Forward-Looking Statements

This presentation may contain forward-looking statements regarding future events, plans or the expected financial performance of our company, including our expectations regarding our products, technology, strategy, customers, markets, acquisitions and investments. These statements reflect management's current expectations, estimates and assumptions based on the information currently available to us. These forward-looking statements are not guarantees of future performance and involve significant risks, uncertainties and other factors that may cause our actual results, performance or achievements to be materially different from results, performance or achievements expressed or implied by the forward-looking statements contained in this presentation.

For additional information about factors that could cause actual results to differ materially from those described in the forward-looking statements made in this presentation, please refer to our periodic reports and other filings with the SEC, including the risk factors identified in our most recent quarterly reports on Form 10-Q and annual reports on Form 10-K, copies of which may be obtained by visiting the Splunk Investor Relations website at www.investors.splunk.com or the SEC's website at www.sec.gov. The forward-looking statements made in this presentation are made as of the time and date of this presentation. If reviewed after the initial presentation, even if made available by us, on our website or otherwise, it may not contain current or accurate information. We disclaim any obligation to update or revise any forward-looking statement based on new information, future events or otherwise, except as required by applicable law.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. We undertake no obligation either to develop the features or functionalities described, in beta or in preview (used interchangeably), or to include any such feature or functionality in a future release.

Detection Technique Deep Dive

SEC1428B

Doug Brown

Senior Threat Hunter | CrowdStrike

splunk> .conf22

May this presentation improve the security of organisations great and small.

Doug Brown “trustedsubject”

Senior Threat Hunter | CrowdStrike

Detection Technique Deep Dive

- This presentation builds upon .conf 2018 SEC1038 Detection Technique Deep Drive (https://www.splunk.com/en_us/resources/videos/detection-technique-deep-dive.html) which I recommend watching after this presentation for completeness.

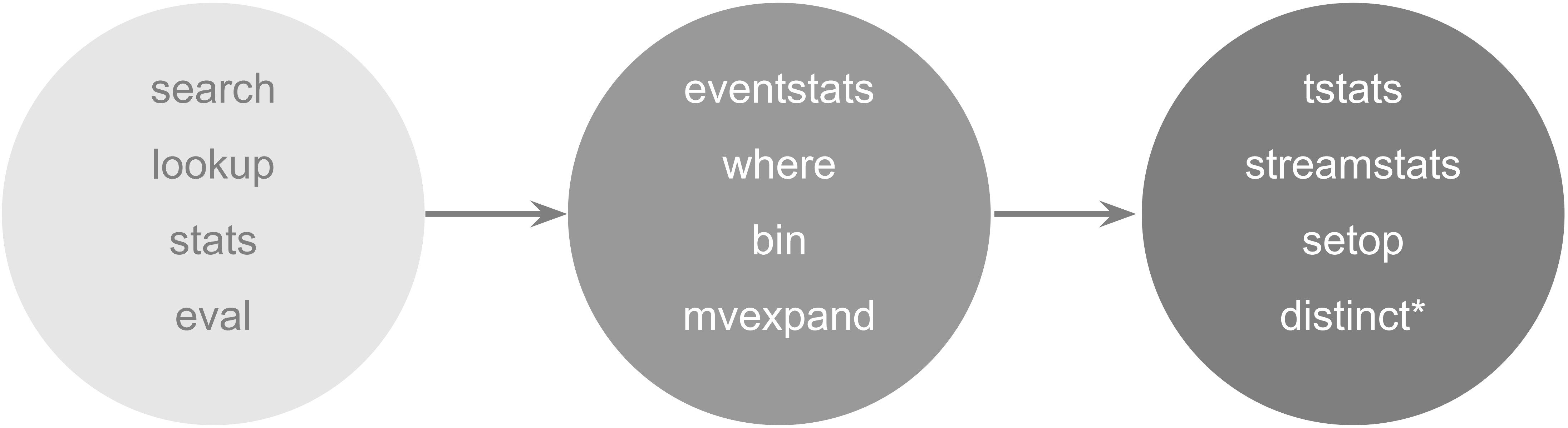
- The case studies you see in this session can be found in the app: https://splunkbase.splunk.com/app/4209

- So we can test our SPL™, every time you run a search using the spike `casestudy` macros, they generate new data that has roughly the same statistical properties.

- We use the Set Operations Technology Add-On (https://splunkbase.splunk.com/app/3516/)

SPL™ for Detection

The only commands you'll ever need



search

lookup

stats

eval

eventstats

where

bin

mvexpand

tstats

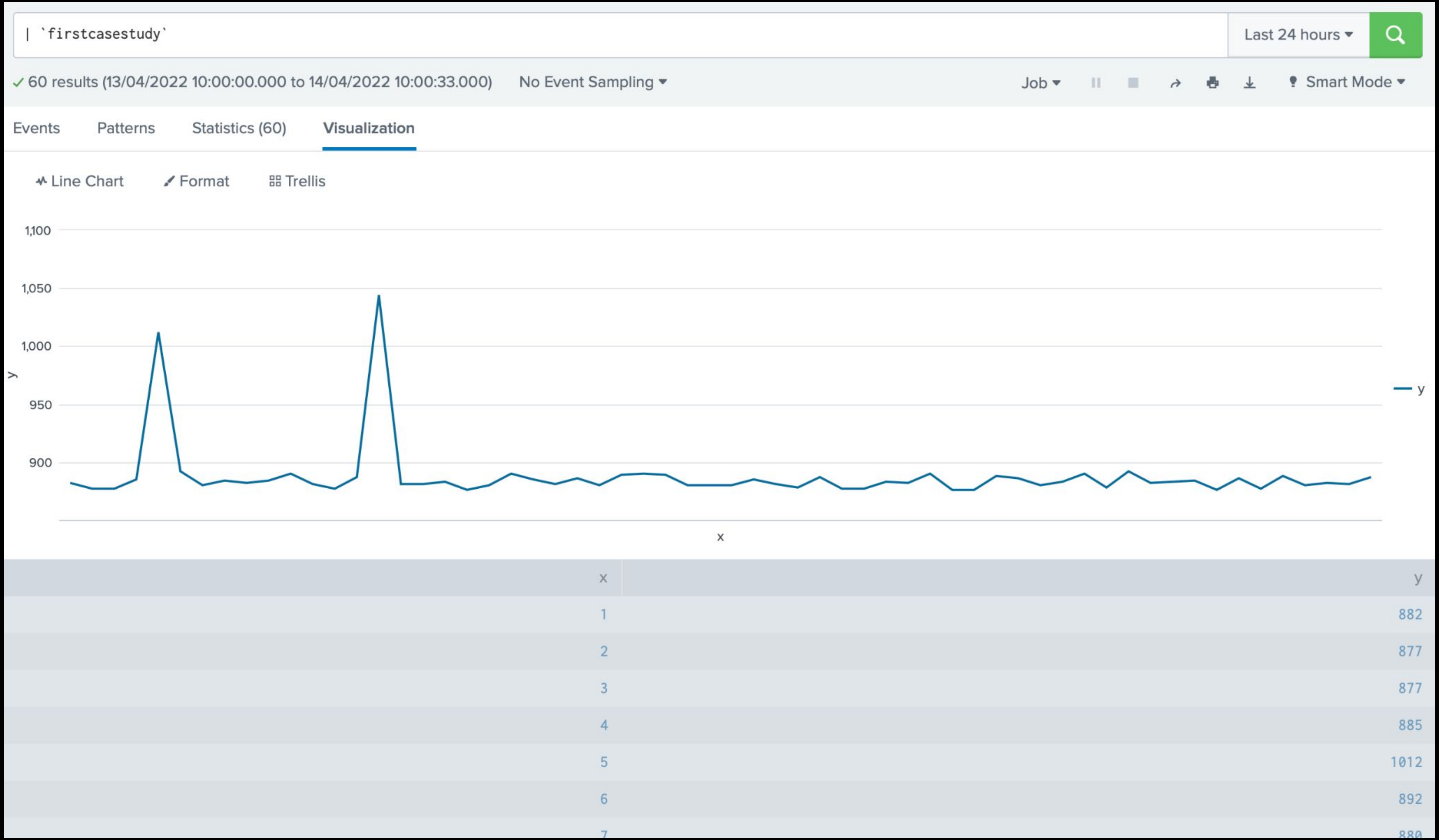
streamstats

setop

distinct*

* Provided by the Set Operations Technology Add-on

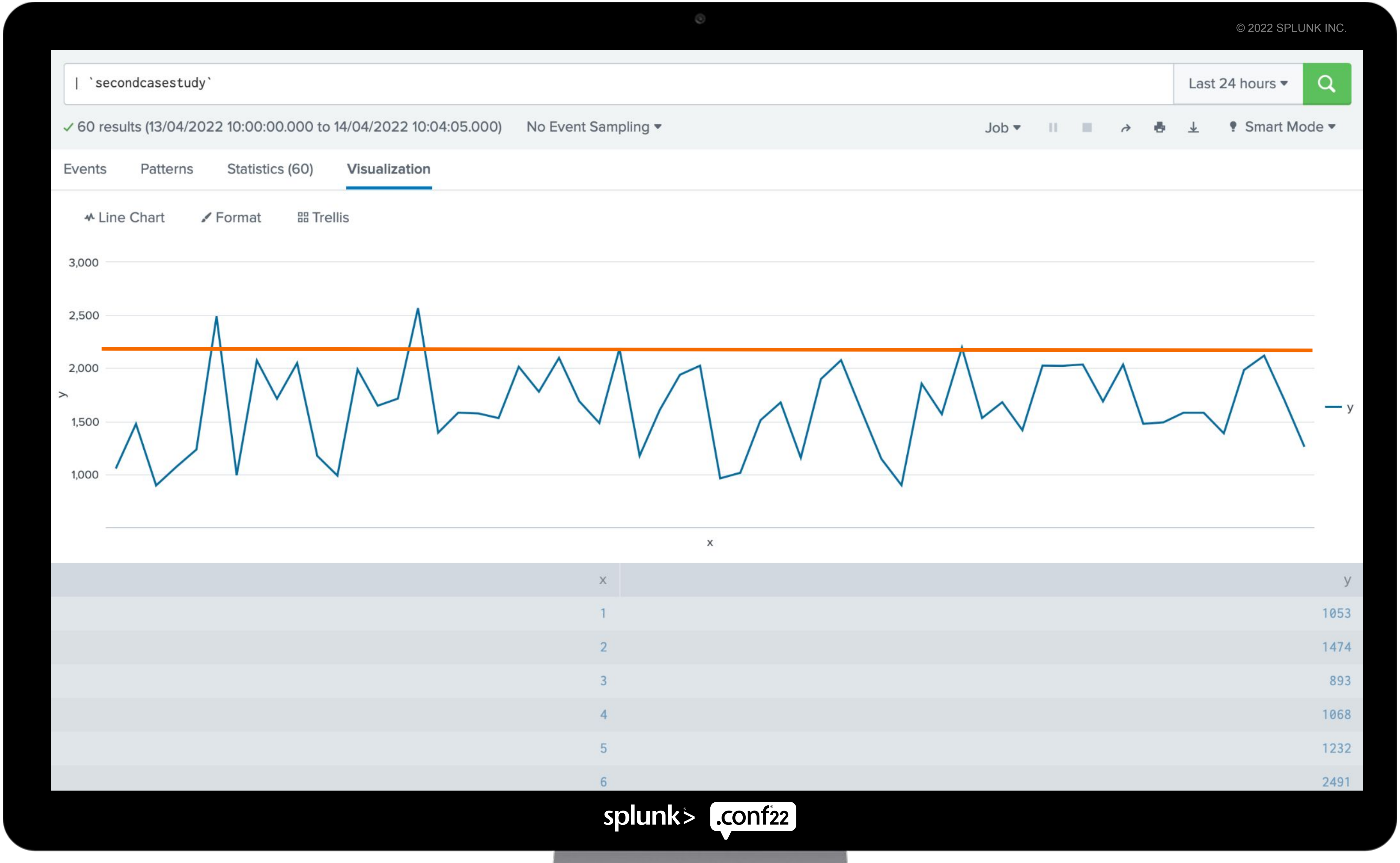
Spikes



```
| `firstcasestudy`
```

✓ 60 results (13/04/2022 10:00:00.000 to 14/04/2022 10:00:33.000)

x	y
1	882
2	877
3	877
4	885
5	1012
6	892
7	880



```
| `secondcasestudy`
```

Last 24 hours ▾

✓ 60 results (13/04/2022 10:00:00.000 to 14/04/2022 10:04:05.000) No Event Sampling ▾

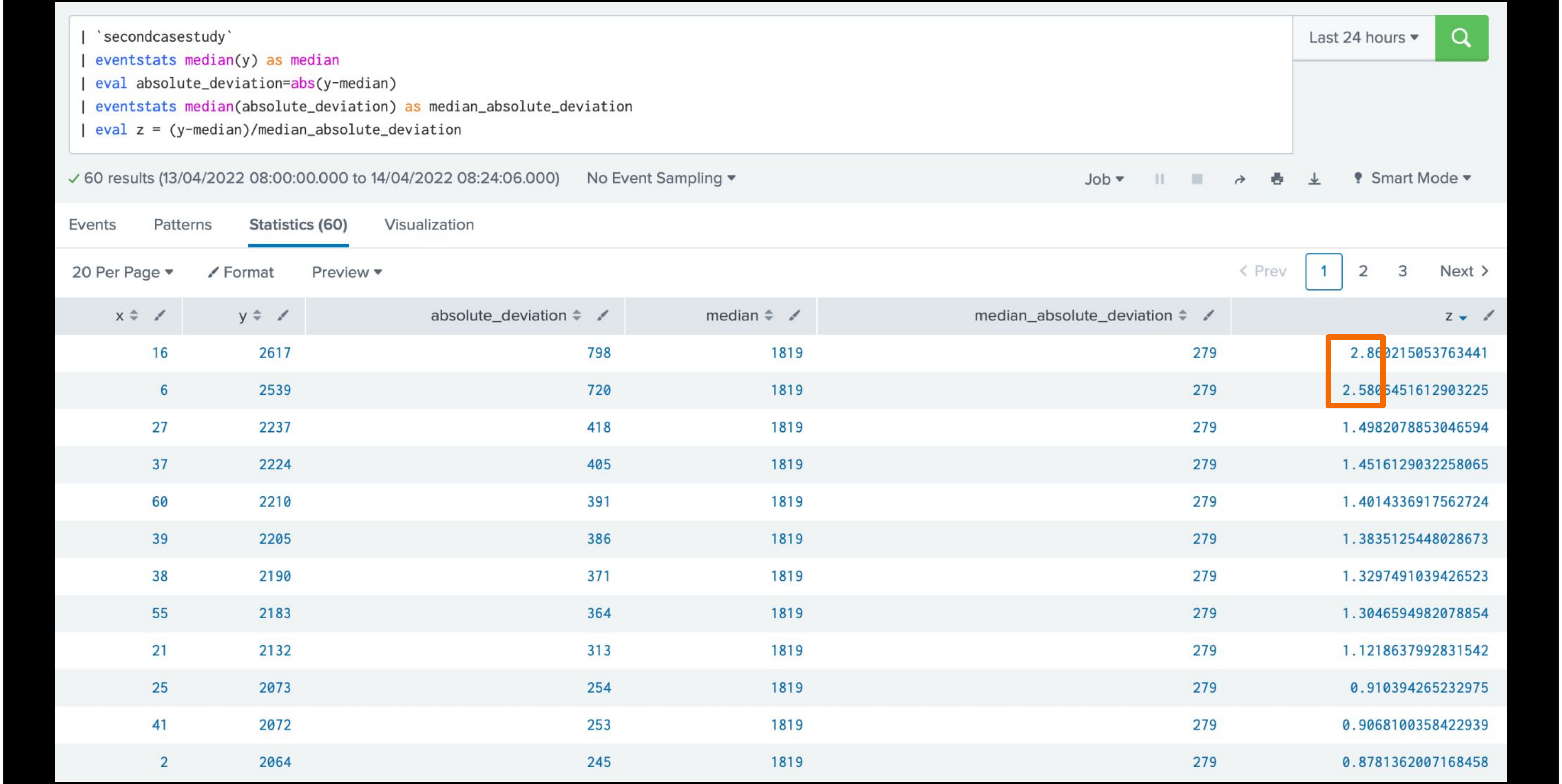
x	y
1	1053
2	1474
3	893
4	1068
5	1232
6	2491

Spikes

Median Absolute Deviation

```
...
eventstats median(y) as median
eval absolute_deviation=abs(y-median)
eventstats median(absolute_deviation) as median_absolute_deviation
where y>median+median_absolute_deviation*#
```

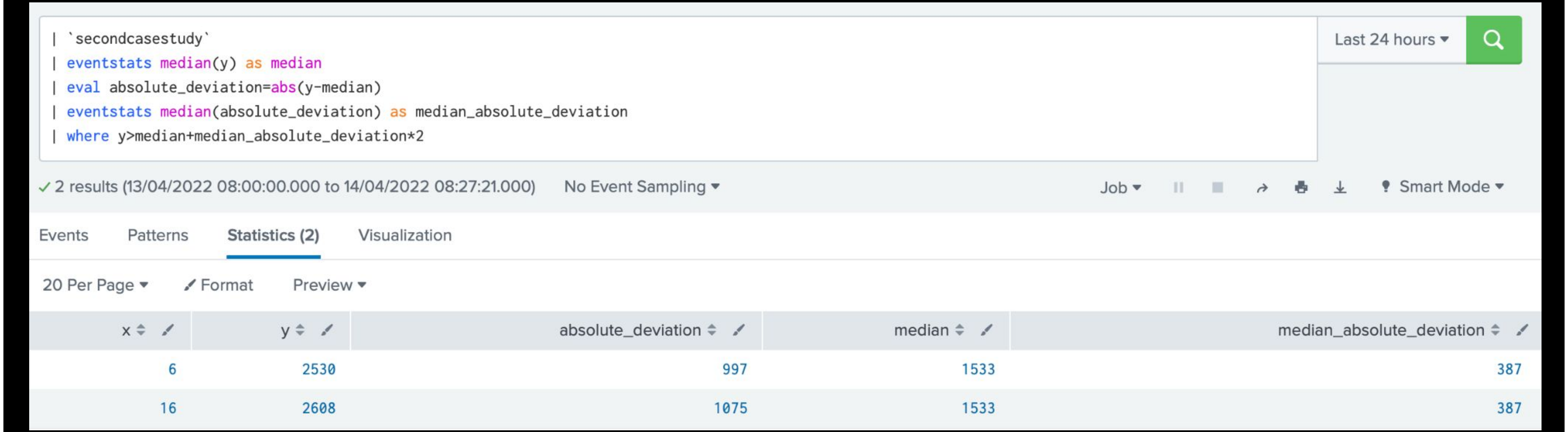
- Let your data guide you to the appropriate # number
- Use fields such as src_ip or user as ‘by’ fields in the eventstats as required



```
| `secondcasestudy`
| eventstats median(y) as median
| eval absolute_deviation=abs(y-median)
| eventstats median(absolute_deviation) as median_absolute_deviation
| eval z = (y-median)/median_absolute_deviation
```

✓ 60 results (13/04/2022 08:00:00.000 to 14/04/2022 08:24:06.000) No Event Sampling ▾

x	y	absolute_deviation	median	median_absolute_deviation	z
16	2617	798	1819	279	2.860215053763441
6	2539	720	1819	279	2.5806451612903225
27	2237	418	1819	279	1.4982078853046594
37	2224	405	1819	279	1.4516129032258065
60	2210	391	1819	279	1.4014336917562724
39	2205	386	1819	279	1.3835125448028673
38	2190	371	1819	279	1.3297491039426523
55	2183	364	1819	279	1.3046594982078854
21	2132	313	1819	279	1.1218637992831542
25	2073	254	1819	279	0.910394265232975
41	2072	253	1819	279	0.9068100358422939
2	2064	245	1819	279	0.8781362007168458



```
| `secondcasestudy`
| eventstats median(y) as median
| eval absolute_deviation=abs(y-median)
| eventstats median(absolute_deviation) as median_absolute_deviation
| where y>median+median_absolute_deviation*2
```

Last 24 hours

✓ 2 results (13/04/2022 08:00:00.000 to 14/04/2022 08:27:21.000)

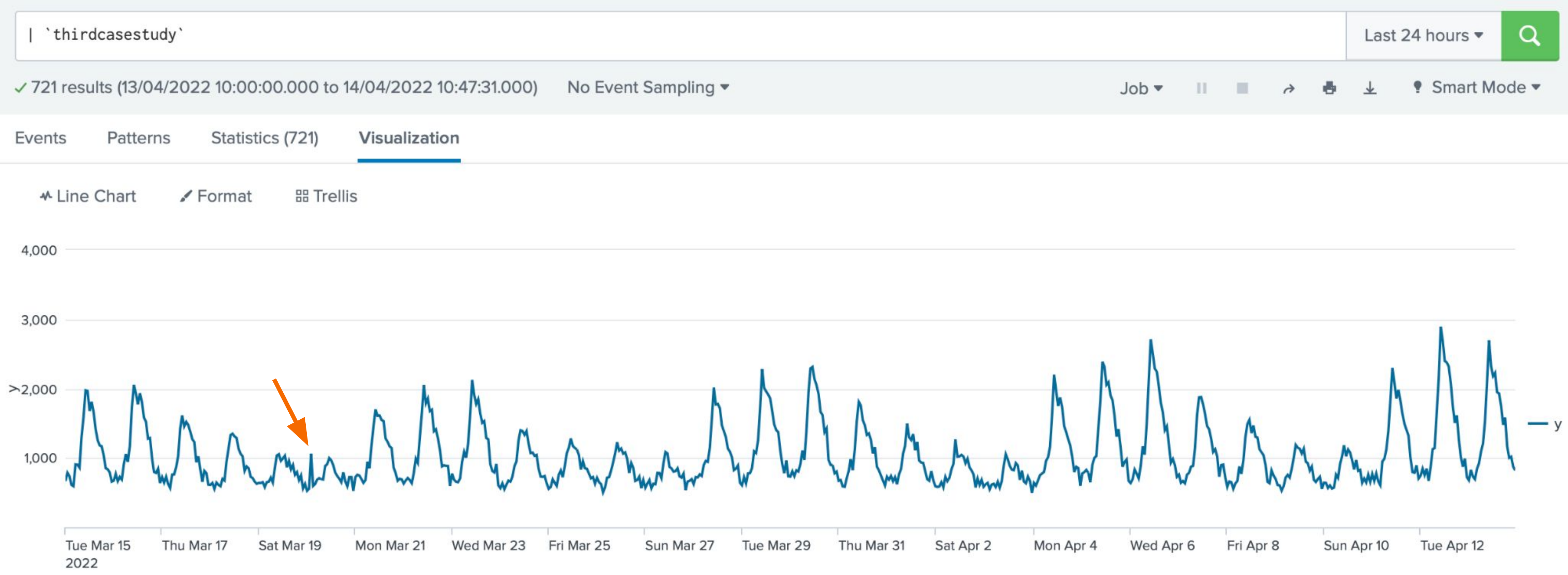
x	y	absolute_deviation	median	median_absolute_deviation
6	2530	997	1533	387
16	2608	1075	1533	387

Spikes

Data with Seasonality

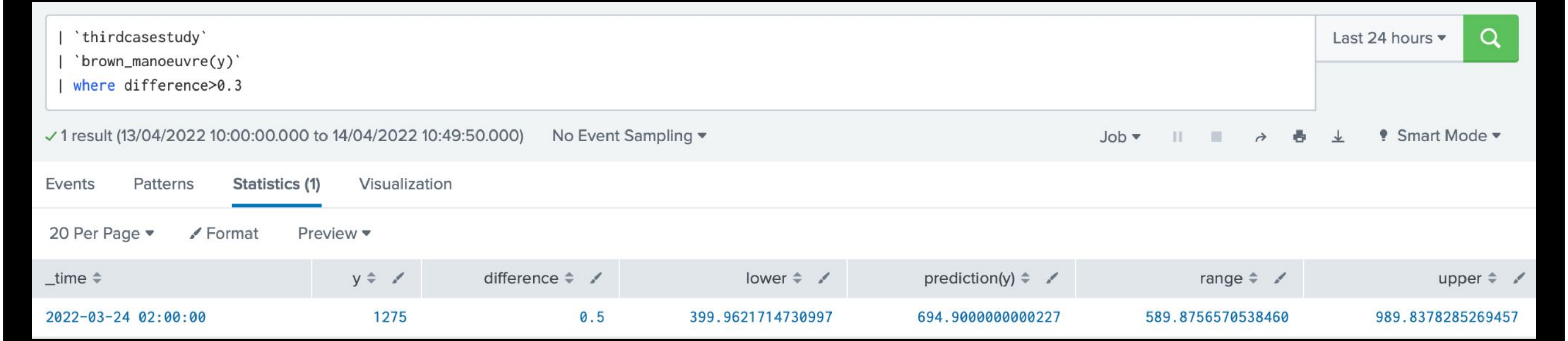
Detecting spikes in data with seasonality is out of this presentation scope, however:

- Consider fields such as date_wday and date_hour as ‘by’ fields to account for seasonality when using statistical methods such as Standard Deviation or Median Absolute Deviation.
- For more complex seasonality, try `brown_manoeuvre(y)` from the .conf 2018 session.



```
| `thirdcasestudy`
```

✓ 721 results (13/04/2022 10:00:00.000 to 14/04/2022 10:47:31.000)



```
| `thirdcasestudy`
| `brown_manoeuvre(y)`
| where difference>0.3
```

✓ 1 result (13/04/2022 10:00:00.000 to 14/04/2022 10:49:50.000) No Event Sampling

_time	y	difference	lower	prediction(y)	range	upper
2022-03-24 02:00:00	1275	0.5	399.9621714730997	694.9000000000227	589.8756570538460	989.8378285269457

First-Time Events

First-Time Events

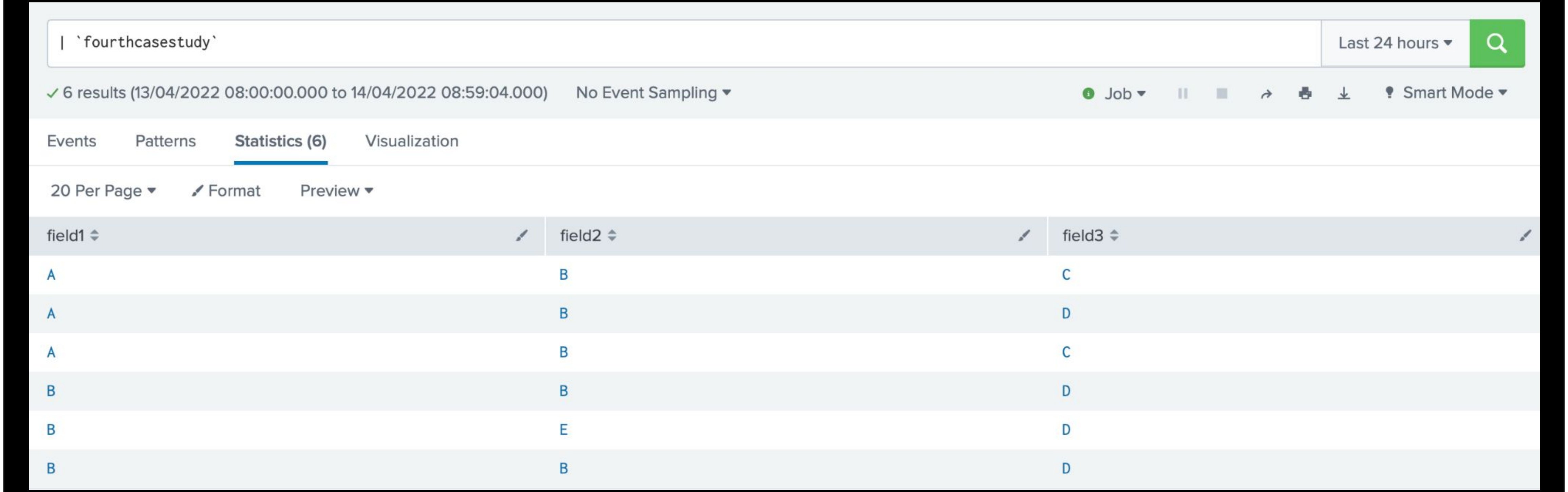
When a new value is seen in a field or combination of fields

```
...
eventstats count by fieldA ... fieldn
where count == 1
```

In practice, we usually use streamstats rather than eventstats and add `_time>relative_time(now(),"-1h")` to the where predicate. Needless to say, use fields such as src_ip and user as 'by' fields.

In some cases, a higher threshold (such as < 3) may be necessary to prevent false-negatives. Consider finding rare rather than unique values using something like:

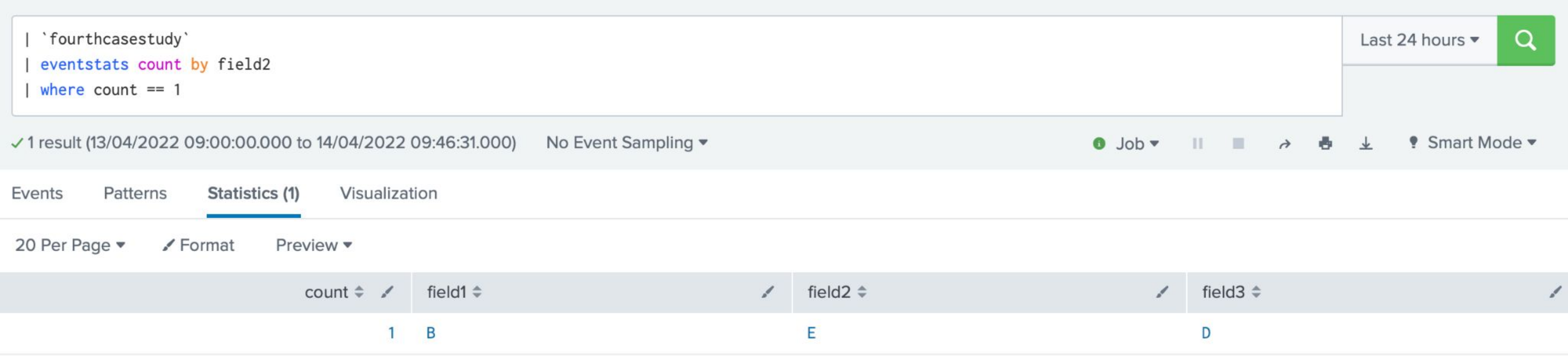
```
...
streamstats count as datapoint
streamstats count by fieldA
where count/datapoint < 0.01 AND _time>relative_time(now(),"-1h")
```



```
| `fourthcasestudy`
```

✓ 6 results (13/04/2022 08:00:00.000 to 14/04/2022 08:59:04.000) No Event Sampling

field1	field2	field3
A	B	C
A	B	D
A	B	C
B	B	D
B	E	D
B	B	D



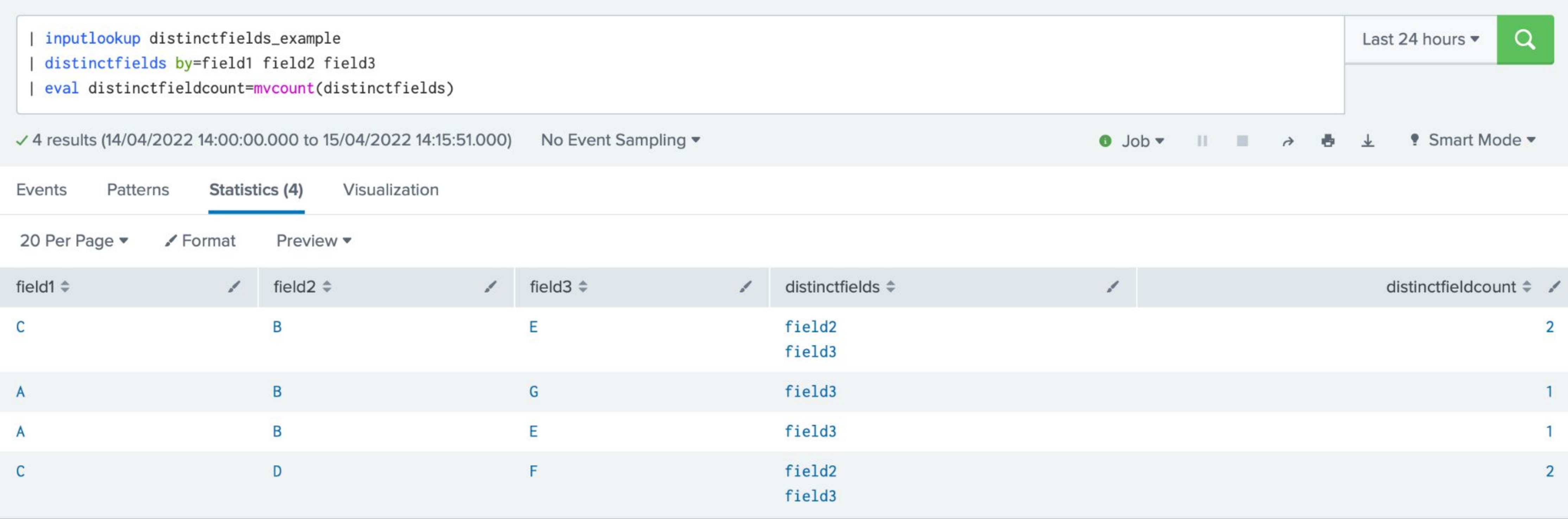
```
| `fourthcasestudy`
| eventstats count by field2
| where count == 1
```

count	field1	field2	field3
1	B	E	D

First-Time Events

Distinct fields

- If you want to determine which *fields* contain unique values, the `distinctfields` and `distinctstream` commands are used.
- The cardinality of the *distinctfields* field can then be used to measure behaviour change.
- We have limited time so will focus more on sequences in this session. For more on the use of distinct fields, please see the .conf 2018 presentation and SetOps documentation: https://github.com/doksu/setops/wiki#usage



```
| inputlookup distinctfields_example
| distinctfields by=field1 field2 field3
| eval distinctfieldcount=mvcount(distinctfields)
```

✓ 4 results (14/04/2022 14:00:00.000 to 15/04/2022 14:15:51.000)

field1	field2	field3	distinctfields	distinctfieldcount
C	B	E	field2 field3	2
A	B	G	field3	1
A	B	E	field3	1
C	D	F	field2 field3	2

Simple Sequences

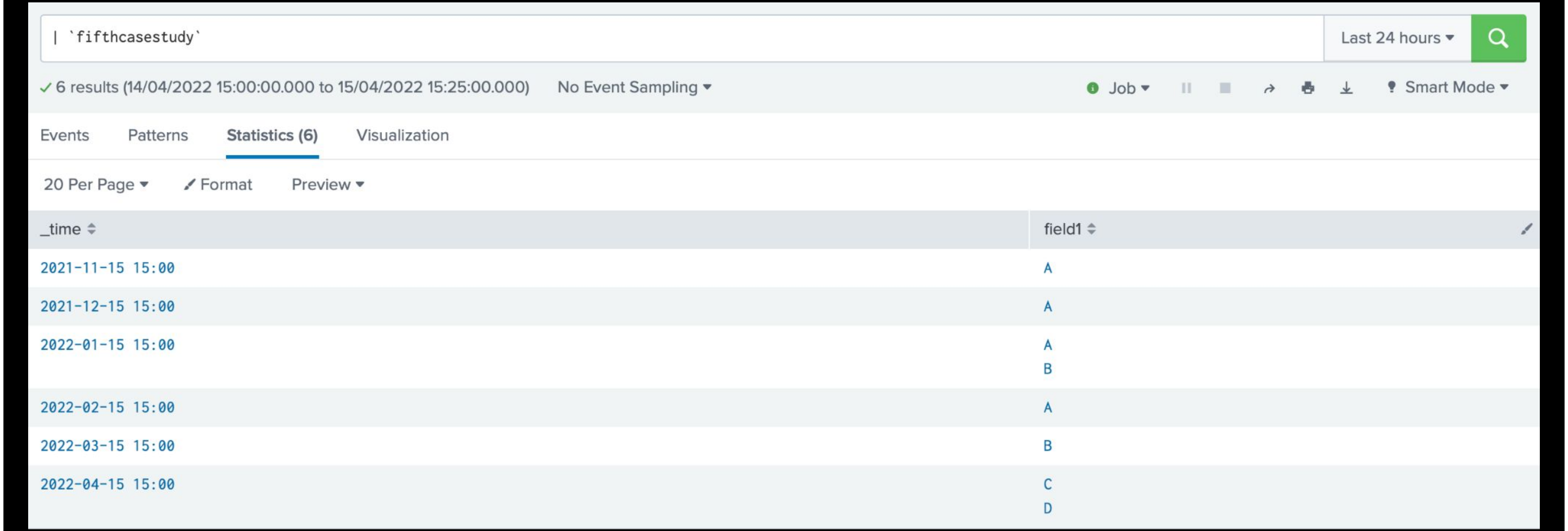
Detecting if “this then that” happens

splunk> .conf22

Simple Sequences

Detecting if “this then that” happens

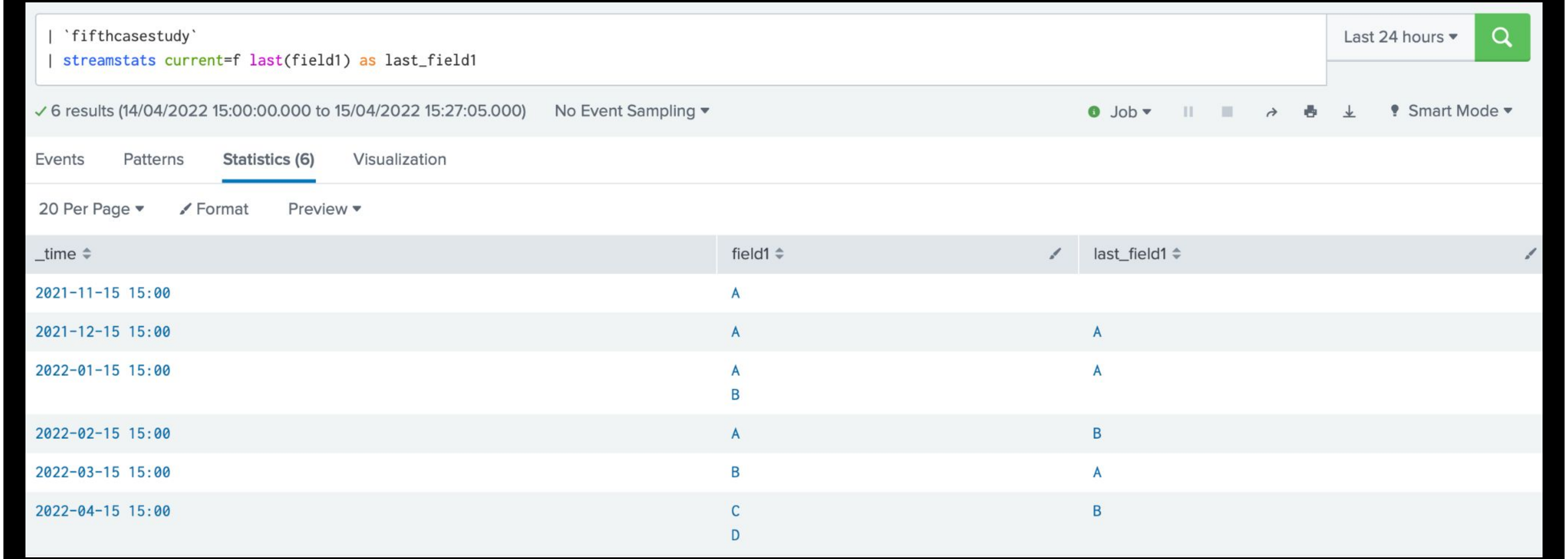
```
...
streamstats current=f last(fieldA) as last_fieldA by ...
where fieldA != last_fieldA AND _time>relative_time(now(),"-1h")
```



```
| `fifthcasestudy`
```

✓ 6 results (14/04/2022 15:00:00.000 to 15/04/2022 15:25:00.000)

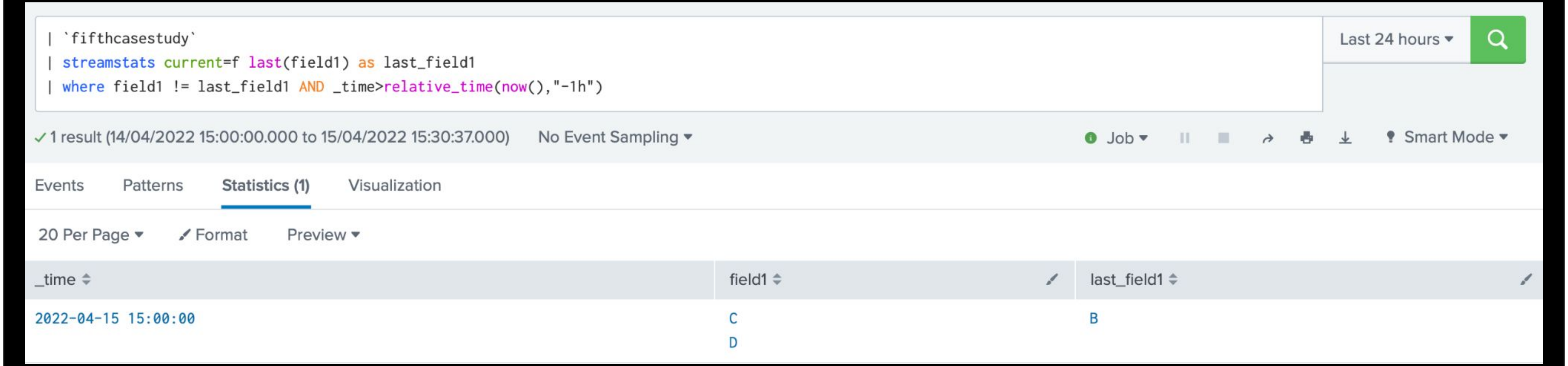
_time	field1
2021-11-15 15:00	A
2021-12-15 15:00	A
2022-01-15 15:00	A B
2022-02-15 15:00	A
2022-03-15 15:00	B
2022-04-15 15:00	C D



```
| `fifthcasestudy`
| streamstats current=f last(field1) as last_field1
```

✓ 6 results (14/04/2022 15:00:00.000 to 15/04/2022 15:27:05.000)

_time	field1	last_field1
2021-11-15 15:00	A	
2021-12-15 15:00	A	A
2022-01-15 15:00	A B	A
2022-02-15 15:00	A	B
2022-03-15 15:00	B	A
2022-04-15 15:00	C D	B



```
| `fifthcasestudy`
| streamstats current=f last(field1) as last_field1
| where field1 != last_field1 AND _time>relative_time(now(),"-1h")
```

_time	field1	last_field1
2022-04-15 15:00:00	C D	B

splunk> .conf22

T1071.004 DNS Command and Control

Possible C2 Beacon Detected Through Domain Parking

```
| tstats `summariesonly` values("DNS.answer") as answer FROM
  datamodel=Network_Resolution WHERE (nodename=DNS AND DNS.src="10.*" AND
  DNS.query!="*.in-addr.arpa") BY _time "DNS.src", "DNS.query" span=5m

| `drop_dm_object_name("DNS")`
| mvexpand answer
| where match(answer,"^\d+\.\d+\.\d+\.\d+$")
| streamstats current=f window=1 global=f earliest(_time) as previous_time,
  last(answer) as previous_answer by query

| where answer!=previous_answer
  AND match(answer,"^127\.") XOR match(previous_answer,"^127\.")
| eval minutes_difference=round((_time-previous_time)/60)
| lookup dnslookup clientip AS src OUTPUT clienthost AS src_host
```

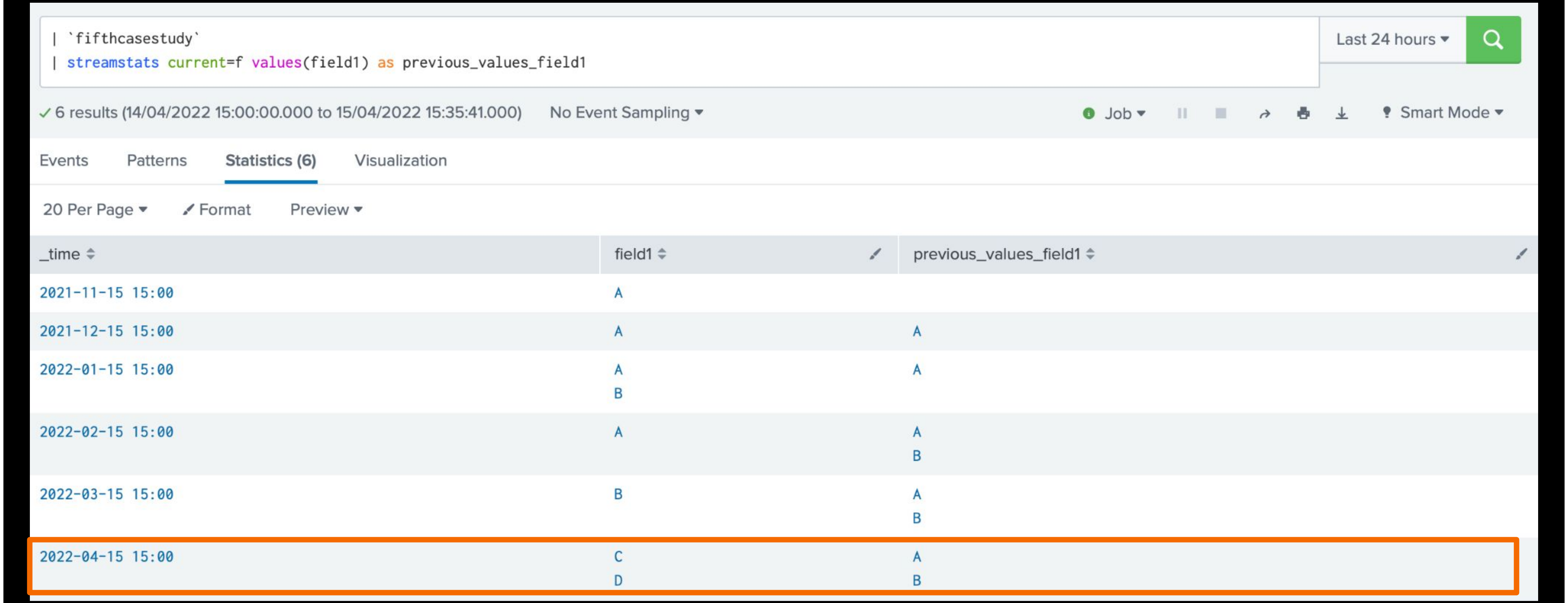
Simple Sequences

Using Set Operations

Simple Sequences

Using Set Operations

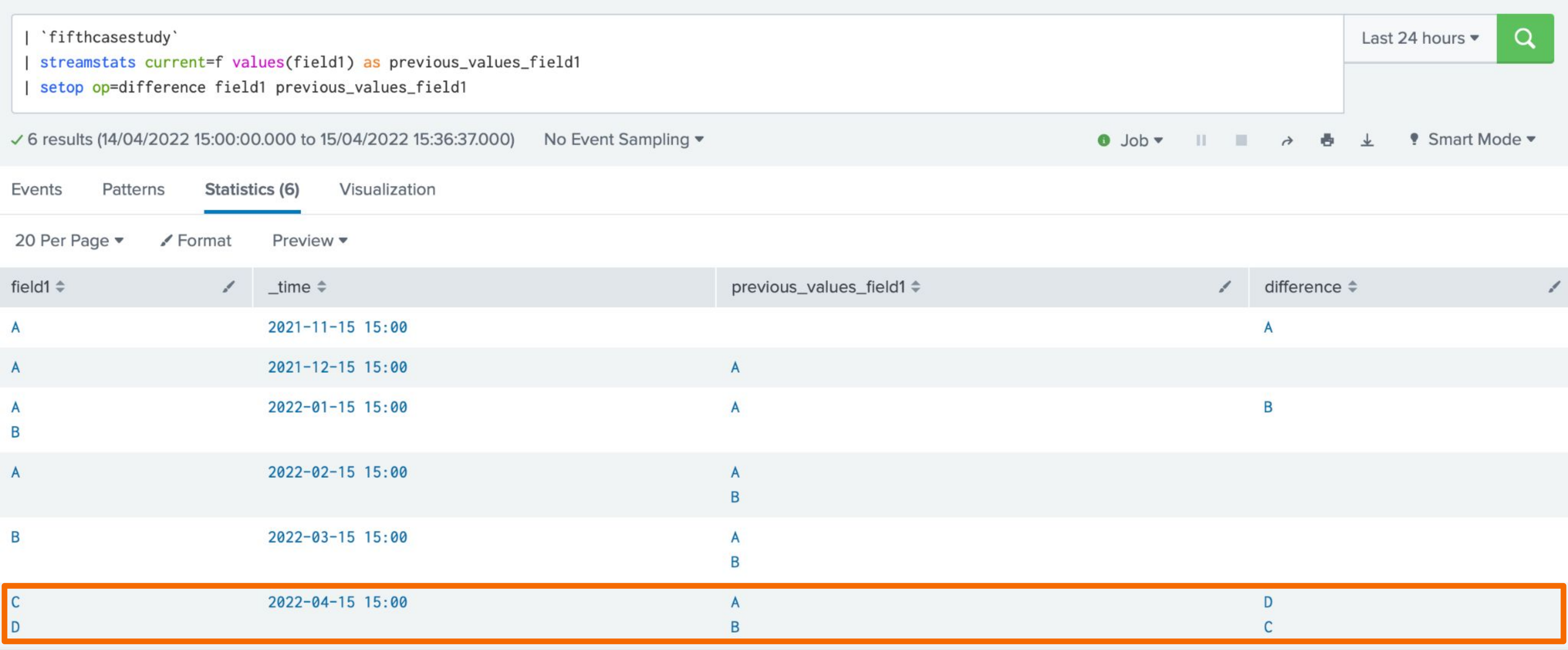
```
...
streamstats current=f values(fieldA) as previous_values_fieldA by ...
setop op=difference fieldA previous_values_fieldA
where mvcount(difference) > 1 AND _time>relative_time(now(),"-1h")
```



```
| `fifthcasestudy`
| streamstats current=f values(field1) as previous_values_field1
```

✓ 6 results (14/04/2022 15:00:00.000 to 15/04/2022 15:35:41.000)

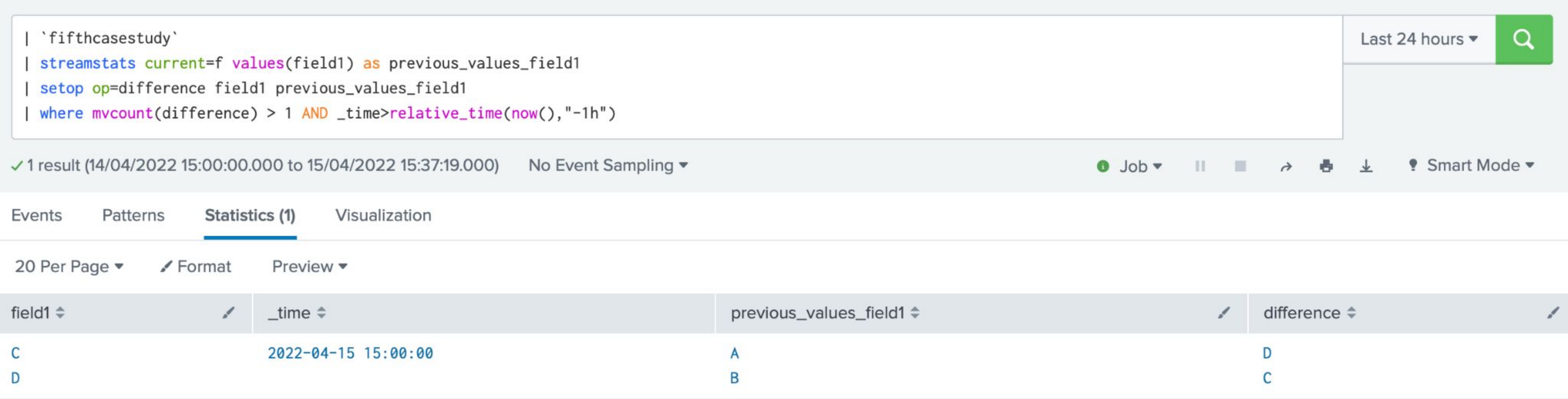
_time	field1	previous_values_field1
2021-11-15 15:00	A	
2021-12-15 15:00	A	A
2022-01-15 15:00	A B	A
2022-02-15 15:00	A	A B
2022-03-15 15:00	B	A B
2022-04-15 15:00	C D	A B



```
| `fifthcasestudy`
| streamstats current=f values(field1) as previous_values_field1
| setop op=difference field1 previous_values_field1
```

✓ 6 results (14/04/2022 15:00:00.000 to 15/04/2022 15:36:37.000) No Event Sampling

field1	_time	previous_values_field1	difference
A	2021-11-15 15:00		A
A	2021-12-15 15:00	A	
A B	2022-01-15 15:00	A	B
A	2022-02-15 15:00	A B	
B	2022-03-15 15:00	A B	
C D	2022-04-15 15:00	A B	D C



```
| `fifthcasestudy`
| streamstats current=f values(field1) as previous_values_field1
| setop op=difference field1 previous_values_field1
| where mvcount(difference) > 1 AND _time>relative_time(now(),"-1h")
```

✓ 1 result (14/04/2022 15:00:00.000 to 15/04/2022 15:37:19.000)

No Event Sampling

field1	_time	previous_values_field1	difference
C D	2022-04-15 15:00:00	A B	D C

T1021.001 Lateral Movement with RDP

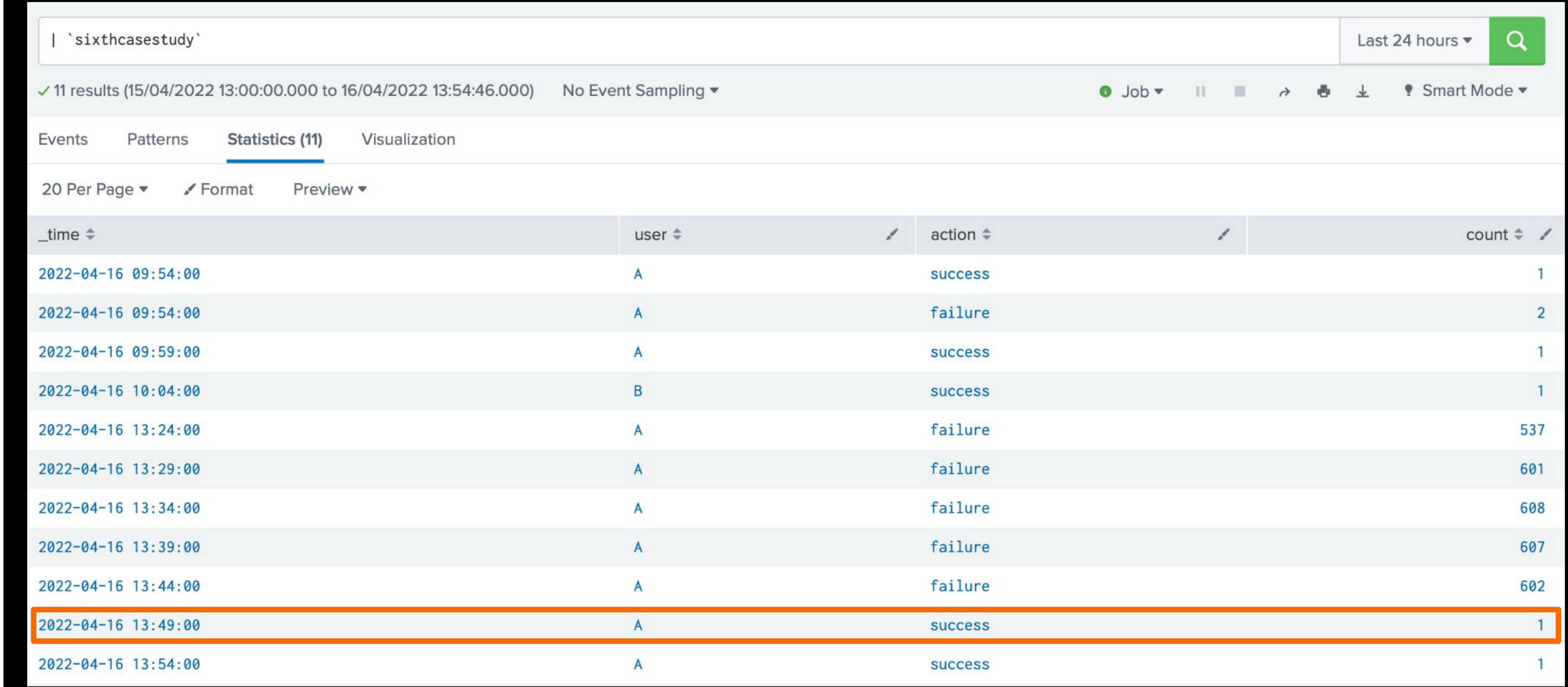
Multiple Servers RDPed to for First Time by User

```
source="WinEventLog:Security" EventCode=4624 Logon_Type=10

| bin _time span=1d
| eval user=lower(user), ComputerName=lower(ComputerName)
| stats values(ComputerName) as dest_host, values(src_ip) as src_ip
   by _time, user

| streamstats current=f values(dest_host) as previous_dest_host by user
| setop op=difference dest_host previous_dest_host
| eval difference_count=mvcount(difference)

| where difference_count>1 AND _time>relative_time(now(),"-24h")
| eval risk_object=user, risk_object_type="user",
   risk_score=difference_count*20
```

Complex Sequences

An example of what can be achieved in SPL™

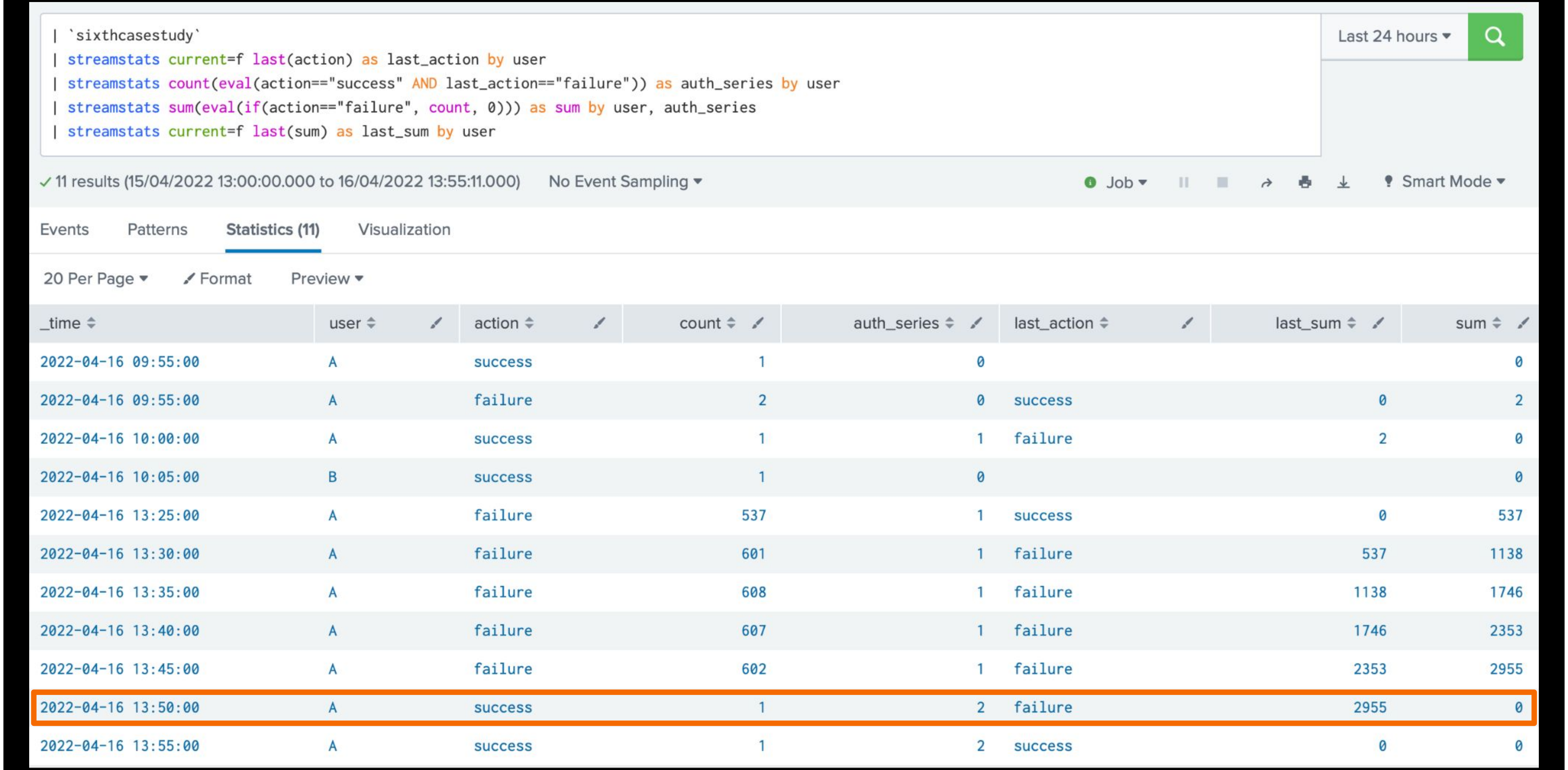
splunk> .conf22



```
| `sixthcasestudy`
```

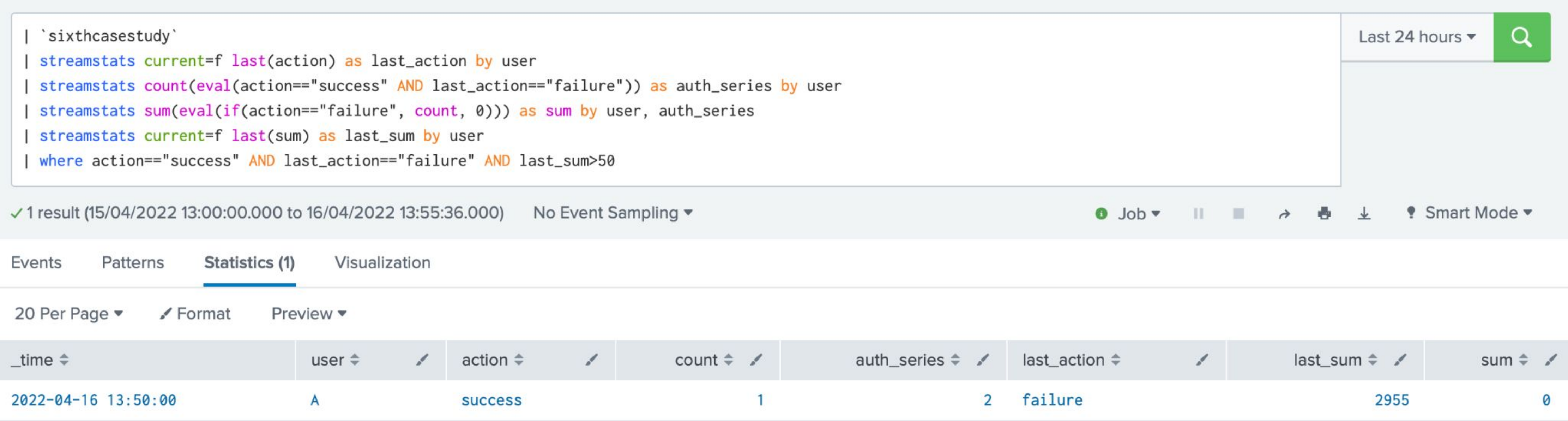
✓ 11 results (15/04/2022 13:00:00.000 to 16/04/2022 13:54:46.000)

_time	user	action	count
2022-04-16 09:54:00	A	success	1
2022-04-16 09:54:00	A	failure	2
2022-04-16 09:59:00	A	success	1
2022-04-16 10:04:00	B	success	1
2022-04-16 13:24:00	A	failure	537
2022-04-16 13:29:00	A	failure	601
2022-04-16 13:34:00	A	failure	608
2022-04-16 13:39:00	A	failure	607
2022-04-16 13:44:00	A	failure	602
2022-04-16 13:49:00	A	success	1
2022-04-16 13:54:00	A	success	1



```
| `sixthcasestudy`
| streamstats current=f last(action) as last_action by user
| streamstats count(eval(action=="success" AND last_action=="failure")) as auth_series by user
| streamstats sum(eval(if(action=="failure", count, 0))) as sum by user, auth_series
| streamstats current=f last(sum) as last_sum by user
```

_time	user	action	count	auth_series	last_action	last_sum	sum
2022-04-16 09:55:00	A	success	1	0			0
2022-04-16 09:55:00	A	failure	2	0	success	0	2
2022-04-16 10:00:00	A	success	1	1	failure	2	0
2022-04-16 10:05:00	B	success	1	0			0
2022-04-16 13:25:00	A	failure	537	1	success	0	537
2022-04-16 13:30:00	A	failure	601	1	failure	537	1138
2022-04-16 13:35:00	A	failure	608	1	failure	1138	1746
2022-04-16 13:40:00	A	failure	607	1	failure	1746	2353
2022-04-16 13:45:00	A	failure	602	1	failure	2353	2955
2022-04-16 13:50:00	A	success	1	2	failure	2955	0
2022-04-16 13:55:00	A	success	1	2	success	0	0



```
| `sixthcasestudy`
| streamstats current=f last(action) as last_action by user
| streamstats count(eval(action=="success" AND last_action=="failure")) as auth_series by user
| streamstats sum(eval(if(action=="failure", count, 0))) as sum by user, auth_series
| streamstats current=f last(sum) as last_sum by user
| where action=="success" AND last_action=="failure" AND last_sum>50
```

Last 24 hours

✓ 1 result (15/04/2022 13:00:00.000 to 16/04/2022 13:55:36.000)

No Event Sampling

_time	user	action	count	auth_series	last_action	last_sum	sum
2022-04-16 13:50:00	A	success	1	2	failure	2955	0

T1110 Brute Force Attack

Successful Authentication by User After High Number of Failures

```
| tstats `summariesonly` count FROM datamodel=Authentication WHERE
  nodename=Authentication AND Authentication.action IN
  ("success","failure") NOT Authentication.user IN ("unknown", "root")
  BY _time, "Authentication.user", "Authentication.action" span=1s

| `drop_dm_object_name("Authentication")`
| streamstats current=f last(action) as last_action by user
| streamstats count(eval(action=="success" AND last_action=="failure"))
  as auth_series by user
| streamstats sum(eval(if(action=="failure", count, 0))) as sum
  by user, auth_series
| streamstats current=f last(sum) as last_sum by user

| where action=="success" AND last_action=="failure" AND last_sum>50
| eval risk_object=user, risk_object_type="user",
  risk_score=round(last_sum/100)*25
```

Highly Complex Sequences

Highly Complex Sequences

- The need to detect highly complex sequences is as rare as it is complex.
- State machines can be used to detect if a complex sequence of events has occurred or if a certain depth in a state machine has been reached.
- State machines are outside the scope of this presentation but if you’re interested, please see: https://www.youtube.com/watch?v=5ToTZYm5bjw

Thank You

splunk> .conf22