splunk>

Splunk Fundamentals 2 – Lab Exercises

Lab typographical conventions:

`{student ID}` indicates you should replace this with your student number.

[`sourcetype=vendor_sales`] OR [`cs_mime_type`] indicates either a source type or the name of a field.

NOTE: This is a lab environment driven by data generators with obvious limitations. This is not a production environment. Screenshots approximate what you should see.

There are a number of source types used in these lab exercises.

Index	Type	Sourcetype	Interesting Fields
web	Online sales	access_combined	action, bytes, categoryId, clientip, itemId, JSESSIONID, price, productId, product_name, referer, referer_domain, sale_price, status, user, useragent
security	Active Directory	winauthentication_security	LogName, SourceName, EventCode, EventType, User
	Badge reader	history_access	Address_Description, Department, Device, Email, Event_Description, First_Name, last_Name, Rfid, Username
	Web server	linux_secure	action, app, dest, process, src_ip, src_port, user, vendor_action
sales	Business Intelligence server	sales_entries	AcctCode, CustomerID, TransactionID
	Retail sales	vendor_sales	AcctID, categoryId, product_name, productId, sale_price, Vendor, VendorCity, VendorCountry, VendorID, VendorStateProvince
network	Email security data	cisco_esa	dcid, icid, mailfrom, mailto, mid
	Web security appliance data	cisco_wsa_squid	action, cs_method, cs_mime_type, cs_url, cs_username, sc_bytes, sc_http_status, sc_result_code, severity, src_ip, status, url, usage, x_mcafee_virus_name, x_wbrs_score, x_webcat_code_abbr
	Firewall data	cisco_firewall	bcg_ip, dept, Duration, fname, IP, lname, location, rfid, splunk_role, splunk_server, Username

games	Game logs	SimCubeBeta	date_hour, date_mday, date_minute, date_month, date_second, data_wday, data_year, date_zone, eventtype, index, linecount, punct, splunk_server, timeendpos, timestartpos

splunk>

Lab Exercise 1 – Beyond Search Fundamentals

Description

This exercise reviews the concepts presented in Module 1, including using the Job Inspector.

NOTE: If at any point you do not see results, check your search syntax and/or expand your time range.

Questions

Examine these searches. Which searches would not return results?

1. index=security sourcetype=linux_secure
2. index=web Sourcetype=access_combined [No results]
3. index=web sourcetype=AcceSS_Combined
4. index=security sourcetype=linux_se% [No results]

What is the most efficient filter?

time

Identify the 3 Selected Fields that Splunk returns by default for every event.

host source sourcetype

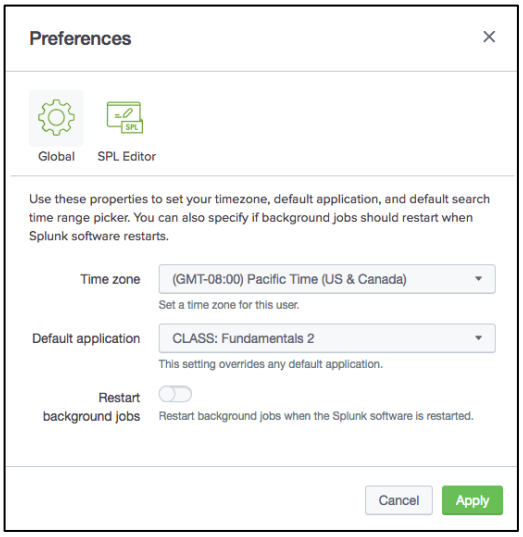
Steps

Task 1: Log into Splunk on the classroom server.

1. Direct your web browser to the class lab system.
2. Log in with the credentials your instructor assigned.

Task 2: Make the CLASS: Fundamentals 2 your default app and change your account time zone setting to reflect your local time.

3. Click your login name on the navigation bar and select **Account Settings**.
4. In the **Full name** field, type your full name and click **Save**.
5. Click the refresh button on your browser and ensure that your name now appears in the Splunk bar.
6. Click your name on the navigation bar and select **Preferences**.
7. From the **Time zone** dropdown, select your local time zone.
8. From the Default app dropdown, select CLASS: Fundamentals 2.



9. Click **Apply**.

NOTE:	**CLASS: Fundamentals 2** is a custom app designed specifically for this training course. It contains custom menu options, such as the Presentation menu, which contains all of the search strings used in the slides. Only searches saved in this app count towards completing the class. When you're in the **CLASS: Fundamentals 2** app, it will be indicated on the right side of the app navigation bar at the top of your screen.
NOTE:	**Do not copy and paste text** from the lab document except when instructed to do so, as quotes and double quotes may not copy as intended.

Task 3: Use the Search Job Inspector to troubleshoot problems.

10. Navigate to the **CLASS: Fundamentals 2 app**. Perform and save all your searches in this app.
11. Search for `index=web sourcetype=access_combined productid=*` over the **last 15 minutes**. Be sure to type exactly as shown, retaining case (i.e., lower case rather than upper case).
    Are any results returned? _____ no
12. Click **Job > Inspect Job** to open the Search Job Inspector and inspect the results.
13. Now, search for `index=web sourcetype=access_combined productId=*` over the **last 15 minutes**. Be sure to retain case.
    Are any results returned? _____ yes
14. Open the Search Job Inspector again and inspect the results.

Scenario: IT wants to check for issues with customer purchases in the online store.

15. Search for online sales transactions (`index=web sourcetype=access_combined action=purchase status=200`) during the **last 30 days**. Using the `table` command, display only the customer IP [`clientip`], the customer action [`action`], and the http status [`status`] of each event.
    Be sure to include an index in your search.

```
index=web sourcetype=access_combined action=purchase status=200
| table clientip, action, status
```

Task 4: Use Search Job Inspector to view performance.

16. Search for `index=web sourcetype=access_combined` over the **last 30 days** using the Verbose search mode, then open the Job Inspector (Job > Inspect Job). How much time did it take for the search to complete? _____
17. Run the same search using the Fast search mode. How much time did it take for the search job to complete? _____
18. Switch the default search mode back to Smart Mode.

NOTE: Given the small amount of data in our lab environment, comparing the Fast mode and Smart mode completion times probably won't produce useful data.

Lab Exercise 2 – Using Transforming Commands for Visualizations

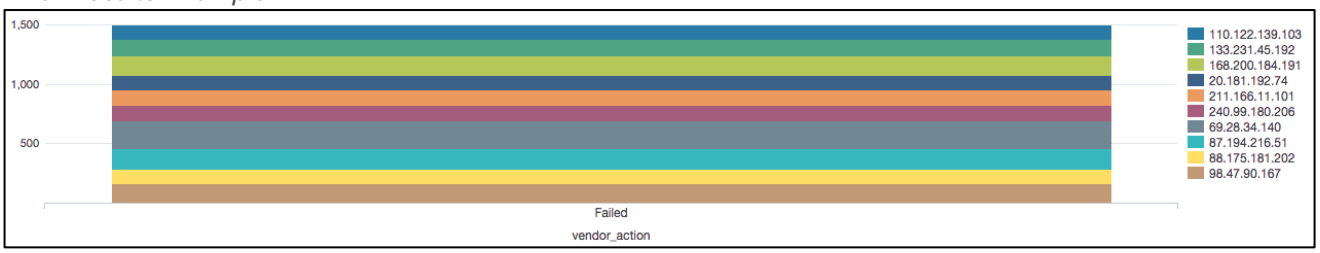
Description

In this lab exercise, you use the `chart` and `timechart` commands.

Steps

Task 1: Report the top ten completed events on the web server during the last 24 hours and add it to a new security dashboard as a column chart.

Final Results Example:



1. Search the web server [`sourcetype=linux_secure`] for events where the [`vendor_action`] is not equal to "`session opened`" during the **last 24 hours**.

   index=security sourcetype=linux_secure vendor_action!="session opened"

Results Example:

i	Time	Event
>	7/24/19 8:04:59.000 PM	Wed Jul 24 2019 20:04:59 www0 sshd[37002]: Failed password for user myuan from 133.166.61.223 port 5826 ssh2 host = www1 source = /opt/log/www1/secure.log sourcetype = linux_secure
>	7/24/19 8:04:56.000 PM	Wed Jul 24 2019 20:04:56 www0 sshd[94890]: Failed password for user myuan from 133.166.61.223 port 5826 ssh2 host = www1 source = /opt/log/www1/secure.log sourcetype = linux_secure
>	7/24/19 8:04:53.000 PM	Wed Jul 24 2019 20:04:53 www0 sshd[91204]: Failed password for user myuan from 133.166.61.223 port 5826 ssh2 host = www1 source = /opt/log/www1/secure.log sourcetype = linux_secure

2. Using the `chart` command, display a count for each of these actions by IP [`src_ip`].

   Hint: Use `over … by`

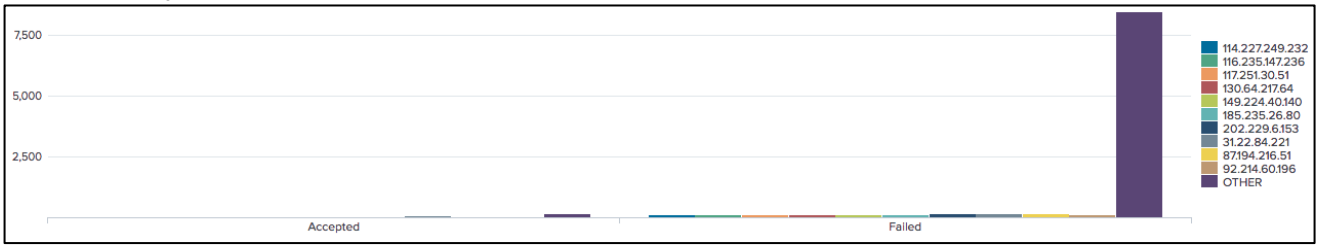
   index=security sourcetype=linux_secure vendor_action!="session opened"
   | chart count over vendor_action by src_ip

Results Example:

vendor_action	114.227.249.232	116.235.147.236	117.251.30.51	130.64.217.64	149.224.40.140	185.235.26.80	202.229.6.153	31.22.84.221	87.194.216.51	92.214.60.196	OTHER
Accepted	33	21	30	21	23	48	29	80	0	52	188
Failed	121	146	131	128	134	131	151	175	176	132	8478

3. Click on the **Visualization** tab and make sure **Column Chart**  is selected.

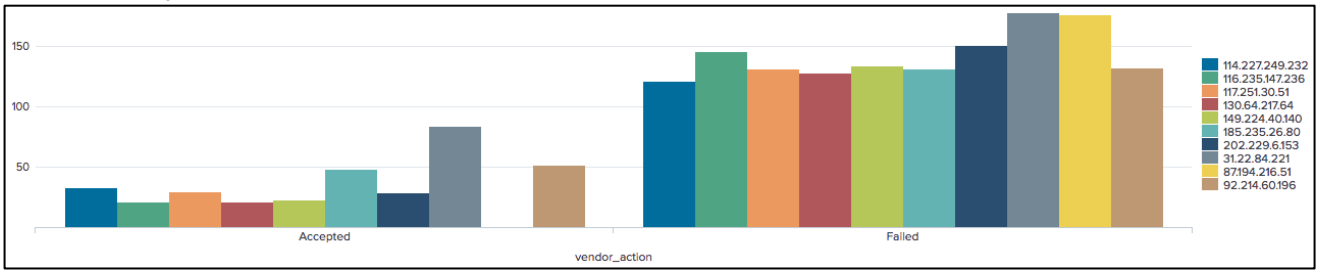
Results Example:



4. As you can see, there is an OTHER column at the end of the Failed results that overwhelms all the other data on the chart, making the other data difficult to see. Set the `useother` option to `f` in order to remove this column.

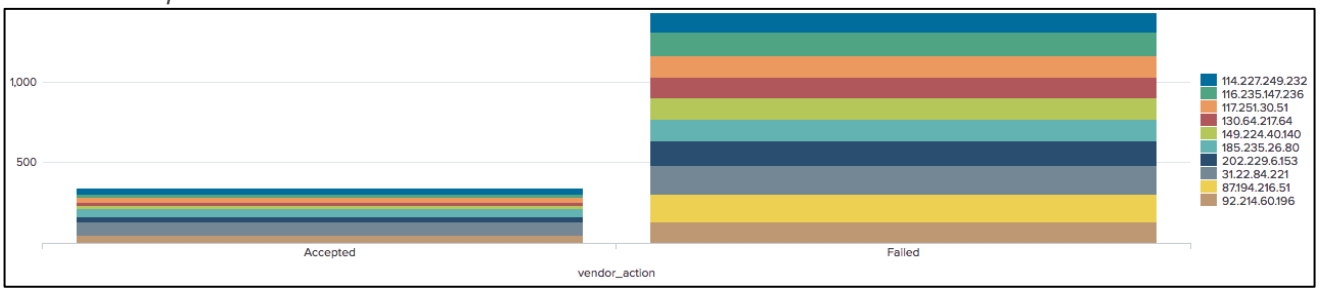
```
index=security sourcetype=linux_secure vendor_action!=”session opened”
| chart count over vendor_action by src_ip useother=f
```

Results Example:



5. Click **Format**; in the General section, set the Stack Mode to **Stacked**.

Results Example:

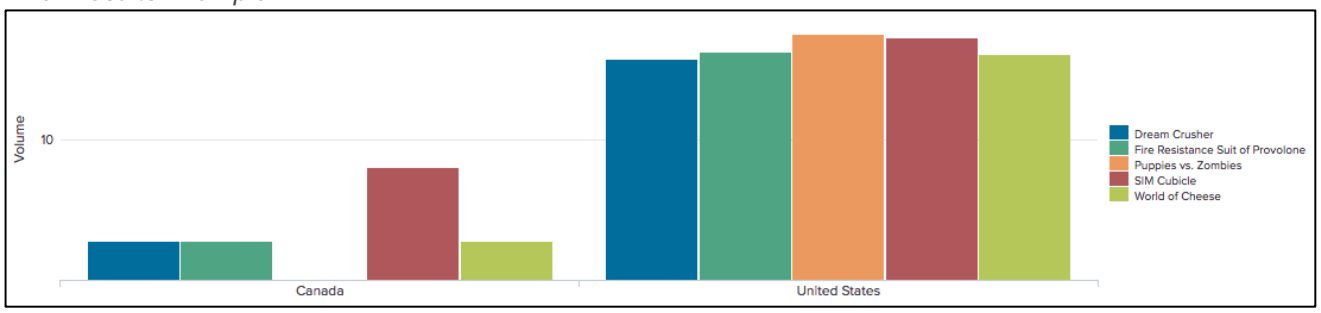


6. Click **Save As** and choose **Report**.
7. Name your report **L2S1** and click **Save**.

8. On the Your Report Has Been Created screen, click **Add to Dashboard**.
9. Save the dashboard with these values:
   - Dashboard: *New*
   - Dashboard Title: *IT Ops*
   - Panel Title**:** Potential Security Breaches
   - Panel Powered By: *Report*
10. Click **Save** and view your dashboard.
11. Mouse over your column chart and click one of the bars. Notice that, by default, the drilldown feature is not activated.
12. Click the **Edit** button.
13. Click the More actions icon  on the top right of the panel.
14. Click **Edit Drilldown**.
15. In the Drilldown Editor, choose **Link to search** from the **On click** dropdown menu.
16. Click **Apply.**
17. Click **Save** to save the dashboard.
18. Mouse over your column chart and click one of the bars. Notice that the drilldown feature is now activated.
19. Use your browser's Back button to return to your dashboard. (This is the easiest way to return to the dashboard from a drilldown.)

Task 2: Chart by country the five best selling products for the vendors in North America during the last 7 days.

Final Results Example:



- VendorID:

- 1000-2999 USA
- 3000-3999 Canada
- 4000-4999 Caribbean, Central & South America
- 5000-6999 Europe and the Middle East
- 7000-8999 Asia and Pacific Region
- 9000-9900 Africa
- 9901-9999 Outliers, such as the South Pole

20. Search for retail store events [`vendor_sales`] from North America (United States and Canada) during the **last 7 days**.

    index=sales sourcetype=vendor_sales VendorID<4000

Results Example:

i	Time	Event
>	2/5/18 9:19:28.000 AM	[05/Feb/2018:17:19:28] VendorID=1106 Code=F AcctID=xxxxxxxxxxx1352 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales
>	2/5/18 9:19:08.000 AM	[05/Feb/2018:17:19:08] VendorID=3106 Code=H AcctID=xxxxxxxxxxx0271 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales
>	2/5/18 9:17:12.000 AM	[05/Feb/2018:17:17:12] VendorID=1149 Code=N AcctID=xxxxxxxxxxx9840 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales

21. Using the `chart` command, count the events over `VendorCountry`.

```
index=sales sourcetype=vendor_sales VendorID<4000
| chart count over VendorCountry
```

Results Example:

VendorCountry	count
Canada	303
United States	4839

22. To see the count of each product sold in each country, add a `by` clause to further split the data by `product_name`.

```
index=sales sourcetype=vendor_sales VendorID<4000
| chart count over VendorCountry by product_name
```

Results Example:

VendorCountry	Dream Crusher	Final Sequel	Fire Resistance Suit of Provolone	Holy Blade of Gouda	Manganiello Bros.	Manganiello Bros. Tee	OTHER	Puppies vs. Zombies	SIM Cubicle	World of Cheese	World of Cheese Tee
Canada	22	17	24	17	36	9	101	7	24	31	15
United States	538	297	404	308	306	311	747	517	536	565	314

23. Use the `limit` option to include only the 5 best-selling products.

NOTE: Splunk automatically calculates the top products by totaling each column and taking the top *n* results (*n* being the number you specify in your limit).

```
index=sales sourcetype=vendor_sales VendorID<4000
| chart count over VendorCountry by product_name limit=5
```

Results Example:

VendorCountry	Dream Crusher	Holy Blade of Gouda	Puppies vs. Zombies	SIM Cubicle	World of Cheese	OTHER
Canada	1	3	0	2	3	27
United States	68	51	67	71	68	304

24. Remove the **OTHER** column from your table.

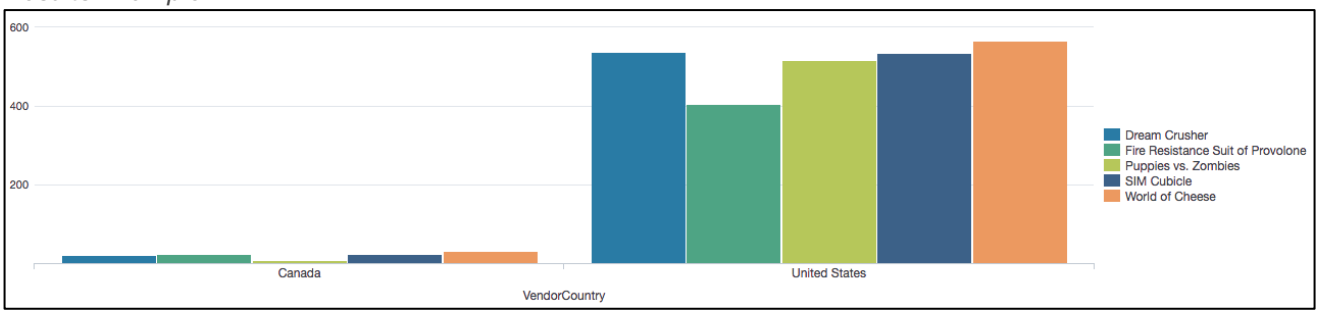
```
index=sales sourcetype=vendor_sales VendorID<4000
| chart count over VendorCountry by product_name limit=5 useother=f
```

Results Example:

VendorCountry	Dream Crusher	Fire Resistance Suit of Provolone	Puppies vs. Zombies	SIM Cubicle	World of Cheese
Canada	22	24	7	24	31
United States	538	404	517	536	565

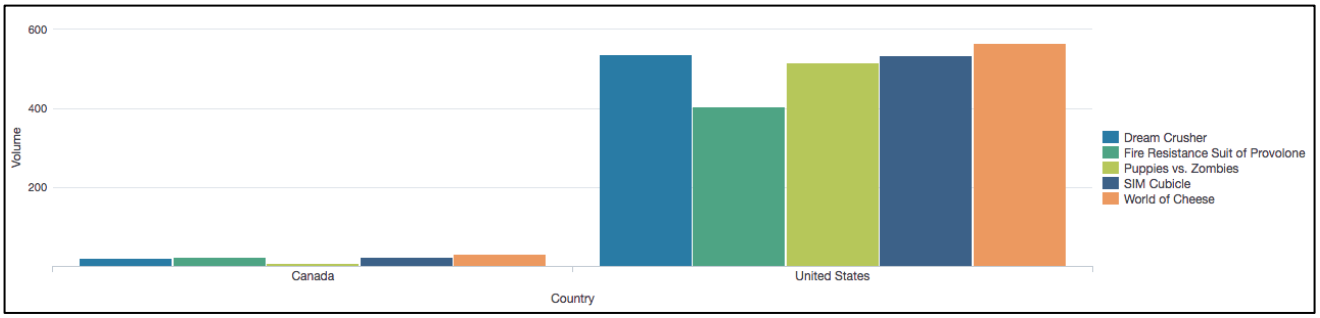
25. Switch to the **Visualization** tab and, if a column chart was not automatically shown, set the chart type to **Column Chart**.

Results Example:

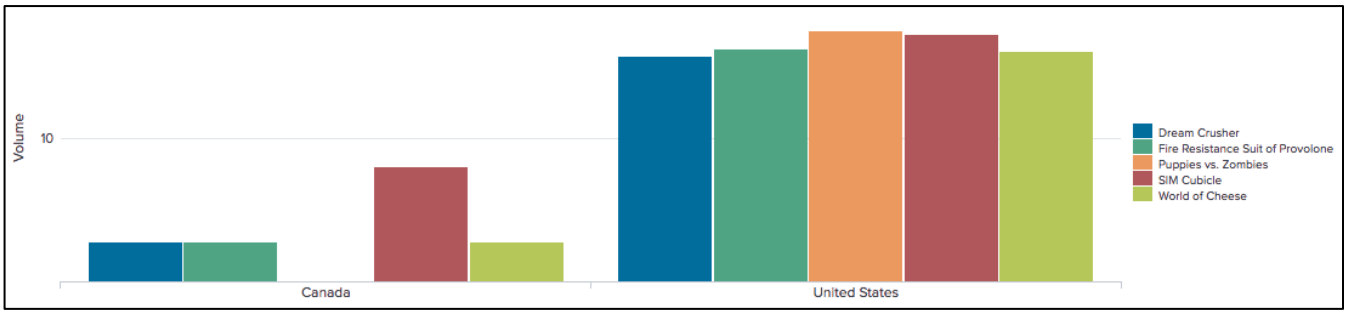


26. Use the **Format** options to define custom labels of **Country** and **Volume** for the X and Y axes, respectively.

Results Example:



27. Use the **Format** option to change the scale of the Y axis from linear to logarithmic (Log).



28. Save your search as report, **L2S2**.

Task 3: Display Internet usage in a timechart during the last 24 hours.

29. Click **Search** to clear the previously set **Format** options.
30. Search for web appliance events [`cisco_wsa_squid`] during the **last 24 hours**.

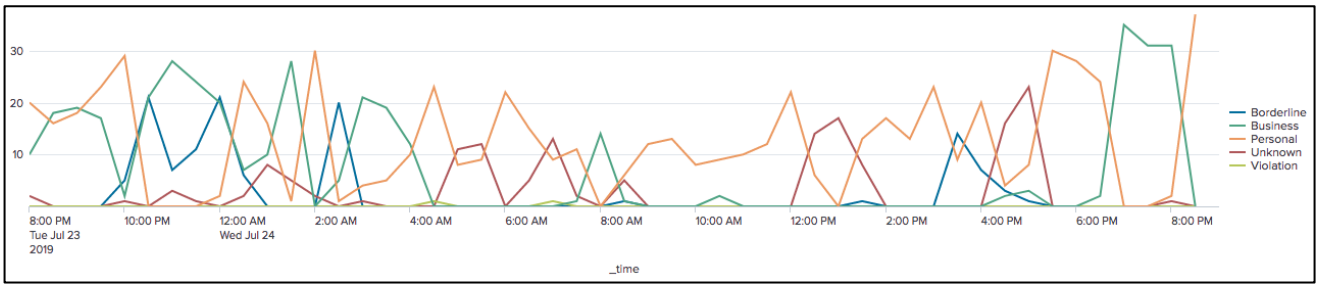
```
index=network sourcetype=cisco_wsa_squid
```

31. Use the `timechart` command to count the events by `usage`.

```
index=network sourcetype=cisco_wsa_squid
| timechart count by usage
```

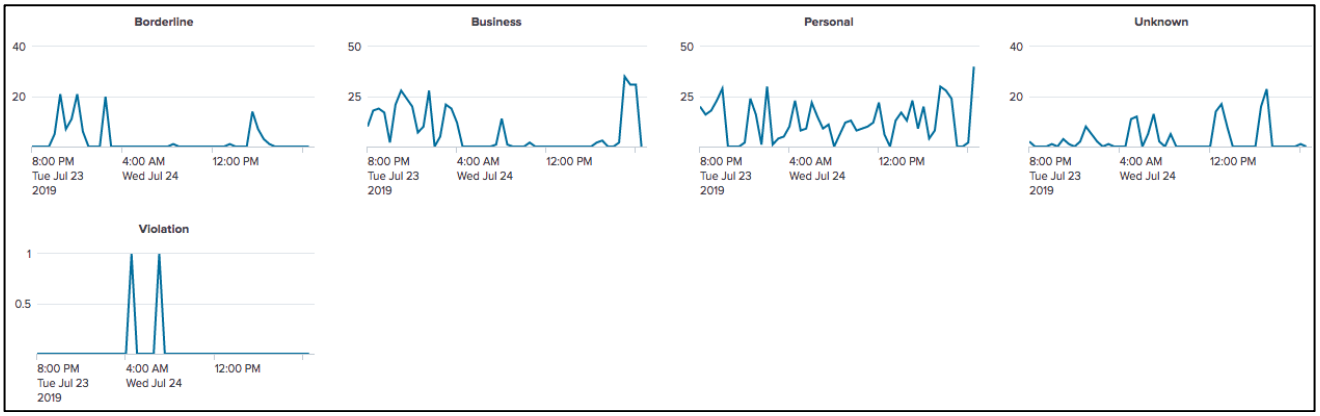
32. Change the visualization to **Line Chart**.

Results Example:



33. Save the search as report, **L2S3**.
34. Add this report to your *IT Ops* dashboard in a panel named: **Internet Usage - Last 24 Hours**. Do **not** click the button to view the dashboard; instead, close the Your Dashboard Panel Has Been Created window by clicking the x in the upper right corner. (If you accidentally do click **View Dashboard**, click your browser's Back button to get back to the L2S3 report.)
35. Click on **Trellis**.
36. Click the **Use Trellis Layout** checkbox.
37. For Scale, click **Independent**.

Results Example:



38. Save the search as a report, **L2S4**.
39. Add this report to your IT Ops dashboard in a panel named: **Internet Usage by Category**.
40. Edit your dashboard and arrange your panels so that the dashboard looks like this:

Results Example:

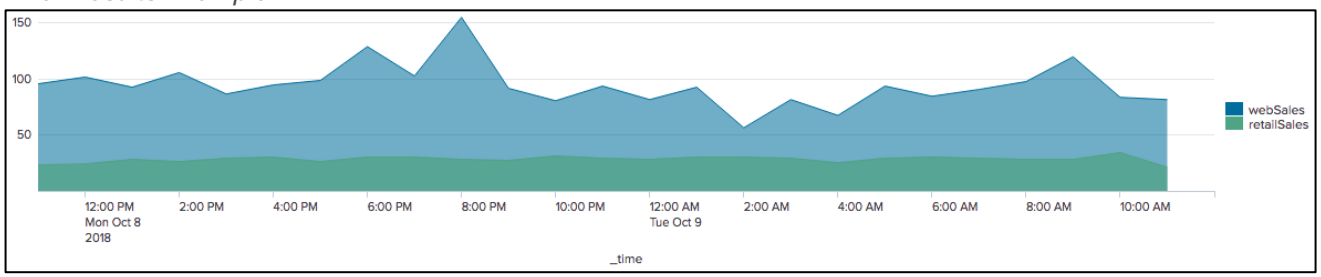
41. Click **Save**.

NOTE:	Visualization formatting options persist until you turn them off or change them. So the next time you do a visualization, by default, it will appear as a line chart with the Trellis option, because that's what you chose previously. And if that's not what you want, just change the options—turn off the Trellis option, choose a different type of visualization, etc.

CHALLENGE Exercise:

Display and compare online and vendor sales during the last 24 hours.

Final Results Example:



42. Search for successful online purchase events [`access_combined`] during the **last 24 hours** and enclose the entire search string in parentheses. (As you continue to modify this search string in the upcoming lab steps, the parentheses will be helpful.)

    (index=web sourcetype=access_combined action=purchase status=200)

43. Modify the search string to also search for all retail sales [`vendor_sales`]. Enclose this new clause in a separate set of parentheses.

    Hint: Use OR to view events from multiple indexes and sourcetypes (not AND).

    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)

44. Use `timechart` to count the sales events by `sourcetype`. Change the sampling interval to 1 hour.
    Hint: View the results in the **Statistics** tab to see the time values.

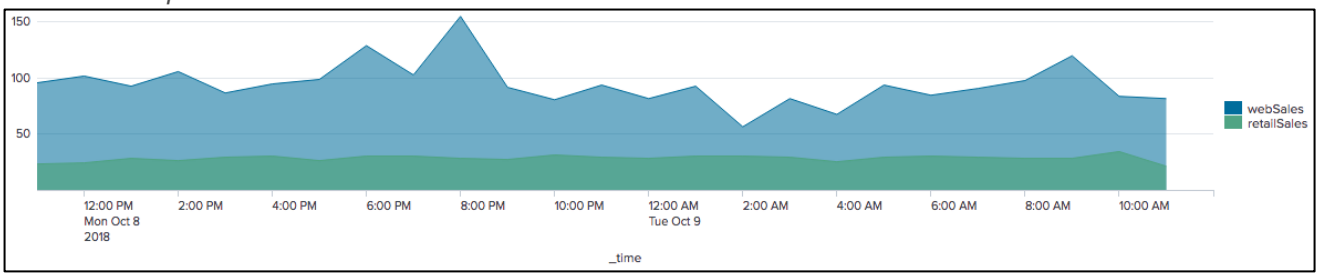
    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)
    | timechart span=1h count by sourcetype

45. Rename the access_combined column to webSales and the vendor_sales column to retailSales.

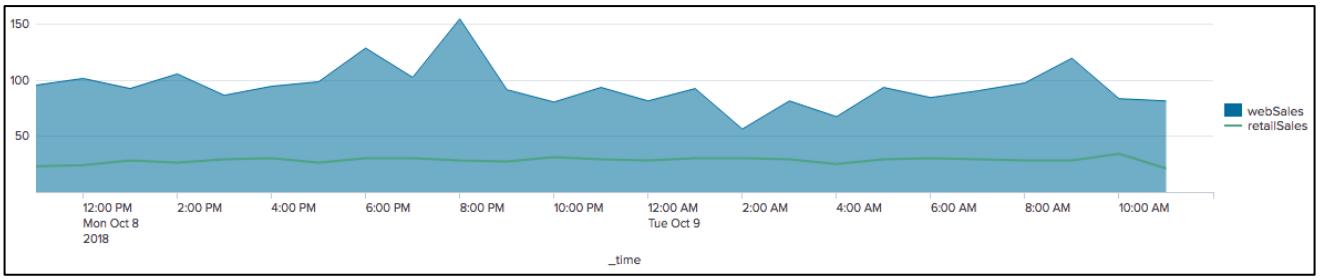
    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)
    | timechart span=1h count by sourcetype
    | rename access_combined as webSales, vendor_sales as retailSales

46. Display the results as an **Area Chart**.

Results Example:



47. Save the search as report, **L2C1**.
48. Optionally, revise the formatting to show `retailSales` as a chart overlay, and save as **L2C2**.



Lab Exercise 3 – Using Trendlines, Mapping, and Single Value Commands

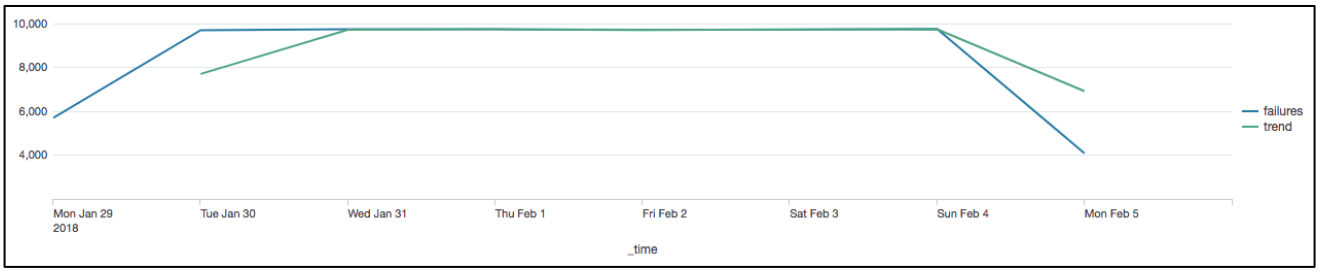
Description

In this lab exercise, use `trendline`, `iplocation`, `geostats`, `geom` and `addtotals` commands – as well as the single-value, choropleth map, and cluster map visualizations.

Steps

Task 1: Display web server failures during the last 7 days in a timechart with a trendline.

Final Example:



1. Search for failed password attempts on the web server [`linux_secure`] during the **last 7 days**.

```
index=security sourcetype=linux_secure fail*
```

Results Example:

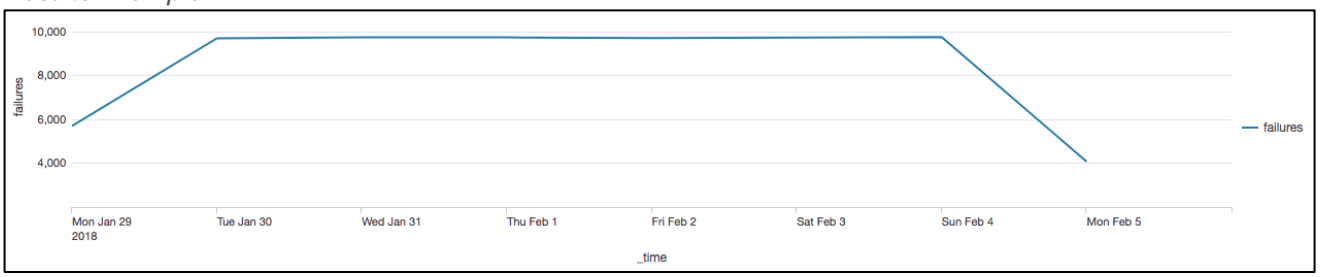
i	Time	Event
>	2/5/18 10:02:05.000 AM	Mon Feb 05 2018 18:02:05 www1 sshd[1224]: Failed password for root from 223.205.219.67 port 3411 ssh2 host = www1 source = /opt/log/www1/secure.log sourcetype = linux_secure
>	2/5/18 10:02:05.000 AM	Mon Feb 05 2018 18:02:05 www3 sshd[2063]: Failed password for invalid user perl from 202.179.8.245 port 2722 ssh2 host = www3 source = /opt/log/www3/secure.log sourcetype = linux_secure

2. Using `timechart`, count the events for each day and rename this new column as `failures`.

```
index=security sourcetype=linux_secure fail*
| timechart span=1d count as failures
```

3. Change the visualization to **Line Chart**.

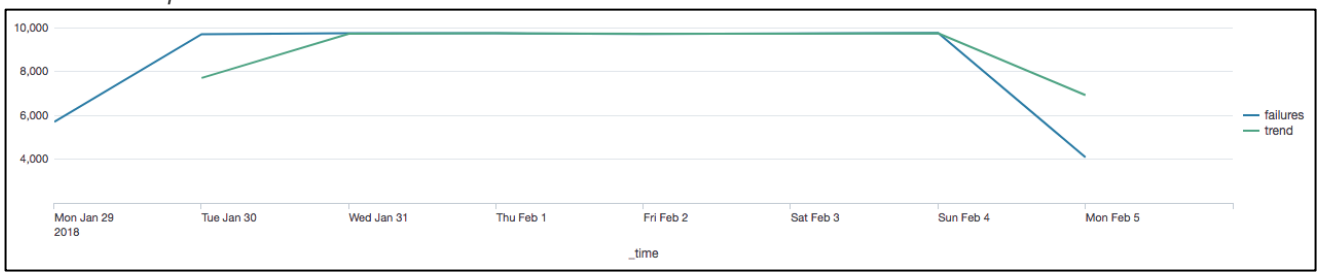
Results Example:



4. Find the `trendline` of failures using a simple moving average (`sma2`) and name the field as `trend`.

```
index=security sourcetype=linux_secure fail*
| timechart span=1d count as failures
| trendline sma2(failures) as trend
```

Results Example:



5. Save your search as report, **L3S1**

Task 2: Display the sales count of strategy games per day at Buttercup Games physical sales locations (i.e., not online) during the previous week.

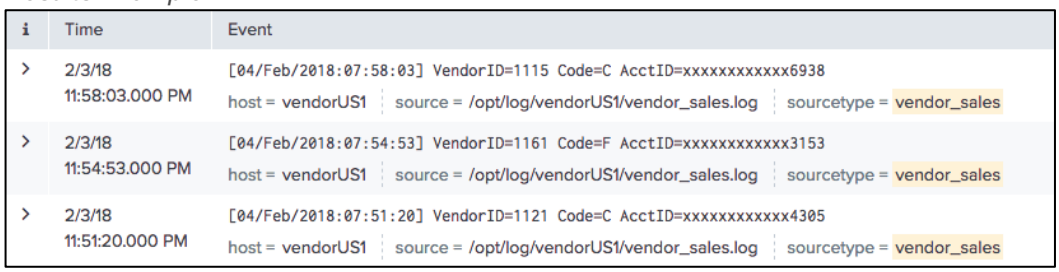
Final Results Example:

6. Search for retail sales [`vendor_sales`] of strategy games [`categoryId`="STRATEGY"] during the **previous week**.

```
index=sales sourcetype=vendor_sales categoryId= "STRATEGY"
```

NOTE: Since the `categoryId` comes from a lookup, the value being matched is case-sensitive. Therefore, be sure to type "STRATEGY" in all uppercase.

Results Example:



i	Time	Event
>	2/3/18 11:58:03.000 PM	[04/Feb/2018:07:58:03] VendorID=1115 Code=C AcctID=xxxxxxxxxxx6938 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales
>	2/3/18 11:54:53.000 PM	[04/Feb/2018:07:54:53] VendorID=1161 Code=F AcctID=xxxxxxxxxxx3153 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales
>	2/3/18 11:51:20.000 PM	[04/Feb/2018:07:51:20] VendorID=1121 Code=C AcctID=xxxxxxxxxxx4305 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales

7. Using `timechart`, count the sales per day of strategy games.

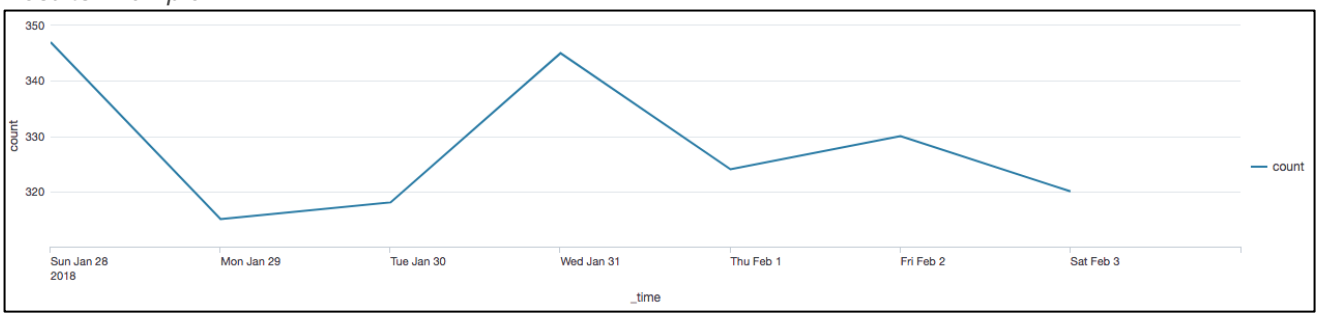
```
index=sales sourcetype=vendor_sales categoryId="STRATEGY"
| timechart span=1d count
```

Results Example

_time	count
2018-01-28	347
2018-01-29	315
2018-01-30	318
2018-01-31	345
2018-02-01	324
2018-02-02	330
2018-02-03	320

8. Change the visualization to **Line Chart**.

Results Example



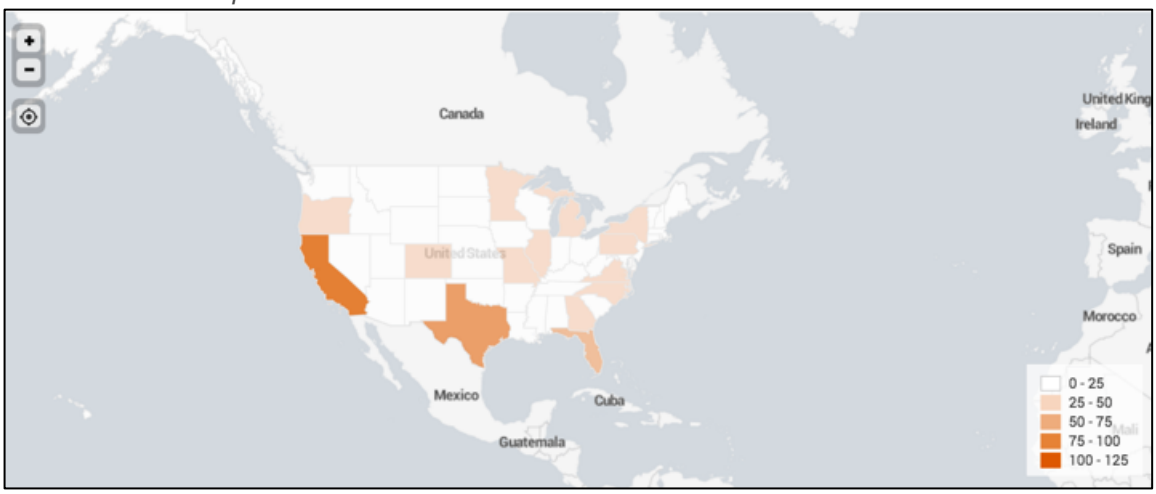
9. Change the visualization to **single value** with the following format:
   - Caption: Strategy Games Sales – Previous Day
   - Show Trend Indicator: *Yes*
   - Show Sparkline: *Yes*
   - Use Colors *Yes*
   - Color By: Trend
   - Color Mode: Set so that the background shows the color based on the trend (e.g., green for an increasing trend and red for a decreasing trend)

Results Example:

10. Save your search as report, **L3S2**.

Task 3: Display a choropleth map of United States retail sales during the last 7 Days.

Final Results Example:



11. Search for United States retail sales during the **last 7 Days**.

    Hint: United States vendors have a VendorID less than 3000.

    index=sales sourcetype=vendor_sales VendorID < 3000

Results Example:

i	Time	Event
>	2/5/18 10:19:38.000 AM	[05/Feb/2018:18:19:38] VendorID=1145 Code=A AcctID=xxxxxxxxxxx9888 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales
>	2/5/18 10:17:57.000 AM	[05/Feb/2018:18:17:57] VendorID=1205 Code=I AcctID=xxxxxxxxxxx5233 host = vendorUS1 source = /opt/log/vendorUS1/vendor_sales.log sourcetype = vendor_sales

12. Using the `chart` command, count the events over `VendorStateProvince`.

    index=sales sourcetype=vendor_sales VendorID<3000
    | chart count over VendorStateProvince

Results Example:

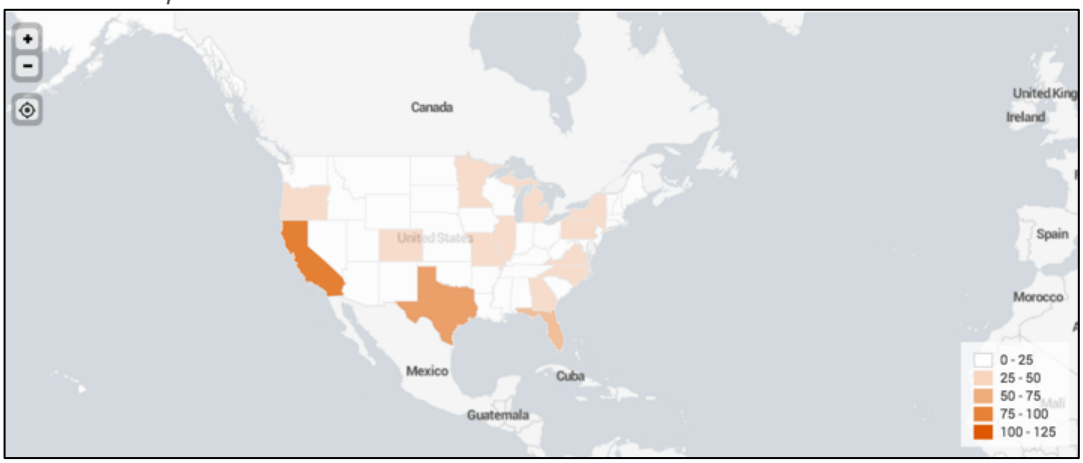
VendorStateProvince	count
Alabama	54
Alaska	81
Arizona	75
Arkansas	54
California	527

13. To display the data as a choropleth map, use the `geom` command to map `VendorStateProvince` to the geo_us_states KMZ file (`geom geo_us_states featureIdField=VendorStateProvince`).

```
index=sales sourcetype=vendor_sales VendorID<3000
| chart count by VendorStateProvince
| geom geo_us_states featureIdField=VendorStateProvince
```

14. Click the **Visualization** tab.
15. Change the visualization to use the **Choropleth Map**. 
16. Zoom in on the map so you can clearly see the United States.

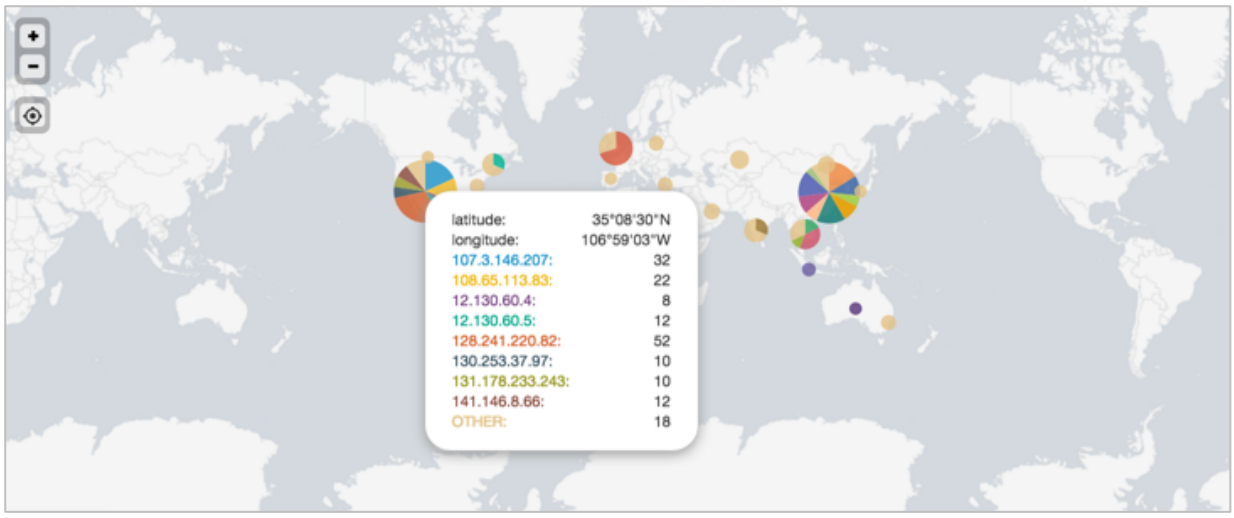
Results Example:



17. Click **Format**.
18. Click **Tiles**.
19. Click Populate from preset configuration.
20. Click Open Street Map.
21. Save your search as report, **L3S3**.

Task 4: Display a map of online sales by country during the previous week.

Final Results Example:



22. Find successful online purchases [`access_combined`] during the **Previous week**.
    Hint: You can use the Fields sidebar to narrow your search results. From `action`, select purchase and from `status`, 200.

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

Results Example:

i	Time	Event
>	2/3/18 11:58:53.000 PM	67.170.226.218 - - [04/Feb/2018:07:58:53] "POST /cart/success.do?JSESSIONID=SD4SL1FF9ADFF4965 HTTP 1.1" 200 379 "http://www.buttercupgames.com/cart.do?action=purchase&itemId=EST-26" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 756 host = www3 source = /opt/log/www3/access.log sourcetype = access_combined
>	2/3/18 11:58:53.000 PM	67.170.226.218 - - [04/Feb/2018:07:58:53] "POST /cart.do?action=purchase&itemId=EST-26&JSESSIONID=SD4SL1FF9ADFF4965 HTTP 1.1" 200 2892 "http://www.buttercupgames.com/cart.do?action=addtocart&itemId=EST-26&categoryId=SIMULATION&productId=SC-MG-G10" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 129 host = www3 source = /opt/log/www3/access.log sourcetype = access_combined

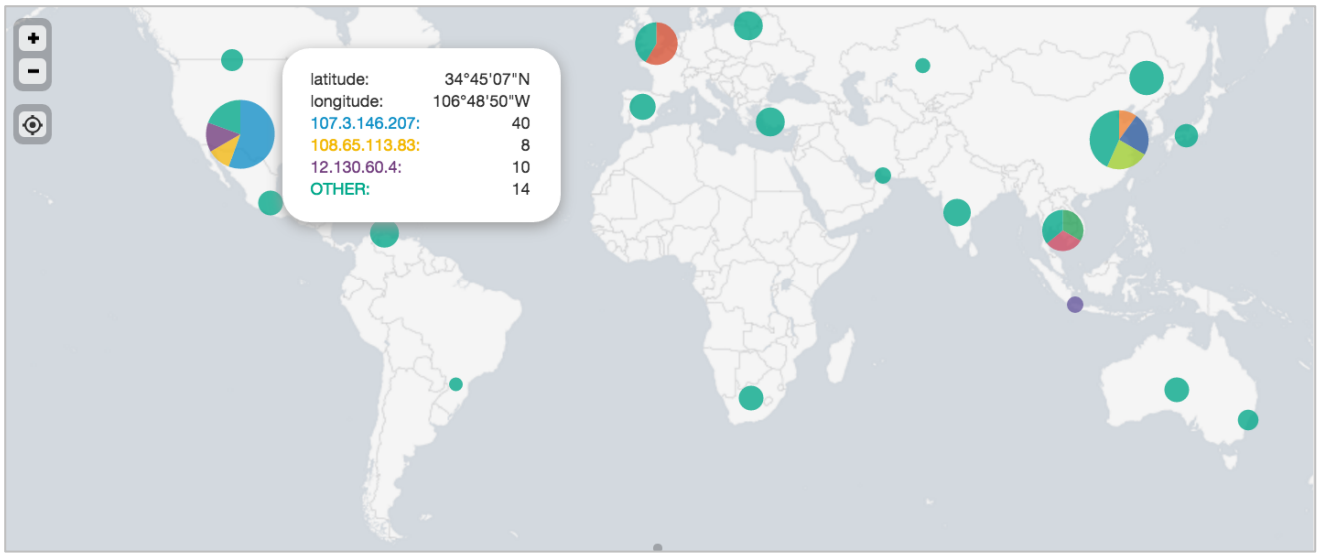
23. Use `iplocation` to extract the location of the purchases based on **clientip**. (You will see the `lat` and `lon` fields on the Fields sidebar.)

    index=web sourcetype=access_combined action=purchase status=200
    | iplocation clientip

24. To place the events on a map, use `geostats` to count by `clientip`. (Note that you may need to manually change the visualization to a Cluster Map,  .)

    index=web sourcetype=access_combined action=purchase status=200
    | iplocation clientip
    | geostats count by clientip

Results Example:



25. Save your search as report, **L3S4**.

Task 5: Count the retail sales units sold by country and include a grand total row.

26. Count the number of retail store purchases [`vendor_sales`] by `VendorCountry` during the **last 4 hours**, and rename the new column to "Units Sold."

```
index=sales sourcetype=vendor_sales
| stats count as "Units Sold" by VendorCountry
```

Results Example:

VendorCountry	Units Sold
Argentina	1
Australia	2
Belarus	1
Bermuda	1

27. Use `addtotals` with the col and row options to display the column total and suppress the row total. Modify the search to include a `Total` label for the last row of the table.

```
index=sales sourcetype=vendor_sales
| stats count as "Units Sold" by VendorCountry
| addtotals col=t row=f labelfield="VendorCountry"
```

28. Scroll to the bottom of the last page of the results to see the last row of the table, as shown below.

Results Example:

Sweden	1
The Bahamas	1
Turkey	1
Ukraine	2
United Kingdom	4
United States	107
Venezuela	1
Vietnam	3
Total	177

29. Save your search as report, **L3S5**.

Lab Exercise 4 – Filtering Results and Manipulating Data

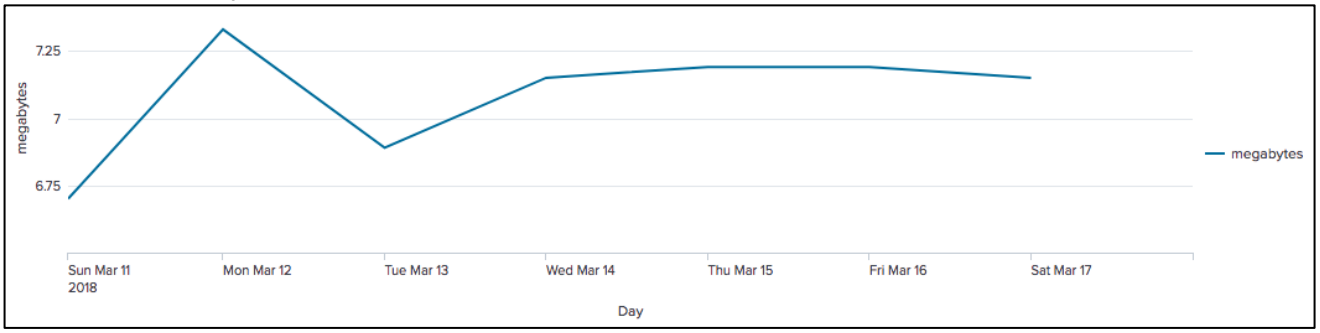
Description

In this lab exercise, you use `eval`, `search`, and `where` commands.

Steps

Task 1: Chart the total daily volume (in MB) of the web servers during the previous week.

Final Results Example:



1. Search online sales [`access_combined`] during the **previous week**.

   ```
   index=web sourcetype=access_combined
   ```

2. Use `timechart` to calculate the total `bytes` and name the field: `bytes`

   ```
   index=web sourcetype=access_combined
   | timechart sum(bytes) as bytes
   ```

Results Example:

_time	bytes
2018-03-11	7028552
2018-03-12	7685197
2018-03-13	7225343
2018-03-14	7501807
2018-03-15	7539912
2018-03-16	7543386
2018-03-17	7492738

3. Use `eval` to convert the `bytes` field to `megabytes`.

   ```
   sourcetype=access_combined
   | timechart sum(bytes) as bytes
   | eval megabytes=bytes/(1024*1024)
   ```

Results Example:

_time	bytes	megabytes
2018-03-11	7028552	6.702949523925781
2018-03-12	7685197	7.329174995422363
2018-03-13	7225343	6.890624046325684
2018-03-14	7501807	7.154280662536621
2018-03-15	7539912	7.190620422363281
2018-03-16	7543386	7.193933486938477
2018-03-17	7492738	7.145631790161133

4. Use the `round` function to round the `megabytes` field values to two decimal places.

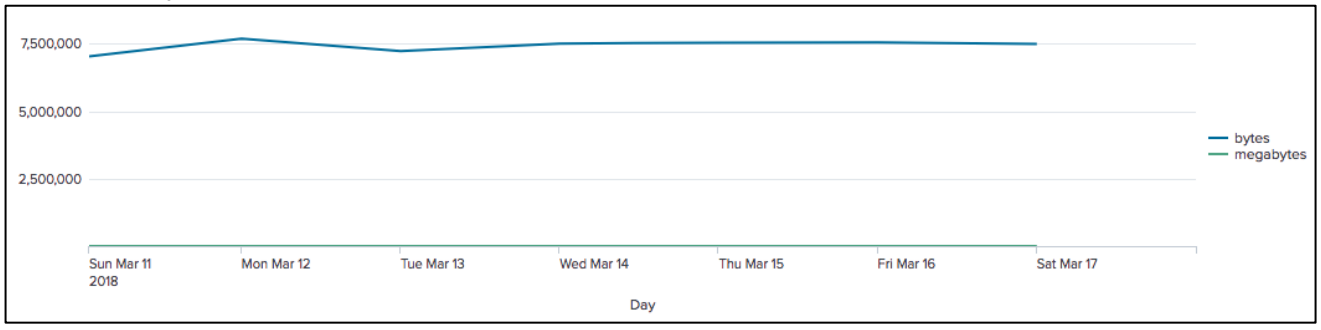
```
index=web sourcetype=access_combined
| timechart sum(bytes) as bytes
| eval megabytes=round(bytes/(1024*1024),2)
```

Results Example:

_time	bytes	megabytes
2018-03-11	7028552	6.70
2018-03-12	7685197	7.33
2018-03-13	7225343	6.89
2018-03-14	7501807	7.15
2018-03-15	7539912	7.19
2018-03-16	7543386	7.19
2018-03-17	7492738	7.15

5. Switch to the **Visualization** tab and display the data as a **Line Chart**. Set the X-axis label to **Day**. Notice that the `bytes` field still displays.

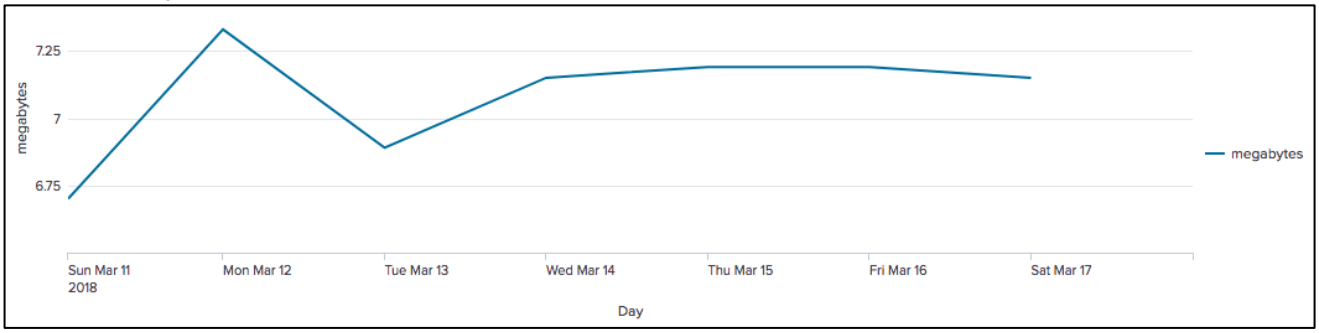
Results Example:



6. Use the `fields` command to remove the `bytes` field.

```
index=web sourcetype=access_combined
| timechart sum(bytes) as bytes
| eval megabytes=round(bytes/(1024*1024),2)
| fields - bytes
```

Results Example:



7. Save your search as report, **L4S1**.

Task 2: Calculate the ratio of GET requests to POST requests for each web server.

Final Results Example:

host	GET	POST	Ratio
www1	709	381	1.86
www2	766	456	1.68
www3	782	466	1.68

8. Search for all events in the online store [`access_combined`] during the **last 24 hours**.

```
index=web sourcetype=access_combined
```

9. Use `chart` to count events over `host` by `method`.

```
index=web sourcetype=access_combined
| chart count over host by method
```

Results Example:

host	GET	POST
www1	709	381
www2	766	456
www3	780	461

10. Use `eval` to create a new column called `Ratio`, which divides `GET` by `POST`.

```
index=web sourcetype=access_combined
| chart count over host by method
| eval Ratio=GET/POST
```

Results Example:

host	GET	POST	Ratio
www1	709	381	1.8608923884514437
www2	766	456	1.6798245614035088
www3	780	461	1.6919739696312364

11. Round the `Ratio` field to two decimal places.

```
index=web sourcetype=access_combined
| chart count over host by method
| eval Ratio=round(GET/POST,2)
```

Results Example:

host	GET	POST	Ratio
www1	709	381	1.86
www2	766	456	1.68
www3	782	466	1.68

12. Save your search as report, **L4S2**.

Task 3: Identify users with more than 3 failed logins during the last 60 minutes and sort in descending order.

Final Results Example:

user	count
myuan	105
nsharpe	51
root	16
djohnson	12
operator	11

13. Search the web server [linux_secure] for failed password attempts during the **last 60 minutes**.

```
index=security sourcetype=linux_secure fail*
```

Results Example:

i	Time	Event
>	2/5/18 11:53:29.000 AM	Mon Feb 05 2018 19:53:29 www1 sshd[5493]: Failed password for nobody from 147.213.138.201 port 4206 ssh2 host = www1 source = /opt/log/www1/secure.log sourcetype = linux_secure
>	2/5/18 11:53:29.000 AM	Mon Feb 05 2018 19:53:29 www2 sshd[2826]: Failed password for invalid user operator from 94.230.166.185 port 3791 ssh 2 host = www2 source = /opt/log/www2/secure.log sourcetype = linux_secure

14. Use `stats` to count the number of failed password attempts by user.

```
index=security sourcetype=linux_secure fail*
| stats count by user
```

Results Example:

user	count
admin	8
administrator	2
agushto	1
apache	1
art	1
backup	2

15. Using the `search` command, filter the results to include only users with more than three failures and sort in descending order.

```
index=security sourcetype=linux_secure fail*
| stats count by user
| search count>3
| sort -count
```

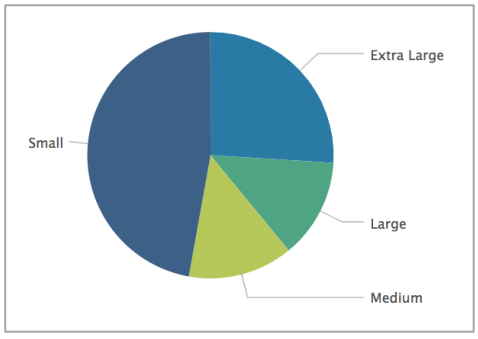
Results Example:

user	count
myuan	105
nsharpe	51
root	16
djohnson	12
operator	11

16. Save your search as report, **L4S3**.

Scenario: **Evaluate and classify the number of bytes associated with each web server event during the last 24 hours as a pie chart. (Event sizes should be categorized as follows: Small, < 2000 bytes; Medium, from 2000 to 2500 bytes; Large, from 2500 to 3000 bytes; Extra Large, over 3000 bytes.)**

Example of final output:

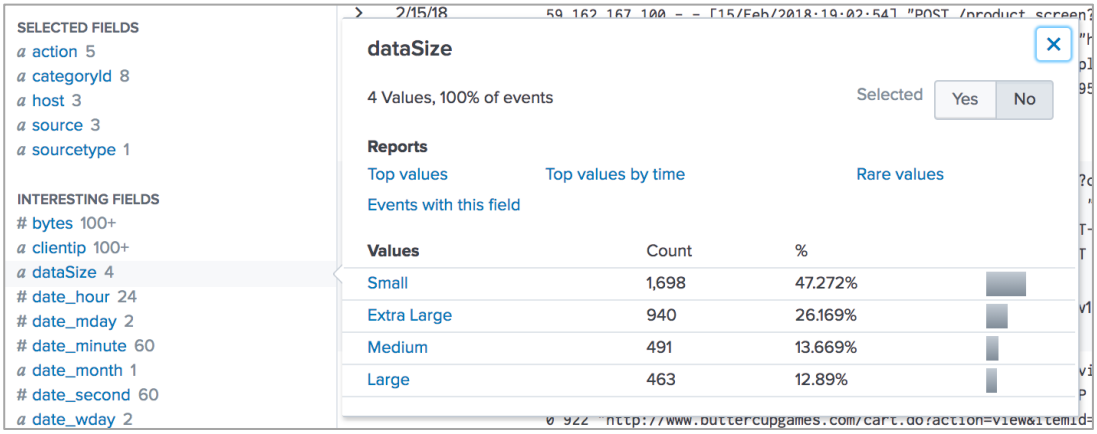


17. Search online transactions `[access_combined]` during the **last 24 hours** and—using the `case` function of the `eval` command—classify the size `(bytes)` of events into a field called `dataSize`. If the event is

less than 2,000 bytes, classify it as Small; if 2,000 or more but less than 2,500 bytes, classify as Medium; finally, if 2,500 or more but less than 3,000 bytes, classify as Large. Include a default value of Extra Large for all events where the bytes value is 3,000 or greater.

```
index=web sourcetype=access_combined
| eval dataSize = case(bytes<2000,"Small",
bytes<2500,"Medium",
bytes<3000,"Large",
true(),"Extra Large")
```

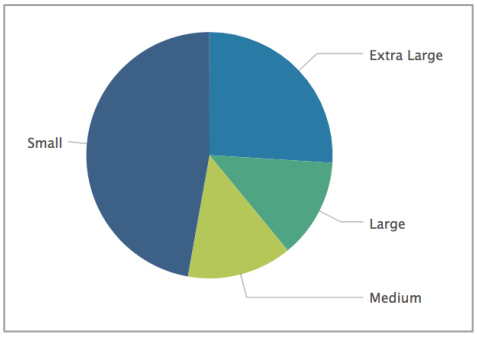
Results example:



18. Using `chart` or `stats`, count the events by `dataSize` and display the results as a pie chart.

```
index=web sourcetype=access_combined
| eval dataSize = case(bytes<2000,"Small", bytes<2500,"Medium",bytes<3000,"Large",true(),"Extra Large")
| chart count by dataSize
```

Results example:

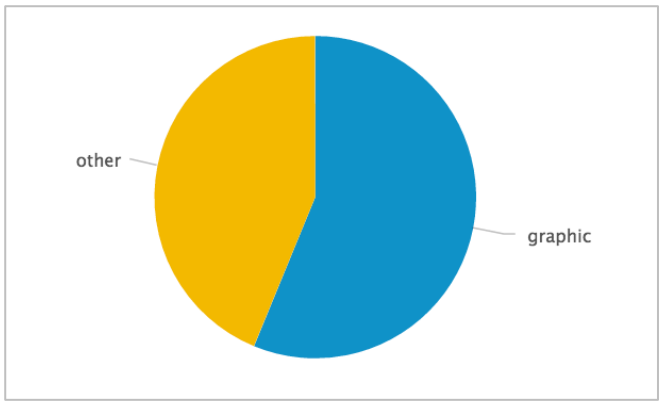


19. Save your search with the name **L4S4**.

CHALLENGE Exercise:

Classify and report employee web traffic by content type during the previous business week.

Final Results Example:



20. Search web appliance data [`cisco_wsa_squid`] during the **previous business week**.

```
index=network sourcetype=cisco_wsa_squid
```

21. Use `stats` or `chart` to count events by the `http_content_type` field.

```
index=network sourcetype=cisco_wsa_squid
| stats count by http_content_type
```

NOTE: In this case, `stats` and `chart` are interchangeable—they use the same syntax and return the same results.

Results Example:

http_content_type	count
-	818
application/javascript	111
application/octet-stream	63
application/x-dosexec	1
application/x-javascript	446
application/x-shockwave-flash	34
image/bmp	6

22. Use the `if` function of `eval` to create a new column named `type`. If the `http_content_type` value begins with “image”, set the `type` field to “graphic”. Otherwise, set the value to “other”.

    Hint: Use the LIKE operator and the % wildcard to define the expression as follows:
    `http_content_type LIKE "image%"`

```
index=network sourcetype=cisco_wsa_squid
| stats count by http_content_type
| eval type=if(http_content_type LIKE "image%","graphic","other")
```

Results Example:

http_content_type	count	type
-	818	other
application/javascript	111	other
application/octet-stream	63	other
application/x-dosexec	1	other
application/x-javascript	446	other
application/x-shockwave-flash	34	other
image/bmp	6	graphic

23. Use another `stats` or `chart` command to sum the `count` column by the `type` field. Rename the sum of the `count` calculation to `total`.

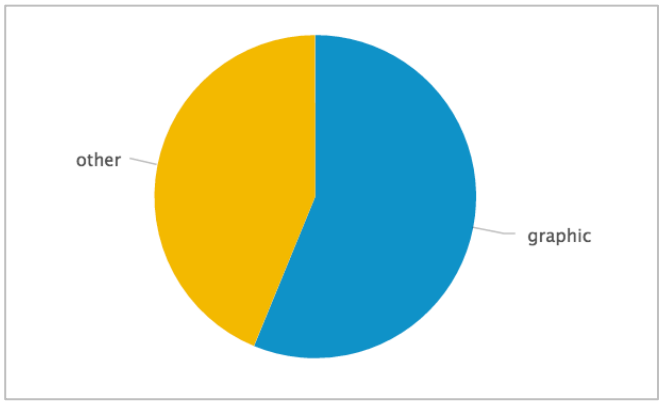
```
index=network sourcetype=cisco_wsa_squid
| stats count by http_content_type
| eval type=if(http_content_type LIKE "image%","graphic","other")
| stats sum(count) as total by type
```

Results Example:

type	total
graphic	3583
other	2296

24. Change the visualization to a **Pie Chart**.

Results Example:



25. Save your search as report, **L4C1**.

CHALLENGE Exercise:

Report which products sold twice as much in the Buttercup Games online store than in the retail store, or vice versa, during the previous week. Show the name of each of these products, as well as the number of units sold online and in the retail store.

Final Results Example:

product_name	access_combined	vendor_sales
Benign Space Debris	470	178
Orvil the Wolverine	556	220

26. Search online sales data [`access_combined`] and retail sales data [`vendor_sales`] for successful purchases during the **previous week**.

    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)

27. Chart a count of productId over product_name by sourcetype.

    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)
    | chart count(productId) as Count over product_name by sourcetype

Results Example:

product_name	access_combined	vendor_sales
Benign Space Debris	470	178
Curling 2014	339	221
Dream Crusher	685	465
Final Sequel	551	288
Fire Resistance Suit of Provolone	663	397
Holy Blade of Gouda	516	334
Manganiello Bros.	571	324
Manganiello Bros. Tee	577	375

28. Use a `where` command to keep only rows where the value in `access_combined` is more than twice the value in `vendor_sales` or the value in `vendor_sales` is more than twice the value in `access_combined`.

    (index=web sourcetype=access_combined action=purchase status=200) OR (index=sales sourcetype=vendor_sales)
    | chart count(productId) as Count over product_name by sourcetype
    | where access_combined > vendor_sales*2 OR vendor_sales > access_combined*2

Results Example:

product_name	access_combined	vendor_sales
Benign Space Debris	470	178
Orvil the Wolverine	556	220

29. Save your search as report, **L4C2**.
30. Modify your previous search to use `search` instead of `where`. Observe that the search produces no results. Why does this search produce no results?

```
(index=web sourcetype=access* action=purchase status=200) OR (index=sales
sourcetype=vendor_sales)
| chart count(productId) as Count over product_name by sourcetype
| search access_combined > vendor_sales*2 OR vendor_sales > access_combined*2
```

No results are found because the `search` command cannot compare values from two different fields. (As you saw earlier, the `where` command can do this.)

Lab Exercise 5 – Correlating Events

Description

Use the transaction command to correlate events.

Steps

Task 1: Analyze transactions in the online store during the last 60 minutes.

Final Results Example:

JSESSIONID	clientip	action
SD7SL8FF6ADFF4957	86.9.190.90	addtocart purchase view
SD6SL9FF5ADFF4961	81.18.148.190	addtocart purchase view
SD2SL10FF2ADFF4963	194.215.205.19	addtocart purchase remove

1. Search for all events in the online store [`access_combined`] during the **last 60 minutes**.

   index=web sourcetype=access_combined

2. Display a table that shows the `_time`, `clientip`, `JSESSIONID`, and the `action`. Note that the actions are listed in reverse chronological order (most to least recent.)

   index=web sourcetype=access_combined
   | table _time, clientip, JSESSIONID, action

Results Example:

_time	clientip	JSESSIONID	action
2018-02-05 12:40:03	211.166.11.101	SD0SL3FF5ADFF4950	
2018-02-05 12:39:45	211.166.11.101	SD0SL3FF5ADFF4950	
2018-02-05 12:37:35	211.245.24.3	SD6SL7FF4ADFF4956	
2018-02-05 12:37:18	211.245.24.3	SD6SL7FF4ADFF4956	addtocart
2018-02-05 12:28:05	91.199.80.24	SD1SL10FF7ADFF4953	
2018-02-05 12:27:55	91.199.80.24	SD1SL10FF7ADFF4953	purchase

3. Modify your search to only include events with a value in the `action` field.

   index=web sourcetype=access_combined action=*
   | table _time, clientip, JSESSIONID, action

Results Example:

_time	clientip	JSESSIONID	action
2018-02-05 12:44:02	195.2.240.99	SD0SL6FF5ADFF4959	view
2018-02-05 12:43:51	195.2.240.99	SD0SL6FF5ADFF4959	addtocart
2018-02-05 12:37:18	211.245.24.3	SD6SL7FF4ADFF4956	addtocart
2018-02-05 12:27:55	91.199.80.24	SD1SL10FF7ADFF4953	purchase
2018-02-05 12:27:55	91.199.80.24	SD1SL10FF7ADFF4953	purchase

4. Remove the `table` command and all the arguments being passed to it. Using the `transaction` command, create groups of transactions based on the `JSESSIONID` field.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
```

Results Example:

i	Time	Event
>	2/5/18 12:46:10.000 PM	194.215.205.19 - - [05/Feb/2018:20:46:10] "POST /cart.do?action=addtocart&itemId=EST-19&productId=PZ-SG-G05&JSESSIONID=SD2SL10FF2ADFF4963 HTTP 1.1" 200 3407 "http://www.buttercupgames.com/product.screen?productId=PZ-SG-G05" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 647 194.215.205.19 - - [05/Feb/2018:20:46:14] "POST /cart.do?action=purchase&itemId=EST-19&JSESSIONID=SD2SL10FF2ADFF4963 HTTP 1.1" 200 3746 "http://www.buttercupgames.com/cart.do?action=addtocart&itemId=EST-19&categoryId=STRATEGY&productId=PZ-SG-G05" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 936 194.215.205.19 - - [05/Feb/2018:20:46:14] "POST /cart/success.do?JSESSIONID=SD2SL10FF2ADFF4963 HTTP 1.1" 200 3014 "http://www.buttercupgames.com/cart.do?action=purchase&itemId=EST-19" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 911 194.215.205.19 - - [05/Feb/2018:20:46:23] "POST /cart.do?action=addtocart&itemId=EST-15&productId=MB-AG-T01&JSESSIONID=SD2SL10FF2ADFF4963 HTTP 1.1" 200 3572 "http://www.buttercupgames.com/product.screen?productId=MB-AG-T01" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 420 194.215.205.19 - - [05/Feb/2018:20:46:25] "POST /cart.do?action=purchase&itemId=EST-15&JSESSIONID=SD2SL10FF2ADFF4963 HTTP 1.1" 200 2743 "http://www.buttercupgames.com/cart.do?action=addtocart&itemId=EST-15&categoryId=TEE&productId=MB-AG-T01" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 830 Show all 9 lines host = www1 source = /opt/log/www1/access.log sourcetype = access_combined

5. Modify your search to display the transactions in a table. Include `JSESSIONID`, `clientip`, and `action`.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
| table JSESSIONID, clientip, action
```

Results Example:

JSESSIONID	clientip	action
SD6SL9FF5ADFF4961	81.18.148.190	addtocart purchase view
SD8SL6FF5ADFF4954	59.162.167.100	changequantity view
SD2SL10FF2ADFF4963	194.215.205.19	addtocart purchase remove
SD0SL6FF5ADFF4959	195.2.240.99	addtocart remove view

NOTE: By default, the values in the action column are ordered alphabetically, ignoring duplicates.

6. View only transactions that contain at least one purchase event. Use the `search` command to find transactions containing a purchase.

NOTE: The search command must be downstream from the transaction command.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
| table JSESSIONID, clientip, action
| search action=purchase
```

Results Example:

JSESSIONID	clientip	action
SD7SL8FF6ADFF4957	86.9.190.90	addtocart purchase view
SD6SL9FF5ADFF4961	81.18.148.190	addtocart purchase view
SD2SL10FF2ADFF4963	194.215.205.19	addtocart purchase remove

7. Save your search as report, **L5S1**.

Task 2: Display the online store purchase transactions lasting more than one minute and include the number of events in each transaction.

Final Results Example:

JSESSIONID	clientip	action	durationMinutes	eventcount
SD7SL8FF6ADFF4957	86.9.190.90	addtocart purchase view	1.3	11
SD1SL10FF7ADFF4953	91.199.80.24	addtocart purchase remove view	2.7	13
SD3SL8FF9ADFF4955	195.69.252.22	addtocart purchase remove view	1.4	9

8. If not already displayed, run your **L5S1** search again.
9. Set the search mode to **Verbose Mode**, which will re-execute your search.
10. Click the Events tab. Notice the new fields generated by the `transaction` command: `duration` and `eventcount`.

11. Modify your search to add the `duration` and `eventcount` fields to your table after the `clientip` field. Run your search in **Smart Mode**.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
| table JSESSIONID, clientip, duration, eventcount, action
| search action=purchase
```

Results Example:

JSESSIONID	clientip	duration	eventcount	action
SD7SL8FF6ADFF4957	86.9.190.90	77	11	addtocart purchase view
SD6SL9FF5ADFF4961	81.18.148.190	32	5	addtocart purchase view
SD2SL10FF2ADFF4963	194.215.205.19	46	9	addtocart purchase remove

12. Use `eval` to create a new field named `durationMinutes`, which is the rounded value of `duration` divided by 60. Round to one decimal place.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
| table JSESSIONID, clientip, duration, eventcount, action
| search action=purchase
| eval durationMinutes=round(duration/60,1)
```

Results Example:

JSESSIONID	clientip	duration	eventcount	action	durationMinutes
SD7SL8FF6ADFF4957	86.9.190.90	77	11	addtocart purchase view	1.3
SD6SL9FF5ADFF4961	81.18.148.190	32	5	addtocart purchase view	0.5
SD2SL10FF2ADFF4963	194.215.205.19	46	9	addtocart purchase remove	0.8

13. Modify your search to find data where the `durationMinutes` is greater than one minute. Also, remove the `duration` field from the table.

```
index=web sourcetype=access_combined action=*
| transaction JSESSIONID
| search action=purchase
| eval durationMinutes=round(duration/60,1)
| table JSESSIONID, clientip, action, durationMinutes, eventcount
| where durationMinutes > 1
```

Results Example:

JSESSIONID	clientip	action	durationMinutes	eventcount
SD7SL8FF6ADFF4957	86.9.190.90	addtocart purchase view	1.3	11
SD1SL10FF7ADFF4953	91.199.80.24	addtocart purchase remove view	2.7	13
SD3SL8FF9ADFF4955	195.69.252.22	addtocart purchase remove view	1.4	9

14. Save your search as report, **L5S2**.

Task 3: Search for online store transactions that begin with an `addtocart` action and end with a `purchase` action.

Final Results Example:

clientip	JSESSIONID	product_name	action	duration	eventcount	price
199.15.234.66	SD10SL10FF2ADFF4963	Dream Crusher	addtocart purchase	4	2	39.99
86.9.190.90	SD7SL8FF6ADFF4957	World of Cheese Tee	addtocart purchase	1	2	9.99
86.9.190.90	SD7SL8FF6ADFF4957	Holy Blade of Gouda	addtocart purchase	3	2	5.99

15. Search for all events from the online store [`access_combined`] in the **last 60 minutes** and correlate the events based on `clientip`.

```
index=web sourcetype=access_combined
| transaction clientip
```

16. Use the `startswith` and `endswith` options of the `transaction` command to display transactions that begin with an `addtocart` action and end with a `purchase` action.

```
index=web sourcetype=access_combined
| transaction clientip startswith=action="addtocart" endswith=action="purchase"
```

17. In a table, display clientip, JSESSIONID, product_name, action, duration, eventcount, and price.

```
index=web sourcetype=access_combined
| transaction clientip startswith=action="addtocart" endswith=action="purchase"
| table clientip, JSESSIONID, product_name, action, duration, eventcount, price
```

Results Example:

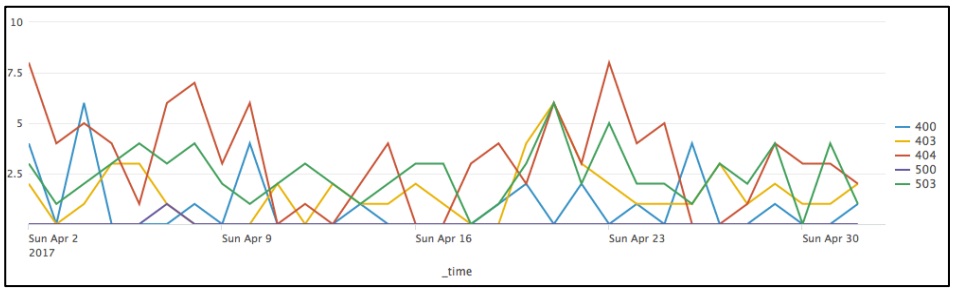
clientip	JSESSIONID	product_name	action	duration	eventcount	price
199.15.234.66	SD10SL10FF2ADFF4963	Dream Crusher	addtocart purchase	4	2	39.99
86.9.190.90	SD7SL8FF6ADFF4957	World of Cheese Tee	addtocart purchase	1	2	9.99
86.9.190.90	SD7SL8FF6ADFF4957	Holy Blade of Gouda	addtocart purchase	3	2	5.99

18. Save your search as report, **L5S3**.

CHALLENGE Exercise:

Report common HTTP status errors that occurred during the last 30 days on the online sales web servers and the internal web appliance within a proximity of 5 minutes or less. Only include days with more than 5 common errors.

Final Results Example:



1. Search HTTP status error events from the online sales web servers [`access_combined`] and the web appliance [`cisco_wsa_squid`] during the **last 30 days**. For best performance, limit extracted fields to only `sourcetype` and `status`.

```
(index=network sourcetype=cisco_wsa_squid) OR
(index=web sourcetype=access_combined) status>399
| fields sourcetype, status
```

2. Create transactions based on `status` field values and limit the span to 5 minutes.

NOTE: If you do not see results, increase the `maxspan` value.

```
(index=network sourcetype=cisco_wsa_squid) OR
(index=web sourcetype=access_combined) status>399
| fields sourcetype, status
| transaction status maxspan=5m
```

3. Limit the results to only transactions that contain at least one event from each sourcetype.

```
(index=network sourcetype=cisco_wsa_squid) OR (index=web sourcetype=access_combined)
status>399
| fields sourcetype, status
| transaction status maxspan=5m
| search sourcetype=access_combined AND sourcetype=cisco_wsa_squid
```

4. Use `timechart` to count events by `status`.

```
(index=network sourcetype=cisco_wsa_squid) OR
(index=web sourcetype=access_combined) status>399
| fields sourcetype, status
| transaction status maxspan=5m
| search sourcetype=access_combined AND sourcetype=cisco_wsa_squid
| timechart count by status
```

Results Example:

_time	400	403	404	503
2018-01-06	3	2	3	0
2018-01-07	0	0	1	1
2018-01-08	0	6	3	3
2018-01-09	0	1	4	5

5. Discard rows that have fewer than 5 errors for all `status` values.
   Hint: Use `addtotals`.

```
(index=network sourcetype=cisco_wsa_squid) OR
(index=web sourcetype=access_combined) status>399
| fields sourcetype, status
| transaction status maxspan=5m
| search sourcetype=access_combined AND sourcetype=cisco_wsa_squid
| timechart count by status
| addtotals
| search Total>4
```

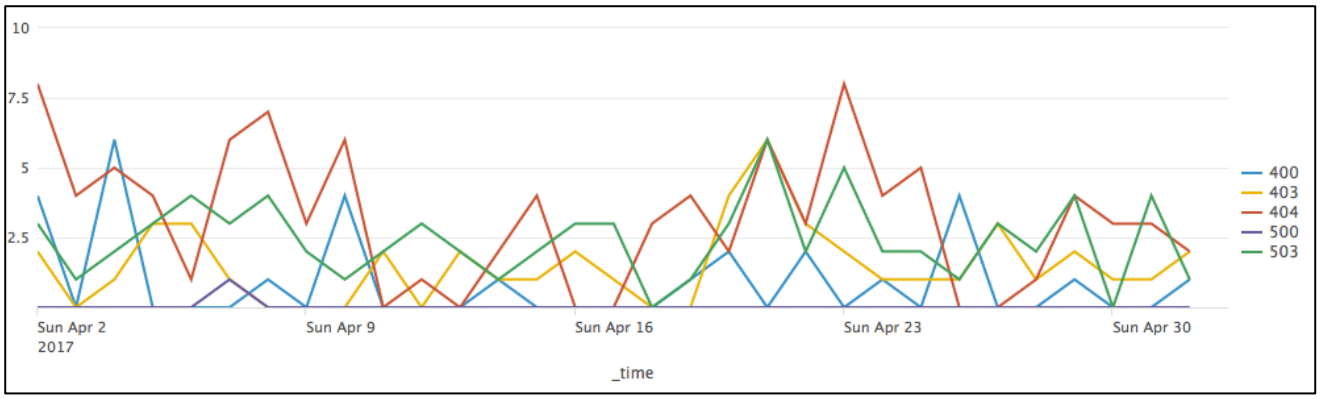
Results Example:

_time	400	403	404	503	Total
2018-01-06	3	2	3	0	8
2018-01-08	0	6	3	3	12
2018-01-09	0	1	4	5	10
2018-01-10	0	3	1	2	6

6. Remove the `Total` column and display the data as a **Line chart**.

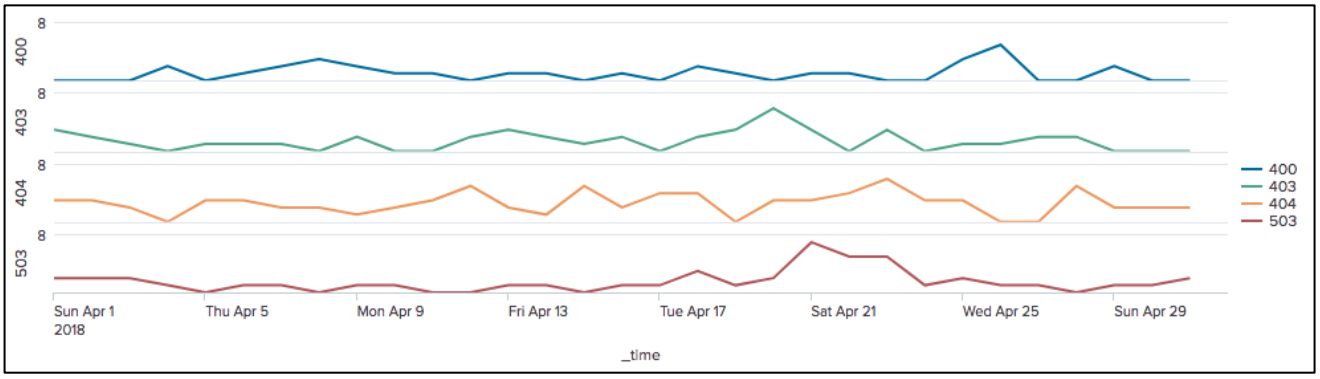
```
(index=network sourcetype=cisco_wsa_squid) OR
(index=web sourcetype=access_combined) status>399
| fields sourcetype, status
| transaction status maxspan=5m
| search sourcetype=access_combined AND sourcetype=cisco_wsa_squid
| timechart count by status
| addtotals
| search Total>4
| fields - Total
```

Results Example:



7. Save your search as report, **L5C1**.
8. Optionally, for this line chart, set **Multi-series Mode** to **Yes**. Observe the change in how the lines are represented.

Hint: It's one of the **Format** options on the **General** tab.



Lab Exercise 7: Creating and Managing Fields

Description

This lab exercise walks you through the process of creating field extractions based on either a Regular Expression (regex) or Delimiters.

Steps

Scenario: Access to the Linux server needs to be monitored.

Task 1: Use the Field Extractor (FX) to extract the IP address and port fields using the Regular Expression method.

1. Search for all events in the **last 24 hours** for the `linux_secure` sourcetype that contain the keyword *port*.

   index=security sourcetype=linux_secure port
2. View the event details to see all the extracted fields. Click the > arrow under the i icon in the first event that contains an IP address value.
3. Click Event Actions > Extract Fields.
4. Select the **Regular Expression** method and click **Next**.
5. Highlight the IP address value in the sample event.
6. In the **Field name** box, type `src`.
7. Click Add Extraction.
8. Scroll down to the Preview section and verify that the correct information is being extracted. You may see that "**::**" is extracted as a src value. But within this **particular** set of data, "::" actually represents an **invalid** IP address. You'll remove this value in the Validate process (Steps 12-13).
9. Highlight the port value.
10. In the **Field name** box, type `port`.
11. Click **Add Extraction** and click **Next**.
12. In the **Validate** step, click on the src tab. You will see "::" listed as a valid value. In the filter field, type `src=::` and click **Apply**.
13. Click the "x" next to the highlighted value of "**::**" for the src field. (It doesn't matter which event you choose.) The event sample will now show that "::" is an invalid value for the src field.

14. Click **Next**.
15. Review the Extractions Name and click **Finish**.

NOTE: Depending on what events you choose as examples, Splunk may not be able to generate the regex for both field extractions at once. If you encounter difficulties, try creating two separate extractions, one for each field.

16. Wait for about a minute, then search for events in the `linux_secure` sourcetype in the **last 24 hours**. List the top ports by IP address.

```
index=security sourcetype=linux_secure | top port by src
```

NOTE: It may take a few moments for the newly extracted fields to appear in the search because the training environment uses a search head cluster, and it takes a minute for knowledge objects to replicate across the cluster. (For details, see the Splunk Cluster Administration course.) This is also true of all the other knowledge objects you'll create in Fundamentals 2. In general, it's best to wait about a minute after object creation before submitting your search.

Results Example:

src	port	count	percent
107.3.146.207	3057	2	3.703704
107.3.146.207	4950	1	1.851852
107.3.146.207	4929	1	1.851852
107.3.146.207	4822	1	1.851852
107.3.146.207	4800	1	1.851852
107.3.146.207	4779	1	1.851852
107.3.146.207	4550	1	1.851852
107.3.146.207	4506	1	1.851852
107.3.146.207	4141	1	1.851852
107.3.146.207	4131	1	1.851852
108.50.217.115	8677	100	87.719298
108.50.217.115	7238	1	0.877193

Scenario: **The engineering team launched the beta of a new game called SimCube. To make improvements to the game, engineers want to see how users are playing the game. However, the log file doesn't contain headers and the fields are not auto-extracted.**

Task 2: Use FX to extract fields using the delimiters method.

17. Search for all events in the **last 30 days** for the SimCubeBeta sourcetype in the games index.

```
index=games sourcetype=SimCubeBeta
```

18. View the event details to see which fields are extracted.
19. In the fields sidebar, underneath the Interesting Fields section, click **+ Extract New Fields**.

20. Click the first event to select it as a sample event.
21. Click **Next**.
22. Select the **Delimiters** method and click **Next**.
23. For the Delimiter type, select **Comma**.
24. Rename all the fields as follows (in this order):
    - field1 > time
    - field2 > src
    - field3 > version
    - field4 > misc
25. After all the fields are renamed, click **Next**.
26. For Extractions Name, enter `simgame_log` and click **Finish>**.
27. Using the regex field extraction method, run the same search as you did in step 17 and extract the remaining fields (see results example below):
    - user
    - CharacterName
    - action
    - role

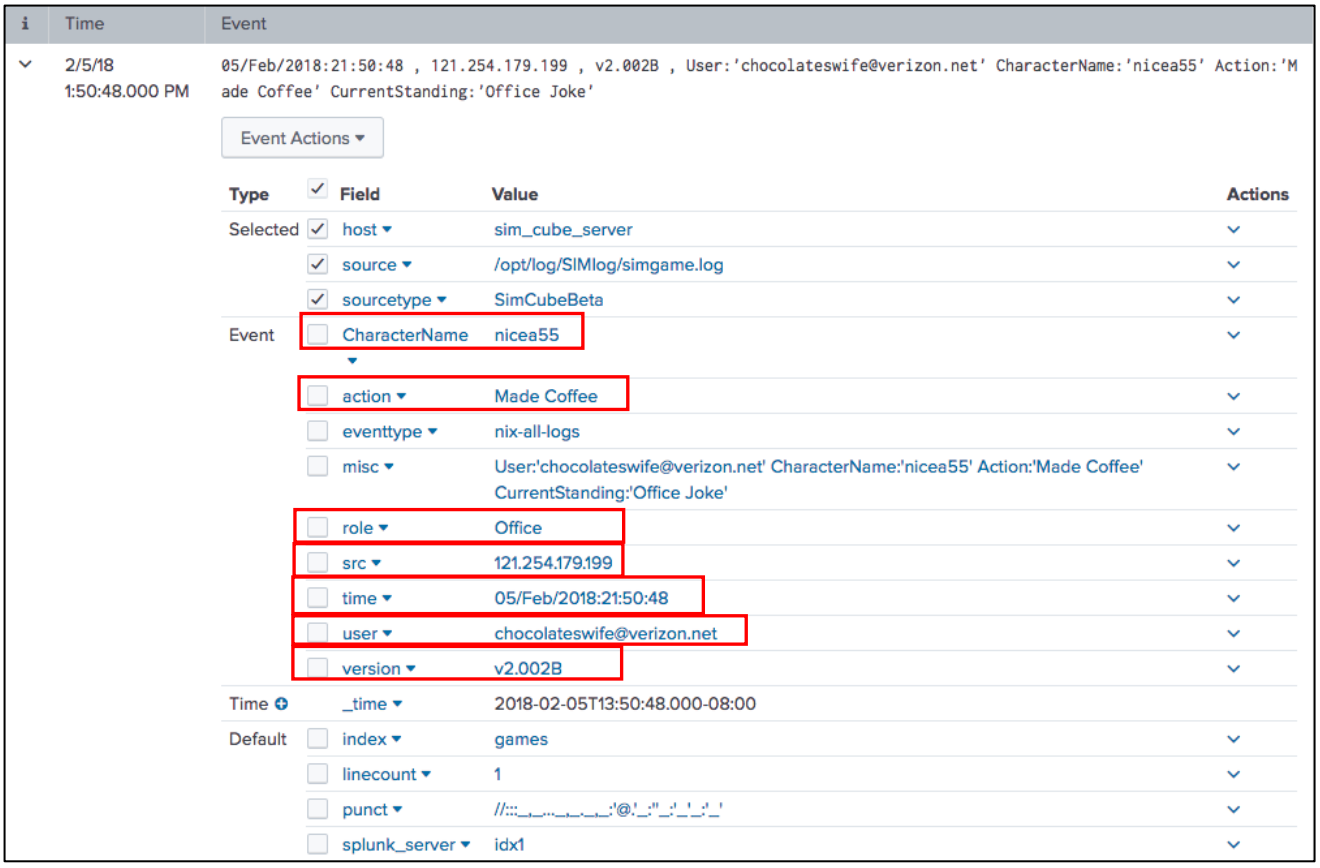
NOTE: Be sure to capture all the characters **between** the single quotes, but **not** the single quotes themselves. Some versions of Internet Explorer actually won't allow you to exclude the single quotes. If you're using IE and you encounter this problem, you must switch to another browser in order to complete the exercise.

28. While still on the **Select fields** step (before the validation stage), click on **Non-Matches** to see whether any relevant events are being excluded. (If no events display when you click **Non-Matches**, proceed to step 31.)
29. Hover your cursor over any excluded event that you want to include, and click **+ Add sample event**.
30. Highlight each relevant value in the sample event and click **Select a Field**. For each value, choose the field name you want associated with that value and click **Add Extraction**.
31. Repeat steps 28 – 29 for each excluded event until there are no more **Non-Matches**.
32. Click **Next** to proceed to the **Validate** step.
33. When you're satisfied with your result, click **Next**.

NOTE: Be sure to thoroughly check your results during the validation stage. It's important to ensure you've captured all characters inside the single quotes for the fields you've extracted.

34. Accept the prefilled Extractions Name and click **Finish>** to save.
35. Wait for about a minute, then run your search again and check that all expected fields appear.

Results Example:



NOTE: It may take a minute before the newly extracted fields appear in the search.

splunk>

Lab Exercise 8: Working with Field Aliases and Calculated Fields

Description

This lab exercise walks you through the process of creating field aliases and calculated fields.

Steps

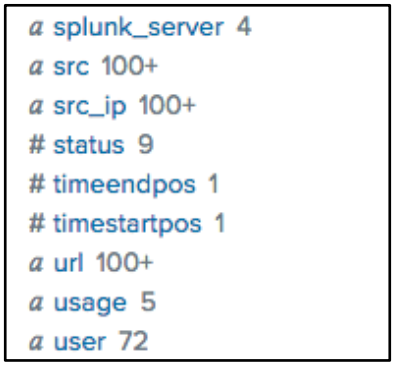
Scenario: **The IT Ops team runs reports for all employee access but the user name field is not consistent across the different source types.**

Task 1: Create a field alias so that *cs_username* also appears as *user*.

1. Search for all events in the `cisco_wsa_squid` sourcetype over the **last 7 days**.

   index=network sourcetype=cisco_wsa_squid
2. Note the `cs_username` field values.
3. Go to **Settings > Fields > Field aliases**. Create a field alias with the following values:
   - Destination app: class_Fund2
   - Name: cisco_wsa_squid_aliases
   - Apply to: sourcetype
   - Named: cisco_wsa_squid
   - Field aliases: cs_username = user
4. Click **Save**.
5. Return to the **CLASS: Fundamentals 2** app. Re-run your search and examine the user field and values.

Results Example:



6. Search for all events in the `cisco_firewall` sourcetype over the **last 30 days**.
7. Note the `Username` field values.
8. Create another field alias for sourcetype `cisco_firewall` with the following values:
   - Destination app: class_Fund2
   - Name: cisco_firewall_aliases
   - Apply to: sourcetype
   - Named: cisco_firewall
   - Field aliases: Username = user
9. Perform the following search: `index=network sourcetype=cisco* user=*` over the last 30 days. Do you receive results from the `cisco_wsa_squid` and `cisco_firewall` sourcetypes?

   Yes, you should see both source types.

NOTE: It may take a minute before the field aliases are applied and appear in searches.

Scenario: **The IT Ops team is monitoring bandwidth usage for all users for the last month, but the data is reported in bytes. The team needs the usage to be measured in megabytes.**

Task 2: Create a calculated field that converts bytes to MB.

10. Search for all events in the **last 7 days** for the `cisco_wsa_squid` sourcetype.

    index=network sourcetype=cisco_wsa_squid

11. Note the `sc_bytes` field. This field displays the amount of bytes used for that event.
12. Go to Settings > Fields > Calculated fields.
13. Create a calculated field named **sc_megabytes** that converts the value of `sc_bytes` to MB with the following values:
    - Destination app: class_Fund2
    - Apply to: sourcetype
    - Named: cisco_wsa_squid
    - Name: sc_megabytes
    - Eval expression: sc_bytes/(1024*1024)
    -
14. Return to the **CLASS: Fundamentals 2** app. Perform a search on the `cisco_wsa_squid` sourcetype that shows the total bandwidth by usage.

    index=network sourcetype=cisco_w* | stats sum(sc_megabytes) as "Bandwidth (MB)" by usage

Results Example:

usage	Bandwidth (MB)
Borderline	6.86968708038330100000
Business	17.08714580535888700000
Personal	54.93885517120361000000
Unknown	17.56064128875732400000
Violation	0.8761548995971680000

Supplemental Exercise:

Scenario: **The IT Ops team wants to correlate data from multiple source types using the `http_action` and `http_method` fields. In the `access_combined` source type, these fields are currently called `action` and `method`.**

Task 1: Create two field aliases for the access_combined sourcetype called http_action and http_method, based on the existing access_combined fields action and method.

1. Create the field aliases.
2. Run a search to verify that the field aliases were created correctly.

Lab Exercise 9: Creating Tags and Event Types

Description

This lab exercise walks you through the steps to create tags and event types.

Steps

Scenario: **The IT Operations team needs to monitor failed login attempts made with any variation of admin/administrator user accounts to their network devices. To avoid lengthy searches, include all events with these user accounts and create tags.**

Task 1: Create tags to identify all admin accounts.

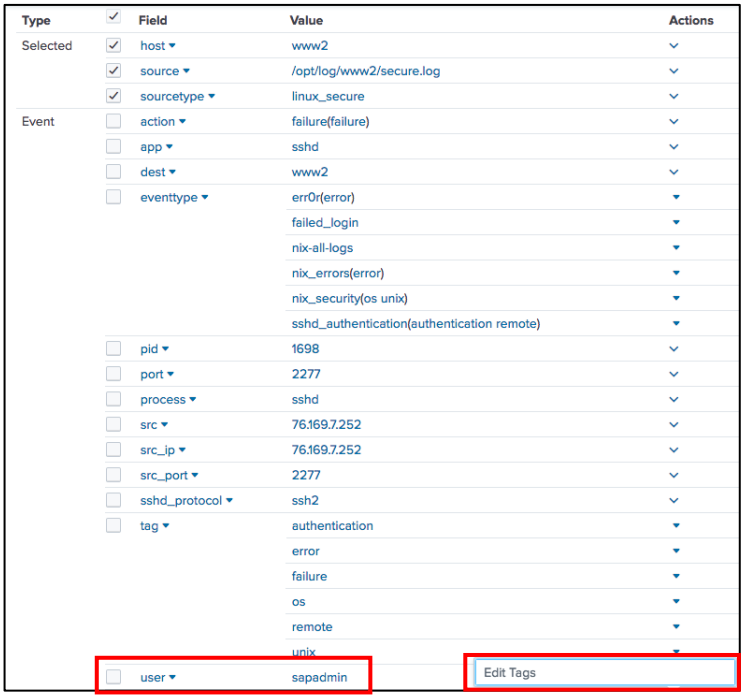
1. Run a search over the **Last 24 hours** for all failed login attempts for any variation of the user *admin* under the security index. You should see the following five users: admin, administrator, sysadmin, itmadmin, and sapadmin.

   index=security failed user=*admin*

NOTE: Only trailing wildcards make efficient use of indexes. For that reason, it's generally a best practice *not* to use wildcards at the beginning of a string, as such searches have to scan all events within the specified time frame. However, doing a search with a wildcard at the beginning of a string is *possible* and sometimes necessary in particular scenarios. Be advised, however, that such searches are inefficient and, in general, should be avoided.

2. Expand an event and find the row for the **user** field. Click the **down arrow** under the **Actions** column and select **Edit Tags**.

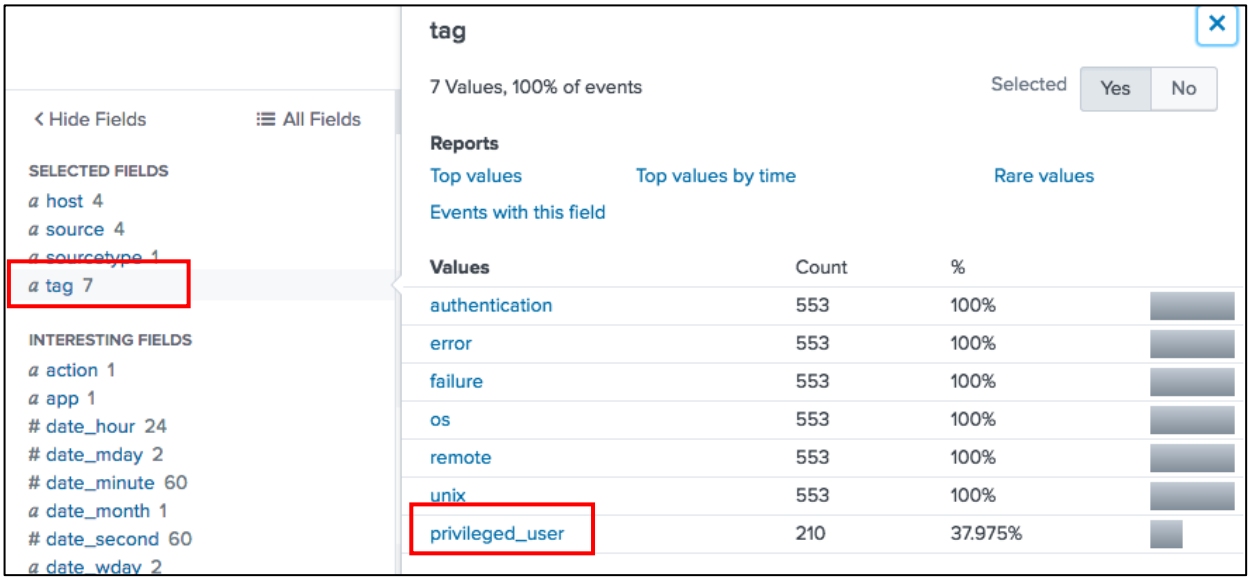
Example:



3. In the **Tag(s)** field, type **privileged_user** and click **Save**.
4. Create tags for each variation of the user *admin (*admin, administrator, sysadmin, itmadmin, and sapadmin). You can create the subsequent tags the same way you created the first one, from the Events tab of the search results. Alternatively, you can also create the subsequent tags by going to the **Settings** > **Tags** > **List by tag name** screen, choosing the newly created **privileged_user** tag, adding the other four types of admins, and clicking **Save**.
5. Run the search again and check to see that the privileged_user tag was created.

   index=security failed user=*admin*
6. If it isn't already, add **tag** to your list of Selected Fields.

Results Example:



Task 2: Use tags in a search.

7. Search for all failed login attempts by privileged user accounts for the **Last 7 days**. You should see the following five users: admin, administrator, sysadmin, itmadmin, sapadmin

   index=security failed tag=privileged_user

Scenario: Customers are reporting issues trying to purchase items from the Buttercup Games online store and internal users get errors trying to access the internet. IT Ops wants an easy way to determine if there is any correlation when both systems encounter problems.

Task 3: Create an event type for status errors greater than or equal to 500 on web servers/devices.

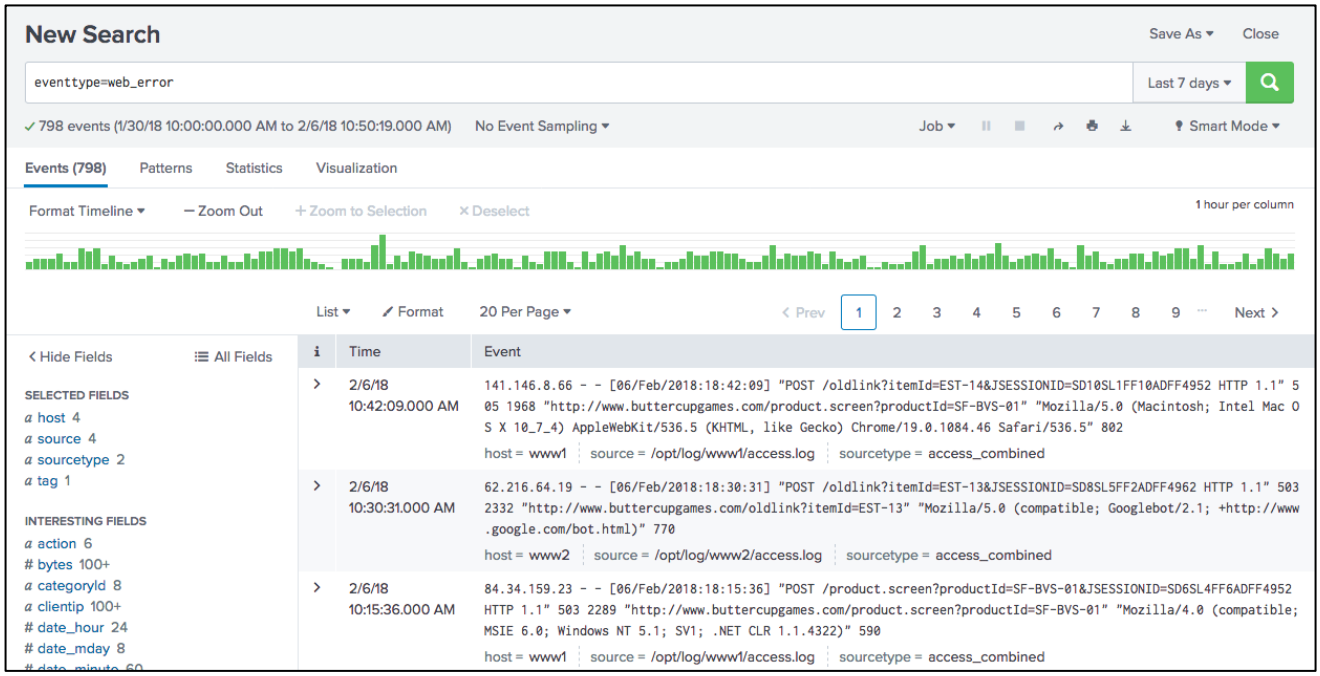
8. Search for all online sales and Web security appliance data with status error codes greater than 500 in the **last 7 days.**

   (index=web sourcetype=access_combined) OR (index=network sourcetype=cisco_wsa_squid) status>=500

9. Select **Save As** > **Event Type**.
10. Name your event type: `web_error`
11. Leave the **Priority** set to 1 (Highest).
12. Click **Save**.
13. Perform a search for the web_error event type for the **Last 7 days**.

    eventtype=web_error
14. Expand an event and click the checkbox next to **eventtype** to add it to the Selected fields.
15. How many sourcetypes are returned?

Results Example:



NOTE: Depending upon add-ons or apps you have installed, additional event types may be displayed.

splunk>

Lab Exercise 10: Creating and Using Macros

Description

This lab exercise walks you through the steps for creating a basic macro and a macro with arguments.

Steps

Scenario: **The VP of Sales wants to run ad-hoc searches to determine the value of products sold in a given month in various countries. He also wants to easily convert international sales to US Dollars based on current exchange rates.**

Task 1: Write a basic macro to create a table displaying the total sales of each product sold in Europe.

1. Using the `stats` command, create a table showing the total retail sales for each product sold in Europe (combining sales from Germany, France, and Italy) over the **Last 30 days** and rename the total sales column as USD.

   ```
   index=sales sourcetype=vendor_sales VendorCountry=Germany OR VendorCountry=France OR VendorCountry=Italy  | stats sum(price) as USD by product_name
   ```

2. Using the `eval` command, convert the numeric values in the total sales column to strings and concatenate them with a $ sign.

   ```
   index=sales sourcetype=vendor_sales VendorCountry=Germany OR VendorCountry=France OR VendorCountry=Italy  | stats sum(price) as USD by product_name | eval USD = "$" + tostring(USD,"commas")
   ```

   Hint: After typing this search string, you may want to copy it into a notepad, as you'll be using it to create a macro later in this exercise.

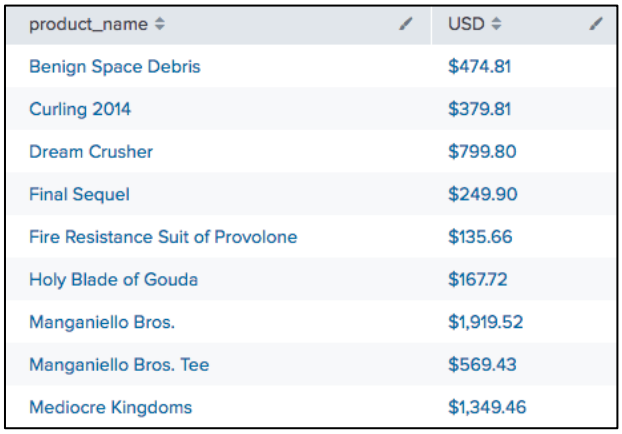
3. Navigate to Settings > Advanced search > Search macros.
4. Click New Search Macro.
5. Verify the Destination app is set to **class_Fund2**.
6. Name the macro: Europe_sales
7. In the **Definition** field, type or paste the search string from Step 2.
8. Save the macro.

Task 2: Use a basic macro.

9. Return to the CLASS: Fundamentals 2 app.
10. In the search bar, type `` `Europe_sales` `` and search over the **Last 30 days**. Examine the results.

NOTE: Remember to type the macro name between backticks, not single quotes.

Results Example:



product_name	USD
Benign Space Debris	$474.81
Curling 2014	$379.81
Dream Crusher	$799.80
Final Sequel	$249.90
Fire Resistance Suit of Provolone	$135.66
Holy Blade of Gouda	$167.72
Manganiello Bros.	$1,919.52
Manganiello Bros. Tee	$569.43
Mediocre Kingdoms	$1,349.46

Task 3: Create a macro that enables users to specify currency when performing a search. This macro uses currency, currency symbol, and rate as variables (arguments).

11. Run the following search to determine total sales for each product from vendors in Europe in the **last 30 days**:

```
sourcetype=vendor_sales VendorCountry=Germany OR VendorCountry=France OR
VendorCountry=Italy
| stats sum(price) as USD by product_name
| eval euro = "€" + tostring(round(USD*0.79,2), "commas"), USD = "$" +
tostring(USD, "commas")
```

Now you're going to use the second portion of this search string, where the evaluations are done, to create a dynamic macro with arguments.

12. Navigate to Settings > Advanced search > Search macros.
13. Click New Search Macro.
14. Verify the Destination app is set to **class_Fund2**.
15. Name the macro: convert_sales(3)
16. To make things easy for the user, the currency, currency symbol and exchange rate are arguments. Enter the following search string (the arguments are encapsulated by the $ signs):

```
stats sum(price) as USD by product_name
| eval $currency$="$symbol$".tostring(round(USD*$rate$,2),"commas"),USD="$" +
tostring(USD,"commas")
```

NOTE: Be sure to include the pipe symbol (|) before the `eval` command. You can copy/paste the € symbol from this document or go to the following website for the keyboard shortcuts:

http://bit.ly/2BqMmR0

17. In the **Arguments** field, type the arguments, separated by commas.
    Hint: currency,symbol,rate (order of variables must match the search string)
18. Save the macro.

Task 4: Use your macro with arguments in a search.

19. Return to the CLASS: Fundamentals 2 app.
20. Perform a search for `sourcetype=vendor_sales` where the `VendorCountry` is Germany, France, or Italy. Use the macro and pass the arguments `euro, €,` and `0.79` for results in the **Last 30 days**.
    Hint: `convert_sales(currency,symbol,rate)`

    index= sales sourcetype=vendor_sales VendorCountry=Germany OR VendorCountry=France OR VendorCountry=Italy | `convert_sales(euro,€,.79)`
21. Run the search again for sales in the UK with the following arguments `GBP, £,` and `0.64`. Copy/paste the £ symbol from this document.

    index=sales sourcetype=vendor_sales VendorCountry="United Kingdom" |`convert_sales(GBP,£,.64)`

Results Example:

product_name	USD	GBP
Benign Space Debris	$374.85	£239.90
Curling 2014	$259.87	£166.32
Dream Crusher	$479.88	£307.12
Final Sequel	$74.97	£47.98
Fire Resistance Suit of Provolone	$95.76	£61.29
Holy Blade of Gouda	$101.83	£65.17
Manganiello Bros.	$759.81	£486.28
Manganiello Bros. Tee	$199.80	£127.87
Mediocre Kingdoms	$349.86	£223.91
Orvil the Wolverine	$399.90	£255.94
Puppies vs. Zombies	$4.99	£3.19
SIM Cubicle	$319.84	£204.70
World of Cheese	$499.80	£319.87
World of Cheese Tee	$169.83	£108.69

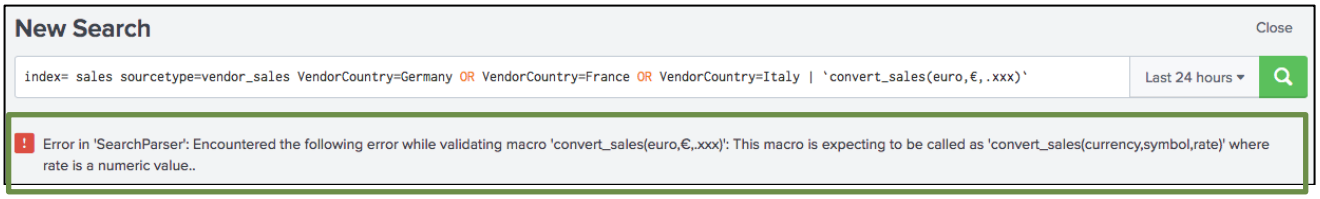
Task 5: Edit your macro and use the isnum expression to validate the rate field.

22. Navigate to Settings > Advanced search > Search macros.
23. Choose your user name from the Owner dropdown list.
24. Click on the **convert_sales(3)** link.
25. In the Validation Expression text box, type: isnum($rate$)
26. In the Validation Error Message text box, type: This macro is expecting to be called as 'convert_sales(currency,symbol,rate)' where rate is a numeric value.
27. Click **Save**.

28. Return to the CLASS: Fundamentals 2 app.
29. Perform a search for `sourcetype=vendor_sales` for the **Last 30 days** where the `VendorCountry` is Germany, France, or Italy. Use the macro, but deliberately pass a non-numeric value for the rate argument (for example, pass the arguments `euro, €,` and `.xxx`).

    index= sales sourcetype=vendor_sales VendorCountry=Germany OR VendorCountry=France OR VendorCountry=Italy | `convert_sales(euro,€,.xxx)`
30. Check to see that your error message displays.

Results Example:



Lab Exercise 11: Creating and Using Workflow Actions

Description

These steps create GET, POST, and Search workflow actions.

Steps

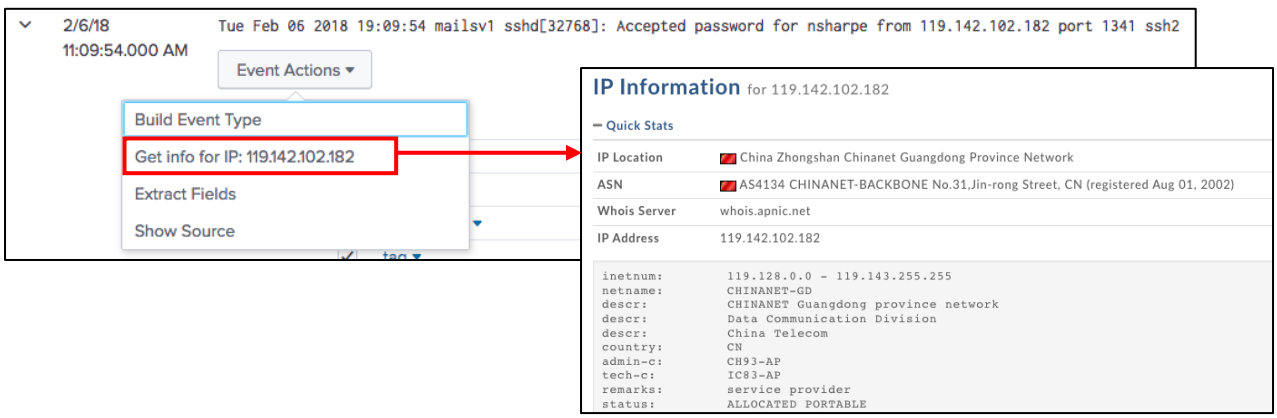
Scenario: **Hackers are continually trying to log into the Linux server. IT Ops analysts need to track ongoing attempts by external sources trying to log in with invalid credentials.**

Task 1: Create a GET workflow action that opens a new browser window with information about the source IP address.

1. Navigate to Settings > Fields > Workflow actions.
2. Click **New Workflow Action** to create a workflow action.
3. For the Destination App, select class_Fund2.
4. For **Name**, type: get_whois_info
5. For **Label**, type: Get info for IP: src_ip
6. For Apply only to the following fields, type: src_ip
7. For **Action type**, make sure link is selected.
8. For **URI**, type: http://who.is/whois-ip/ip-address/src_ip
9. From the **Open link in** dropdown menu, verify New window is selected.
10. From the **Link Method** dropdown menu, verify get is selected.
11. Save your workflow action.
12. Verify your workflow action works as expected. Return to the **CLASS: Fundamentals 2** app and search for `index=security sourcetype=linux_secure src_ip=*` over the **last 24 hours**. (You may need to refresh your browser for the workflow action to appear.)
13. Expand the first event containing a value for `src_ip` and click **Event Actions**.
14. Click **Get info for IP**: {src_ip}. A secondary browser window or tab should open to the URI and display the IP address information.

NOTE: If whois is not behaving as expected, try http://whois.domaintools.com/src.

Results Example:



Scenario: **The revenue accounting department is having issues with sales transactions not posting to the accounting system. This issue is causing revenue recognition discrepancies and the IT department is tasked with notifying the accounting system administrators when there is a transaction error in the system.**

Task 2: Create a POST workflow action that will use fields from events with errors to create a ticket in the IT ticket tracking system.

15. Perform a search on the `sales_entries` sourcetype for events posting errors.

    index=sales sourcetype=sales_entries error

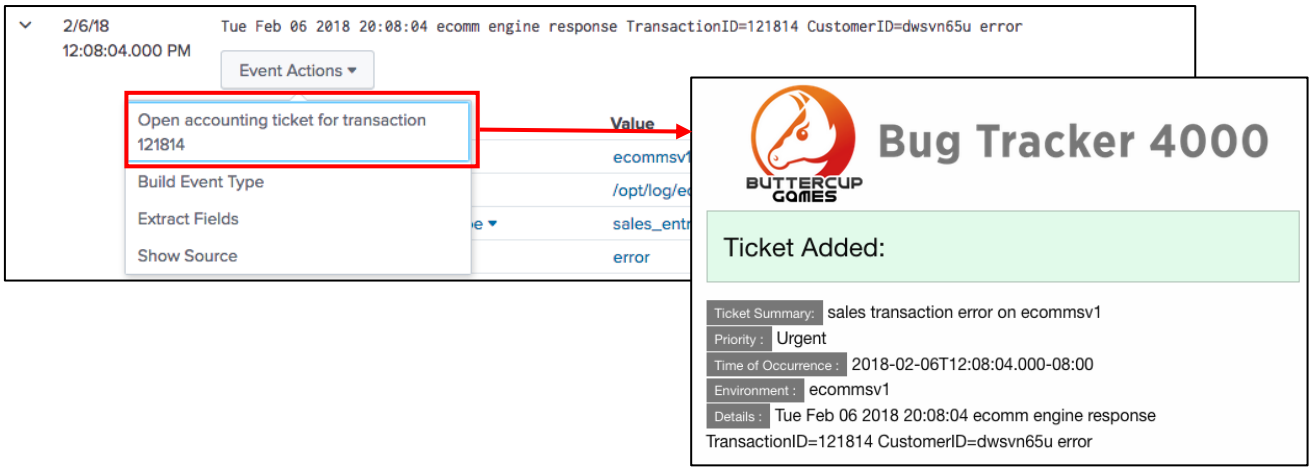
    These events contain two fields that are needed when creating tickets in the tracking system: `TransactionID` and `CustomerID`.

16. Create a field extraction with a field name of **result** for the string "error." This allows you to easily search for events where **result=error**.

NOTE: If you don't recall how to create a field extraction, please refer to Lab Exercise 7. If the **result=error** field extraction isn't done, the rest of this task will **not** work.

17. Navigate to Settings > Fields > Workflow actions.
18. Select New Workflow Action.
19. For the Destination App, select **class_Fund2**.
20. For **Name**, type: Create accounting system ticket
21. For **Label**, type: Open accounting ticket for transaction $TransactionID$
22. For Apply only to the following fields, type: result
23. For **Show Action in**, select Event menu.
24. For **Action type**, make sure link is selected.
25. For **URI**, type: http://52.3.246.206
26. From the **Open link in** dropdown menu, select **New window**.
27. From the **Link Method** dropdown menu, select **post**.
28. Enter the following values for the **Post arguments**:
    - details = $_raw$
    - environment = $host$
    - occurred = $_time$
    - priority = Urgent
    - summary = sales transaction error on $host$
29. Click **Save**.
30. Rerun your search for events where **result=error** and view the details of one of the returned events. Does your POST workflow action appear?
31. Click on your workflow action. A new browser window should appear with the ticket details.

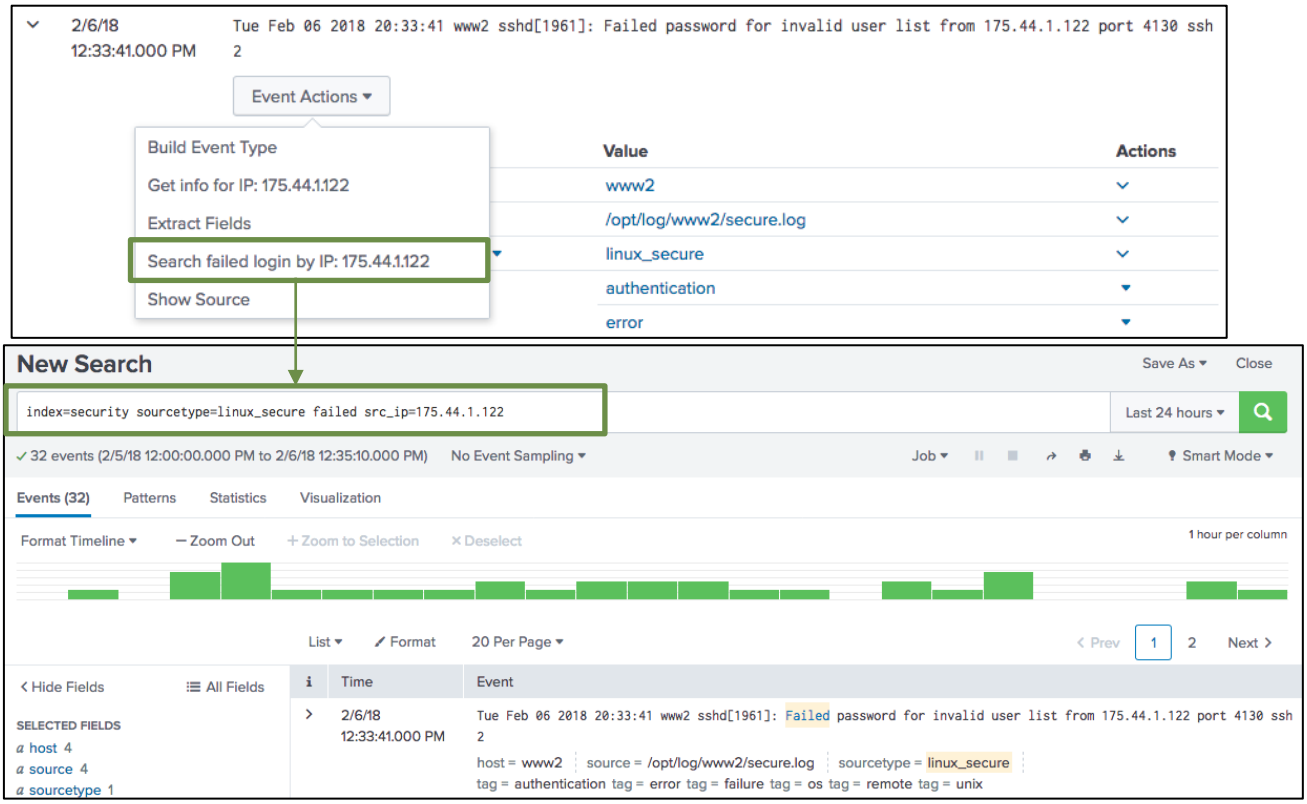
Results Example:



Task 3: Create a Search workflow action that performs a search for all failed password events associated with a specific IP address.

32. Navigate to Settings > Fields > Workflow actions.
33. Click New Workflow Action.
34. For the Destination App, select **class_Fund2**.
35. For **Name**, type: search_access_by_ipaddress
36. For **Label**, type: Search failed login by IP: src_ip
37. For Apply only to the following fields, type: src_ip
38. From the **Action Type** dropdown menu, select search.
39. In the **Search string** field, type: `index=security sourcetype=linux_secure failed src_ip=$src_ip$`
40. From the **Run in app** dropdown, select **class_Fund2**.
41. From the **Run search in** dropdown menu, verify New window is selected.
42. Select the Use the same time range as the search that created the field listing checkbox.
43. Save your workflow action.
44. Verify your workflow action works as expected. Return to the **CLASS: Fundamentals 2** app and search for `index=security sourcetype=linux_secure src_ip=*` over the **last 24 hours**. (You may need to refresh your browser for the workflow action to appear.)
45. Expand an event with an IP address field and click **Event Actions**.
46. Select Search failed login by IP: {src_ip}
47. A secondary search window should open with the search results for the IP address.

Results Example:



splunk>

Lab Exercise 12: Creating Data Models

Description

This exercise walks you through the process of creating a data model. After the data model is created, create a pivot to verify your data model provides the expected results.

Steps

Scenario: **The VP of Sales wants to run reports based on daily activity from the online store, but doesn't have the time to learn the search language.**

Task 1: Add the Web Requests root event. The root event will be the base search for all child events.

1. Navigate to Settings > Data models.
2. Click New Data Model.
3. In the **Title field**, type: Buttercup Games Site Activity
4. For **App**, make sure **Search & Reporting** is selected.

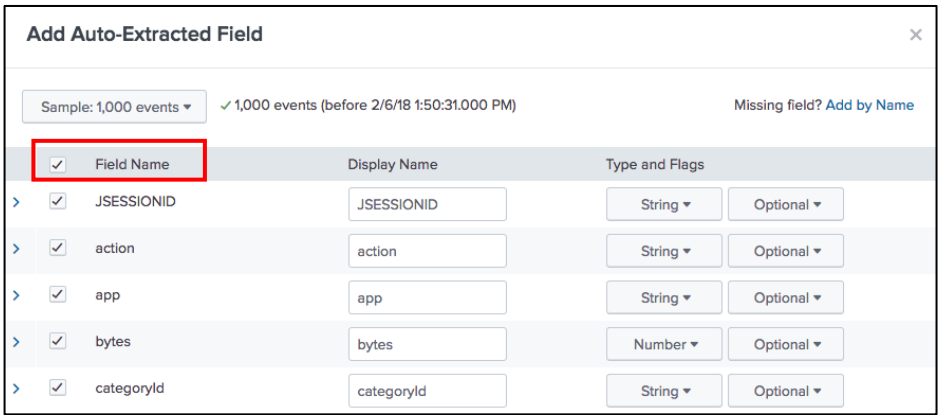
NOTE: Students are logged in with the power role and in this environment, power users have read-only permissions. Therefore, students can only create data models in the default Search & Reporting app, not in the CLASS: Fundamentals 2 app.

5. Click **Create**.
6. Click **Add Dataset** and select Root Event.
7. In the **Dataset Name** field, type: Web requests.
8. In the **Constraints** field, type: `index=web sourcetype=access_combined`
9. Click **Preview** to see a sampling of the events.
10. After the data has been verified, save the root event.

Task 2: Add auto-extracted fields.

11. Make sure the root Web requests dataset is selected.
12. Click **Add Field** and select **Auto-Extracted**. A dialog box opens and displays all auto-extracted fields.
13. Select all fields by checking the **Field Name** checkbox. Selecting this box selects all auto-extracted fields.

Example:

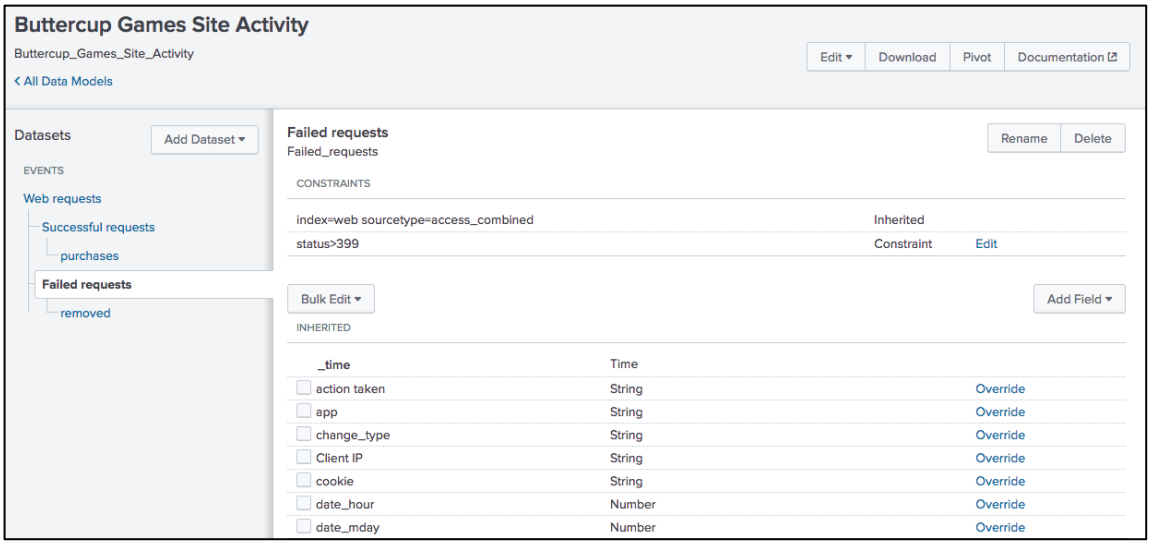


14. Rename the following fields for pivot users:
    - action > action taken
    - bytes > size
    - categoryId > product category
    - clientip > client IP
    - productId > product ID
    - product_name > product name
    - req_time > request time
15. Click **Save**.

Task 3: Add two child events, one for actions that were successful and one for actions that failed.

16. Click **Add Dataset** and select Child.
17. In the **Dataset Name** field, type: Successful requests
18. In the **Additional Constraints** field, type: `status<400`
19. Click **Preview** to see a test sample of your results.
20. **Save** the child dataset.
21. Select the Successful requests dataset. Add a child dataset called **purchases** with an **Additional Constraints** value of `action=purchase productId=*`. Preview your results before clicking **Save**.
22. Select the Web requests event and add a child dataset named: Failed requests**.**
23. In the **Additional Constraints** field, type: `status>399`
24. Click **Preview** to receive a test sample of your results.
25. **Save** the child dataset.
26. Under the Failed requests dataset, add a child dataset named **removed** with an **Additional Constraints** value of `action=remove productId=*`. Remember to click **Save**.

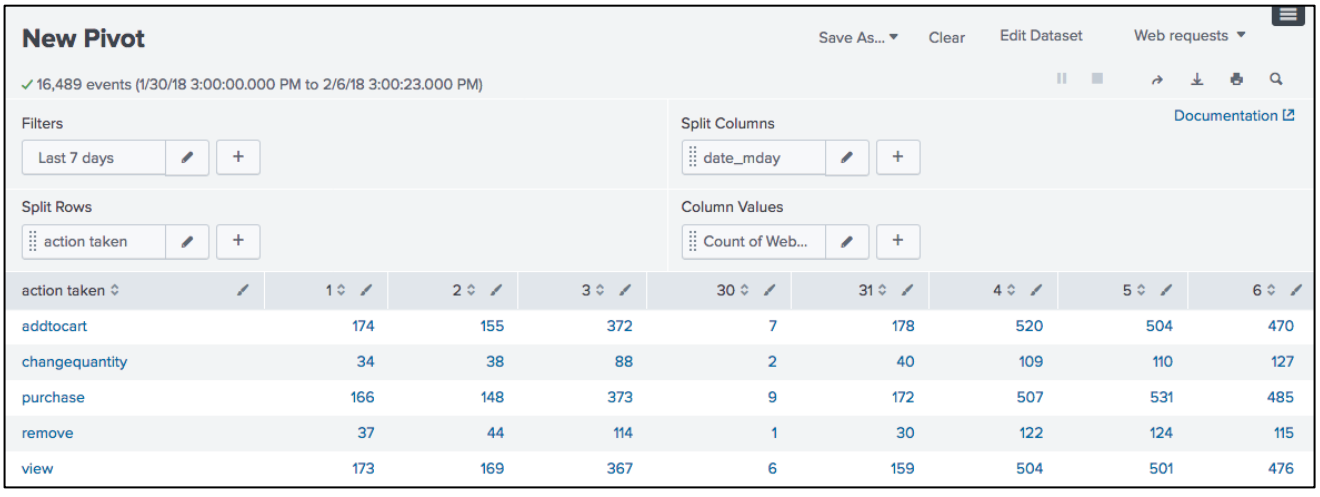
Results Example:



Task 4: Test your data model by creating a pivot.

27. Click **Pivot** in the upper right hand corner to test the data model.
28. Select the Web requests dataset**.**
29. In the **New Pivot** window, change the following:
    - Filter on the Last 7 days
    - Split Rows by action taken and click **Add To Table**
    - Split Columns by date_mday and click **Add To Table**

Results Example:



action taken	1	2	3	30	31	4	5	6
addtocart	174	155	372	7	178	520	504	470
changequantity	34	38	88	2	40	109	110	127
purchase	166	148	373	9	172	507	531	485
remove	37	44	114	1	30	122	124	115
view	173	169	367	6	159	504	501	476

Task 5: Add a field that uses an eval expression. The eval expression will display events chronologically by date and day of the week.

30. Select Edit Dataset.

31. Make sure Web requests is selected.
32. From the **Add Field** dropdown, select **Eval Expression**.
33. In the **Eval Expression** field, type: strftime(_time,"%m-%d %A")

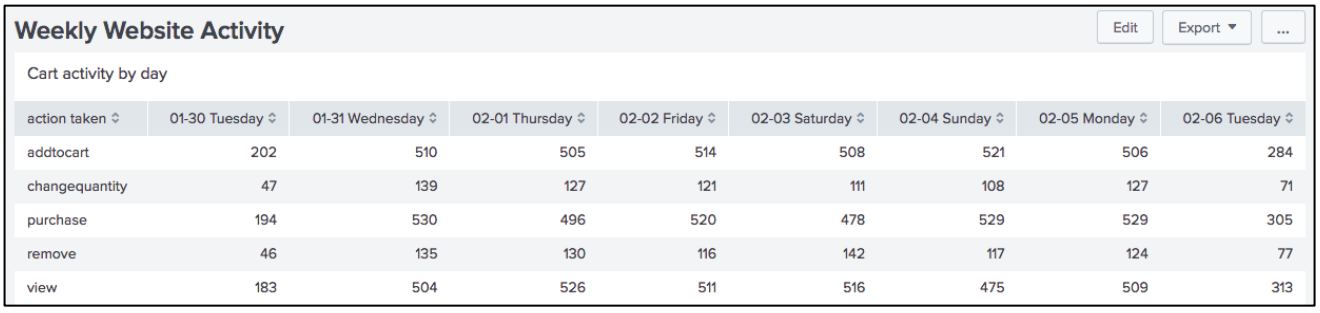
NOTE: strftime is a function that converts epoch time to a readable format. You'll learn more about it in Splunk Fundamentals 3.

34. For **Field Name**, type: day
35. For **Display Name**, type: day
36. Click **Preview** to verify your eval expression returns results.
37. **Save** the eval expression.

Task 6: Verify the eval expression works as expected by using Pivot to create a dashboard.

38. Click **Pivot**.
39. Select the Web requests dataset.
40. Change the time filter to the **Last 7 days**.
41. **Split Rows** by action taken.
42. Click Add To Table.
43. Split Columns by day.
44. Click Add To Table.
45. Click Save As and select Dashboard Panel.
46. For **Dashboard Title**, type: Weekly Website Activity
47. For **Panel Title**, type: Cart activity by day
48. Click **Save**.
49. Click **View Dashboard**. You should see the web requests categorized and counted by day.

Results Example:



action taken	01-30 Tuesday	01-31 Wednesday	02-01 Thursday	02-02 Friday	02-03 Saturday	02-04 Sunday	02-05 Monday	02-06 Tuesday
addtocart	202	510	505	514	508	521	506	284
changequantity	47	139	127	121	111	108	127	71
purchase	194	530	496	520	478	529	529	305
remove	46	135	130	116	142	117	124	77
view	183	504	526	511	516	475	509	313

Task 7: Add fields from a lookup. The lookup table will provide descriptions of status codes.

50. Verify that you are still in the **Search & Reporting** app. If necessary, click the dropdown list next to the **splunk>** logo at the top left of the window and choose **App: Search & Reporting**.
51. Navigate to Settings > Data models.
52. Select the Buttercup Games Site Activity data model.

53. Make sure the Web requests root dataset is selected.
54. Click **Add Field** and select **Lookup**.
55. From the **Lookup Table** dropdown list, select **http_status_lookup**.
56. For the **Input** section in the **Field in Lookup** dropdown, select **code**.
57. From the **Field in Dataset** dropdown**,** select **status**. This maps the `status` field in your indexed data to the `code` column in the lookup table.
58. For the lookup **Output** section in the **Field in Lookup** field, check the **description** checkbox.
59. In the **Display Name** field, type: status description
60. Click the **Preview** button. You should see a **description** column in the results.
61. Click **Save**.

Task 8: Verify the lookup works properly by creating a Pivot report.

62. Click **Pivot**.
63. Select the **Web requests** dataset.
64. Change the Filter to **Last 7 days**.
65. From **Split Rows**, add the status description attribute and click **Add To Table**.
66. Click the **+** button to split by another row and add the **status** attribute. Click **Add To Table**.

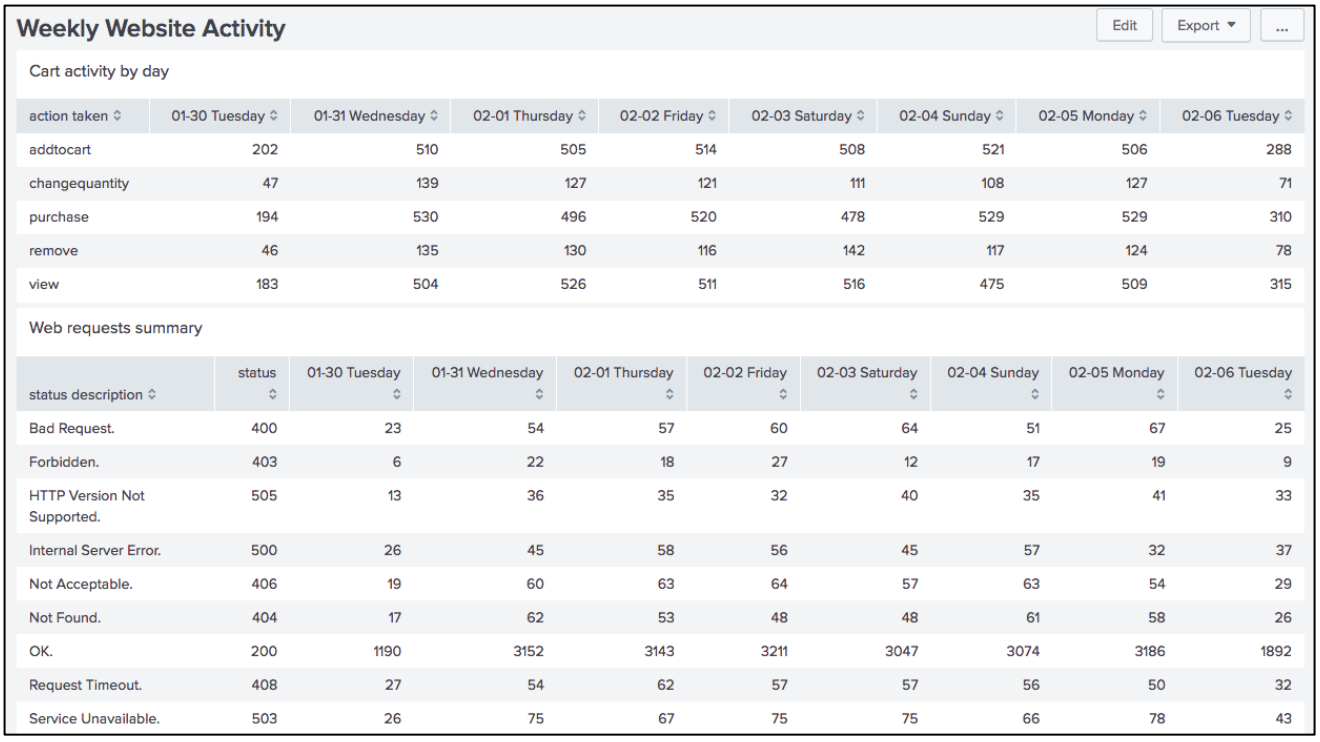
NOTE: This is a double row split, not a column split.

Results Example:

status description	status	Count of Web requests
Bad Request.	400	204
Forbidden.	403	56
HTTP Version Not Supported.	505	146
Internal Server Error.	500	170
Not Acceptable.	406	201
Not Found.	404	192
OK.	200	11119
Request Timeout.	408	192
Service Unavailable.	503	261

67. Split Columns by day and click Add To Table.
68. Click Save As and select Dashboard Panel.
69. Select Existing Dashboard and select Weekly Website Activity.
70. For the **Panel Title**, type: Web requests summary
71. Click **Save**.
72. Click View Dashboard.

Results Example:



Weekly Website Activity Edit Export ...

Cart activity by day

action taken	01-30 Tuesday	01-31 Wednesday	02-01 Thursday	02-02 Friday	02-03 Saturday	02-04 Sunday	02-05 Monday	02-06 Tuesday
addtocart	202	510	505	514	508	521	506	288
changequantity	47	139	127	121	111	108	127	71
purchase	194	530	496	520	478	529	529	310
remove	46	135	130	116	142	117	124	78
view	183	504	526	511	516	475	509	315

Web requests summary

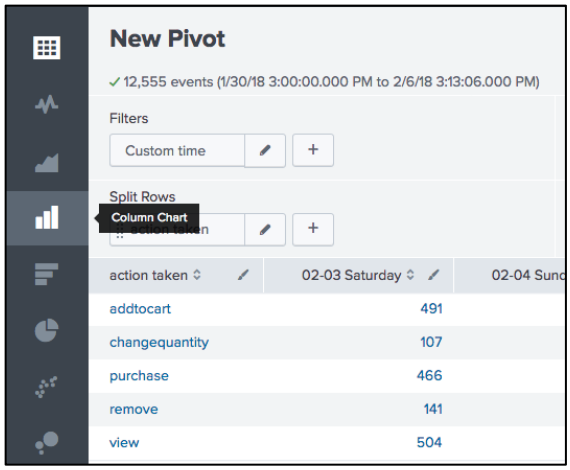
status description	status	01-30 Tuesday	01-31 Wednesday	02-01 Thursday	02-02 Friday	02-03 Saturday	02-04 Sunday	02-05 Monday	02-06 Tuesday
Bad Request.	400	23	54	57	60	64	51	67	25
Forbidden.	403	6	22	18	27	12	17	19	9
HTTP Version Not Supported.	505	13	36	35	32	40	35	41	33
Internal Server Error.	500	26	45	58	56	45	57	32	37
Not Acceptable.	406	19	60	63	64	57	63	54	29
Not Found.	404	17	62	53	48	48	61	58	26
OK.	200	1190	3152	3143	3211	3047	3074	3186	1892
Request Timeout.	408	27	54	62	57	57	56	50	32
Service Unavailable.	503	26	75	67	75	75	66	78	43

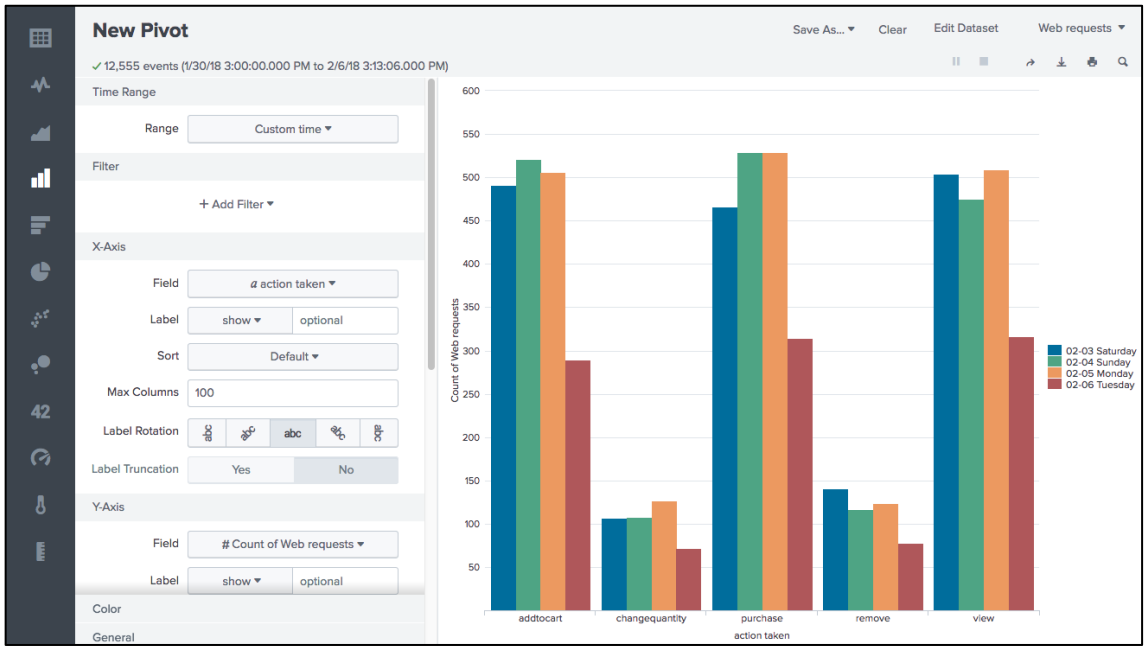
Supplemental Exercise:

Task 1: From the pivot editor, add a filter to narrow your results.

1. Hover your mouse in the lower right corner of the **Cart Activity by day** dashboard panel. Click the **Open in Pivot** icon  .
2. Refine your search results by selecting the **Column chart** icon from the table formats on the left.

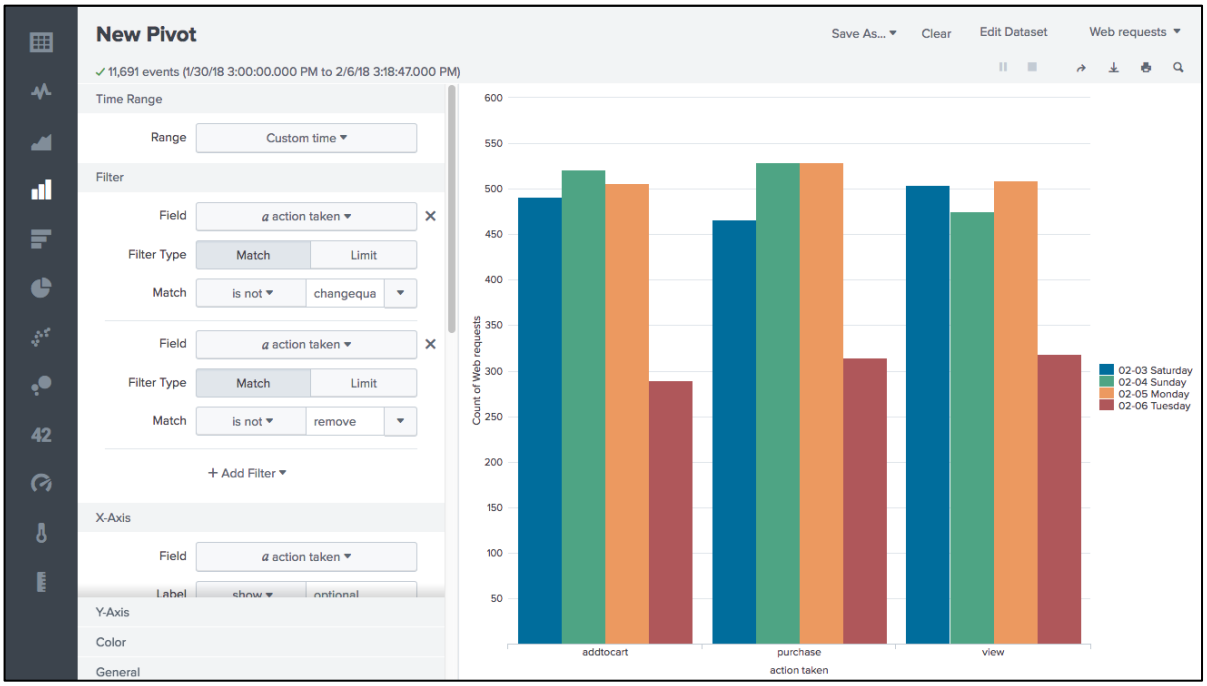
Results Examples:





3. Click Add Filter and choose action taken.
4. For Filter Type, select **Match**.
5. For **Match**, change the operator to **is not**, then select **changequantity**.
6. Add another filter and again choose **action taken**.
7. For the **Filter Type**, select **Match**.
8. For **Match**, change the operator to **is not** and then select **remove**.

Results Example:



9. Click Save As and select Dashboard Panel.
10. Save to the **Weekly Website Activity** dashboard.
11. For **Panel Title**, type: Add purchase view
12. **Save** and **view** your dashboard.
13. Rearrange the panels to your liking and admire your work!

splunk>

Lab Exercise 13: Using the Common Information Model (CIM) Add-On

Description

In this lab exercise, you normalize your data to the Splunk Common Information Model (CIM) using the CIM add-on.

Steps

Scenario: **The Buttercup Games sales team wants to correlate sales data across multiple data sources, but not all source types use the same field names. To ensure that all data is reported correctly, the IT team has installed the CIM app to use as a standard for field names.**

Task 1: Examine your data.

1. Return to the CLASS: Fundamentals 2 app.
2. Search for all action types related to online transactions over the **last 4 hours**.

   ```
   index=web sourcetype=access_combined action=*
   ```
3. Examine the values of the following fields. These fields are required for your dashboard:
   - host
   - action
   - clientip
   - status
   - useragent
4. In a separate browser tab or window, examine the Web data model in the CIM Reference Tables from the following link:
   https://docs.splunk.com/Documentation/CIM/latest/User/Howtousethesereferencetables
5. In the browser you opened in step 4, select **Web** from the data model list on the left.
6. Examine the **Fields for Web event datasets** table. Based on the fields in `access_combined`, which fields in the data model match the fields needed for your dashboard?

Field name in source type	Field in Data Model
host	dest
action	action
clientip	src
status	status
useragent	http_user_agent

7. Using the `datamodel` command, are the fields in your data populated in the Web data model?

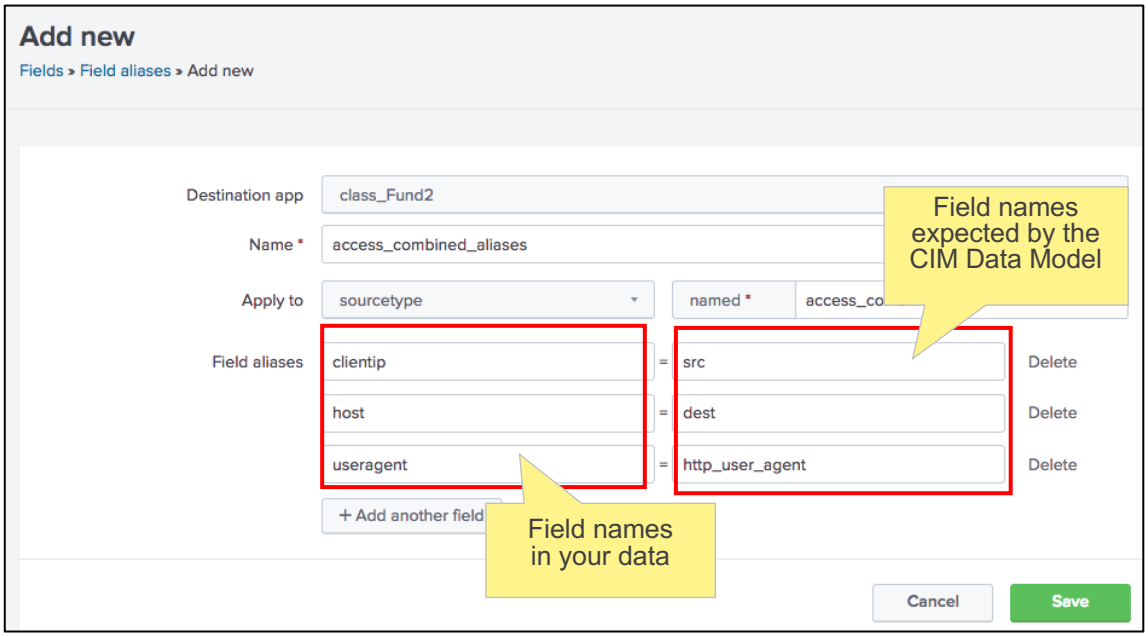
   | datamodel Web Web search | fields Web*

   Hint: Refer to the example on the **`datamodel` Command – Example** slide and then check which fields are included in your result.

Field in Your Data	Matching Attribute	Data Model Field Populated?
host	dest	No
action	action	Yes
clientip	src	No
status	status	Yes
useragent	http_user_agent	No

Task 2: Create field aliases for the fields that aren't populated in the data model.

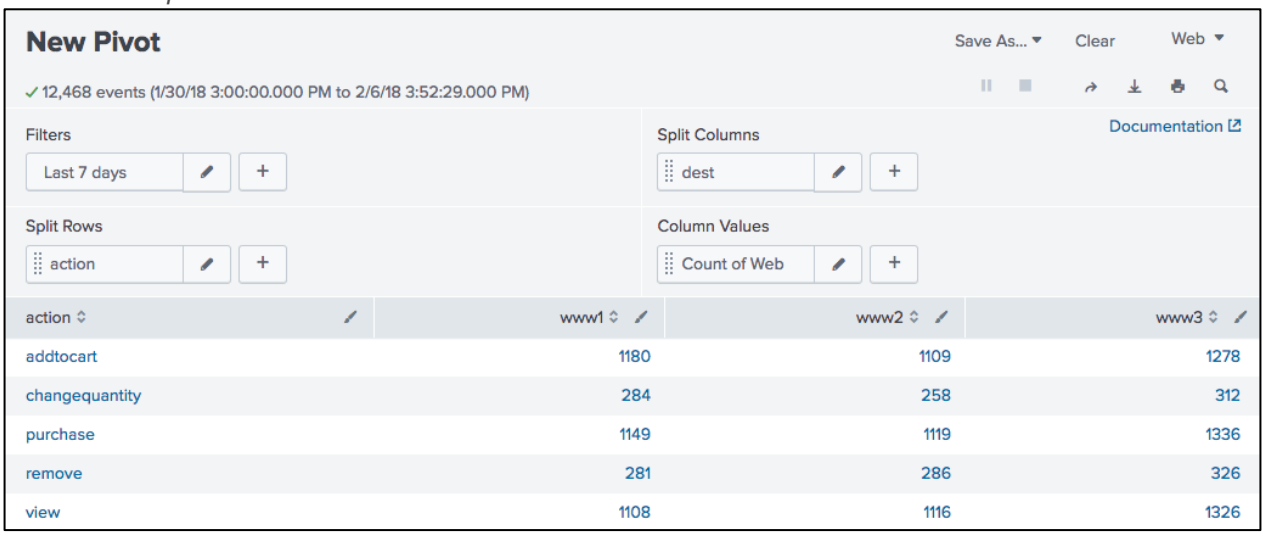
8. Create field aliases for the needed attributes that didn't populate.
   a) Navigate to Settings > Fields > Field aliases.
   b) Click New Field Alias.
   c) Verify Destination app is: **class_Fund2**
   d) In the Name box, type: `access_combined_aliases`
   e) From the Apply dropdown, make sure **sourcetype** is selected.
   f) In the **named** field, type: `access_combined`
   g) In the **Field aliases** left box, type: `clientip`
   h) In the **Field aliases** right box, type: `src`
   i) Click Add another field.
   j) Repeat the previous steps for the remaining fields and field aliases:
   k) host = dest
   l) useragent = http_user_agent
   m) Make sure your page looks identical to the example shown, and then click **Save**.



Task 5: Validate your data against the CIM Web data model.

9. Return to the CLASS: Fundamentals 2 app.
10. Navigate to Settings > Data models.
11. Using the **Web** data model, select **Pivot**.
12. Select the **Web** dataset object.
13. Filter on the Last 7 days and Split Rows by action and Split Columns by dest.

Results Example:



14. Change your pivot to **Split Rows** by **src**. Then change Split Columns by **status**. Are you able to split on all the expected fields in the Web data model?

NOTE: If your data model fields are not populating, delete the field alias and create it again. Be careful to avoid typos.