4. Dashboard Cookbook - Advanced Techniques

2023-07-07 10:59:21 AM to 2023-07-07 11:59:21 AM

errorCode Client.RequestLimitExceeded keywords * cat_lookup dev

Welcome to the dashboards demo for dashboards!

This is one of a series of demonstration dashboards see:

1. Basics
2. Time Series
3. Advanced Analytics
4. Advanced Techniques

Checkout these micro learning videos:

- Create a Dashboard - Create a Simple Dashboard - Customize a Dashboard - Share a Dashboard Inside Your Organization

This is a set of dashboards that demonstrate how to write and format search output, and match that to various panel types.

Advanced Techniques

This is a selection of more advanced dashboard techniques and panel types.

Template Variables

Add one or more a template variables to the dashboard to maximise flexibility. The variables appear in the UI as a list if you have derived list values but a user can enter any value in the parameter and press enter. You can also set default values such as here defaults are set for these variables.

For example:

- a parameter for env in sourcecategory
- a service
- keywords or customer name

Add the variable in the search syntax. At search time it's replaced with value.

```
_sourcecategory={{variable}}/foo/bar {{variable}}
| where field matches /{{variable}}/
```

You can also use {{variables}} in the title of panels , inside text panels, or in other config fields for panels such as the series alias field in overrides. Right now the errorCode variable for this dashboard is: 'Client.RequestLimitExceeded'.

More info: template vars - moustache format

Changing a variable type changes the query string so it's possible to direct link into a dashboard in sumo with specific parameters. For example: https://service.sumologic.com/ui/#/dashboardv2/?variables=errorCode:AccessDenied;keywords:s3

You can use lookup tables as the source of a list using cat command see the cat and lookup panel example below

Table for variables: errorcode=Client.RequestLimitExceeded and keywords=*

	errorCode	eventname	eventsource	_count
1	Client.RequestLimitExceeded	DescribeReservedInstancesListings	ec2.amazonaws.com	5
2	Client.RequestLimitExceeded	DescribeVpcs	ec2.amazonaws.com	1
3	Client.RequestLimitExceeded	RevokeSecurityGroupIngress	ec2.amazonaws.com	1
4	Client.RequestLimitExceeded	AuthorizeSecurityGroupIngress	ec2.amazonaws.com	1

Tip: Linked Dashboards

You can link dashboards together to quickly view related data. Each panel can have links to other dashboards. When you select a data point on the panel you'll have an option to click on linked dashboards. This allows you to quickly reference other related dashboards to investigate.

More info: https://help.sumologic.com/Visualizations-and-Alerts/Dashboard_(New)/Link_Dashboard_(New)

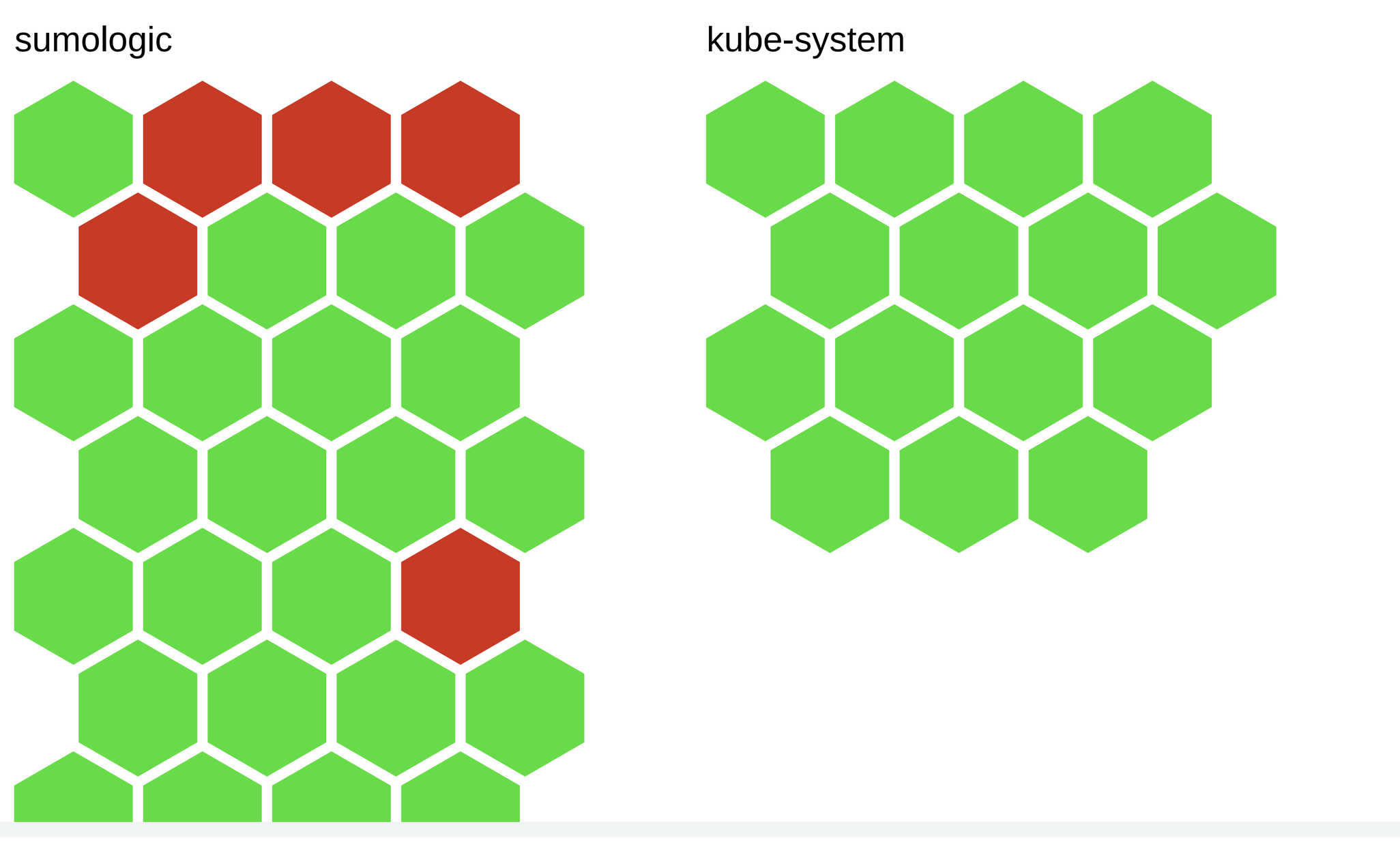
Tip: Entity Explorer

Sumo Logic observability solutions create an entity heirarchy. You can click on a node in one of these node views to open the entity explorer. From there you can drill down to further content like relevant logs or metrics.

More info: https://help.sumologic.com/Visualizations-and-Alerts/Dashboard_(New)/Drill_down_to_discover_root_causes

Pods Running - Click on a Pod To Open Entity Explorer

2023-07-07 11:28:14 AM to 2023-07-07 11:43:14 AM



Scheduled Views To Accelerate Reporting

Scheduled views are a key administrative /power user tool for speeding up load time for a known search use case. For slow aggregate reporting dashboarding over days/weeks this it is one of the most powerful acceleration tools. For large scale aggregation use cases where you want to speed up query load create a scheduled view and use this view to drive the dashboard panels. This can be 100x or more faster than running the query every time vs raw data. Common use cases for views:

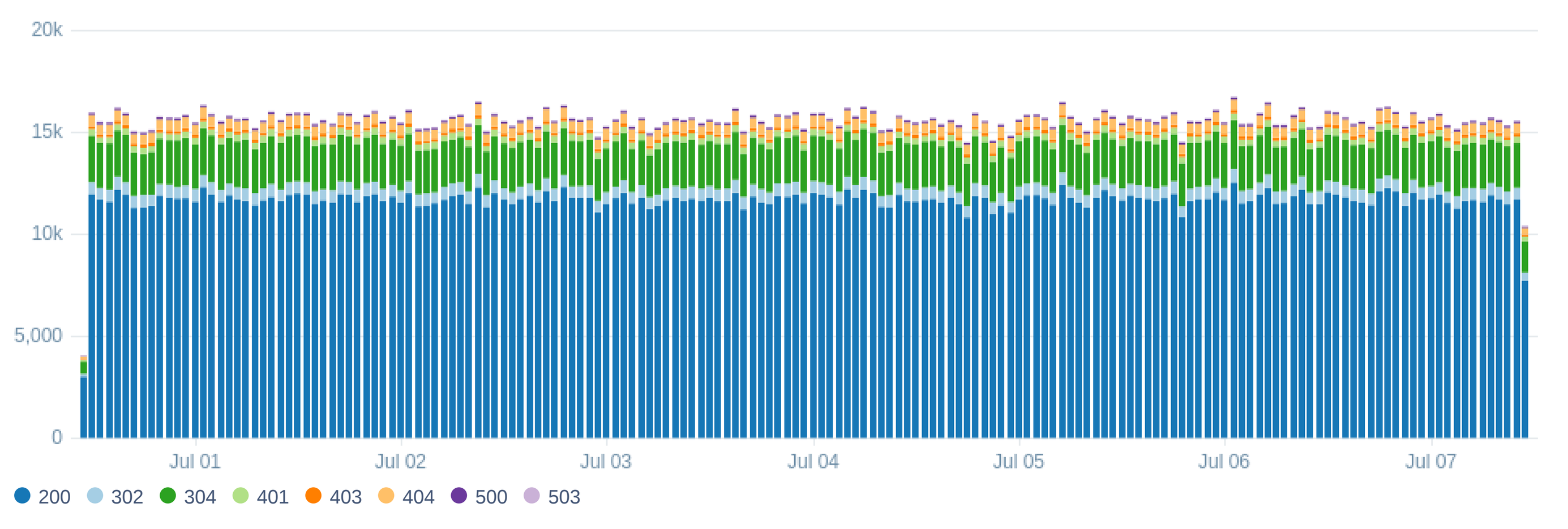
- aggregates by statuscode, uri, node for a web service
- pre computing geopip or threatip lookups and storing historically accurate lookups in the view
- pre computing data volume information (because the data volume parse regex multi is very i/o intensive)

```
_view=status_count_apache_access
| timeslice 1h
| sum(_count) as requests by status_code,_timeslice
| transpose row _timeslice column status_code
```

Scheduled views can be defined Manage Data / Logs / Scheduled Views or by using 'save to index' as the output of a scheduled search. Advanced users might even have views that summarized views into such as one saved search per hour and another per day.

Scheduled View Provides Fast Panel Load Over Long Times or Large Data Sets

2023-06-30 11:43:16 AM to 2023-07-07 11:43:16 AM



Tip: Overrides

Features: we can use the overrides panel to have multiple axes or special per series formatting. Common override use cases:

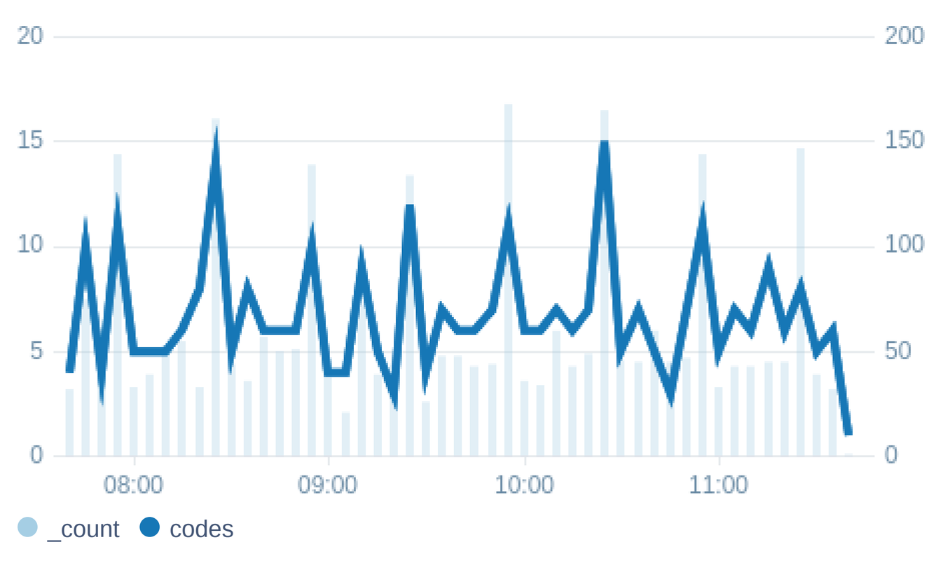
- left/right axis
- custom line color / types
- alias for series eg. #a renamed to {{loadbalancer}}

Tip: Time Ranges Per Panel

The time range of a panel can 'inherit' that global setting for the dashboard. You can also set a separate time range in the individual panel as this one does ----->

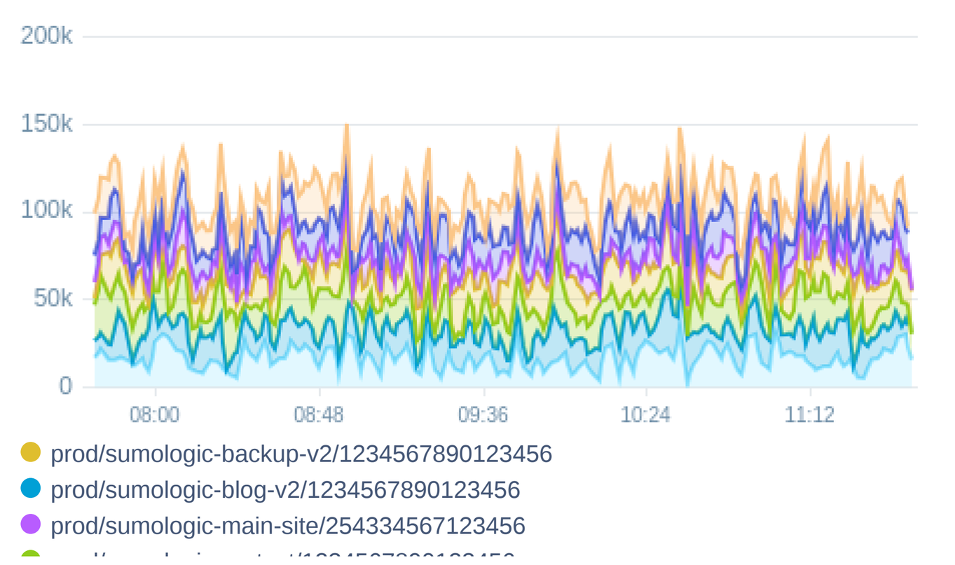
time series two aggregates and two axis L R

2023-07-07 7:43:15 AM to 2023-07-07 11:43:15 AM



Requests Served by ALB

2023-07-07 7:43:14 AM to 2023-07-07 11:43:14 AM



Logs and Metrics on Same Chart

We can put both metrics and logs series on the same chart. #A is logs, #B is metrics

here we have a logs query counting errors in instance logs, and logs showing avg cpu and logs

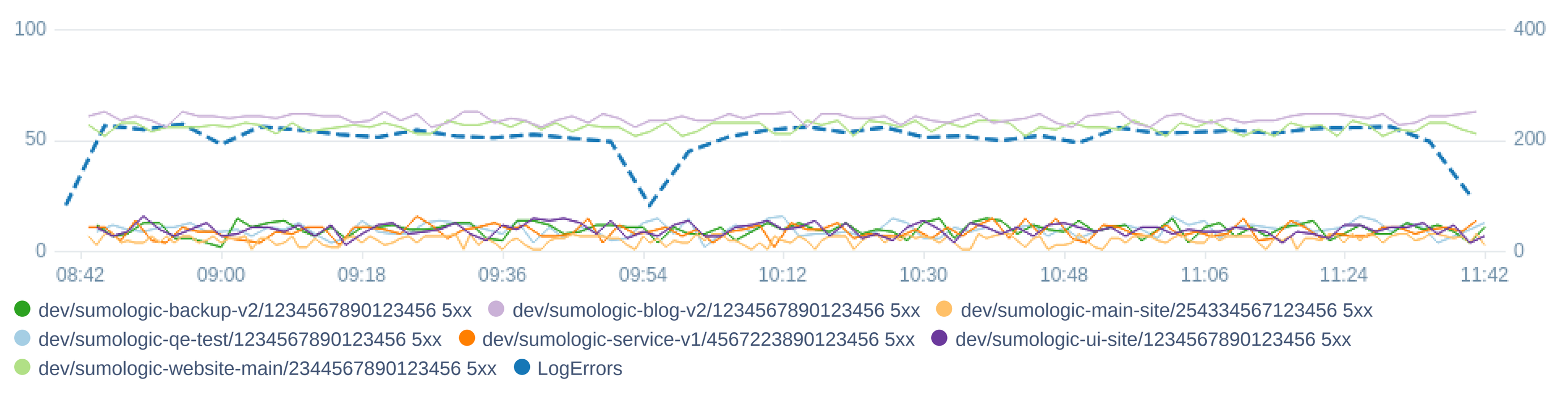
```
_sourcecategory=labs/apache/error error | timeslice
| count as LogErrors by _timeslice
```

metrics:

```
Namespace=AWS/ApplicationELB (metric=HTTPCode_ELB_5XX_Count)
| sum by loadbalancer
```

Logs and metrics chart

2023-07-07 8:43:15 AM to 2023-07-07 11:43:15 AM



100
50
0
400
200
0
08:42
09:00
09:18
09:36
09:54
10:12
10:30
10:48
11:06
11:24
11:42
dev/sumologic-backup-v2/1234567890123456 5xx
dev/sumologic-blog-v2/1234567890123456 5xx
dev/sumologic-main-site/254334567123456 5xx
dev/sumologic-qe-test/1234567890123456 5xx
dev/sumologic-service-v1/4567223890123456 5xx
dev/sumologic-ui-site/1234567890123456 5xx
dev/sumologic-website-main/2344567890123456 5xx
LogErrors

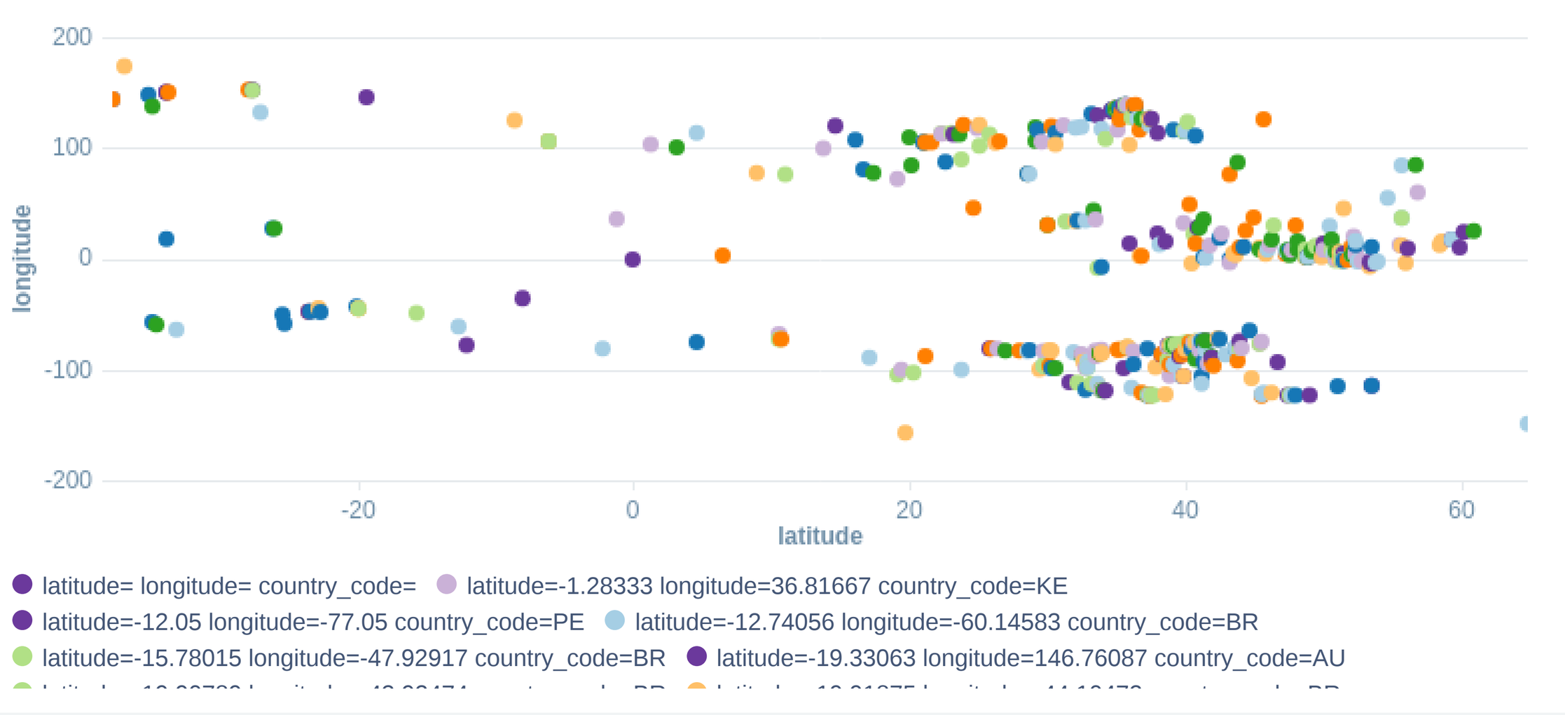
Scatter

Scatter charts display two independent numeric fields allowing you to see any correlation between them. You can visually determine how your fields relate to and affect one another.

The aggregate field is displayed as a collection of points. Each point requires two numerical fields for the X and Y axes.

```
_sourceCategory=service "message=User logged in" remote_ip
| parse "[remote_ip=*]" as remote_ip
| lookup latitude, longitude from geo://location on ip = remote_ip
| count as logins by latitude, longitude
```

Scatter



200
100
0
-100
-200
longitude
-20
0
20
40
60
latitude
latitude= longitude= country_code=
latitude=-1.28333 longitude=36.81667 country_code=KE
latitude=-12.05 longitude=-77.05 country_code=PE
latitude=-12.74056 longitude=-60.14583 country_code=BR
latitude=-15.78015 longitude=-47.92917 country_code=BR
latitude=-19.33063 longitude=146.76087 country_code=AU

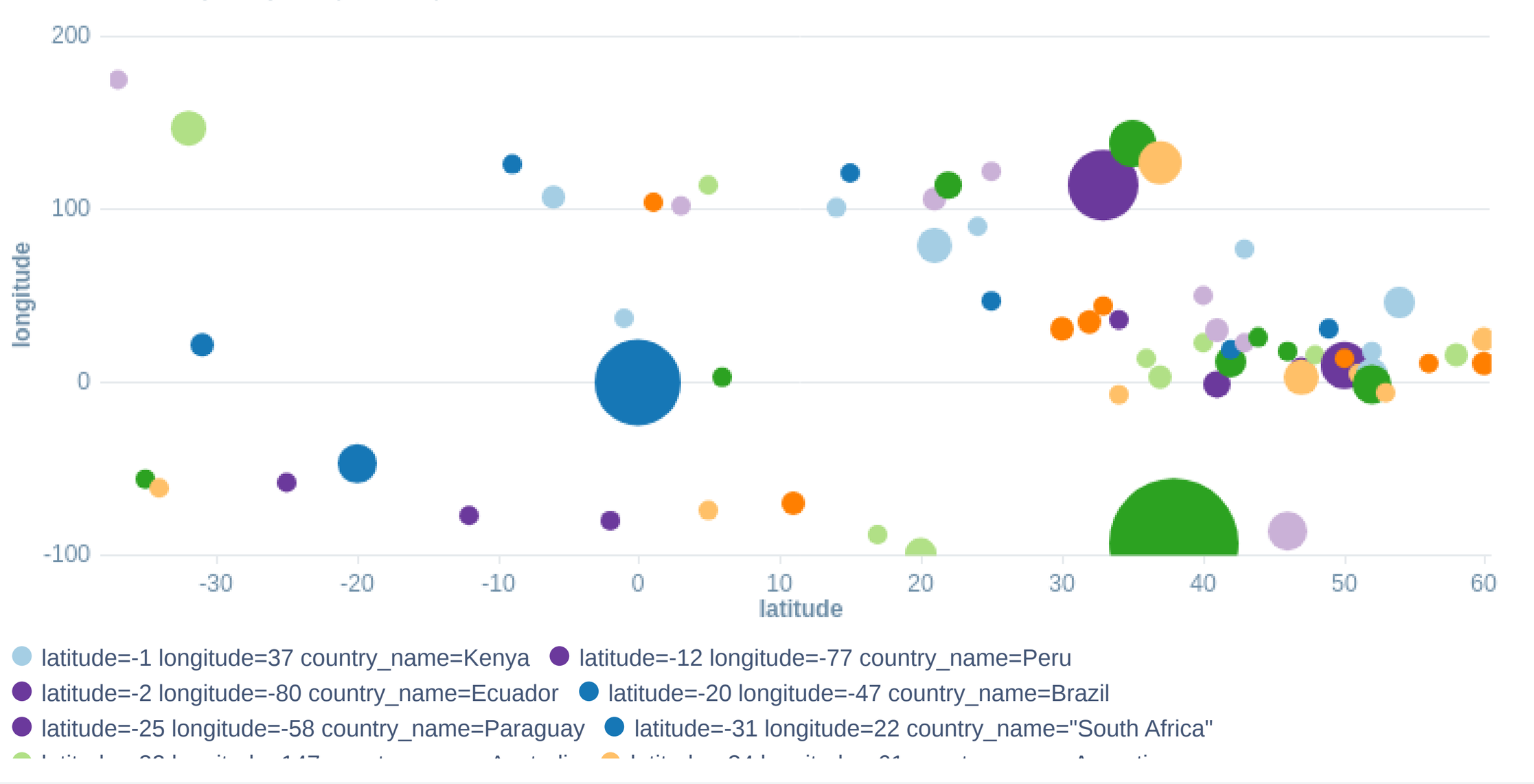
Bubble

Bubble charts display three dimensions of data. A bubble chart is a two dimensional scatter chart where each data point is represented by its size, the third dimension. This allows you to visualize the counts associated with each point. Bubble charts require at least one aggregate dimension and two other numeric dimensions.

The X dimension must be numeric and is displayed against the X axis of the bubble chart. The Y dimension must be numeric and is displayed against the Y axis of the bubble chart. The Z dimension is normally the aggregate field and shows the value of each bubble.

```
_sourceCategory=service "message=User logged in" remote_ip | parse "[remote_ip=*]" as
remote_ip | lookup latitude, longitude, city, state from geo://location on ip = remote_ip
| count as logins by city, latitude, longitude
```

Bubble: lat x long x logins by country code



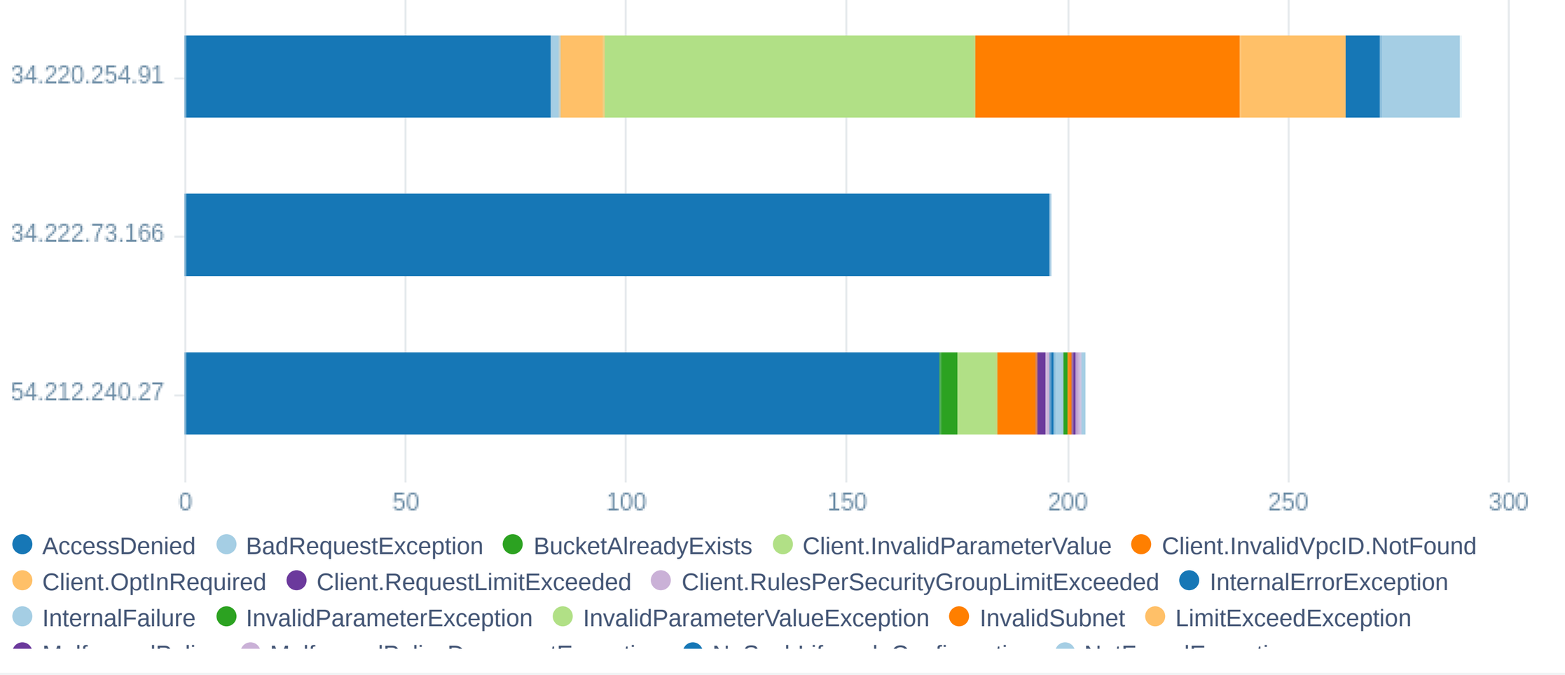
200
100
0
-100
longitude
-30
-20
-10
0
10
20
30
40
50
60
latitude
latitude=-1 longitude=37 country_name=Kenya
latitude=-12 longitude=-77 country_name=Peru
latitude=-2 longitude=-80 country_name=Ecuador
latitude=-20 longitude=-47 country_name=Brazil
latitude=-25 longitude=-58 country_name=Paraguay
latitude=-31 longitude=22 country_name="South Africa"

Transpose For Non Time Series Data

we can also use the transpose operator to stack by something other than a time series. Useful for building high density charts where we have a number of both row and column categories.

```
_sourceCategory=labs/aws/cloudtrail* errorcode
| json field=_raw "errorCode" | count by errorcode, _sourcehost
 | transpose row _sourcehost column errorcode
```

Transposing Data



34.220.254.91
34.222.73.166
54.212.240.27
0
50
100
150
200
250
300
AccessDenied
BadRequestException
BucketAlreadyExists
Client.InvalidParameterValue
Client.InvalidVpcID.NotFound
Client.OptInRequired
Client.RequestLimitExceeded
Client.RulesPerSecurityGroupLimitExceeded
InternalErrorException
InternalFailure
InvalidParameterException
InvalidParameterValueException
InvalidSubnet
LimitExceedException

Constructing a link using tourl()

you can create a link to any url with the tourl() and urlencode() functions. These are displayed as clickable links in search or dashboards that open a new panel.

You can also use this method to link to another dashboard (since template params are in the query string), or to launch context sensitive links in other tools.

Clever power users of sumo can context link with tourl into other dashboards within the same sumo org because template variable values that filter the dash are part of the query string.

This means you can constuct a 'tourl' to make a row in a search table clicable to drill down to another sumo dashboard using operators like concat and urlencode.

```
https://service.sumologic.com/ui/#/dashboardv2/<somedash>
?variables=errorCode:Client.RequestLimitExceeded;keywords:*;cat_lookup:dev
```

IPs With High Crowdstrike Malicious Confidence Score. Click link to search

	interfaceID	accountid	src	dest	malicious_confidence	threat_labels
1	eni-4b118871	382844716374	156.196.207.35	172.31.10.8	high	["KillChain/C2","MaliciousConfidence/ ","Malware/njRAT","ThreatType/Comm y"]
2	eni-4b118871	382844716374	189.19.104.178	209.114.111.1	high	["KillChain/C2","MaliciousConfidence/ ","Malware/njRAT","ThreatType/Comm y"]
3	eni-4b118872	382844716374	2.11.255.62	209.114.111.1	high	["ThreatType/Targeted","ThreatType/C entialHarvesting","ThreatType/Crimin illChain/C2","Malware/DarkComet","T Type/TargetedCrimeware","Malicious dence/High","ThreatType/Information er","ThreatType/Ransomware","Threa e/Downloader"]
4	eni-4b118871	382844716374	177.192.254.44	209.114.111.1	high	["KillChain/C2","MaliciousConfidence/ ","Malware/njRAT","ThreatType/Comm y"]
5	eni-4b118871	382844716374	208.73.22.100	12.219.43.133	high	["KillChain/C2","MaliciousConfidence/ ","Malware/njRAT","ThreatType/Comm y"]
6	eni-4b118871	382844716374	182.77.77.77	209.114.111.1	high	["KillChain/C2","MaliciousConfidence/ ","Malware/njRAT","ThreatType/Comm y"]
7	eni-4b118871	382844716374	213.136.88.99	12.219.43.133	high	["Actor/MYTHICLEOPARD","CSD/CS 17011","KillChain/C2","MaliciousConf ce/High","ThreatType/Targeted"]

1 2

Custom JSON Settings

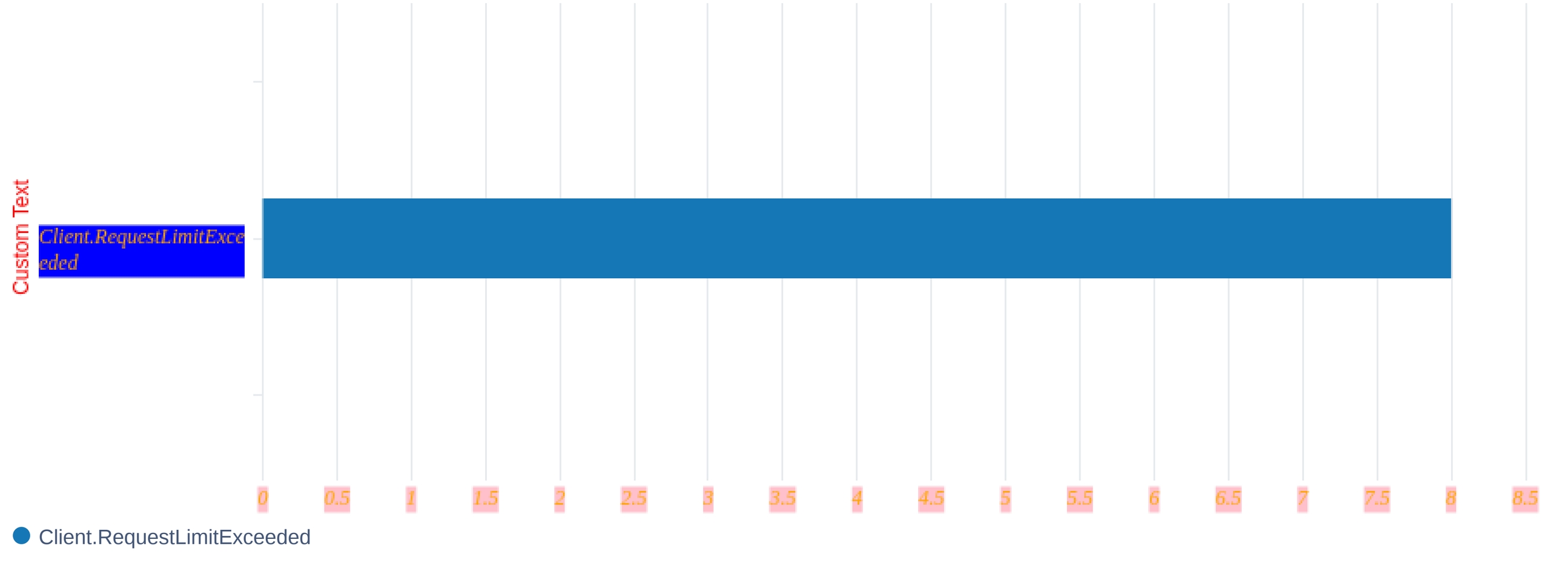
In a dashboard edit panel you can open the JSON tab to make custom changes to properties of the chart. Many settings can be used that are not supported in the UI.

example config:

```
"axisX": {
   "title": "Custom Text",
   "titleFontSize": 12,
   "labelFontSize": 12,
   "titleFontColor": "red",
   "labelFontColor": "green",
   "labelFontStyle": "italic",
   "labelFontFamily": "Calibri"
 },
```

for more info see the canvas.js docs page

Custom JSON Edited Chart



Status Using Emojis

The sumo UI will render many emoji characters in a table - which enables us to put a color coded status icon in a table column.

for encoding use utf-16 format

HTTP 5xx Error Pct - Emoji Status Icons

	url	requests	http5xx	Pct5xx	status
1	/Lists/b/ref=sva_videos_2?ie=UTF	62	43	69.4	✕
2	/testimonials/ref=vfgb_sdsd_4	73	38	52.1	✕
3	/shopping/cart/view.jsp	17	8	47.1	▯
4	/_downloads/Case_Study.pdf	295	20	6.8	▯
5	/shopping/cart/checkout.jsp	225	7	3.1	▯
6	/shopping/cart/confirm.jsp	207	3	1.4	▯
7	/_css/master.1334356838.css	484	0	0	▯
8	/favicon.ico	801	0	0	▯
9	/_media/bio_nir.jpg	587	0	0	▯
10	/_includes/follow/latest_tweet.php?	505	0	0	▯
11	/_includes/follow/follow_us.php	807	0	0	▯
12	/aboutus/	819	0	0	▯
13	/blog/index.php	1,337	0	0	▯
14	/	1,227	0	0	▯

HTTP 5xx Error Pct - Emoji Progress Bar Type

	url	requests	http5xx	Pct5xx	icon
1	/Lists/b/ref=sva_videos_2?ie=UTF	62	43	69.4	■■■■■■
2	/testimonials/ref=vfgb_sdsd_4	73	38	52.1	■■■■■
3	/shopping/cart/view.jsp	17	8	47.1	■■■■
4	/_downloads/Case_Study.pdf	295	20	6.8	■
5	/shopping/cart/checkout.jsp	225	7	3.1	■
6	/shopping/cart/confirm.jsp	207	3	1.4	■
7	/_css/master.1334356838.css	484	0	0	■
8	/favicon.ico	801	0	0	■
9	/_media/bio_nir.jpg	587	0	0	■
10	/_includes/follow/latest_tweet.php?	505	0	0	■

Distributed Tracing: Traces

This is a special panel type if you have Open Telemetry trace data in Sumo. Click on a trace to open the detailed Traces View

You can use {{vars}} in the scope of the trace panel for example:

```
application = {{my_app}}
```

Coming soon!

We are working hard to provide PNG/PDF export of this panel soon.

Distributed Tracing: Service Map

This is a special panel type for Distributed Tracing services.

Click on a node to bring up entity explorer

Coming soon!

We are working hard to provide PNG/PDF export of this panel soon.

Distributed Tracing: Custom Span Analy…

Open telemtry span data is exposed in the spans index dashboarding. The tags field contains all default and cu existing fields for service,operation etc.

```
_index=_trace_spans  error
| json field=tags "$['deployment.env
| where environment matches "{{env}}
| json field=tags "$['http.url']" as
| count by service,statuscode,status
| sort _count
```

Span Analytics: _index=trace_spans

	service	statuscode	statusmessage	operation
1	accounts	ERROR	500	/accounts/{name}
2	payments-service	ERROR		HTTP GET
3	payments-service	ERROR	500	/payments/transact/{from}
4	the-coffee-bar	ERROR	404	HTTP POST
5	accounts	ERROR	500	ApplicationFilterChain.doF
6	the-coffee-bar	ERROR	503	HTTP POST
7	the-coffee-machine	ERROR	503	HTTP POST
8	the-coffee-bar-frontend	ERROR	404	HTTP POST
9	the-coffee-machine	ERROR	502	HTTP POST
10	the-coffee-bar	ERROR	502	HTTP POST
11	water-svc	ERROR	503	POST /get_water
12	machine-svc	ERROR	503	POST /prepare_coffee

Using cat, lookups and subquery as a filter in panel

You can combine cat and subquery with template variables to filter panels.

For example say I have a lookup table saved of aws account ids and names a be able to pick a 'name' and filter by 'id' First make a template variable with a l the list of values and awsaccount as the value for the list.

```
cat /shared/lookups/example_awsaccounts | count by a
| sort account asc
```

Then use this as a subqery to filter the panel so a user can pick an account na by accountid value.

```
_sourcecategory=*cloudtrail* errorcode
[subquery: cat /shared/lookups/example_awsaccounts
| where account matches "{{my_var}}"
| count by accountid
| compose accountid keywords]
| json "errorCode","recipientAccountId","errorMessag
| count by account,errorcode,recipientaccountid,erro
```

This table is filtered by the cat_lookup var: where account matches "dev"

	account	errorCode	recipientAccountId	errorMessage	_count
1	dev	NoSuchLifecycleConfiguration	951234567898	The lifecycle configuration does not exist	8
2	dev	AccessDenied	951234567898	User: arn:aws:iam::951234567898:user/James Gordon is not authorized to perform: rds:RestoreDBInstanceFromDBSnapshot on resource: arn:aws:rds:us-east-2:951234567898:db:vcs-2017-12-12-snapshot	3
3	dev	AccessDenied	951234567898	User: arn:aws:sts::951234567898:assumed-role/config-role/AWSConfig-Describe is not authorized to perform: acm:ListCertificates	8
4	dev	AccessDenied	951234567898	User: arn:aws:iam::951234567898:user/jschmo is not authorized to perform: rds:RestoreDBInstanceFromDBSnapshot on resource: arn:aws:rds:us-east-3:951234567898:db:vcs-2017-12-12-snapshot	1
5	dev	AccessDenied	951234567898	User: arn:aws:iam::951234567898:user/kevin is not authorized to perform: rds:RestoreDBInstanceFromDBSnapshot on resource: arn:aws:rds:us-east-1:951234567898:db:vcs-2017-12-12-snapshot	1

Timeless Dashboards Using Lookup Tables

You can cat a v1 or v2 lookup as the basis for a dashboard. This effectively makes the dashboard 'timeless' because lookups don't have a time series value.

```
cat /shared/lookups/example_awsaccounts
| count by accountid,account | fields -_count | sort account asc
```

If you do have a time column in a lookup table (such as say last login time by account). You can trick sumo into dashboard that by converting the time value into a long var _messagetime before you do a timeslice. In such a use case the dashboard time selector will not work but you can make a 'days_ago' variable.

for example

```
| where now() - tolong(time) <  (1000 * 60 * 60 * 24 * {{my_var}} ) // last n days
| formatdate(tolong(time),"YYYY-MM-dd HH:mm:ss ZZZ") as update_time
| tolong(time) as _messagetime | timeslice 1d
```

One very advanced use case for a timeless dashboard might be:

- an 15m scheduled search saves the most recent activity by each id to a v2 lookup table using 'save to index'
- the lookup table will hold the most recent activity time with type,status,description columns for each id (the primary key), and a TTL of 365d

This panel is a 'timeless' panel that just shows a lookup table

	accountid	account
1	821986006950	821986006950
2	951234567898	dev
3	111111111111	prod
4	123456789033	stage
5	524597330176	teststack