mastery program

> DevOps

Monitoreo con Prometheus

> Chuy Lerma

PromQL

> Basics
> Aggregation
> Operators

Basics

PromQL

> Prometheus QL.
> Diferente de SQL.

Aggregation Basics

- > Gauges
- > Counter
- > Summary

Gauges

Usualmente cuando usas aggregators quieres sum, avg, max y min.
Para calcular el tamaño total del filesystem en cada máquina:

Gauges

Tomemos como ejemplo la métrica que nos reporta el tamaño de cada uno de nuestros file systems montados

> node_filesystem_size_bytes

Gauges

Usualmente cuando usas aggregators quieres sum, avg, max y min.
Para calcular el tamaño total del filesystem en cada máquina:

```
> sum without(device, fstype, mountpoint)(node_filesystem_size_bytes)
```

Gauges

Max puede ser usado para el tamaño del filesystem más grande montado en cada máquina:

```
> max without(device, fstype,
mountpoint)(node_filesystem_size_byt
es)
```

Gauges

Avg puede ser usado para encontrar el promedio de file descriptors (fds) abiertos:

```
> avg without(instance,
job)(process_open_fds)
```

Counters

Lo que realmente quieres obtener es que tan rapido el counter ha cambiado sobre el tiempo. Para eso usas rate, increase e irate.

Counters

Por ejemplo, para calcular la cantidad de tráfico recibido en la red por segundo:

```
>rate(node_network_receive_bytes_total[5m])
```

Counters

Como lo que obtienes es un gauge puedes hacer aggregations:

```
>sum
without(device)(rate(node_network_re
ceive_bytes_total[5m]))
```

Counters

O filtrar por device:

```
>sum
without(instance)(rate(node_network_
receive_bytes_total{device="eth0"}[5
m]))
```

Summary

Por ejemplo
prometheus_http_response_size_bytes_
Contiene la cantidad de data que cada llamada regresa

>_count contiene el número de peticiones
>_sum contiene el número de byt

Summary

Esta consulta nos da la tasa de solicitudes HTTP totales por seg

```
>sum
without(handler)(rate(prometheus_htt
p_response_size_bytes_count[5m]))
```

Summary

Esta consulta nos da la tasa de solicitudes HTTP totales por seg

```
>sum
without(handler)(rate(prometheus_htt
p_response_size_bytes_sum[5m]))
```

Summary

```
> sum
without(handler)(rate(prometheus_htt
p_response_size_bytes_sum[5m])) /
  sum
without(handler)(rate(prometheus_htt
p_response_size_bytes_count[5m]))
```

Selectors

> Matchers.
> Range Vector.
> Offset

Matchers

> = equality
> != negative equality matcher
> =~ regular expression matcher
> !~ negative regular expression matcher

Matchers

```
>node_filesystem_size_bytes{job="node",mountpoint=~"/run/.*"}
>node_filesystem_size_bytes{job="node",mountpoint!~"/run/user/.*"}

>node_filesystem_size_bytes{job="node",mountpoint=~"/run/.*",
    mountpoint!~"/run/user/.*"}
```

Range Vectors

Regresa varios ejemplos por time serie. Siempre se usan con rate functions.

```
>process_cpu_seconds_total[1m]
>rate(process_cpu_seconds_total[1m])
```

Offset

Tiempo de evaluación de una query.
Por ejemplo:

```
>process_resident_memory_bytes{job="
node"} offset 15m
```

El uso de memoria 15 minutos atrás.

HTTP API

> Query.

Query.

```
http://localhost:9090/api/v1/query?q
uery=process_resident_memory_bytes
```

Aggregation

Grouping

> without
> by

Without.

```
sum without(fstype,
mountpoint)(node_filesystem_size_byt
es)
```

By.

```
sum by(job, instance,
device)(node_filesystem_size_bytes)
```

By.

```
>sum by(job, instance,
device)(node_filesystem_size_bytes)

>count by(release)(node_uname_info)
```

Operators

> sum
> count
> avg
> min and max

Sum

```
>sum without(fstype, mountpoint,
device)(node_filesystem_size_bytes)
>sum
without(device)(rate(node_disk_read_
bytes_total[5m]))
```

It’s important for counter to add a rate before the sum

Count

```
>sum without(fstype, mountpoint,
device)(node_filesystem_size_bytes)
>sum
without(device)(rate(node_disk_read_
bytes_total[5m]))
```

It’s important for counter to add a rate before the sum

Avg

```
>avg
without(cpu)(rate(node_cpu_seconds_t
otal[5m]))
>sum
without(device)(rate(node_disk_read_
bytes_total[5m]))
```

Min and Max

```
>max without(device, fstype, mountpoint)(node_filesystem_size_bytes)
>min without(device, fstype, mountpoint)(node_filesystem_size_bytes)
```

It’s important for counter to add a rate before the sum

Binary Operators

Working with Scalars

> Arithmetic Operators
> Comparison Operators
> BOOL MODIFIER

Arithmetic Operators

```
+ addition
- subtraction
● Multiplication (*)
/ division
% modulo
^ exponentiation
```

Comparison Operators

```
== equals
!= not equals
> greater than
< less than
>= greater than or equal to
<= less than or equal to
```

Recursos importantes

>github.com/infinityworks/prometheus-example-queries
>github.com/chop-dbhi/prometheus-sql
>github.com/samber/awesome-prometheus-alerts
>percona.com/sites/default/files/presentations/Prometheus-MySQL-2101.pdf