Lab Exercises

Lab typographical conventions:

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

The lab instructions refer to these source types by the types of data they represent:

Type	Sourcetype	Fields of interest
Web Application	`access_combined_wcookie`	`action, bytes, categoryId, clientip, itemId, JSESSIONID, productId, referer, referer_domain, status, useragent, file`
Database	`db_audit`	`Command, Duration, Type`
Web server	`linux_secure`	`COMMAND, PWD, pid, process`

Lab Module 9 – Transforming Commands

NOTE: This lab document has two sections. The first section includes the instructions without answers. The second section includes instructions with the expected search string (answer) in red.

Description

In this lab, you will be using some of the Splunk's transforming commands to derive statistics from the data.

Steps

Scenario: The sales group would like a report of the five best-selling products by productId.

Task 1: Use the top command to get a list of the best-selling products.

1. Navigate to the Search view. (If you are in the **Home** app, click **Search & Reporting** from the column on the left side of the screen. You can also access the Search view by clicking the **Search** menu option on the bar at the top of the screen.)

NOTE: For this course, you will be searching across all time using the main index. This is NOT a best practice in a production environment, but needed for these labs due to the nature of the limited dataset.

2. Enter a search that returns all web application events where an item was successfully purchased. Remember that if the `success.do` file is successfully returned to a user, a purchase was made.

Results Example:

i	Time	Event
>	5/21/18 11:57:14.000 PM	109.169.32.135 - - [21/May/2018:23:57:14] "POST /cart/success.do?JSESSIONID=SD1SL7FF6ADFF89341&productId=FI-AG-G08 HTTP 1.1" 200 3767 "http://www.buttercupgames.com/cart.do?action=purchase&" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 986 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie
>	5/21/18 11:57:13.000 PM	109.169.32.135 - - [21/May/2018:23:57:13] "POST /success.do?action=purchase&categoryId=SHOOTER&productId=WC-SH-G04&JSESSIONID=SD1SL7FF6ADFF89341 HTTP 1.1" 200 268 "http://www.buttercupgames.com/cart.do?action=addtocart&categoryId=SHOOTER&productId=WC-SH-G04" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 448 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie
>	5/21/18 11:53:43.000 PM	198.35.3.23 - - [21/May/2018:23:53:43] "POST /success.do?action=purchase&categoryId=ARCADE&productId=MB-AG-G07&JSESSIONID=SD8SL8FF6ADFF4957 HTTP 1.1" 200 2915 "http://www.buttercupgames.com/cart.do?action=addtocart&categoryId=ARCADE&productId=MB-AG-G07" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 448 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie

3. Use the `top` command to find the best-selling productIds for all time.

Results Example:

productId	count	percent
WC-SH-G04	1360	8.426792
DB-SG-G01	1319	8.172749
DC-SG-G02	1308	8.104591
MB-AG-T01	1205	7.466386
MB-AG-G07	1204	7.460190
WC-SH-A02	1192	7.385836
FS-SG-G03	1155	7.156577
WC-SH-A01	1100	6.815788
WC-SH-T02	1076	6.667080
PZ-SG-G05	1012	6.270525

4. Notice that ten rows were returned. You were asked to only return the top 5.
5. Use the `limit` argument to only return the number of rows requested.

Results Example:

productId	count	percent
WC-SH-G04	1360	8.426792
DB-SG-G01	1319	8.172749
DC-SG-G02	1308	8.104591
MB-AG-T01	1205	7.466386
MB-AG-G07	1204	7.460190

6. Use the `showperc` option of `top` to remove `percent` from the display.

Results Example:

productId	count
WC-SH-G04	1360
DB-SG-G01	1319
DC-SG-G02	1308
MB-AG-T01	1205
MB-AG-G07	1204

7. What is the productId of the best-selling product? Please take note as it might be in the quiz.

Scenario: There are times when rare events can give you some of the best information. For example rarely accessed files, these can be files that someone internal to the company has made publically available, or a backdoor that few people know about. The security team has asked you to find files that are accessible online, but get very little traffic.

Task 2: Use the rare command to see what files get accessed the least amount in our web application.

8. Enter a search that returns all web application events where a file was successfully served to the user.
9. Use the `rare` command to find the files that show up the least amount of times in our events.

Results Example:

file	count	percent
api	1	0.000858
account	2	0.001716
userlist	10	0.008582
passwords.pdf	235	0.201670
error.do	1795	1.540416
success.do	16139	13.850009
oldlink	19642	16.856179
category.screen	19958	17.127361
cart.do	29328	25.168416
product.screen	29417	25.244793

10. Do you see anything that might be of a concern for the security team? Make the report even more granular using the `by` clause to split the `rare` events by the month in which they happened (`date_month`).

Scenario: The sales team would like to know the number of items added to carts, versus the number of items that get purchased.

Task 3: Use the count function of the stats command to find out how many items were added to a cart versus being purchased.

11. Enter a search that returns all web application events where an item was successfully added to a cart, or purchased. Remember, when an item is added to the cart the `cart.do` file is served; the `success.do` is served when the item is purchased.

12. Use the `stats count` function with a `by` clause to count events by the file that was served.

Results Example:

file	count
cart.do	29328
success.do	16139

13. Notice that the count column is labeled `count` by default. Use an `as` clause to rename the column to `Transactions`.

Results Example:

file	Transactions
cart.do	29328
success.do	16139

14. Using the `rename` command, change the name of the `file` field to `Function`.

Results Example:

Function	Transactions
cart.do	29328
success.do	16139

Scenario: The marketing manager would like to know how many times users have logged into the web application.

Task 4: Use the distinct count stats function to count how many times sessions were created for users on the system.

15. Use search terms with the `stats dc` function to count all sessions (`JSESSIONID`) that have been used in our web application data.

Results Example:

dc(JSESSIONID)
11455

16. Use the `as` clause to rename the sessions as `Logins`.

Scenario: While being able to see the number of logins, the manager would like a little more information on who is logging in and asked you to report log-ins by IP.

17. Using the `by` clause, split the `Logins` by `clientip`.

Results Example:

clientip	Logins
107.3.146.207	186
108.65.113.83	83
109.169.32.135	172
110.138.30.229	82
110.159.208.78	95

18. Use the `sort` command to sort `Logins` so that the `clientip` with the most `Logins` is displayed at the top of the list. Make a note of the top `clientip` you might be asked about it in the quiz.

Scenario: The IT team is working on its infrastructure budget for the year. It's trying to get a handle on where most of their bandwidth cost is being spent.

Task 5: Use the stats sum function to find the total bytes used for the web application.

19. Craft search terms that return all events where a file was successfully served to a user.
20. Create a field named `TotalBytes` by using the `sum` function of the `stats` command.

Results Example:

TotalBytes
255334688

21. Split the results by the `file` field using the `by` clause.

Results Example:

file	TotalBytes
account	4238
api	1456
cart.do	61311724
category.screen	42250130
error.do	3747647
oldlink	41349801
passwords.pdf	11103985
product.screen	61672339
success.do	33862909
userlist	27690

22. Use the `sort` command to sort the file names into alphabetical order.
23. What is the name of the file that used the least amount of bandwidth? You might be asked about it in the quiz.

Scenario: EMERGENCY! The website has come to a crawl causing users to complain. An application developer tells you that they pushed an update recently and is concerned that there might be a database query that is causing the issue.

Task 6: Use the stats command's average function to find the average time for each database query being run.

24. Search all database events and use the average (`avg`) function of the `stats` command to get an average `Duration` of all queries.

Results Example:

avg(Duration)
239.4764303178484

25. Use `as` and `by` clauses to rename the average field to `time to complete` and split by the `Command`.

Results Example:

Command	time to complete
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 2208	9988
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 1021	9985
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 4011	9985
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 6186	9980
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 2429	9971

26. Sort the `time to complete` so that `Command` values that take the longest are shown first.
27. Notice anything about the `Command` values that are taking the longest to complete?

Scenario: The Web Development team has asked for a list of the browsers that are being used to access the web application.

Task 7: Use the lists and values functions of the stats command to run a report of the browsers users are using to access the web application from.

28. Use the `stats list` function to generate a list of all `useragent` values that have accessed the web application.

Results Example:

```
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.28) Gecko/20120306 YFF3 Firefox/3.6.28 ( .NET CLR 3.5.30729; .NET4.0C)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
```

29. Notice that most of the `useragent` values show up multiple times in the results.

30. Use the `stats values` function to only return one instance of each `useragent.` Add an `as` clause to name the result as `Agents used.`

Results Example:

Agents used
Googlebot/2.1 (http://www.googlebot.com/bot.html)
Googlebot/2.1 (http://www.googlebot.com/bot.html)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET4.0C; .NET4.0E; MS-RTC LM 8; InfoPath.1)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; InfoPath.1; .NET4.0C; .NET4.0E; MS-RTC LM 8)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; MS-RTC LM 8; InfoPath.2)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.2; .NET CLR 1.1.4322; InfoPath.1; MS-RTC LM 8)
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/534.55.3 (KHTML, like Gecko) Version/5.1.5 Safari/534.55.3
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_4) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5

31. This report would be much more useful if we knew the number of times each `useragent` was used. Add a `count` function to the `stats` command that counts the events by `useragent` as `Times used.`

Results Example:

useragent	Agents used	Times used
Opera/9.20 (Windows NT 6.0; U; en)	Opera/9.20 (Windows NT 6.0; U; en)	1557
Opera/9.01 (Windows NT 5.1; U; en)	Opera/9.01 (Windows NT 5.1; U; en)	1890
Mozilla/5.0 (iPad; U; CPU OS 4_3_5 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8L1 Safari/6533.18.5	Mozilla/5.0 (iPad; U; CPU OS 4_3_5 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8L1 Safari/6533.18.5	5300
Mozilla/5.0 (iPad; CPU OS 5_1_1 like Mac OS X) AppleWebKit/534.46 (KHTML, like Gecko) Version/5.1 Mobile/9B206 Safari/7534.48.3	Mozilla/5.0 (iPad; CPU OS 5_1_1 like Mac OS X) AppleWebKit/534.46 (KHTML, like Gecko) Version/5.1 Mobile/9B206 Safari/7534.48.3	5013
Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots)	Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots)	1231
Mozilla/5.0 (compatible; NetcraftSurveyAgent/1.0/cc-prepass-https; +info@netcraft.com)	Mozilla/5.0 (compatible; NetcraftSurveyAgent/1.0/cc-prepass-https; +info@netcraft.com)	1483
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)	11754
Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)	Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)	1575

32. Put the results of `Agents used` and `Times used` into a table.

Results Example:

Agents used	Times used
Googlebot/2.1 (http://www.googlebot.com/bot.html)	1277
Googlebot/2.1 (http://www.googlebot.com/bot.html)	1327
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)	2155
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)	2170
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)	1825

Lab Exercises

Lab typographical conventions:

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

NOTE: Lab work will be done on your personal computer or virtual machine, no lab environment is provided. We suggest you **DO NOT** do the lab work on your production environment.

The lab instructions refer to these source types by the types of data they represent:

Type	Sourcetype	Fields of interest
Web Application	`access_combined_wcookie`	`action, bytes, categoryId, clientip, itemId, JSESSIONID, productId, referer, referer_domain, status, useragent, file`
Database	`db_audit`	`Command, Duration, Type`
Web server	`linux_secure`	`COMMAND, PWD, pid, process`

Lab Module 9 – Transforming Commands with Solutions

NOTE: This lab document has two sections. The first section includes the instructions without answers. The second section includes instructions with the expected search string (answer) in red.

Description

In this lab, you will be using some of the Splunk's transforming commands to derive statistics from the data.

Steps

Scenario: The sales group would like a report of the five best-selling products by productId.

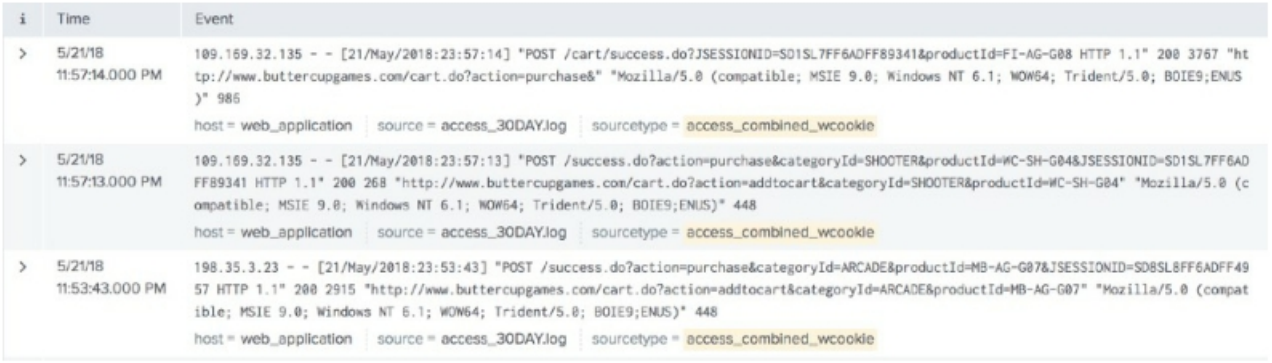
Task 1: Use the top command to get a list of the best-selling products.

1. Navigate to the Search view. (If you are in the **Home** app, click **Search & Reporting** from the column on the left side of the screen. You can also access the Search view by clicking the **Search** menu option on the bar at the top of the screen.)

NOTE: For this course, you will be searching across all time using the main index. This is NOT a best practice in a production environment, but needed for these labs due to the nature of the limited dataset.

2. Enter a search that returns all web application events where an item was successfully purchased. Remember that if the `success.do` file is successfully returned to a user, a purchase was made.
(index=main sourcetype=access_combined_wcookie status=200 file=success.do)

Results Example:



i	Time	Event
>	5/21/18 11:57:14.000 PM	109.169.32.135 - - [21/May/2018:23:57:14] "POST /cart/success.do?JSESSIONID=SD1SL7FF6ADFF89341&productId=FI-AG-G08 HTTP 1.1" 200 3767 "http://www.buttercupgames.com/cart.do?action=purchase&" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 986 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie
>	5/21/18 11:57:13.000 PM	109.169.32.135 - - [21/May/2018:23:57:13] "POST /success.do?action=purchase&categoryId=SHOOTER&productId=WC-SH-G04&JSESSIONID=SD1SL7FF6ADFF89341 HTTP 1.1" 200 268 "http://www.buttercupgames.com/cart.do?action=addtocart&categoryId=SHOOTER&productId=WC-SH-G04" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 448 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie
>	5/21/18 11:53:43.000 PM	198.35.3.23 - - [21/May/2018:23:53:43] "POST /success.do?action=purchase&categoryId=ARCADE&productId=MB-AG-G07&JSESSIONID=SD8SL8FF6ADFF4957 HTTP 1.1" 200 2915 "http://www.buttercupgames.com/cart.do?action=addtocart&categoryId=ARCADE&productId=MB-AG-G07" "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)" 448 host = web_application source = access_30DAY.log sourcetype = access_combined_wcookie

3. Use the `top` command to find the best-selling productIds for all time.

(index=main sourcetype=access_combined_wcookie status=200 file=success.do | top productId)

Results Example:

productId	count	percent
WC-SH-G04	1360	8.426792
DB-SG-G01	1319	8.172749
DC-SG-G02	1308	8.104591
MB-AG-T01	1205	7.466386
MB-AG-G07	1204	7.460190
WC-SH-A02	1192	7.385836
FS-SG-G03	1155	7.156577
WC-SH-A01	1100	6.815788
WC-SH-T02	1076	6.667080
PZ-SG-G05	1012	6.270525

4. Notice that ten rows were returned. You were asked to only return the top 5.
5. Use the `limit` argument to only return the number of rows requested.

(index=main sourcetype=access_combined_wcookie status=200 file=success.do | top productId limit=5)

Results Example:

productId	count	percent
WC-SH-G04	1360	8.426792
DB-SG-G01	1319	8.172749
DC-SG-G02	1308	8.104591
MB-AG-T01	1205	7.466386
MB-AG-G07	1204	7.460190

6. Use the `showperc` option of `top` to remove `percent` from the display.

(index=main sourcetype=access_combined_wcookie status=200 file=success.do | top productId limit=5 showperc=false)

Results Example:

productId	count
WC-SH-G04	1360
DB-SG-G01	1319
DC-SG-G02	1308
MB-AG-T01	1205
MB-AG-G07	1204

7. What is the productId of the best-selling product? Please take note as it might be in the quiz. (WC-SH-G04)

Scenario: There are times when rare events can give you some of the best information. For example rarely accessed files, these can be files that someone internal to the company has made publically available, or a backdoor that few people know about. The security team has asked you to find files that are accessible online, but get very little traffic.

Task 2: Use the rare command to see what files get accessed the least amount in our web application.

8. Enter a search that returns all web application events where a file was successfully served to the user.
(index=main sourcetype=access_combined_wcookie status=200)

9. Use the `rare` command to find the files that show up the least amount of times in our events.
(index=main sourcetype=access_combined_wcookie status=200 | rare file)

Results Example:

file	count	percent
api	1	0.000858
account	2	0.001716
userlist	10	0.008582
passwords.pdf	235	0.201670
error.do	1795	1.540416
success.do	16139	13.850009
oldlink	19642	16.856179
category.screen	19958	17.127361
cart.do	29328	25.168416
product.screen	29417	25.244793

10. Do you see anything that might be of a concern for the security team? Make the report even more granular using the `by` clause to split the `rare` events by the month in which they happened (`date_month`).
(index=main `sourcetype=access_combined_wcookie status=200 | rare file by date_month`)

Scenario: The sales team would like to know the number of items added to carts, versus the number of items that get purchased.

Task 3: Use the count function of the stats command to find out how many items were added to a cart versus being purchased.

11. Enter a search that returns all web application events where an item was successfully added to a cart, or purchased. Remember, when an item is added to the cart the `cart.do` file is served; the `success.do` is served when the item is purchased.
(index=main sourcetype=access_combined_wcookie file=success.do OR file=cart.do status=200)

12. Use the `stats count` function with a `by` clause to count events by the file that was served.

(index=main sourcetype=access_combined_wcookie file=success.do OR file=cart.do status=200 | stats count by file)

Results Example:

file	count
cart.do	29328
success.do	16139

13. Notice that the count column is labeled `count` by default. Use an `as` clause to rename the column to `Transactions`.

(index=main sourcetype=access_combined_wcookie file=success.do OR file=cart.do status=200 | stats count as Transactions by file)

Results Example:

file	Transactions
cart.do	29328
success.do	16139

14. Using the `rename` command, change the name of the `file` field to `Function`.

(index=main sourcetype=access_combined_wcookie file=success.do OR file=cart.do status=200 | stats count as Transactions by file | rename file as Function)

Results Example:

Function	Transactions
cart.do	29328
success.do	16139

Scenario: The marketing manager would like to know how many times users have logged into the web application.

Task 4: Use the distinct count stats function to count how many times sessions were created for users on the system.

15. Use search terms with the `stats dc` function to count all sessions (`JSESSIONID`) that have been used in our web application data.

(index=main sourcetype=access_combined_wcookie | stats dc(JSESSIONID))

Results Example:

dc(JSESSIONID)
11455

16. Use the `as` clause to rename the sessions as `Logins`.

Scenario: While being able to see the number of logins, the manager would like a little more information on who is logging in and asked you to report log-ins by IP.

17. Using the `by` clause, split the `Logins` by `clientip`.

(index=main sourcetype=access_combined_wcookie | stats dc(JSESSIONID) as Logins by clientip)

Results Example:

clientip	Logins
107.3.146.207	186
108.65.113.83	83
109.169.32.135	172
110.138.30.229	82
110.159.208.78	95

18. Use the `sort` command to sort `Logins` so that the `clientip` with the most `Logins` is displayed at the top of the list. Make a note of the top `clientip` you might be asked about it in the quiz.

    (index=main sourcetype=access_combined_wcookie | stats dc(JSESSIONID) as Logins by clientip | sort -Logins)

    (87.194.216.51)

Scenario: The IT team is working on its infrastructure budget for the year. It's trying to get a handle on where most of their bandwidth cost is being spent.

Task 5: Use the stats sum function to find the total bytes used for the web application.

19. Craft search terms that return all events where a file was successfully served to a user.

    (index=main sourcetype=access_combined_wcookie status=200)

20. Create a field named `TotalBytes` by using the `sum` function of the `stats` command.

    (index=main sourcetype=access_combined_wcookie status=200 | stats sum(bytes) as TotalBytes)

Results Example:

TotalBytes
255334688

21. Split the results by the `file` field using the `by` clause.

    (index=main sourcetype=access_combined_wcookie status=200 | stats sum(bytes) as TotalBytes by file)

Results Example:

file	TotalBytes
account	4238
api	1456
cart.do	61311724
category.screen	42250130
error.do	3747647
oldlink	41349801
passwords.pdf	11103985
product.screen	61672339
success.do	33862909
userlist	27690

22. Use the `sort` command to sort the file names into alphabetical order.

    (index=main sourcetype=access_combined_wcookie status=200 | stats sum(bytes) as TotalBytes by file | sort file)

23. What is the name of the file that used the least amount of bandwidth? You might be asked about it in the quiz. (api)

Scenario: EMERGENCY! The website has come to a crawl causing users to complain. An application developer tells you that they pushed an update recently and is concerned that there might be a database query that is causing the issue.

Task 6: Use the stats command's average function to find the average time for each database query being run.

24. Search all database events and use the average (`avg`) function of the `stats` command to get an average `Duration` of all queries.

    (index=main sourcetype=db_audit | stats avg(Duration))

Results Example:

avg(Duration) ⇕
239.4764303178484

25. Use `as` and `by` clauses to rename the average field to `time to complete` and split by the `Command`.

    (index=main sourcetype=db_audit | stats avg(Duration) as "time to complete" by Command)

Results Example:

Command ⇕	time to complete ⇕
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 2208	9988
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 1021	9985
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 4011	9985
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 6186	9980
SELECT * FROM users INNER JOIN creditcards ON users.userid=creditcards.userid INNER JOIN contactinfo ON users.userid=contactinfo.userid INNER JOIN usertracking ON users.userid=usertracking.userid WHERE users.userid = 2429	9971

26. Sort the `time to complete` so that `Command` values that take the longest are shown first.

    (index=main sourcetype=db_audit | stats avg(Duration) as "time to complete" by Command | sort - "time to complete")

27. Notice anything about the `Command` values that are taking the longest to complete?

Scenario: The Web Development team has asked for a list of the browsers that are being used to access the web application.

Task 7: Use the lists and values functions of the stats command to run a report of the browsers users are using to access the web application from.

28. Use the `stats list` function to generate a list of all `useragent` values that have accessed the web application.

    (index=main sourcetype=access_combined_wcookie | stats list(useragent))

Results Example:

```
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.28) Gecko/20120306 YFF3 Firefox/3.6.28 ( .NET CLR 3.5.30729; .NET4.0C)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
```

29. Notice that most of the `useragent` values show up multiple times in the results.

30. Use the `stats values` function to only return one instance of each `useragent`. Add an `as` clause to name the result as `Agents used`.

(index=main sourcetype=access_combined_wcookie | stats values(useragent) as "Agents used")

Results Example:

Agents used
Googlebot/2.1 (http://www.googlebot.com/bot.html)
Googlebot/2.1 (http://www.googlebot.com/bot.html)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET4.0C; .NET4.0E; MS-RTC LM 8; InfoPath.1)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; InfoPath.1; .NET4.0C; .NET4.0E; MS-RTC LM 8)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; MS-RTC LM 8; InfoPath.2)
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.2; .NET CLR 1.1.4322; InfoPath.1; MS-RTC LM 8)
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/534.55.3 (KHTML, like Gecko) Version/5.1.5 Safari/534.55.3
Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_4) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5

31. This report would be much more useful if we knew the number of times each `useragent` was used. Add a `count` function to the `stats` command that counts the events by `useragent` as `Times used`.

(index=main sourcetype=access_combined_wcookie | stats values(useragent) as "Agents used" count as "Times used" by useragent)

Results Example:

useragent	Agents used	Times used
Opera/9.20 (Windows NT 6.0; U; en)	Opera/9.20 (Windows NT 6.0; U; en)	1557
Opera/9.01 (Windows NT 5.1; U; en)	Opera/9.01 (Windows NT 5.1; U; en)	1890
Mozilla/5.0 (iPad; U; CPU OS 4_3_5 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8L1 Safari/6533.18.5	Mozilla/5.0 (iPad; U; CPU OS 4_3_5 like Mac OS X; en-us) AppleWebKit/533.17.9 (KHTML, like Gecko) Version/5.0.2 Mobile/8L1 Safari/6533.18.5	5300
Mozilla/5.0 (iPad; CPU OS 5_1_1 like Mac OS X) AppleWebKit/534.46 (KHTML, like Gecko) Version/5.1 Mobile/9B206 Safari/7534.48.3	Mozilla/5.0 (iPad; CPU OS 5_1_1 like Mac OS X) AppleWebKit/534.46 (KHTML, like Gecko) Version/5.1 Mobile/9B206 Safari/7534.48.3	5013
Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots)	Mozilla/5.0 (compatible; YandexBot/3.0; +http://yandex.com/bots)	1231
Mozilla/5.0 (compatible; NetcraftSurveyAgent/1.0/cc-prepass-https; +info@netcraft.com)	Mozilla/5.0 (compatible; NetcraftSurveyAgent/1.0/cc-prepass-https; +info@netcraft.com)	1483
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)	Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; BOIE9;ENUS)	11754
Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)	Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)	1575

32. Put the results of `Agents used` and `Times used` into a table.

(index=main sourcetype=access_combined_wcookie | stats values(useragent) as "Agents used" count as "Times used" by useragent | table "Agents used", "Times used")

Results Example:

Agents used	Times used
Googlebot/2.1 (http://www.googlebot.com/bot.html)	1277
Googlebot/2.1 (http://www.googlebot.com/bot.html)	1327
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)	2155
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)	2170
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)	1825