# 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.

<table>
<thead>
<tr>
<th>Index</th>
<th>Type</th>
<th>Sourcetype</th>
<th>Interesting Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>web</td>
<td>Online sales</td>
<td>access_combined</td>
<td>action, bytes, categoryId, clientip, itemId, JSESSIONID, price, productId, productName, referer, referer_domain, sale_price, status, user, useragent</td>
</tr>
<tr>
<td>security</td>
<td>Active Directory</td>
<td>winauthentication_security</td>
<td>LogName, SourceName, EventCode, EventType, User</td>
</tr>
<tr>
<td></td>
<td>Badge reader</td>
<td>history_access</td>
<td>Address_Description, Department, Device, Email, Event_Description, First_Name, last_Name, Rfid, Username</td>
</tr>
<tr>
<td></td>
<td>Web server</td>
<td>linux_secure</td>
<td>action, app, dest, process, src_ip, src_port, user, vendor_action</td>
</tr>
<tr>
<td>sales</td>
<td>Business Intelligence server</td>
<td>sales_entries</td>
<td>AcctCode, CustomerID, TransactionID</td>
</tr>
<tr>
<td></td>
<td>Retail sales</td>
<td>vendor_sales</td>
<td>AcctID, categoryId, product_name, productId, sale_price, Vendor, VendorCity, VendorCountry, VendorID, VendorStateProvince</td>
</tr>
<tr>
<td>network</td>
<td>Email security data</td>
<td>cisco_esa</td>
<td>dcid, icid, mailfrom, mailto, mid</td>
</tr>
<tr>
<td></td>
<td>Web security appliance data</td>
<td>cisco_wsa_squid</td>
<td>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_wbres_score, x_webcat_code_abbr</td>
</tr>
<tr>
<td></td>
<td>Firewall data</td>
<td>cisco_firewall</td>
<td>bgc_ip, dept, Duration, fname, IP, lname, location, rfid, splunk_role, splunk_server, Username</td>
</tr>
<tr>
<td>games</td>
<td>Game logs</td>
<td>SimCubeBeta</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>date_hour, date_mday, date_minute, date_month, date_second, data_wday, data_year, date_zone, eventtype, index, linecount, punct, splunk_server, timeendpos, timestartpos</td>
<td></td>
</tr>
</tbody>
</table>
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
3. index=web sourcetype=AcceSS_Combined
4. index=security sourcetype=linux_se%

**What is the most efficient filter?**

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

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.
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 all your searches in this app. Starting with Lab Exercise 2, you will also save 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? _______

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? _______

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.
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, the difference between Fast mode and Smart mode completion times probably won’t be significant.
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.

Results Example:

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

   Hint: Use `over ... by`
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.

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

**Results Example:**

<table>
<thead>
<tr>
<th>VendorCountry</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>303</td>
</tr>
<tr>
<td>United States</td>
<td>4839</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>22</td>
<td>17</td>
<td>24</td>
<td>17</td>
<td>36</td>
<td>9</td>
<td>101</td>
<td>7</td>
<td>24</td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>United States</td>
<td>538</td>
<td>297</td>
<td>404</td>
<td>308</td>
<td>306</td>
<td>311</td>
<td>747</td>
<td>137</td>
<td>536</td>
<td>565</td>
<td>314</td>
</tr>
</tbody>
</table>

**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).

**Results Example:**

<table>
<thead>
<tr>
<th>VendorCountry</th>
<th>Dream Crusher</th>
<th>Holy Blade of Gouds</th>
<th>Puppies vs. Zombies</th>
<th>SIM Cubicle</th>
<th>World of Cheese</th>
<th>OTHER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>United States</td>
<td>68</td>
<td>51</td>
<td>67</td>
<td>71</td>
<td>68</td>
<td>304</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>VendorCountry</th>
<th>Dream Crusher</th>
<th>Fire Resistance Suit of Provolone</th>
<th>Puppies vs. Zombies</th>
<th>SIM Cubicle</th>
<th>World of Cheese</th>
<th>OTHER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>22</td>
<td>24</td>
<td>7</td>
<td>24</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>United States</td>
<td>538</td>
<td>404</td>
<td>917</td>
<td>536</td>
<td>565</td>
<td></td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>VendorCountry</th>
<th>Dream Crusher</th>
<th>Fire Resistance Suit of Provolone</th>
<th>Puppies vs. Zombies</th>
<th>SIM Cubicle</th>
<th>World of Cheese</th>
<th>World of Cheese Tee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canada</td>
<td>22</td>
<td>24</td>
<td>7</td>
<td>24</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>United States</td>
<td>538</td>
<td>404</td>
<td>917</td>
<td>536</td>
<td>565</td>
<td></td>
</tr>
</tbody>
</table>

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:**

![Chart with custom labels](image)

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

![Logarithmic Y-axis chart](image)

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**.
31. Use the **timechart** command to count the events by **usage**.
32. Change the visualization to **Line Chart**.

**Results Example:**

![Line chart](image)
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.)
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).

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.

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

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

   **Results Example:**

   ![Area Chart Example](image)

47. Save the search as report, **L2C1**.

48. Optionally, revise the formatting to show retailSales as a chart overlay, and save as **L2C2**.

   ![Updated Area Chart](image)
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 user authentication 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.

Results Example:

<table>
<thead>
<tr>
<th>i</th>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>2/5/18 10:02:05 AM</td>
<td>Mon Feb 05 2018 18:02:05 www1 sshd[1224]: Failed password for root from 223.205.219.67 port 3411 ssh2 host = www1</td>
</tr>
<tr>
<td>&gt;</td>
<td>2/5/18 10:02:05 AM</td>
<td>Mon Feb 05 2018 18:02:05 www3 sshd[2803]: Failed password for invalid user perl from 202.179.8.245 port 2722 ssh2 host = www3</td>
</tr>
</tbody>
</table>

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

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, and display the sales count and trend for the previous day.

Final Results Example:

![Strategy Games Sales - Previous Day]

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

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:

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
</table>

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

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

**Results Example**

![Line Chart Example](image)

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**:

![Single Value Example](image)

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.

   **Results Example**:

   ![Map of United States with color-coded regions]

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>host = vendorUS1</td>
</tr>
<tr>
<td></td>
<td>host = vendorUS1</td>
</tr>
</tbody>
</table>

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

   **Results Example**:

   ![Table with state counts]

<table>
<thead>
<tr>
<th>VendorStateProvince</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alabama</td>
<td>54</td>
</tr>
<tr>
<td>Alaska</td>
<td>81</td>
</tr>
<tr>
<td>Arizona</td>
<td>75</td>
</tr>
<tr>
<td>Arkansas</td>
<td>54</td>
</tr>
<tr>
<td>California</td>
<td>527</td>
</tr>
</tbody>
</table>

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).

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.

**Results Example:**
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.)

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, [image].)

Results Example:

![Map Image]

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."

Results Example:

<table>
<thead>
<tr>
<th>VendorCountry</th>
<th>Units Sold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argentina</td>
<td>1</td>
</tr>
<tr>
<td>Australia</td>
<td>2</td>
</tr>
<tr>
<td>Belarus</td>
<td>1</td>
</tr>
<tr>
<td>Bermuda</td>
<td>1</td>
</tr>
</tbody>
</table>
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.

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:

<table>
<thead>
<tr>
<th>Country</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweden</td>
<td>1</td>
</tr>
<tr>
<td>The Bahamas</td>
<td>1</td>
</tr>
<tr>
<td>Turkey</td>
<td>1</td>
</tr>
<tr>
<td>Ukraine</td>
<td>2</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>4</td>
</tr>
<tr>
<td>United States</td>
<td>107</td>
</tr>
<tr>
<td>Venezuela</td>
<td>1</td>
</tr>
<tr>
<td>Vietnam</td>
<td>3</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>177</strong></td>
</tr>
</tbody>
</table>

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.

1. Search online sales [access_combined] during the previous week.
2. Use timechart to calculate the total bytes and name the field: bytes

Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-03-11</td>
<td>7028552</td>
</tr>
<tr>
<td>2018-03-12</td>
<td>7685197</td>
</tr>
<tr>
<td>2018-03-13</td>
<td>7225343</td>
</tr>
<tr>
<td>2018-03-14</td>
<td>7501807</td>
</tr>
<tr>
<td>2018-03-15</td>
<td>7539912</td>
</tr>
<tr>
<td>2018-03-16</td>
<td>7543386</td>
</tr>
<tr>
<td>2018-03-17</td>
<td>7492738</td>
</tr>
</tbody>
</table>

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

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

**Results Example:**

<table>
<thead>
<tr>
<th>_time</th>
<th>bytes</th>
<th>megabytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-03-11</td>
<td>7028552</td>
<td>6.70</td>
</tr>
<tr>
<td>2018-03-12</td>
<td>7685197</td>
<td>7.33</td>
</tr>
<tr>
<td>2018-03-13</td>
<td>7225343</td>
<td>6.89</td>
</tr>
<tr>
<td>2018-03-14</td>
<td>7501807</td>
<td>7.15</td>
</tr>
<tr>
<td>2018-03-15</td>
<td>7539912</td>
<td>7.19</td>
</tr>
<tr>
<td>2018-03-16</td>
<td>7543386</td>
<td>7.19</td>
</tr>
<tr>
<td>2018-03-17</td>
<td>7492738</td>
<td>7.15</td>
</tr>
</tbody>
</table>

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.

**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:**

<table>
<thead>
<tr>
<th>host</th>
<th>GET</th>
<th>POST</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>www1</td>
<td>709</td>
<td>381</td>
<td>1.86</td>
</tr>
<tr>
<td>www2</td>
<td>766</td>
<td>456</td>
<td>1.68</td>
</tr>
<tr>
<td>www3</td>
<td>782</td>
<td>466</td>
<td>1.68</td>
</tr>
</tbody>
</table>

8. Search for all events in the online store [access_combined] during the **last 24 hours**.
9. Use **chart** to count events over **host** by **method**.

**Results Example:**

<table>
<thead>
<tr>
<th>host</th>
<th>GET</th>
<th>POST</th>
</tr>
</thead>
<tbody>
<tr>
<td>www1</td>
<td>709</td>
<td>381</td>
</tr>
<tr>
<td>www2</td>
<td>766</td>
<td>456</td>
</tr>
<tr>
<td>www3</td>
<td>780</td>
<td>461</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>host</th>
<th>GET</th>
<th>POST</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>www1</td>
<td>709</td>
<td>381</td>
<td>1.8608923884514437</td>
</tr>
<tr>
<td>www2</td>
<td>766</td>
<td>456</td>
<td>1.6798246614035088</td>
</tr>
<tr>
<td>www3</td>
<td>780</td>
<td>461</td>
<td>1.6919739666312364</td>
</tr>
</tbody>
</table>

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

**Results Example:**
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:**

<table>
<thead>
<tr>
<th>user</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>myuwn</td>
<td>105</td>
</tr>
<tr>
<td>nsharpe</td>
<td>51</td>
</tr>
<tr>
<td>root</td>
<td>16</td>
</tr>
<tr>
<td>djohnson</td>
<td>12</td>
</tr>
<tr>
<td>operator</td>
<td>11</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>i</th>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>2/5/18 11:53:29 AM</td>
<td>Mon Feb 05 2018 19:53:29 sshd[147]: Failed password for nobody from 147.213.138.201 port 4286 ssh2 host = www1 source = /opt/log/wwwsecure.log</td>
</tr>
<tr>
<td>&gt;</td>
<td>2/5/18 11:53:29 AM</td>
<td>Mon Feb 05 2018 19:53:29 sshd[2836]: Failed password for invalid user from 94.238.166.185 port 3791 ssh2 host = www2 source = /opt/log/wwwsecure.log</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>user</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>admin</td>
<td>8</td>
</tr>
<tr>
<td>administrator</td>
<td>2</td>
</tr>
<tr>
<td>agustho</td>
<td>1</td>
</tr>
<tr>
<td>apache</td>
<td>1</td>
</tr>
<tr>
<td>ent</td>
<td>1</td>
</tr>
<tr>
<td>backup</td>
<td>2</td>
</tr>
</tbody>
</table>

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

**Results Example:**
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:

- Small
- Medium
- Large
- Extra Large

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.

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

**Results example:**

![Pie chart example](image)

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:**

![Pie chart example](image)

20. Search web appliance data `[cisco_wsa_squid]` during the **previous business week**.
21. Use `stats` or `chart` to count events by the `http_content_type` field.

**Results Example:**

<table>
<thead>
<tr>
<th><code>http_content_type</code></th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>818</td>
</tr>
<tr>
<td><code>application/javascript</code></td>
<td>111</td>
</tr>
<tr>
<td><code>application/octet-stream</code></td>
<td>63</td>
</tr>
<tr>
<td><code>application/x-dosexec</code></td>
<td>1</td>
</tr>
<tr>
<td><code>application/x-javascript</code></td>
<td>446</td>
</tr>
<tr>
<td><code>application/x-shockwave-flash</code></td>
<td>34</td>
</tr>
<tr>
<td><code>image/bmp</code></td>
<td>6</td>
</tr>
</tbody>
</table>

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

**Results Example:**

<table>
<thead>
<tr>
<th>http_content_type</th>
<th>count</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>818</td>
<td>other</td>
</tr>
<tr>
<td>application/javascript</td>
<td>111</td>
<td>other</td>
</tr>
<tr>
<td>application/octet-stream</td>
<td>63</td>
<td>other</td>
</tr>
<tr>
<td>application/x-dosexec</td>
<td>1</td>
<td>other</td>
</tr>
<tr>
<td>application/x-javascript</td>
<td>446</td>
<td>other</td>
</tr>
<tr>
<td>application/x-shockwave-flash</td>
<td>34</td>
<td>other</td>
</tr>
<tr>
<td>image/bmp</td>
<td>6</td>
<td>graphic</td>
</tr>
</tbody>
</table>

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`.

**Results Example:**

<table>
<thead>
<tr>
<th>type</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>graphic</td>
<td>3583</td>
</tr>
<tr>
<td>other</td>
<td>2296</td>
</tr>
</tbody>
</table>

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

**Results Example:**

![Pie Chart]

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

**CHALLENGE Exercise:**

Report which one-hour periods over the last 24 hours have seen the number of Buttercup Games online sales twice as numerous as the number of sales in retail stores.
Final Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>sales</th>
<th>web</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-09-11 10:00</td>
<td>39</td>
<td>139</td>
</tr>
<tr>
<td>2019-09-11 11:00</td>
<td>40</td>
<td>122</td>
</tr>
<tr>
<td>2019-09-11 12:00</td>
<td>40</td>
<td>174</td>
</tr>
<tr>
<td>2019-09-11 13:00</td>
<td>36</td>
<td>145</td>
</tr>
<tr>
<td>2019-09-11 14:00</td>
<td>36</td>
<td>143</td>
</tr>
<tr>
<td>2019-09-11 15:00</td>
<td>39</td>
<td>142</td>
</tr>
</tbody>
</table>

26. Search online sales data [access_combined] and retail sales data [vendor_sales] for successful purchases during the last 24 hours.

27. Use timechart to count the sales events by index using a sampling interval of 1 hour.

Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>sales</th>
<th>web</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-09-11 11:00</td>
<td>40</td>
<td>122</td>
</tr>
<tr>
<td>2019-09-11 12:00</td>
<td>40</td>
<td>174</td>
</tr>
<tr>
<td>2019-09-11 13:00</td>
<td>36</td>
<td>145</td>
</tr>
<tr>
<td>2019-09-11 14:00</td>
<td>36</td>
<td>143</td>
</tr>
<tr>
<td>2019-09-11 15:00</td>
<td>39</td>
<td>142</td>
</tr>
<tr>
<td>2019-09-11 16:00</td>
<td>39</td>
<td>159</td>
</tr>
</tbody>
</table>

28. Use a where command to keep only rows where the number of web sales are more than twice the number of retail sales.

*Results Example:*
29. Save your search as report, L4C2.

30. Modify your previous search to use search instead of where and observe the results. Why are the results different?
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:

<table>
<thead>
<tr>
<th>JSESSIONID</th>
<th>clientip</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD7SL8FF5ADFF4957</td>
<td>86.9.190.90</td>
<td>addtocart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>purchase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>view</td>
</tr>
<tr>
<td>SD6SL9FF5ADFF4961</td>
<td>81.18.148.190</td>
<td>addtocart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>purchase</td>
</tr>
<tr>
<td></td>
<td></td>
<td>view</td>
</tr>
<tr>
<td>SD2SL10FF2ADFF4963</td>
<td>194.215.205.19</td>
<td>addtocart</td>
</tr>
<tr>
<td></td>
<td></td>
<td>purchase</td>
</tr>
</tbody>
</table>

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

Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>clientip</th>
<th>JSESSIONID</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-02-05 12:40:03</td>
<td>211.166.11.101</td>
<td>SD05L3FF5ADFF4950</td>
<td></td>
</tr>
<tr>
<td>2018-02-05 12:39:45</td>
<td>211.166.11.101</td>
<td>SD05L3FF5ADFF4950</td>
<td></td>
</tr>
<tr>
<td>2018-02-05 12:37:35</td>
<td>211.245.24.3</td>
<td>SD05L7FF4ADFF4956</td>
<td></td>
</tr>
<tr>
<td>2018-02-05 12:37:18</td>
<td>211.245.24.3</td>
<td>SD05L7FF4ADFF4956</td>
<td>addtocart</td>
</tr>
<tr>
<td>2018-02-05 12:28:05</td>
<td>91.199.80.24</td>
<td>SD1SL10FF7ADFF4953</td>
<td></td>
</tr>
<tr>
<td>2018-02-05 12:27:55</td>
<td>91.199.80.24</td>
<td>SD1SL10FF7ADFF4953</td>
<td>purchase</td>
</tr>
</tbody>
</table>

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

Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>clientip</th>
<th>JSESSIONID</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-02-05 12:44:02</td>
<td>195.2.240.99</td>
<td>SD05L6FF5ADFF4959</td>
<td>view</td>
</tr>
<tr>
<td>2018-02-05 12:43:51</td>
<td>195.2.240.99</td>
<td>SD05L6FF5ADFF4959</td>
<td>addtocart</td>
</tr>
<tr>
<td>2018-02-05 12:37:18</td>
<td>211.245.24.3</td>
<td>SD05L7FF4ADFF4956</td>
<td>addtocart</td>
</tr>
<tr>
<td>2018-02-05 12:27:55</td>
<td>91.199.80.24</td>
<td>SD1SL10FF7ADFF4953</td>
<td>purchase</td>
</tr>
<tr>
<td>2018-02-05 12:27:55</td>
<td>91.199.80.24</td>
<td>SD1SL10FF7ADFF4953</td>
<td>purchase</td>
</tr>
</tbody>
</table>
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.

**Results Example:**

![Results Example Table]

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

**Results Example:**

![Results Example Table]

**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.

**Results Example:**
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:**

<table>
<thead>
<tr>
<th>JSESSIONID</th>
<th>clientip</th>
<th>action</th>
<th>durationMinutes</th>
<th>eventcount</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD7SL8FF5ADFF4957</td>
<td>86.9.190.90</td>
<td>addtocart purchase view</td>
<td>1.3</td>
<td>11</td>
</tr>
<tr>
<td>SD6SL9FF5ADFF4961</td>
<td>81.18.148.190</td>
<td>addtocart purchase view</td>
<td>2.7</td>
<td>13</td>
</tr>
<tr>
<td>SD2SL10FF2ADFF4963</td>
<td>194.215.205.19</td>
<td>addtocart purchase remove view</td>
<td>1.4</td>
<td>9</td>
</tr>
</tbody>
</table>

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**.

**Results Example:**

<table>
<thead>
<tr>
<th>JSESSIONID</th>
<th>clientip</th>
<th>duration</th>
<th>eventcount</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD7SL8FF6ADFF4957</td>
<td>86.3.190.90</td>
<td>77</td>
<td>11</td>
<td>addtocart purchase view</td>
</tr>
<tr>
<td>SD6SL9FF5ADFF4961</td>
<td>81.18.148.190</td>
<td>32</td>
<td>5</td>
<td>addtocart purchase view</td>
</tr>
<tr>
<td>SD2SL10FF2ADFF4963</td>
<td>194.215.205.19</td>
<td>46</td>
<td>9</td>
<td>addtocart purchase remove</td>
</tr>
</tbody>
</table>
12. Use \texttt{eval} to create a new field named \texttt{durationMinutes}, which is the rounded value of \texttt{duration} divided by 60. Round to one decimal place.

\textit{Results Example:}

<table>
<thead>
<tr>
<th>JSESSIONID</th>
<th>clientip</th>
<th>duration</th>
<th>eventcount</th>
<th>action</th>
<th>durationMinutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD75L8FF6ADF4957</td>
<td>86.9.190.90</td>
<td>77</td>
<td>11</td>
<td>addtocart purchase view</td>
<td>1.3</td>
</tr>
<tr>
<td>SD65L9FF5ADF4961</td>
<td>8118.148.190</td>
<td>32</td>
<td>5</td>
<td>addtocart purchase view</td>
<td>0.5</td>
</tr>
<tr>
<td>SD25L10FF2ADF4963</td>
<td>194.215.205.19</td>
<td>46</td>
<td>9</td>
<td>addtocart purchase remove</td>
<td>0.8</td>
</tr>
</tbody>
</table>

13. Modify your search to find data where the \texttt{durationMinutes} is greater than one minute. Adjust the table to display only \texttt{JSESSIONID}, \texttt{clientip}, \texttt{action}, \texttt{durationMinutes}, and \texttt{eventcount}, in that order.

\textit{Results Example:}

<table>
<thead>
<tr>
<th>JSESSIONID</th>
<th>clientip</th>
<th>action</th>
<th>durationMinutes</th>
<th>eventcount</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD75L8FF6ADF4957</td>
<td>86.9.190.90</td>
<td>addtocart purchase view</td>
<td>1.3</td>
<td>11</td>
</tr>
<tr>
<td>SD15L10FF7ADF4953</td>
<td>91199.80.24</td>
<td>addtocart purchase remove</td>
<td>2.7</td>
<td>13</td>
</tr>
<tr>
<td>SD35L8FF9ADF4955</td>
<td>195.69.252.22</td>
<td>addtocart purchase remove</td>
<td>1.4</td>
<td>9</td>
</tr>
</tbody>
</table>

14. Save your search as report, \texttt{L5S2}.

\textbf{Task 3: Search for online store transactions that begin with an \texttt{addtocart} action and end with a purchase action.}

\textit{Final Results Example:}

<table>
<thead>
<tr>
<th>clientip</th>
<th>JSESSIONID</th>
<th>product_name</th>
<th>action</th>
<th>duration</th>
<th>eventcount</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>199.15.234.66</td>
<td>SD15L0FF2ADF4957</td>
<td>Dream Crusher</td>
<td>addtocart purchase</td>
<td>4</td>
<td>2</td>
<td>39.99</td>
</tr>
<tr>
<td>86.9.190.90</td>
<td>SD75L8FF6ADF4957</td>
<td>World of Cheese</td>
<td>addtocart purchase</td>
<td>1</td>
<td>2</td>
<td>9.99</td>
</tr>
<tr>
<td>86.9.190.90</td>
<td>SD75L8FF6ADF4957</td>
<td>Holy Blade of Gouds</td>
<td>addtocart purchase</td>
<td>3</td>
<td>2</td>
<td>5.99</td>
</tr>
</tbody>
</table>

15. Search for all events from the online store \texttt{[access_combined]} in the \textbf{last 60 minutes} and correlate the events based on \texttt{clientip}.

16. Use the \texttt{startswith} and \texttt{endswith} options of the \texttt{transaction} command to display transactions that begin with an \texttt{addtocart} action and end with a purchase action.
17. In a table, display clientip, JSESSIONID, product_name, action, duration, eventcount, and price.

Results Example:

<table>
<thead>
<tr>
<th>clientip</th>
<th>JSESSIONID</th>
<th>product_name</th>
<th>action</th>
<th>duration</th>
<th>eventcount</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>199.15.234.66</td>
<td>SD105L0FF2AFDFF4963</td>
<td>Dream Crusher</td>
<td>addcart purchase</td>
<td>4</td>
<td>2</td>
<td>39.99</td>
</tr>
<tr>
<td>86.190.90</td>
<td>SD7186FD4DFDF4967</td>
<td>World of Cheese Tee</td>
<td>addcart purchase</td>
<td>1</td>
<td>2</td>
<td>9.99</td>
</tr>
<tr>
<td>86.190.90</td>
<td>SD7186FD4DFDF4967</td>
<td>Holy Blade of Gouds</td>
<td>addcart purchase</td>
<td>3</td>
<td>2</td>
<td>5.99</td>
</tr>
</tbody>
</table>

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.

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.

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

4. Use timechart to count events by status.

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

   Results Example:

<table>
<thead>
<tr>
<th>_time</th>
<th>400</th>
<th>403</th>
<th>404</th>
<th>503</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>2018-01-06</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>2018-01-08</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>2018-01-09</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>2018-01-10</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
</tbody>
</table>

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

   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.
2. View the event details to see all the extracted fields. Click the > arrow under the 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 may see “::” listed as a valid value. In the filter field, type src=:: and click Apply.
13. If applicable, 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.

**NOTE:** It may take a few moments for the newly extracted fields to appear in the search because the training environment uses an index cluster, and it takes a minute for knowledge objects to replicate across the cluster. (For details, attend 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:**

<table>
<thead>
<tr>
<th>src</th>
<th>port</th>
<th>count</th>
<th>percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>107.3146.207</td>
<td>3057</td>
<td>2</td>
<td>3.703704</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4950</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4929</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4822</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4800</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4779</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4550</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4506</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4141</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>107.3146.207</td>
<td>4131</td>
<td>1</td>
<td>1.851852</td>
</tr>
<tr>
<td>108.50.217.115</td>
<td>8677</td>
<td>100</td>
<td>87.719298</td>
</tr>
<tr>
<td>108.50.217.115</td>
<td>7238</td>
<td>1</td>
<td>0.877193</td>
</tr>
</tbody>
</table>

**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 automatically 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.
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 32.)

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. 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:*
<table>
<thead>
<tr>
<th>i</th>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/5/19 15:48:00 PM</td>
<td>05/Feb/2018:21:50:48, 121.254.179.199, v2.0.02B, User: '<a href="mailto:chocolateswife@verizon.net">chocolateswife@verizon.net</a>' CharacterName: 'nicea55' Action: 'Made Coffee' CurrentStanding: 'Office Joke'</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** It may take a few minutes before the newly extracted fields appear in the search.
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.
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:
```
a splunk_server 4
a src 100+
a src_ip 100+
# status 9
# timeendpos 1
# timestartpos 1
a url 100+
a usage 5
a user 72
```

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?

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.
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.

Results Example:

<table>
<thead>
<tr>
<th>usage</th>
<th>Bandwidth (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Borderline</td>
<td>6.86968708038330100000</td>
</tr>
<tr>
<td>Business</td>
<td>17.08714580535888700000</td>
</tr>
<tr>
<td>Personal</td>
<td>54.93885517120351000000</td>
</tr>
<tr>
<td>Unknown</td>
<td>17.56064128875732400000</td>
</tr>
<tr>
<td>Violation</td>
<td>0.87615489959716800000</td>
</tr>
</tbody>
</table>

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.

   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. Performing an occasional inefficient ad hoc search shouldn’t have too much of a performance impact, but such searches certainly shouldn’t be used in reports, dashboards, dataset constraints, etc.

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:

<table>
<thead>
<tr>
<th>Type</th>
<th>Field</th>
<th>Value</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>host</td>
<td>www2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>source</td>
<td>app/log/secured/log</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sourcetype</td>
<td>linux_secure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>action</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>app</td>
<td>sshd</td>
<td></td>
</tr>
<tr>
<td></td>
<td>dest</td>
<td>www2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>eventtype</td>
<td>admin</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>failed_login</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nsa_logs</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nsa_error</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nsa_security</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sshd_authentication</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pid</td>
<td>1098</td>
<td></td>
</tr>
<tr>
<td></td>
<td>port</td>
<td>2277</td>
<td></td>
</tr>
<tr>
<td></td>
<td>process</td>
<td>sshd</td>
<td></td>
</tr>
<tr>
<td></td>
<td>src</td>
<td>76.169.2.352</td>
<td></td>
</tr>
<tr>
<td></td>
<td>src_ip</td>
<td>76.169.2.352</td>
<td></td>
</tr>
<tr>
<td></td>
<td>src_port</td>
<td>2277</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sshd_protocol</td>
<td>ssh2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>tag</td>
<td>authentication</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>error</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>on</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>remote</td>
<td></td>
</tr>
</tbody>
</table>

| user | sapadmin | Edit Tags |
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, itadmin, 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.

6. If it isn't already, add **tag** to your list of Selected Fields.

**Results example:**

![Results example](image)

**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, itadmin, sapadmin.

**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 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**.

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**.

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:

```
<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/6/18 10:30:31 AM</td>
<td>82.216.64.19 - - [06/Feb/2018:18:38:31] &quot;POST /oldLink?itemid=EST-13&amp;SESSIONID=SDY3S1FF18ADFF4952 HTTP 1.1&quot; 5 853 &quot;<a href="http://www.buttercupgames.com/oldLink?itemid=EST-13">http://www.buttercupgames.com/oldLink?itemid=EST-13</a>&quot; &quot;Mozilla/5.0 (compatible; Googlebot/2.1; <a href="http://www.google.com/bot.html">http://www.google.com/bot.html</a>)&quot; 778</td>
</tr>
<tr>
<td>2/6/18 10:15:36 AM</td>
<td>84.34.159.23 - - [06/Feb/2018:18:15:36] &quot;POST /product.screen?productid=SF-BVS-81&amp;SESSIONID=SDY3S1FF18ADFF4952 HTTP 1.1&quot; 5 583 2289 &quot;<a href="http://www.buttercupgames.com/product.screen?productid=SF-BVS-81">http://www.buttercupgames.com/product.screen?productid=SF-BVS-81</a>&quot; &quot;Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)&quot; 598</td>
</tr>
</tbody>
</table>
```

**NOTE:** Depending upon add-ons or apps you have installed, additional event types may be displayed.
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 US Dollars to the same value in another currency.

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

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:
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 IN (Germany, France, 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.

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.

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.

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)`

**NOTE:** You can copy/paste the € symbol from this document or go to the following website for the keyboard shortcuts: [http://bit.ly/2BqMmR0](http://bit.ly/2BqMmR0)

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.

Results Example:

<table>
<thead>
<tr>
<th>product_name</th>
<th>USD</th>
<th>GBP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benign Space Debris</td>
<td>$374.85</td>
<td>£239.90</td>
</tr>
<tr>
<td>Curling 2014</td>
<td>$259.87</td>
<td>£166.32</td>
</tr>
<tr>
<td>Dream Crusher</td>
<td>$479.88</td>
<td>£307.12</td>
</tr>
<tr>
<td>Final Sequel</td>
<td>$74.97</td>
<td>£47.98</td>
</tr>
<tr>
<td>Fire Resistance Suit of Provolone</td>
<td>$95.76</td>
<td>£51.29</td>
</tr>
<tr>
<td>Holy Blade of Gouda</td>
<td>$101.83</td>
<td>£55.17</td>
</tr>
<tr>
<td>Manganiello Bros.</td>
<td>$759.81</td>
<td>£486.28</td>
</tr>
<tr>
<td>Manganiello Bros. Tee</td>
<td>$199.80</td>
<td>£127.87</td>
</tr>
<tr>
<td>Mediocre Kingdoms</td>
<td>$349.86</td>
<td>£223.91</td>
</tr>
<tr>
<td>Orvil the Wolverine</td>
<td>$399.90</td>
<td>£255.94</td>
</tr>
<tr>
<td>Puppies vs. Zombies</td>
<td>$4.99</td>
<td>£3.19</td>
</tr>
<tr>
<td>SIM Cubide</td>
<td>$319.84</td>
<td>£204.70</td>
</tr>
<tr>
<td>World of Cheese</td>
<td>$499.80</td>
<td>£319.87</td>
</tr>
<tr>
<td>World of Cheese Tee</td>
<td>$169.83</td>
<td>£108.69</td>
</tr>
</tbody>
</table>

**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.
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`).

30. Check to see that your error message displays.

Results Example:

![Search Result Example](image-url)
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 uses 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. 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.

17. Navigate to Settings > Fields > Workflow actions.
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:
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: Create a data model and add a 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. (Notice that this automatically fills in the ID field. **Don't** delete this value. The ID field cannot be blank.)
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. Click the checkboxes to select the following fields, and rename them for pivot users as indicated:
   - action > action taken
   - bytes > size
   - categoryld > product category
   - clientip > client IP
   - date-mday > date-mday (use same name)
   - productld > product ID
   - product_name > product name
   - req_time > request time
   - status > status (use same name)

Example:
14. Click **Save**.

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

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

26. Click **Pivot** in the upper right corner to test the data model.
27. Select the Web requests dataset.
28. 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:

![New Pivot](image)

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

29. Select **Edit Dataset**.
30. Make sure Web requests is selected.
31. From the **Add Field** dropdown, select **Eval Expression**.
32. 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.

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

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

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

**Results Example:**

![Weekly Website Activity Table](image)

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

49. 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**.
50. Navigate to **Settings > Data models**.
51. Select the Buttercup Games Site Activity data model.
52. Make sure the Web requests root dataset is selected.
53. Click Add Field and select Lookup.
54. From the Lookup Table dropdown list, select http_status_lookup.
55. For the Input section in the Field in Lookup dropdown, select code.
56. 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.
57. For the lookup Output section in the Field in Lookup field, check the description checkbox.
58. In the Display Name field, type: status description
59. Click the Preview button. You should see a description column in the results.
60. Click Save.

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

61. Click Pivot.
62. Select the Web requests dataset.
63. Change the Filter to Last 7 days.
64. From Split Rows, add the status description attribute and click Add To Table.
65. 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:

<table>
<thead>
<tr>
<th>status description</th>
<th>code</th>
<th>Count of Web requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bad Request.</td>
<td>400</td>
<td>204</td>
</tr>
<tr>
<td>Forbidden.</td>
<td>403</td>
<td>56</td>
</tr>
<tr>
<td>HTTP Version Not Supported.</td>
<td>505</td>
<td>146</td>
</tr>
<tr>
<td>Internal Server Error</td>
<td>500</td>
<td>170</td>
</tr>
<tr>
<td>Not Acceptable.</td>
<td>406</td>
<td>201</td>
</tr>
<tr>
<td>Not Found.</td>
<td>404</td>
<td>192</td>
</tr>
<tr>
<td>OK</td>
<td>200</td>
<td>1119</td>
</tr>
<tr>
<td>Request Timeout.</td>
<td>408</td>
<td>192</td>
</tr>
<tr>
<td>Service Unavailable</td>
<td>503</td>
<td>261</td>
</tr>
</tbody>
</table>

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

Results Example:
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 *Shopping 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.
9. Click Save As and select Dashboard Panel.
10. Save to the **Weekly Website Activity** dashboard.
11. For **Panel Title**, type: Add – Purchase – View only
12. **Save** and **view** your dashboard.
13. Rearrange the panels to your liking and admire your work!
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.
3. Examine the values of the following fields. These field values are required for your dashboard, but their current names aren’t CIM-compliant.
   - 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?

<table>
<thead>
<tr>
<th>Field name in source type</th>
<th>Field in Data Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>dest</td>
</tr>
<tr>
<td>action</td>
<td>action</td>
</tr>
<tr>
<td>clientip</td>
<td>src</td>
</tr>
<tr>
<td>status</td>
<td>status</td>
</tr>
<tr>
<td>useragent</td>
<td>http_user_agent</td>
</tr>
</tbody>
</table>
7. Using the `datamodel` command, are the fields in your data populated in the Web data model?

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

<table>
<thead>
<tr>
<th>Field in Your Data</th>
<th>Matching Attribute</th>
<th>Data Model Field Populated?</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>dest</td>
<td>No</td>
</tr>
<tr>
<td>action</td>
<td>action</td>
<td>Yes</td>
</tr>
<tr>
<td>clientip</td>
<td>src</td>
<td>No</td>
</tr>
<tr>
<td>status</td>
<td>status</td>
<td>Yes</td>
</tr>
<tr>
<td>useragent</td>
<td>http_user_agent</td>
<td>No</td>
</tr>
</tbody>
</table>

**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.

**Task 3:** Validate your data against the CIM Web data model.

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.