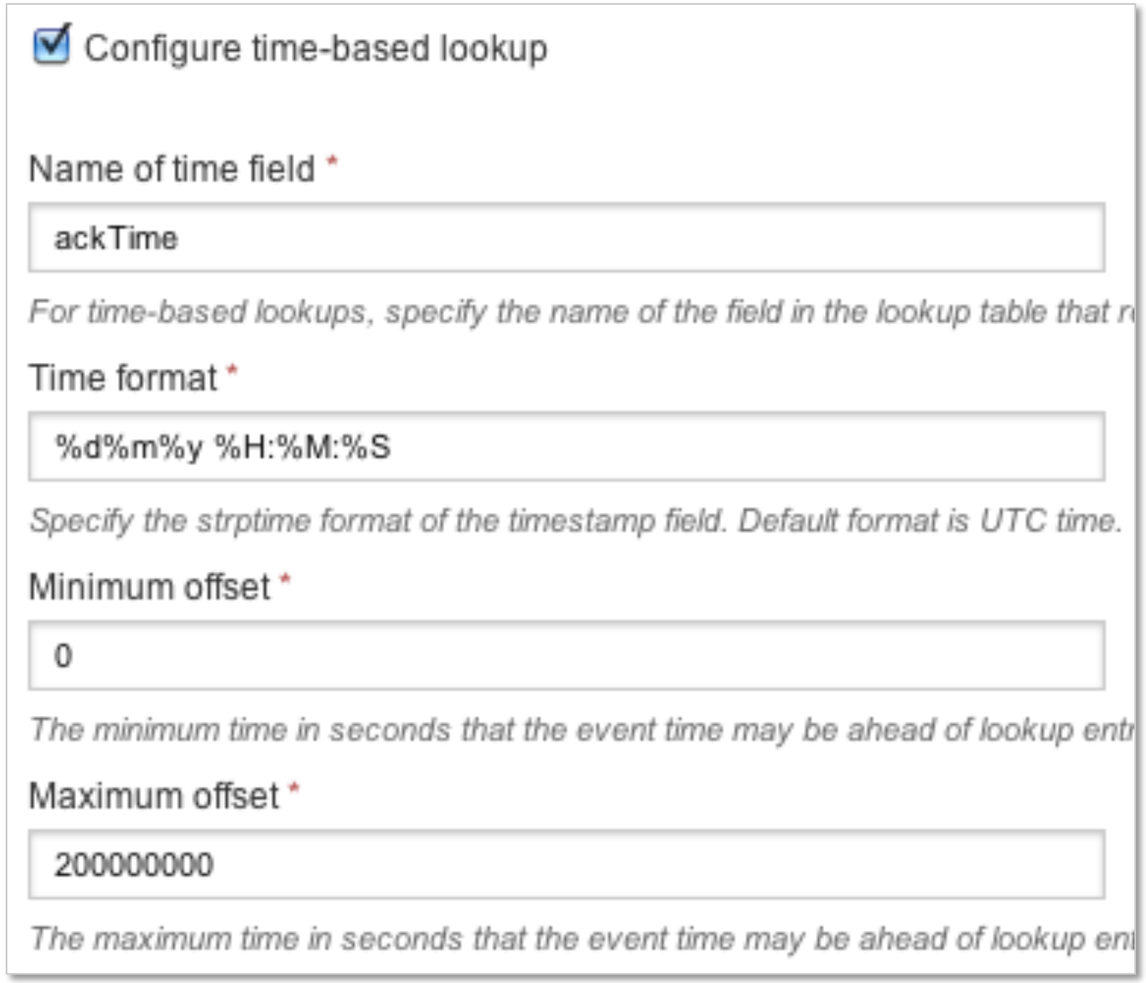
```
ACK:  1 110  11/06/11 11:40:06 171.64.20.120 00:0d:93:b1:9e:d6
```

- .csv file contains the timestamp, IP address, username, and MAC address

```
ackTime,ip,user,macaddress
06NOV2011 11:40:06,171.64.20.120,bwilson,00:0d:93:b1:9e:d6
```

Configuring time-based lookups

1. Specify the name of the time field
2. Enter a strptime format and offset for the time matching
3. Define the minimum offset that an event can be ahead of the time in the lookup (in seconds)
4. Define the maximum offset that an event may be ahead of time in the lookup (in seconds)



Using Splunk DB Connect for lookups

- Splunk DB Connect app enables you to:
  - Easily integrate information from most relational databases, out ot the box support for Oracle Database, Microsoft SQL Server, Sybase, PostgreSQL, and more, with Splunk queries and reports
  - You can also add your own database types by providing JDBC drivers
  - Use Splunk Dbquery and Dbinfo search commands to execute database queries directly from the Splunk Enterprise user interface
- Free Splunk-developed app available on Splunkbase
  http://splunk-base.splunk.com/apps/50803/splunk-db-connect
  - Full documentation available http://docs.splunk.com/documentation/dbx

More lookup options

- In addition to creating and using a file-based lookup, you can also:
  - Populate a lookup table with search results
  - Set up a fields lookup based on an external command or script
  - Set up a fields lookup based on an external database

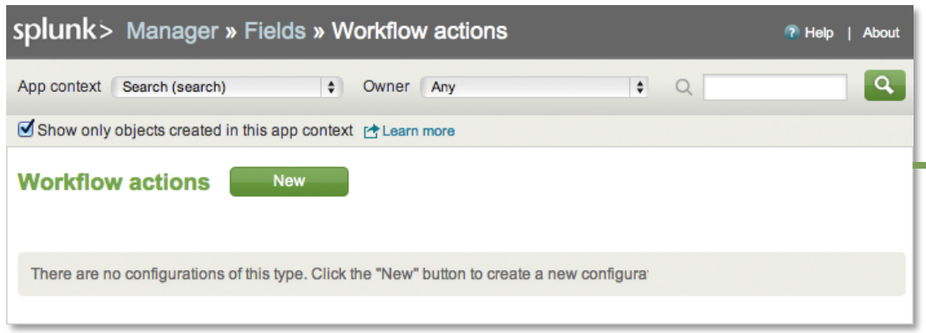
More information on creating and configuring lookups can be found at docs.splunk.com

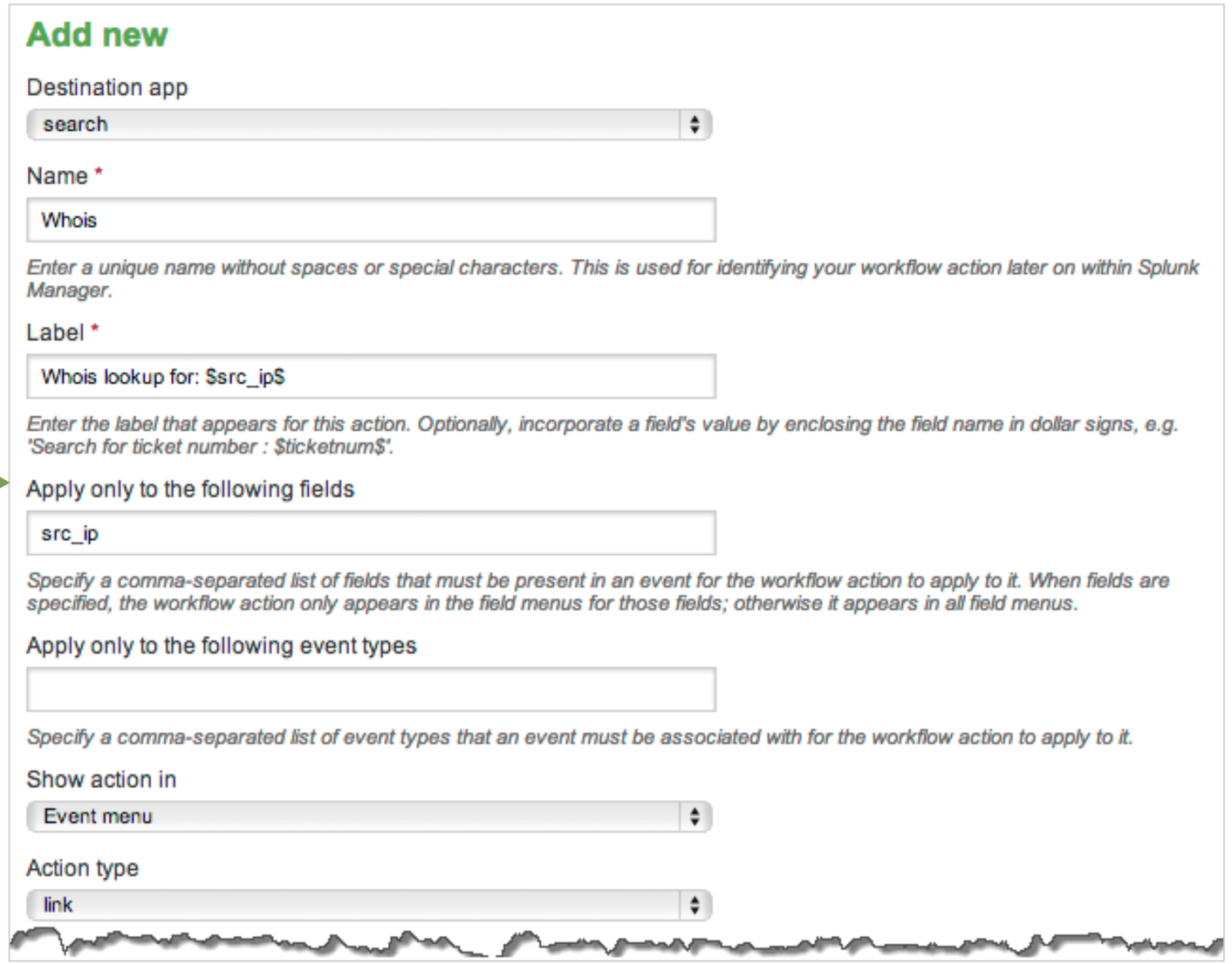
Workflow actions

- Using **Workflow actions** you can set up interactions between specific fields in your events and other applications or web resources
  - Example, when an event displays with a particular field that contains an IP address, open a separate browser window and do an external WHOIS search
- You can set up workflow actions that:
  - Apply only to a particular field, such as an IP address (as opposed to all fields in an event)
  - Apply only to events belonging to a specific event type or group of event types
  - Are accessed either via event dropdown menus, field dropdown menus, or both
  - Perform HTTP GET or POST requests
  - Use field values from a chosen event to launch a secondary search in another browser window

Creating a workflow action – WHOIS lookup - 1

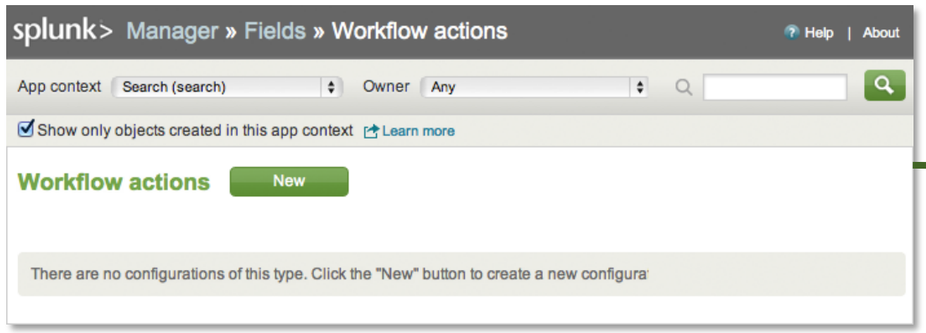
- Build workflow actions through **Manager>Fields> Workflow actions**

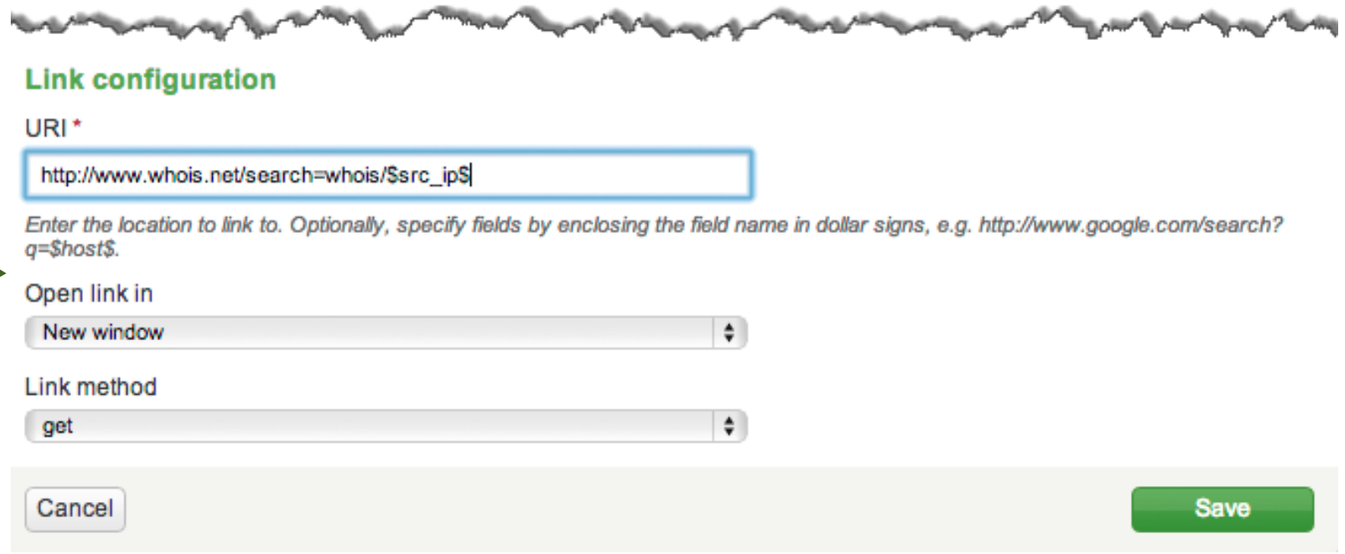




Creating a workflow action – WHOIS lookup - 2

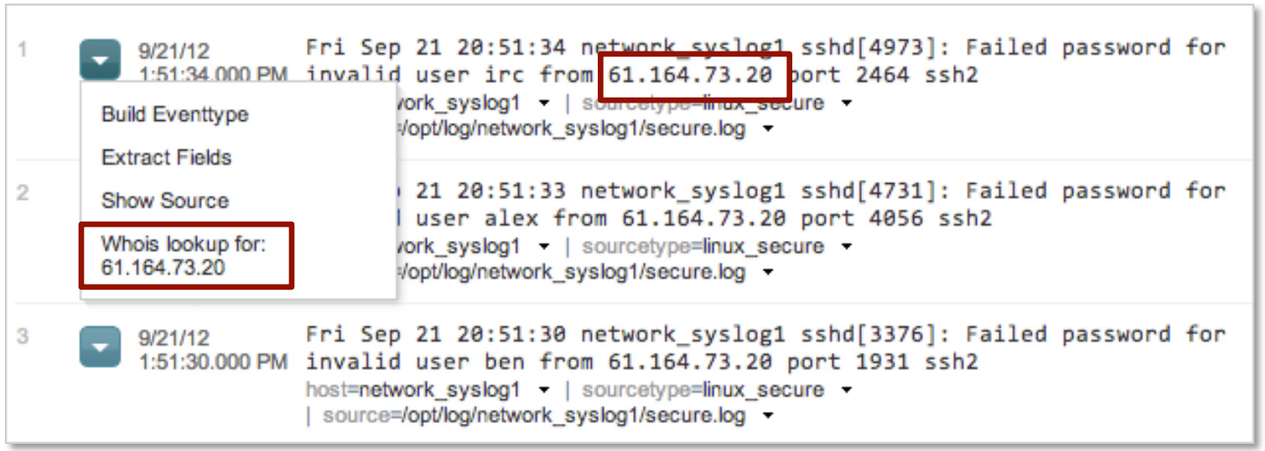
- Build workflow actions through **Manager>Fields> Workflow actions**





Using a workflow action

- If the search returns events containing the `src_ip` field, you can access the workflow from the Event menus
  - Note, because you used the src_ip variable when building the workflow action, the IP address displays in event menu



Lab 6

- **Time:**
  - 20-25 minutes
- **Tasks:**
  - Upload a lookup file
    - `browser_lookup.csv` is a file you downloaded
    - **CAUTION: DO NOT** open this file with Excel as it can corrupt the file
  - Create a lookup definition
  - Create a lookup table
  - Use the lookup in a search
  - Configure the lookup to run automatically
  - Use the automatic lookup in a search
  - Configure and use a workflow action

Section 7:
Report Acceleration

Section objectives

- Describe report acceleration
- Create summaries
- Search against summaries
- Describe summary management

Report acceleration overview

- Reports that cover a large volume of data can:
  - Take long time to complete
  - Consume a great deal of system resources
- You can 'accelerate' a qualifying report when you:
  - Save it
  - Create a dashboard panel based on it
  - Edit a qualifying saved search
- Common use cases include:
  - More efficiently run reports for large datasets over long time ranges
    - Show the number of page views and visitors for each of your web sites over the past 30 days, broken out by site
  - Build a rolling report that shows aggregated statistics over long periods of time
    - Display a running count of downloads for a specific file on a website
    - Calculate the average amount spent per purchase over a year

Report acceleration overview (cont'd)

- To accelerate the search, Splunk creates an acceleration summary
- Acceleration summaries
  - Efficiently report on large volumes of data
  - Qualify future searches against the summary
- To accelerate a report, **Search Mode** must be set to *Smart* or *Fast*
  - Neither the Timeline nor the Fields sidebar display
- By default, only power users can accelerate reports
- If you delete all the searches that use a summary, the summary is deleted
- If an acceleration summary is created from a shared search, other reports that can use it, will use it

Populating search requirements

- Qualifying searches
  - Search must include a reporting command
    - For example: `chart`, `timechart`, `stats`, `top`, and `rare`
  - Any command before the reporting command must be a streaming command, that is a command that applies a transformation to each event returned by the search
    - For example: `eval`, `fields`, `multikv`, `rex`, `rename`, and `replace`

Search examples

- Qualifying search examples:

```
sourcetype=access_* action=purchase status=200
| stats sum(price) as revenue by productId
| eval revenue="$" + revenue
```

```
sourcetype=* | stats count by sourcetype
```

- Non-qualifying search examples

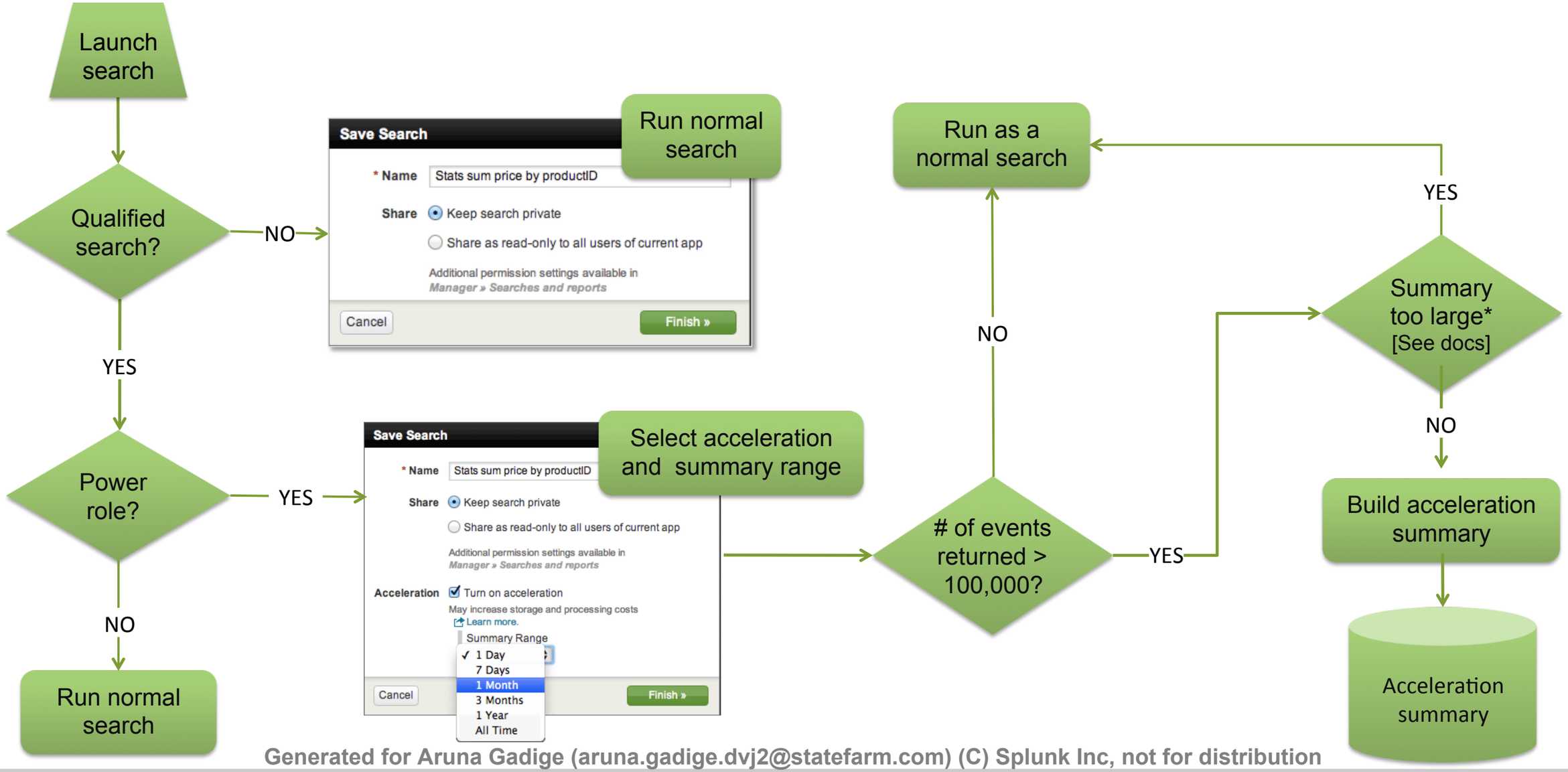
```
sourcetype=access_* action=purchase status=404
```

[No reporting command]

```
sourcetype=access_* | transaction startswith="view" endswith="purchase"
| stats avg(duration)
```

[Transaction is not a streaming command]

Creating acceleration summaries

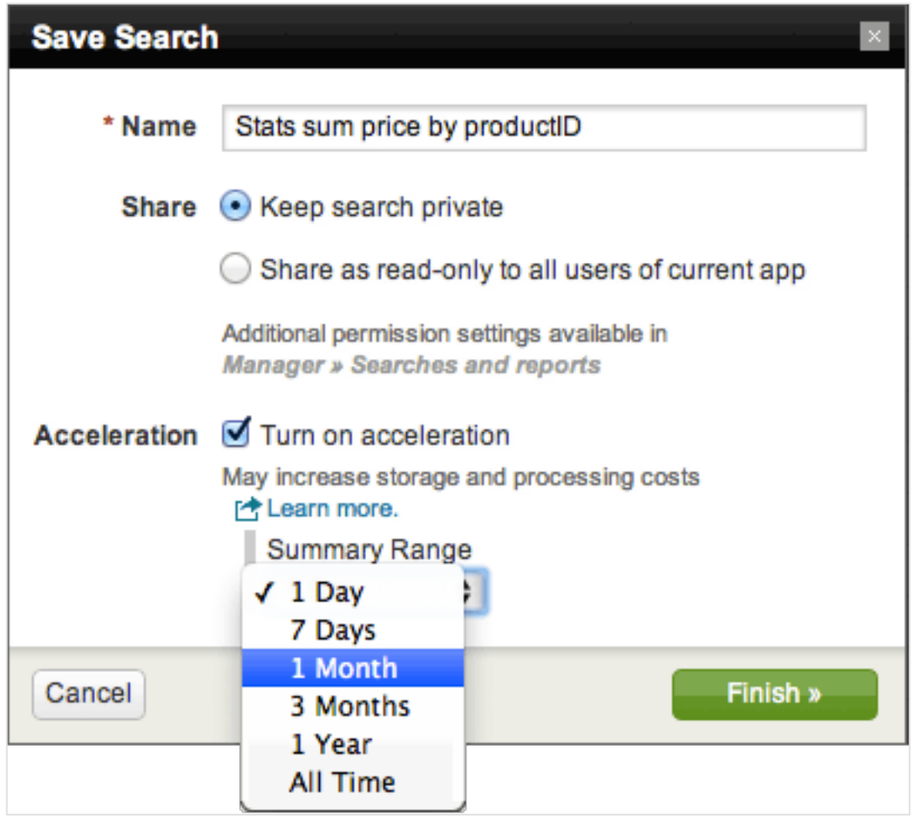


Cases where Splunk will not build a summary

- There are cases where Splunk allows you to "accelerate" a search, but a summary won't be created
- Splunk knows what's most efficient and ***generally*** won't generate a summary if:
  - There are fewer than 100K events in the summary range - it's faster executing the search without a summary
  - Summary size is projected to too large - it's faster executing the search because the main index is smaller
- If a summary is defined and not created for the above reasons, Splunk continues to check periodically, then automatically creates a summary once it meets the requirements

Acceleration summary time ranges

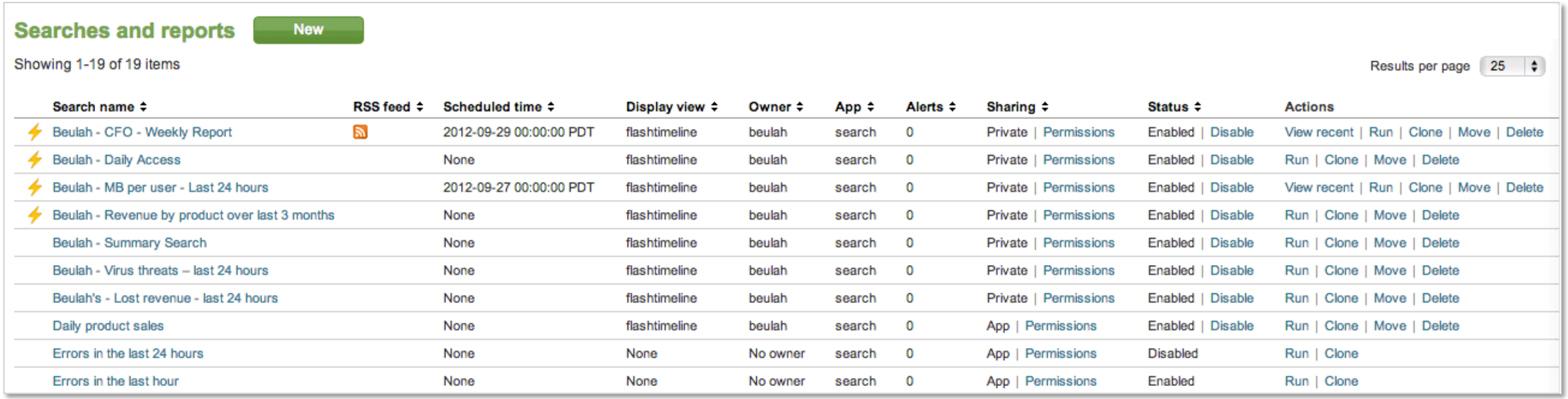
- Summary spans approximate range of time specified in the range
  - Setting a range of one week creates a summary that covers last 7 days, relative to now
- Periodically removes older summary data that passes out of the range



If 'old' [historical] data is important, then use summary indexing as all data is kept

Managing accelerated reports

- You manage accelerated reports from **Manager>Data>Report Acceleration Summaries**
- Accelerated searches are marked with a lightning bolt



Searches and reports New

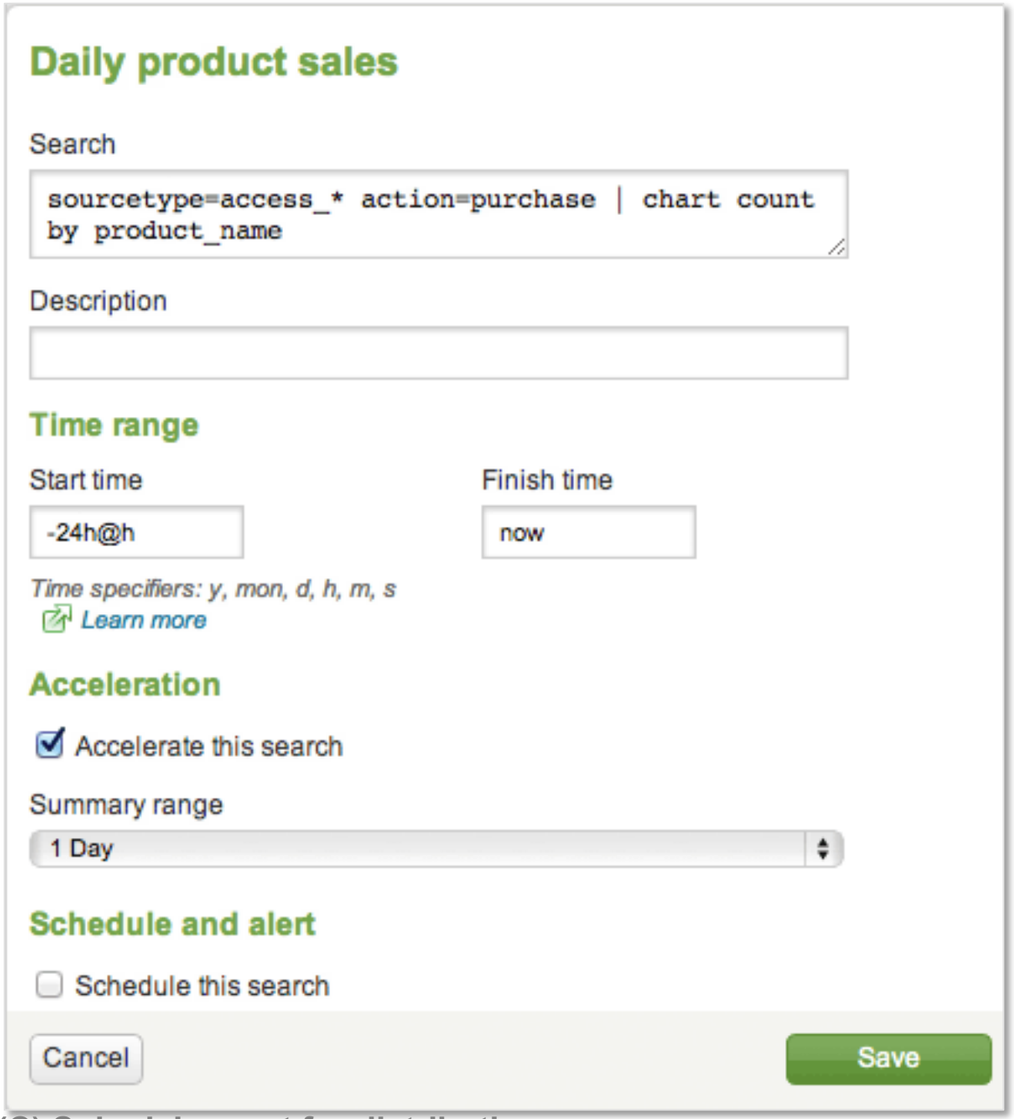
Showing 1-19 of 19 items

Results per page 25

Search name	RSS feed	Scheduled time	Display view	Owner	App	Alerts	Sharing	Status	Actions
Beulah - CFO - Weekly Report		2012-09-29 00:00:00 PDT	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	View recent | Run | Clone | Move | Delete
Beulah - Daily Access		None	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Beulah - MB per user - Last 24 hours		2012-09-27 00:00:00 PDT	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	View recent | Run | Clone | Move | Delete
Beulah - Revenue by product over last 3 months		None	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Beulah - Summary Search		None	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Beulah - Virus threats – last 24 hours		None	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Beulah's - Lost revenue - last 24 hours		None	flashtimeline	beulah	search	0	Private | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Daily product sales		None	flashtimeline	beulah	search	0	App | Permissions	Enabled | Disable	Run | Clone | Move | Delete
Errors in the last 24 hours		None	None	No owner	search	0	App | Permissions	Disabled	Run | Clone
Errors in the last hour		None	None	No owner	search	0	App | Permissions	Enabled	Run | Clone

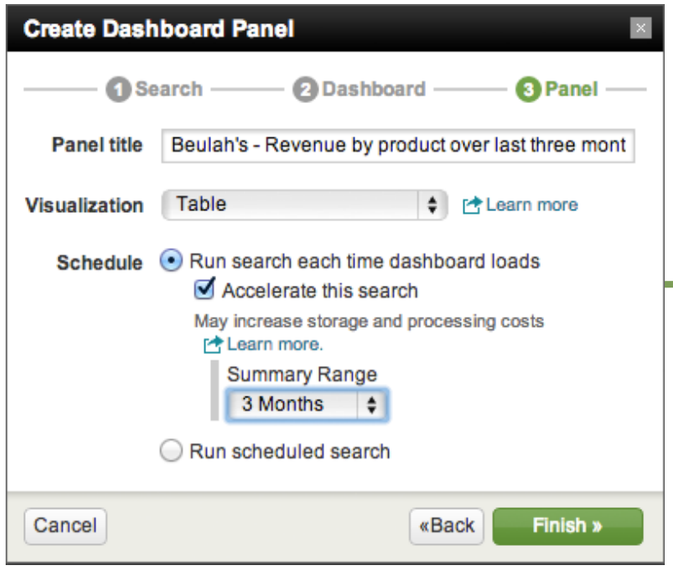
Accelerating an existing saved search

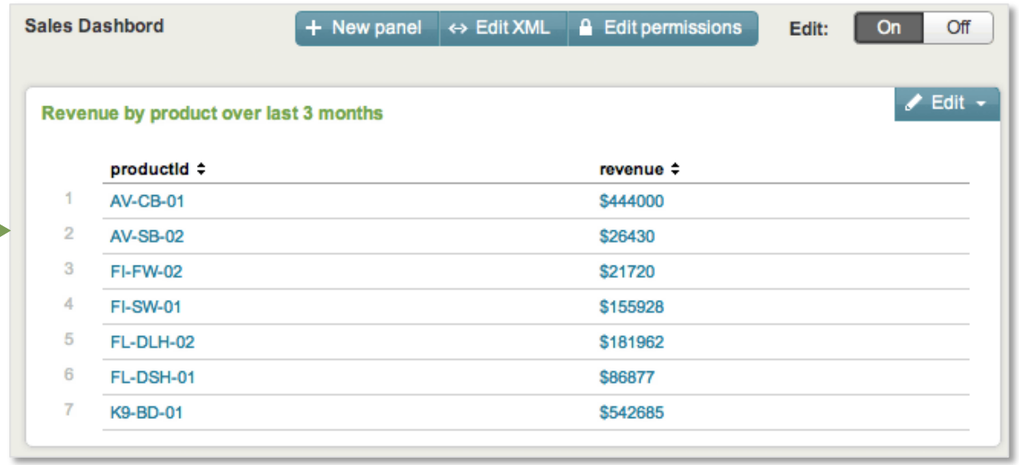
- From **Manager > Searches and reports**, select a saved search
- The Accelerate option is always available, whether the search qualifies or not
  - Splunk will determine if it can be accelerated when you click **Save**
- If you try to accelerate an unqualified search, an error message displays:

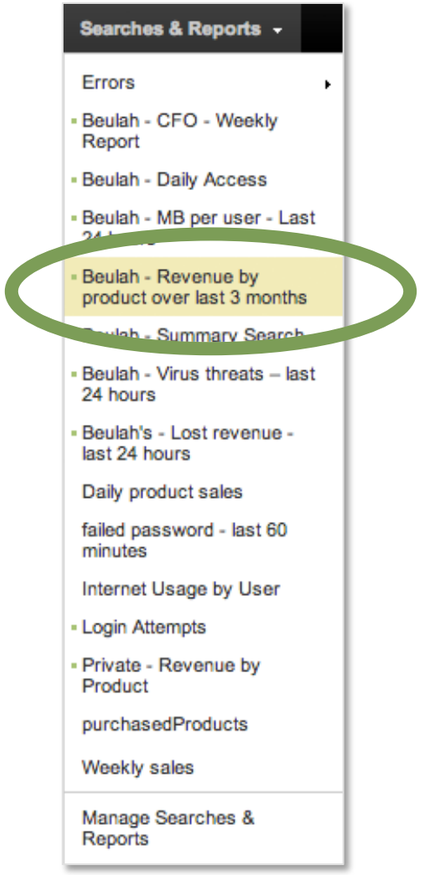


Using an acceleration summary

- Run the saved search from the **Searches & Reports** menu
- Use the saved search in a dashboard, alert, etc.







More ways to search against summary

- Ad-hoc searches can use the summary when:
  - Search criteria matches the populating saved search
  - The time span is greater than or equal to the summary span
    - For time spans that are greater than the span of the summary, Splunk uses as much of the summary as it can
- You can also append the search string with additional reporting commands
  - Example:
    populating search –

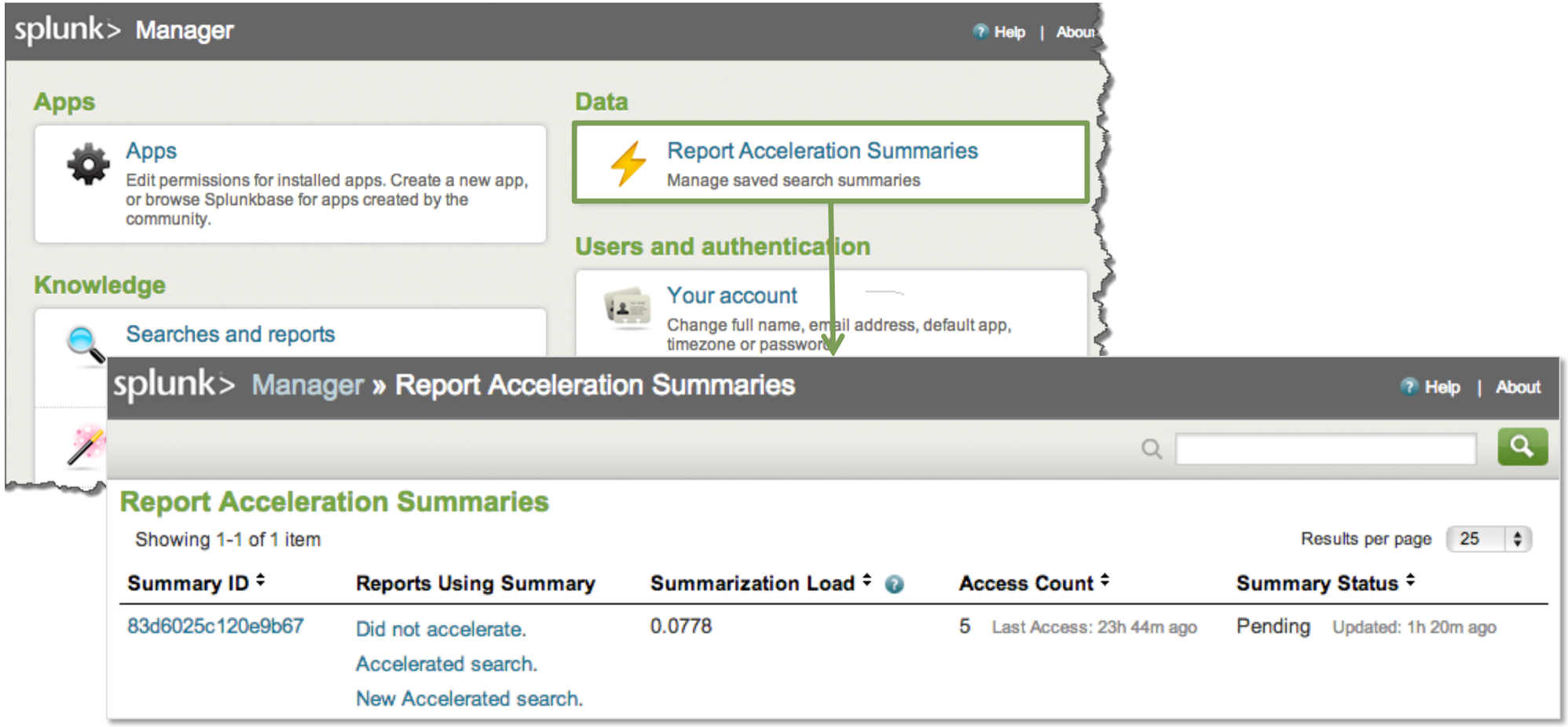
```
sourcetype=access_* | stats count by price
```

    ad hoc search –

```
sourcetype=access_* | stats count by price | eval discount = price/2
```

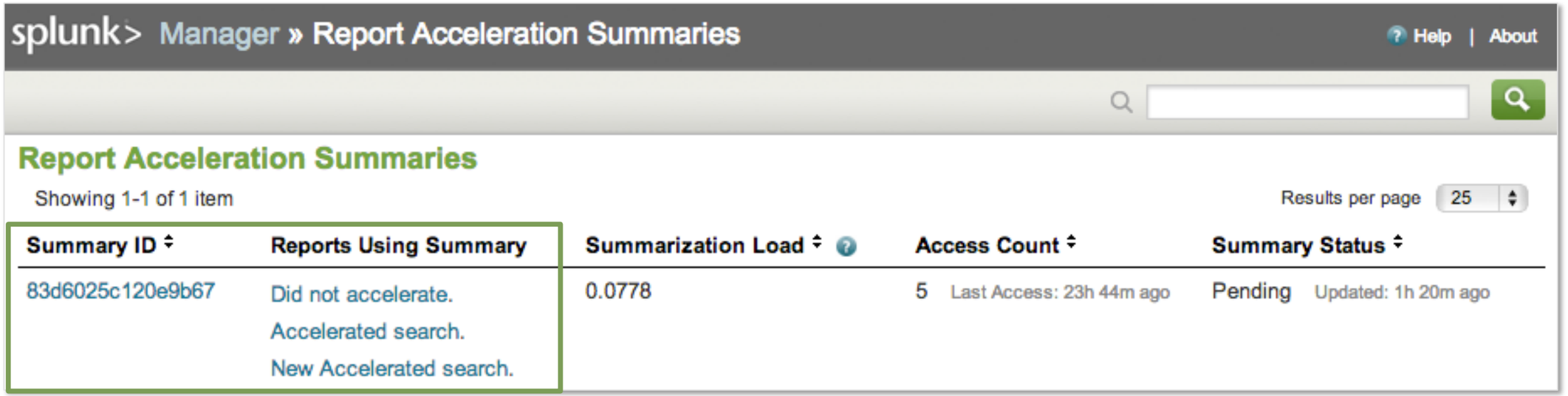
Managing summaries

- Manage summaries from **Manager > Report Acceleration Summaries**



Summary ID and reports

- Each summary can have multiple reports that leverage it
  - To open the saved search in Manager, click the report



Summarization load

- Calculated based on how long it takes for the populating search to run, relative to the schedule
  - Example: Populating search runs every 10 minutes and takes 30 seconds to run
  - Summarization load is 0.05

$$\frac{\text{time to run (seconds)}}{\text{span between searches (seconds)}} = \frac{30}{600} = .05$$

Report Acceleration Summaries

Showing 1-3 of 3 items

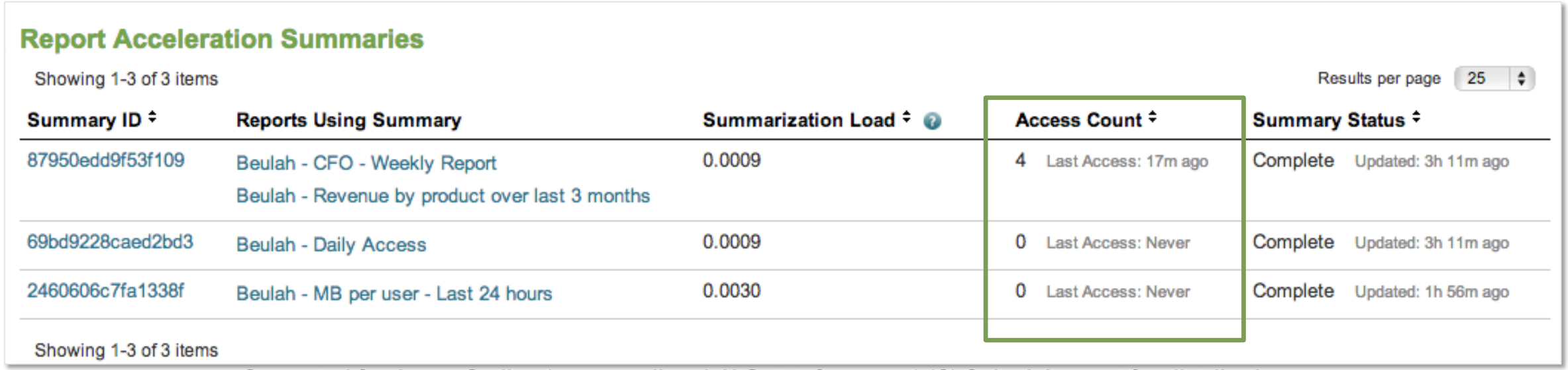
Results per page 25

Summary ID	Reports Using Summary	Summarization Load	Access Count	Summary Status
87950edd9f53f109	Beulah - CFO - Weekly Report Beulah - Revenue by product over last 3 months	0.0009	4 Last Access: 17m ago	Complete Updated: 3h 11m ago
69bd9228caed2bd3	Beulah - Daily Access	0.0009	0 Last Access: Never	Complete Updated: 3h 11m ago
2460606c7fa1338f	Beulah - MB per user - Last 24 hours	0.0030	0 Last Access: Never	Complete Updated: 1h 56m ago

Showing 1-3 of 3 items

Access count

- How many times a summary was used and last time it was used
  - Use this metric along with Summarization Load to determine if a large or high load Summary is worth maintaining/keeping



Report Acceleration Summaries

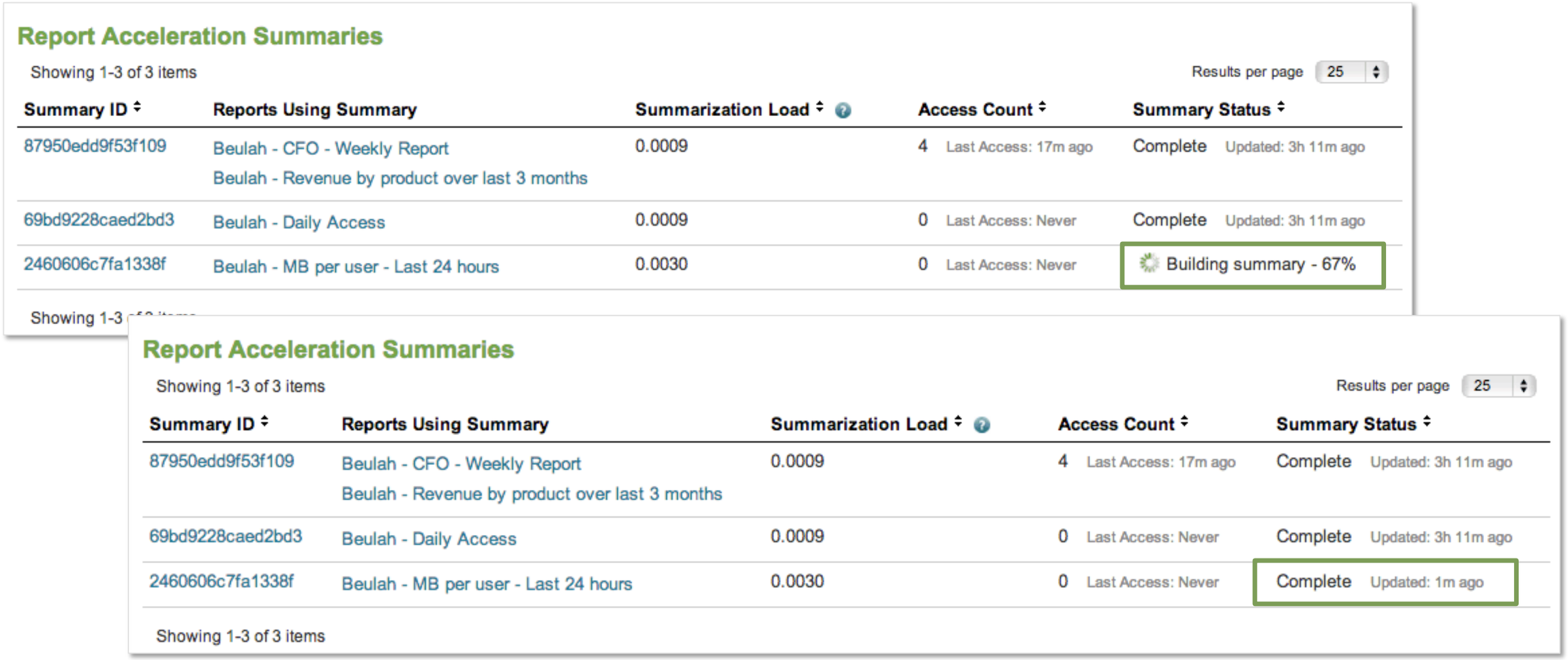
Showing 1-3 of 3 items

Results per page 25

Summary ID	Reports Using Summary	Summarization Load	Access Count	Summary Status
87950edd9f53f109	Beulah - CFO - Weekly Report Beulah - Revenue by product over last 3 months	0.0009	4 Last Access: 17m ago	Complete Updated: 3h 11m ago
69bd9228caed2bd3	Beulah - Daily Access	0.0009	0 Last Access: Never	Complete Updated: 3h 11m ago
2460606c7fa1338f	Beulah - MB per user - Last 24 hours	0.0030	0 Last Access: Never	Complete Updated: 1h 56m ago

Showing 1-3 of 3 items

Summary status



Report Acceleration Summaries

Showing 1-3 of 3 items

Results per page 25

Summary ID	Reports Using Summary	Summarization Load	Access Count	Summary Status
87950edd9f53f109	Beulah - CFO - Weekly Report Beulah - Revenue by product over last 3 months	0.0009	4 Last Access: 17m ago	Complete Updated: 3h 11m ago
69bd9228caed2bd3	Beulah - Daily Access	0.0009	0 Last Access: Never	Complete Updated: 3h 11m ago
2460606c7fa1338f	Beulah - MB per user - Last 24 hours	0.0030	0 Last Access: Never	Building summary - 67%

Showing 1-3 of 3 items

Report Acceleration Summaries

Showing 1-3 of 3 items

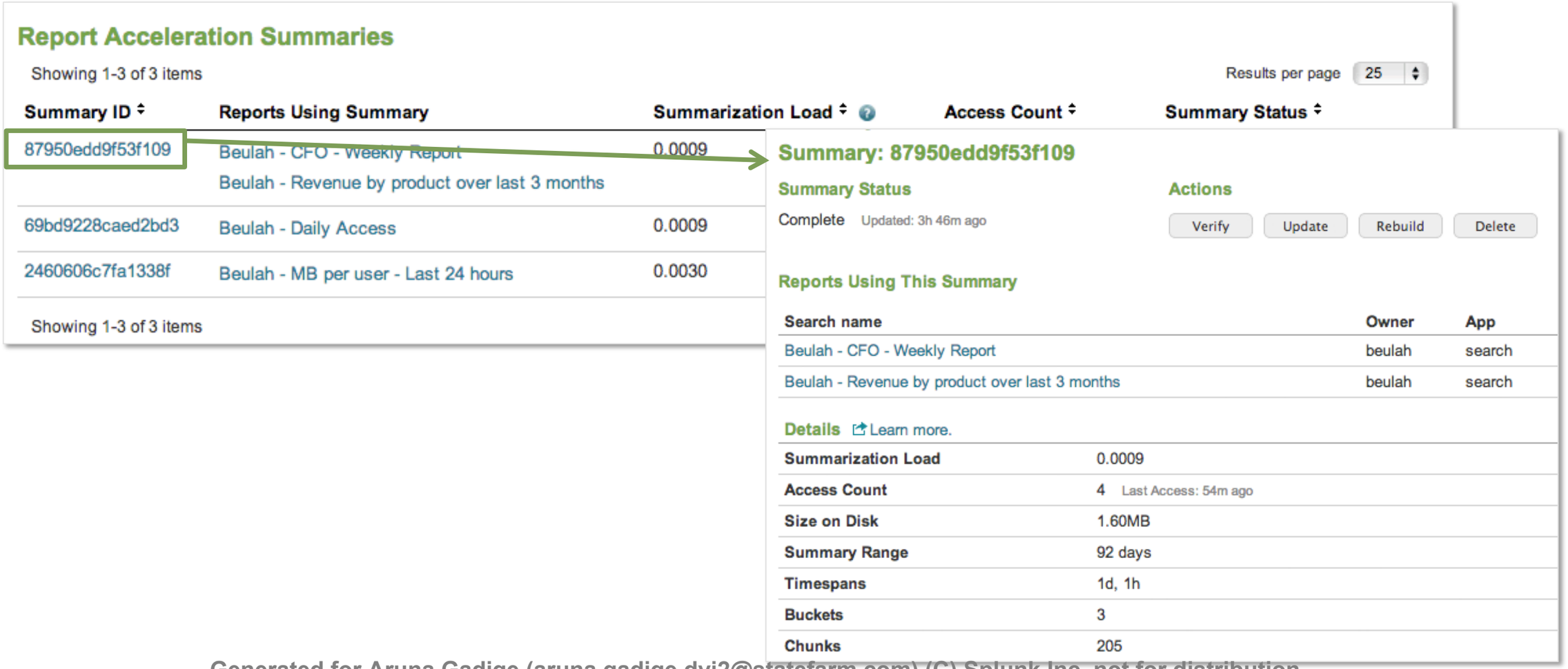
Results per page 25

Summary ID	Reports Using Summary	Summarization Load	Access Count	Summary Status
87950edd9f53f109	Beulah - CFO - Weekly Report Beulah - Revenue by product over last 3 months	0.0009	4 Last Access: 17m ago	Complete Updated: 3h 11m ago
69bd9228caed2bd3	Beulah - Daily Access	0.0009	0 Last Access: Never	Complete Updated: 3h 11m ago
2460606c7fa1338f	Beulah - MB per user - Last 24 hours	0.0030	0 Last Access: Never	Complete Updated: 1m ago

Showing 1-3 of 3 items

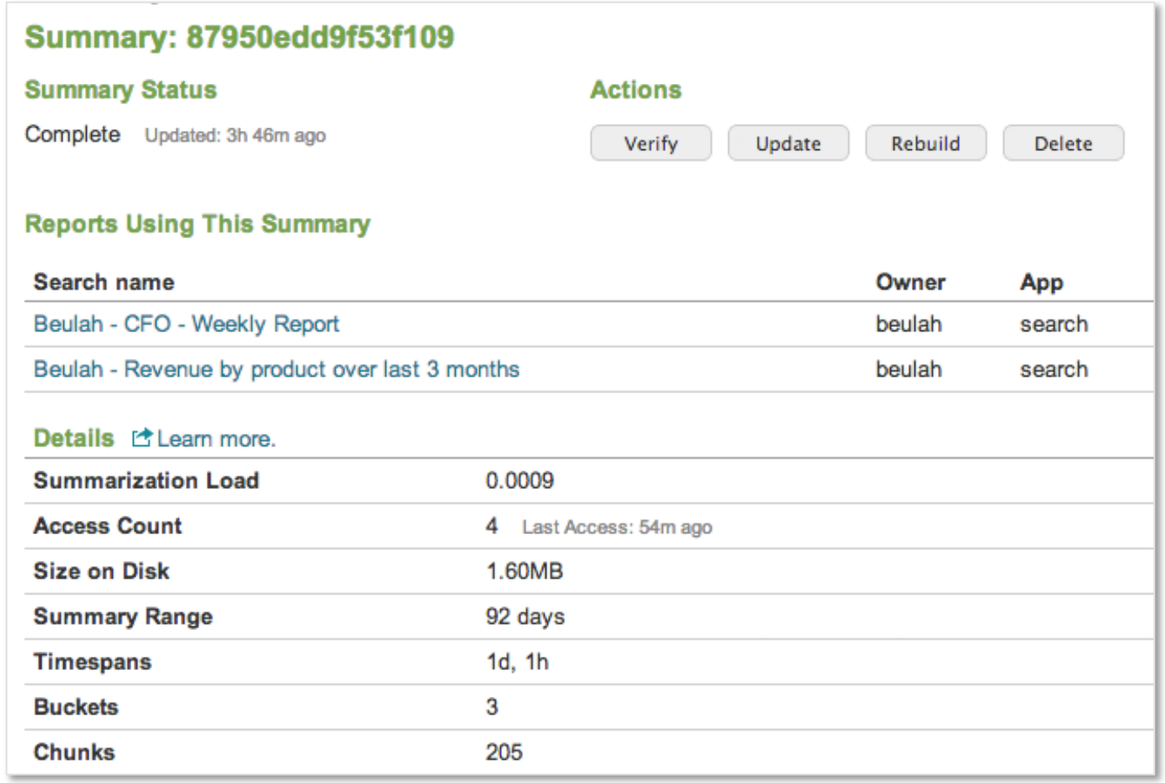
Summary details

- To view summary details, click the Summary ID



Summary details

- Only your searches that use this summary display
  - Searches from other users may use this summary, but do not display here



As data comes in to Splunk, it's stored in the index in **buckets**.
Buckets are covered in detail in the Administrating Splunk class.

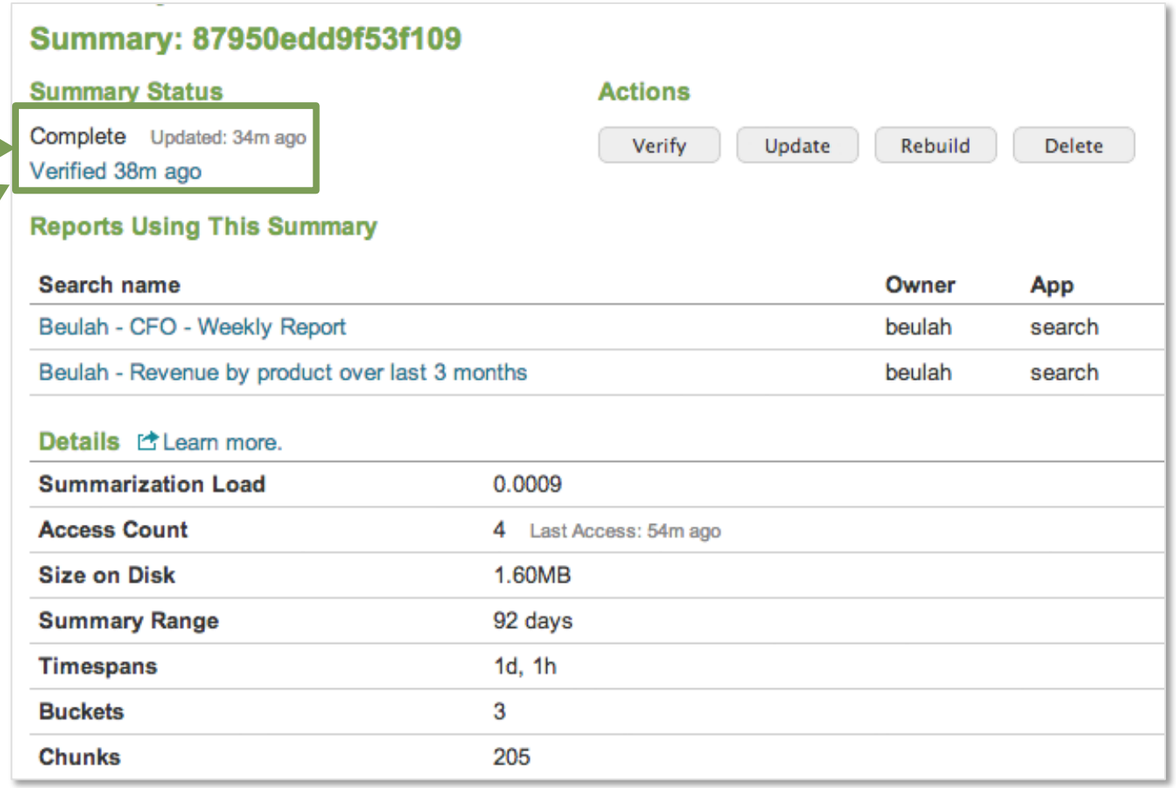
You can also learn more at:

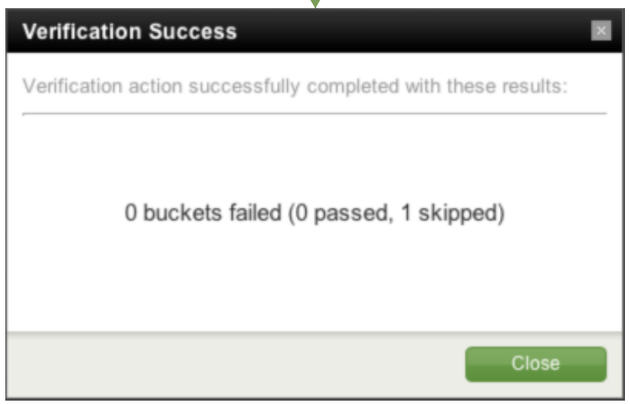
http://docs.splunk.com/Documentation/Splunk/5.0/Indexer/Aboutindexesandindexers

Summary details – status

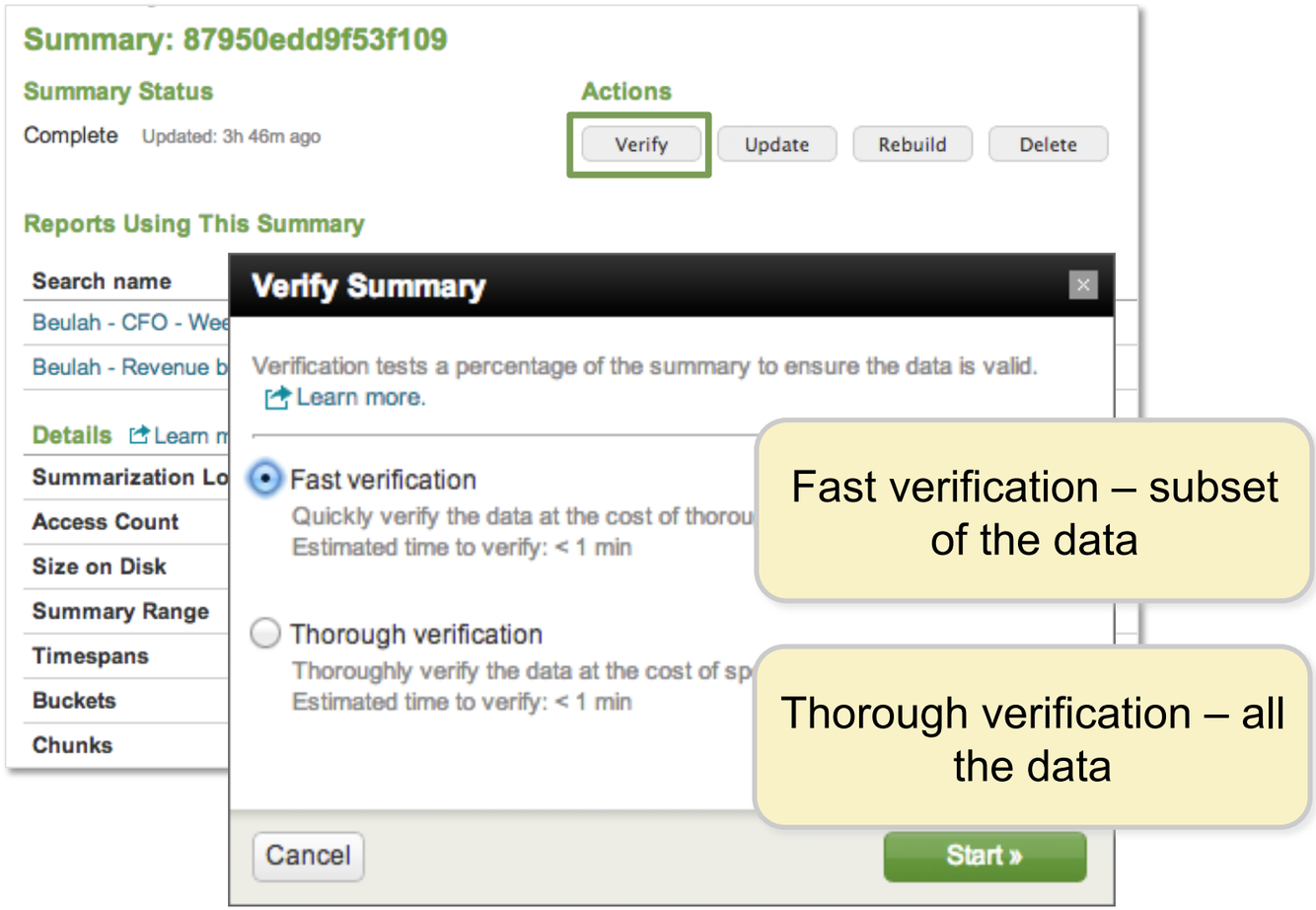
Summary status will either show "complete" with the time the summary was last updated, "suspended" or a percentage of completion

Show results of last verification





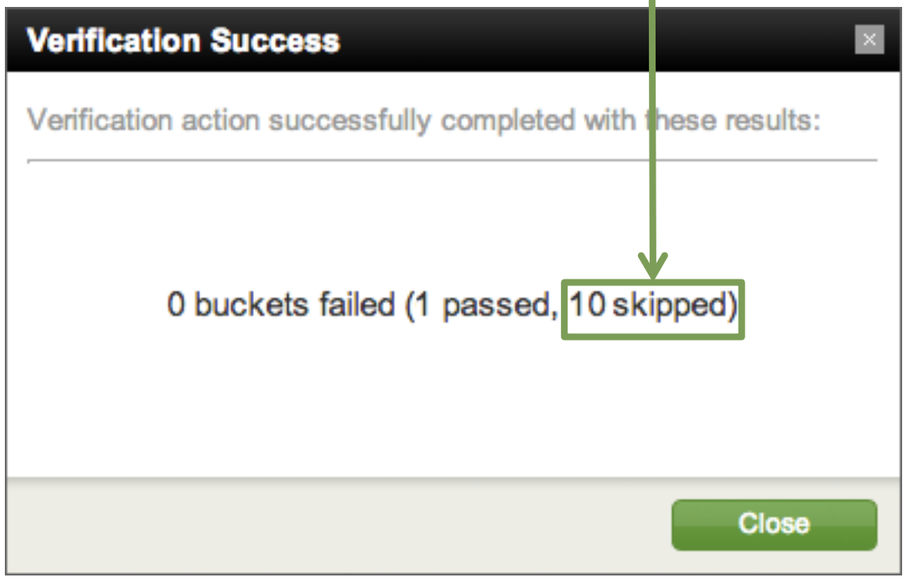
Summary actions – verify



Fast verification – subset of the data

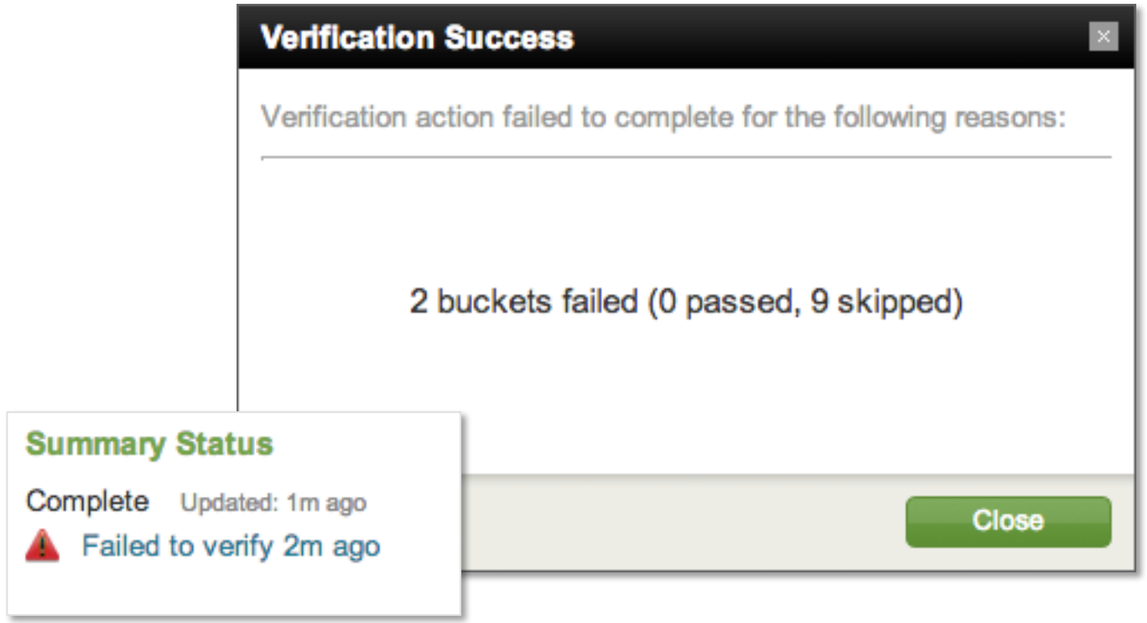
Thorough verification – all the data

Hot buckets or any buckets that are in the process of building are skipped



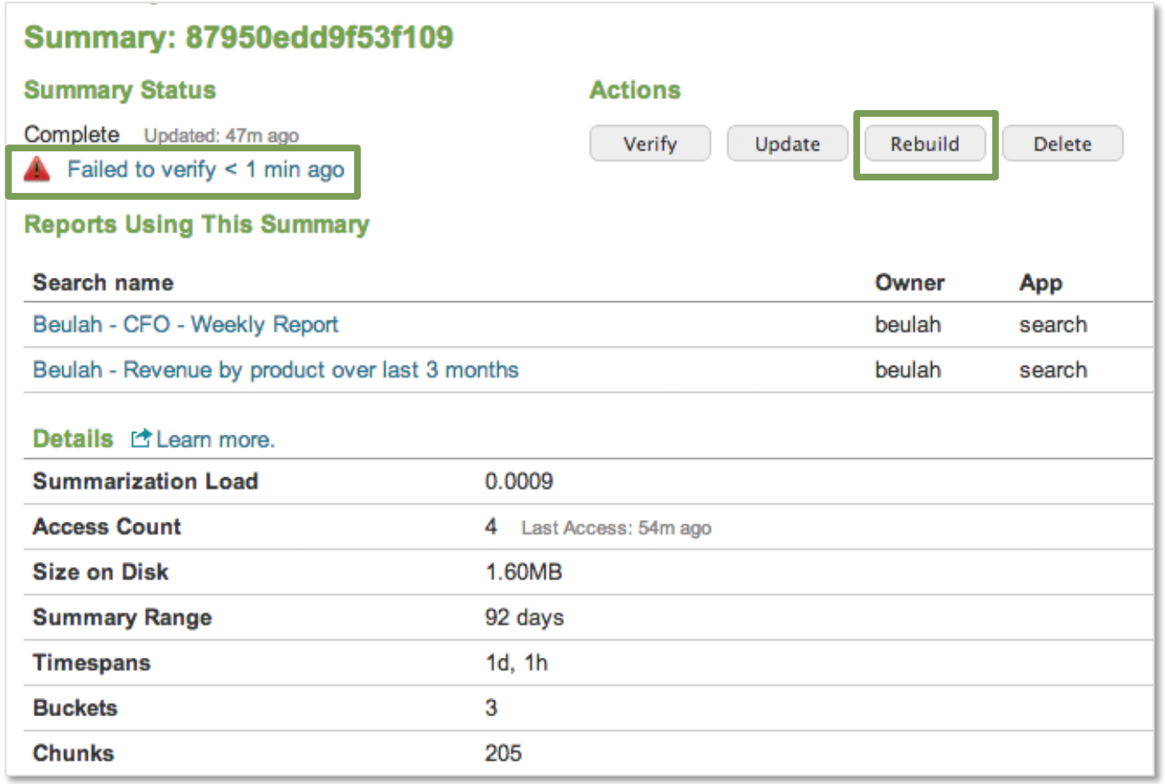
Verification failure

- Summaries can fail verification when the data being returned is not consistent with the data returned when the summary was last built
  - An underlying event type or tag definition may have changed
  - Data deletion may have occurred



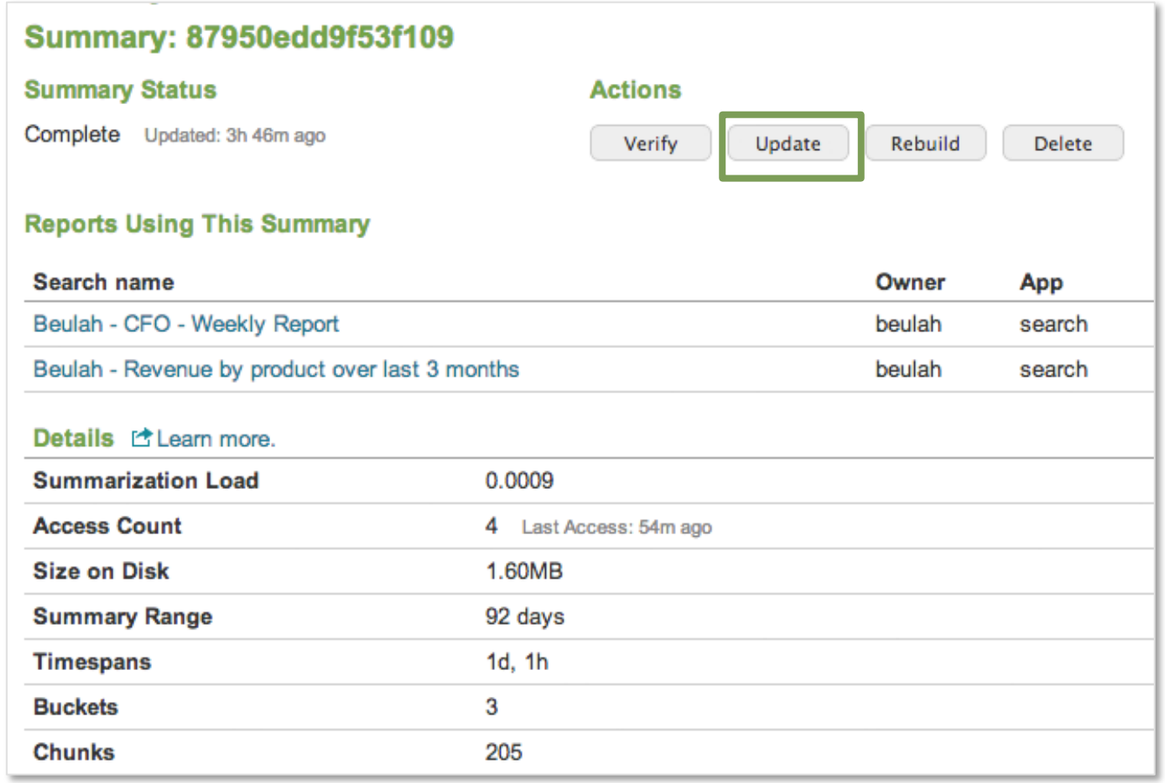
Summary actions – rebuild

- Rebuilds the summary from scratch
- Use this when
  - Summary fails verification
  - You suspect data loss due to system issues or other reasons
  - Underlying knowledge has changed such as tags, event types, field extractions, etc.



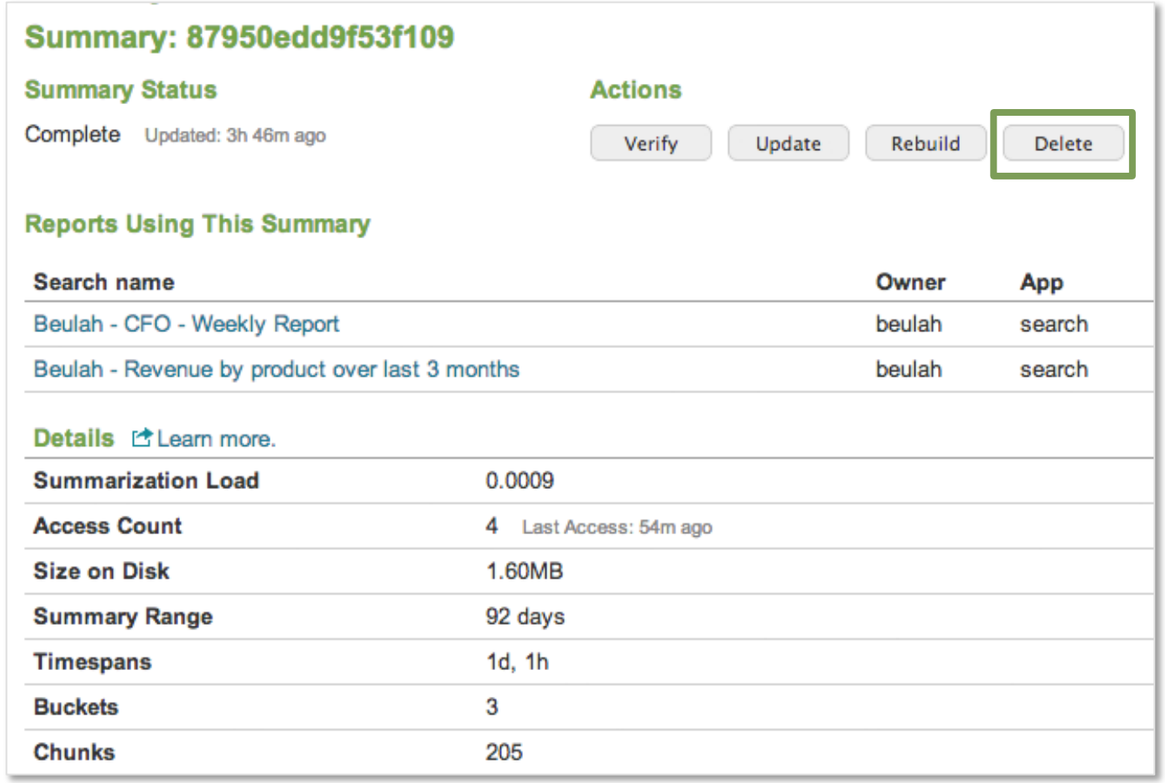
Summary actions – update

- Runs the populating search and updates the summary on-demand



Summary actions – delete

- If this is the only saved search that accesses the summary, then it is deleted
- If there are other searches using the summary:
  - The saved search remains
  - It is not accelerated
  - Can be run manually



Lab 7

- **Time:**
  - 10-15 minutes
- **Tasks:**
  - Save an accelerated saved search
  - Create a dashboard panel with an accelerated saved search
  - Accelerate an existing saved search
  - Use an acceleration summary
  - Access the summary management pages

Section 8:
Creating and Using Macros

Section objectives

- Describe macros
- Manage macros
- Create a basic macro
- Use a basic macro
- Define arguments / variables for a macro
- Add and use arguments with a macro

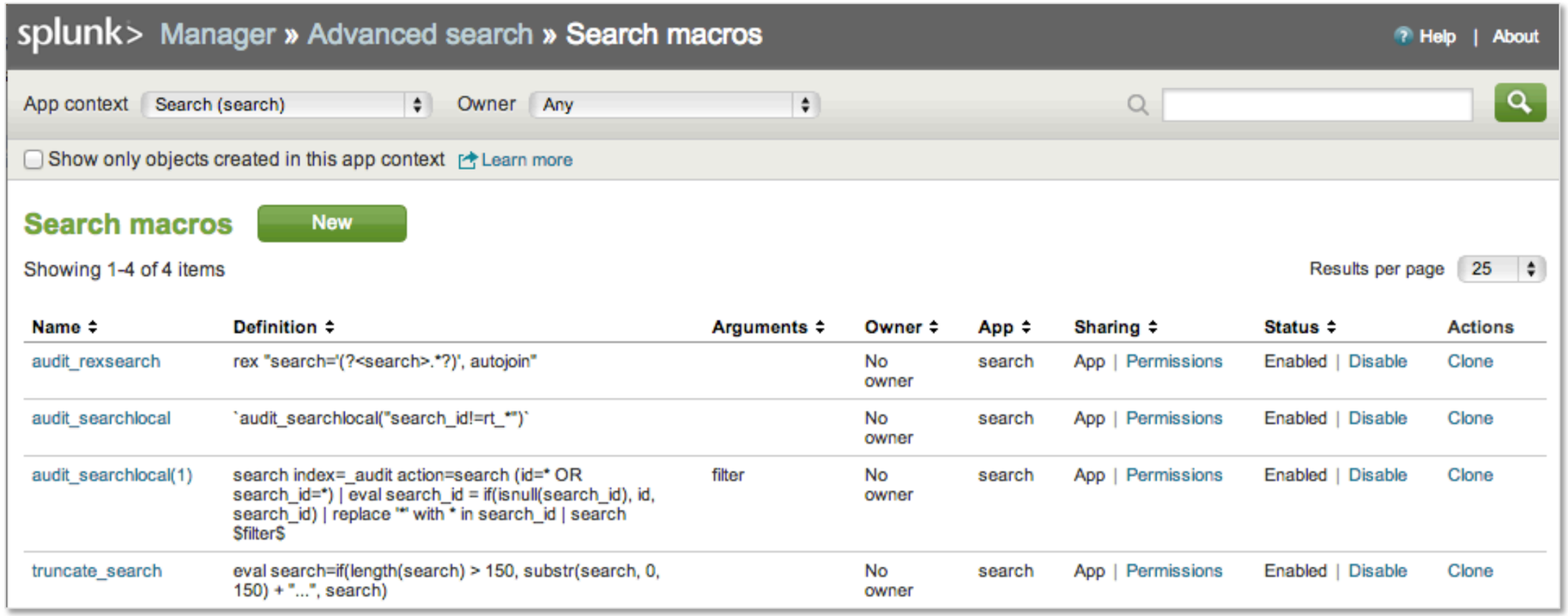
Macros overview

- Useful when you frequently run searches or reports with similar search syntax
- "Portions" of a search you can reuse in multiple places
- Can be any portion of your search string or search command pipeline
- Allows you to define one or more arguments within the search segment
  - Pass values to the search string when using the macro

Managing macros

As with other types of knowledge objects, you can:

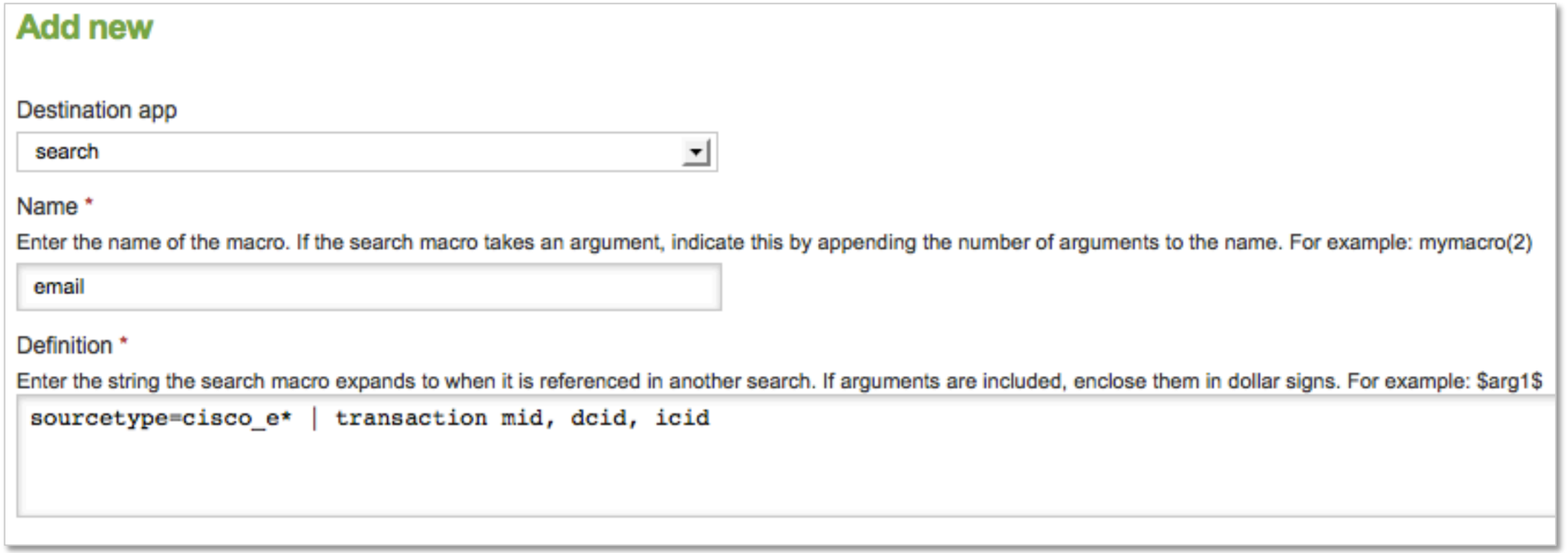
- Create new
- Edit
- Set permissions
- Enable / disable
- Clone
- Move
- Delete



Creating a basic macro

Manager > Advanced search > Search Macros > Add New

1. Select the Destination app
2. Enter a name
3. Type the search string
4. Save



Using a basic macro

- Type the macro name into the search bar, or use it in a saved search or report
- Surround the macro name with the **backtick** (or grave accent) character
  - **`macroname`** != 'macroname'
  - Do not confuse with single-quote character (')
- Pipe to more commands, or precede with search string

```
`email` | chart count by mailto
```

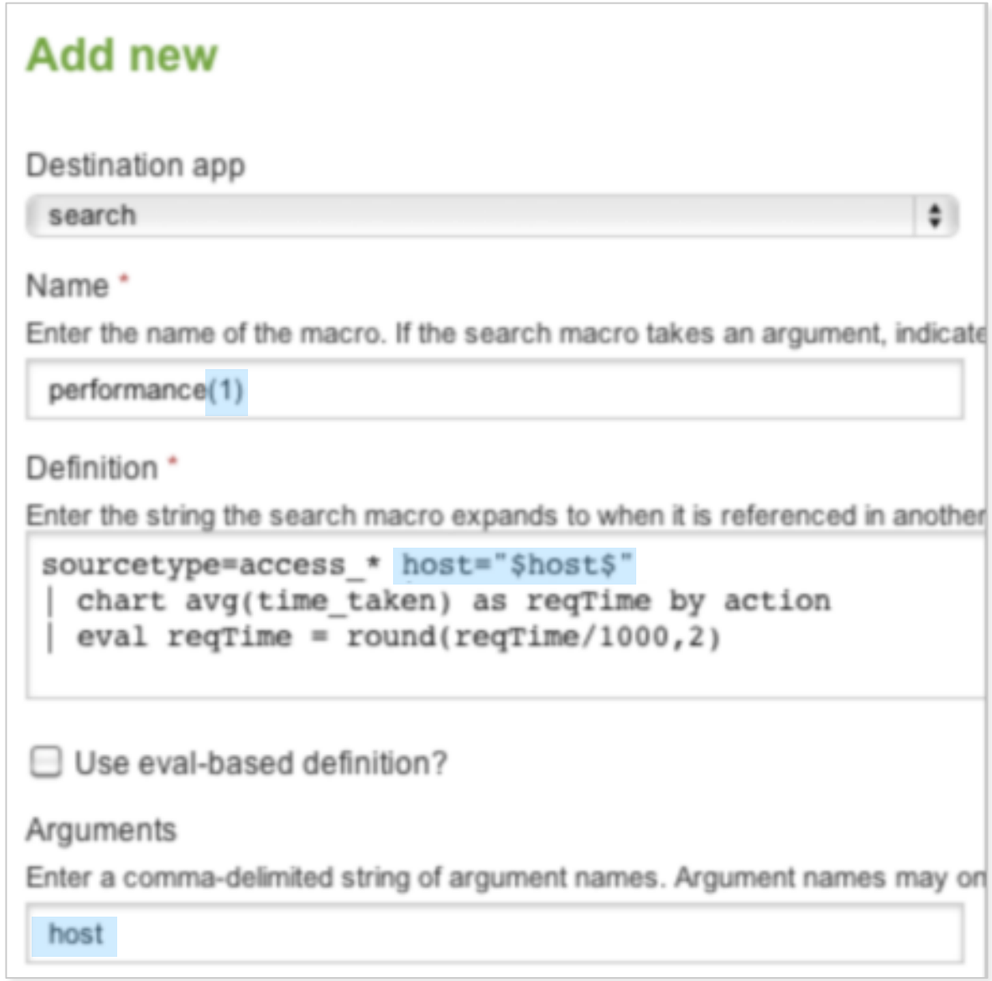
```
sourcetype="cisco_esa" | transaction mid dcid icid
```

Overlay: None

	mailto	count
1	evan@espphotography.com	13
2	bob@bargainave.com	11
3	cedwards@cnesystems.net	10
4	cedwards@napavalleycomputers.com	5
5	ebuceci@gmail.com	5
6	slickdeals@espphotography.com	4
7	lennartvandenende@gmail.com	3
8	news@jebian.com	2
9	sharon@hurstcastle.net	2
10	568_round4_trp@jbrowne.com	1
11	A.Portillo@anaportillo.com	1

Adding arguments

- Change one or more variables of the macro at search time
- Include the number of arguments in parentheses after the macro name
  - `performance(1)`
- Within the search definition, use `$arg$`
  - `host="$host$"`
- In the `arguments` field, enter the name of the argument



Using arguments

- When using a macro with arguments, include the argument in parentheses following the macro name

`performance(www3)`

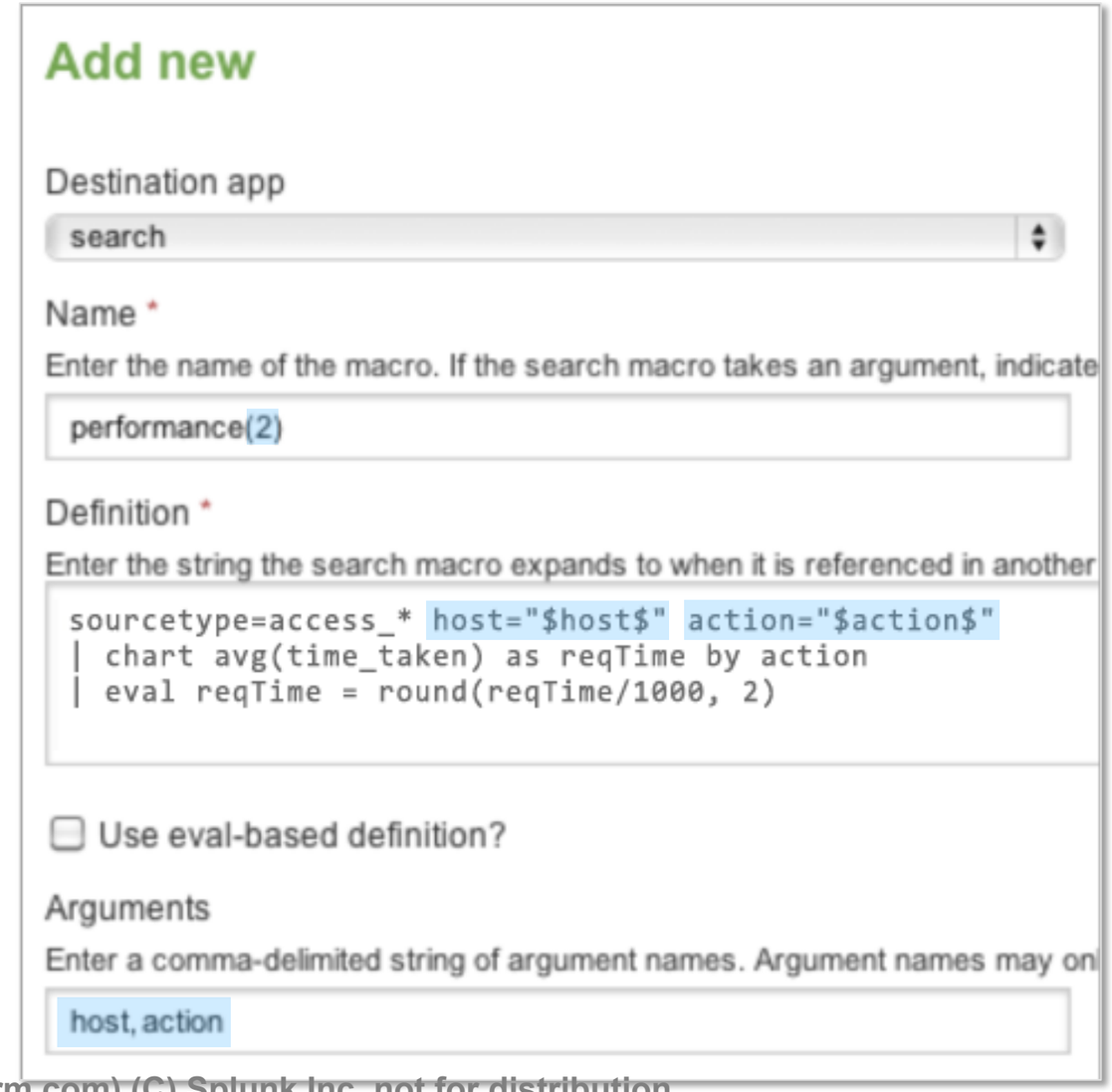
```
sourcetype=access_* host="$host$"
| chart avg(time_taken) as reqTime by action
| eval reqTime = round(reqTime/1000, 2)
```

Overlay: None

	action	reqTime
1	purchase	2.49
2	update	2.50

Adding multiple arguments

- When creating multiple arguments in the same macro:
  - **Name**: Include the number of variables in parenthesis in the macro name
    - performance(2)
  - **Definition**: Include both arguments in the definition
    - host="$host$" action="$action$"
  - **Arguments**: Enter the argument names, separated by commas
    - host, action



Using multiple arguments

- Include the arguments in parenthesis, separated by commas
  - Include arguments in the same order as the macro definition

```
`performance(host=www4, action=purchase)`
```

```
sourcetype=access_* host="$host$" action="$action$"
| chart avg(time_taken) as reqTime by action
| eval reqTime = round(reqTime/1000, 2)
```

Overlay: None

	action	reqTime
1	purchase	2.53

Lab 8

- **Time:**
  - 15-30 minutes
- **Tasks:**
  - Create a basic macro
  - Use the basic macro in a search
  - Create a macro with arguments
  - Use the macro and pass the arguments in a search

Support programs

- **Community**
  - **Splunkbase Answers:** answers.splunk.com
    Post specific questions and get them answered by Splunk community experts.
  - **Splunk Docs:** docs.splunk.com
    These are constantly updated. Be sure to select the version of Splunk you are using.
  - **Wiki:** wiki.splunk.com
    A community space where you can share what you know with other Splunk users.
  - **IRC Channel:** #splunk on the EFNet IRC server Many well-informed Splunk users “hang out” here.
- **Global Support**
  Support for critical issues, a dedicated resource to manage your account – 24 x 7 x 365.
  - **Email**: support@splunk.com
  - **Web:** http://www.splunk.com/index.php/submit_issue
- **Enterprise Support**
  Access you customer support team by phone and manage your cases online 24 x 7 (depending on support contract).

.conf2013: The 4th Annual Splunk WWUC

- Las Vegas: Sept 30-Oct 3, 2013
  - **The Cosmopolitan**
- 1500+ IT & Business Professionals
- 8 Tracks across 8 solution areas
- 3 days of content, 100+ sessions
- 2 days of Splunk University
- 30+ Customer speakers
- 25+ Apps in Splunkbase Labs
- 20+ Technology Partners

http://www.splunk.com/goto/conf

Thank You

Please fill out the class survey

http://www.surveymonkey.com/s/splunkclasses

Appendix A: Summary Indexing

Section objectives

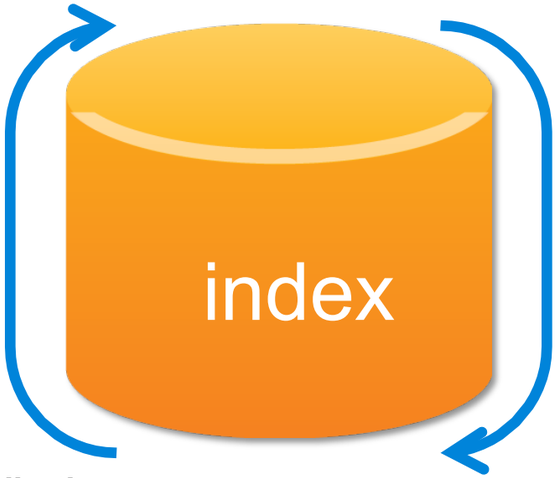
- Define summary indexing
- Create and schedule a summary search
- Populate a summary index
- Run searches against a summary index
- Identify gaps and overlaps in the summary index
- Correct gaps and overlaps in the summary index

Summary indexing overview

- Efficiently report on large volumes of data
- Spread the cost of a computationally expensive report over time
- Common use cases include:
  - Run reports over long time ranges for large datasets more efficiently
    - Example: Show the number of page views and visitors each of your web sites had over the past 30 days, broken out by site
  - Build a rolling report that shows aggregated statistics over long period of time
    - Example: Display a running count of downloads for a specific file on a website
    - Example: Calculate the average amount spent per purchase over a year

Without summary indexing….

- In this example, we're running a search for the top products purchased within the last 30 days
- The site gets millions of hits per day
- Searching the default index for a 30-day time span is *not efficient* as it scans across the entire data set for the specified 30-day period



With summary indexing…

- The pre-calculated statistics you want to report on are incrementally added to a summary index via a saved, scheduled search
- Data is stored in a special table format that optimizes efficiency
  - Each time the saved search runs, it appends the data in the summary index
- Searches or reports that may take several minutes or more to complete can be generated quickly

Summary indexing steps

1. Create a saved search using the si prefix that extracts broad statistics you want to report on
2. Schedule the search to run periodically over an appropriate time interval (hourly, daily, every *n*-minutes) depending on your needs
   - Each time the search runs, it saves the results since the last run into a summary index that you designate
3. Run searches and reports on this smaller, "faster" summary index instead of working with the much larger dataset

Summary indexing flow

3.

```
index=summary search_name=purchasedProducts | sistats count by product_name
```

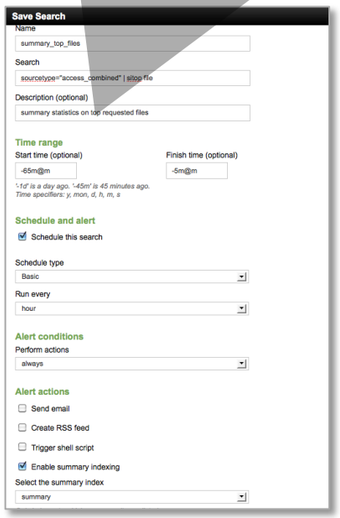
Last 30 days

purchasedProducts search

```
sourcetype="access_combined" | sistats count by product_name
```

Run every hour, time range 1 hour

1.



index

2.

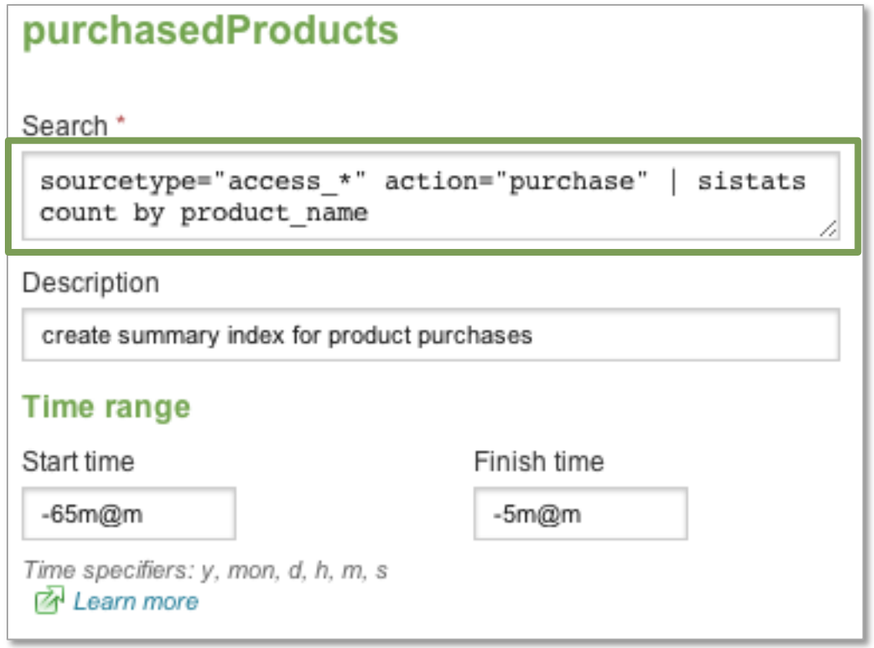
Populating search results are formatted and stored as statistics

Summary index

	product_name	psrsvd_gc	psrsvd_v
1	Birthday Bouquet	2139	1
2	Cake Serving Set	2185	1
3	Chocolate Dreams Confections	2174	1
4	Day Spa Certificate	2210	1
5	Dozen Red Roses	2096	1
6	Greetings Fruit Basket	2087	1
7	Mixed Rose Bouquet	2196	1
8	Sweet Dreams Bouquet	2117	1
9	Sweet Splendor Bouquet	2143	1
10	Tulip Bouquet	2173	1

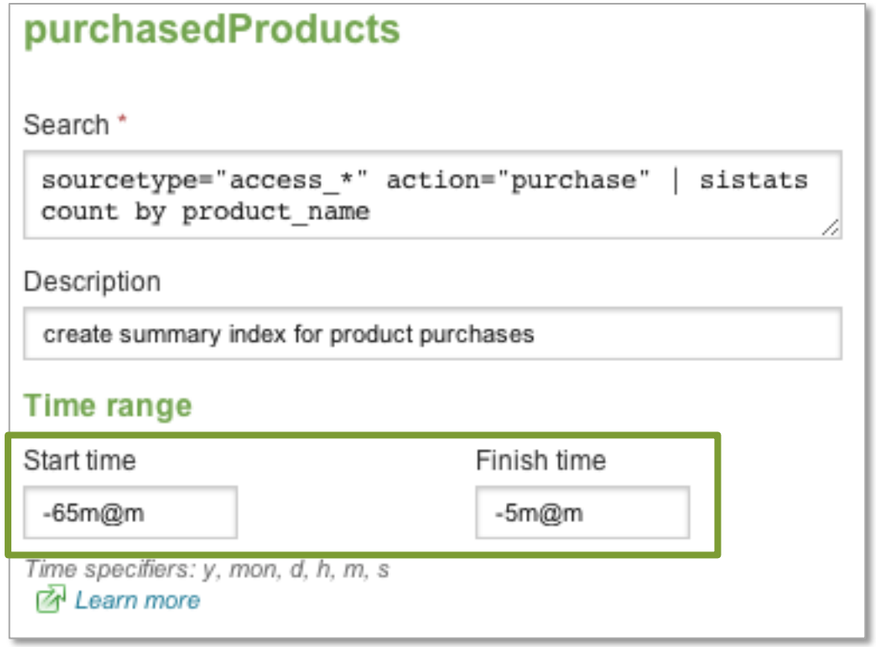
Creating the summary search

- Don't pipe other search operators after the main summary indexing reporting command
  - Save that for the searches you run *against* the summary indexes
- Create a broad enough search to capture the data you want
- Use the proper reporting command to get the statistics you want for further processing later



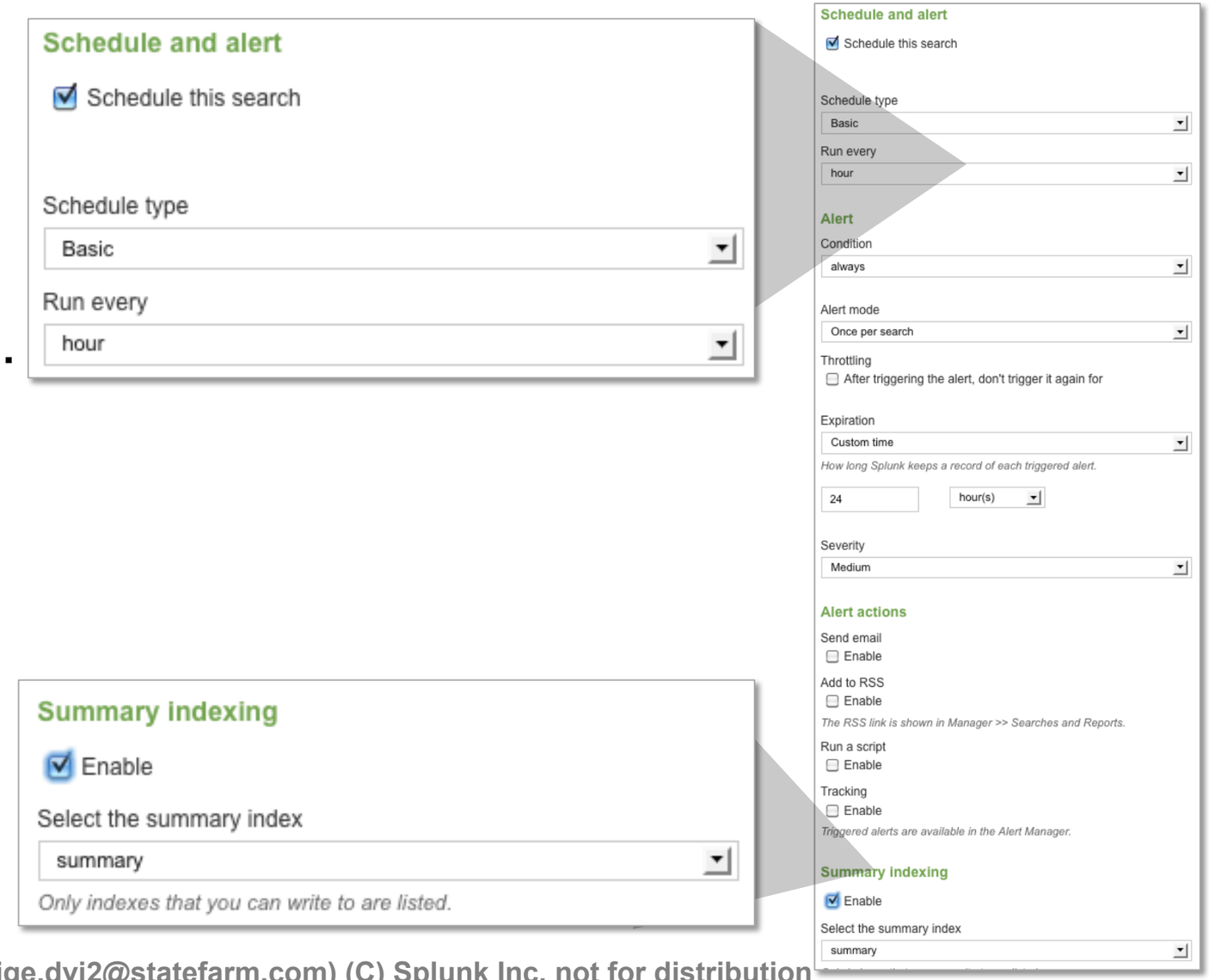
Creating the summary search (cont'd)

- Make the time span of the search smaller than the time span of the reports you want to run against it
  - To report on a month or a year, make the summary search for -24h hours and run every 24 hours
  - For a day, run for an hour timespan every hour



Schedule and save the search

- If the time range you defined is 1 hour, schedule to run every hour
  - For 1 day, schedule 1 day, etc…
- Be sure to enable summary indexing
- Select the default summary index, or a new index you created



Run searches against summary

- You're limited to the initial reporting command you used to create the index, but you can pipe to additional computations or operators
  - Saved search:

```
sourcetype=access_* | sistats count by product_name
```

  - Search on index:

```
index=summary search_name=purchasedProducts
| stats count by product_name
| <additional commands>
```

Summary index gaps

Gaps are usually caused by two main factors:

- A summary index initially only contains events from the point that you start data collection
  - There won't be data from before the summary index collection start date
  - You can put it in there yourself with the backfill script
- splunkd outages
  - If splunkd goes down for a significant amount of time, you could get gaps in your summary data, depending on the populating search schedules

Identifying gaps and overlaps

- To identify gaps and overlaps in your data, run a search against the summary index that uses the `overlap` command
  - Identify suspected gaps / overlaps in a search by specifying a `start time` and `end time` and a saved search name, followed by the `overlap` command in the search string
  - Full details can be found at: http://docs.splunk.com/Documentation/Splunk/latest/Knowledge/Managesummaryindexgapsandoverlaps

Correcting gaps

- Use the backfill script `fill_summary_index.py` to correct gaps or backfill your summary index
  - Specify the App
  - Specify a list of summary searches to backfill
  - Or, specify to backfill all summary searches for the App
- Use `-dedup true` to ensure the backfill script does not create duplicates

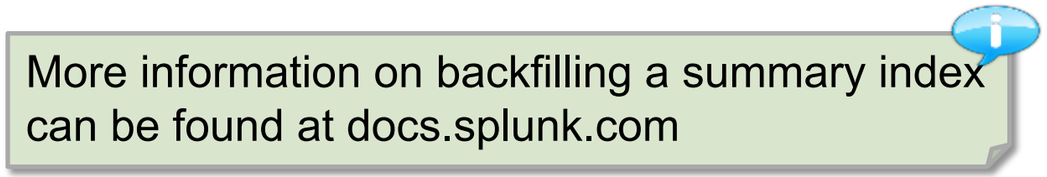
Correcting gaps (cont'd)

- To backfill a summary index that already has data in it:

```
./splunk cmd python fill_summary_index.py -app flowerstore -name "*"
 -et -mon@mon -lt @mon -dedup true -auth admin:changeme
```

  -dedup ensures the backfill script does not create an overlap

- To backfill a summary index that does not have any data, don't use -dedup



Summary index overlaps

- Overlaps are events in a summary index (from the same index-populating search) that share the same timestamp
- Overlapping events skew reports and statistics created from summary indexes
- Overlaps can occur if you set the time range of a saved search to be longer than the scheduled search interval
  - For example, don't arrange for an hourly search to gather data for the past 90 minutes
- Manually delete the overlaps from the summary index by using the search language