Prometheus: Querying basics

DPTP Knowledge Talk

June 7, 2022

1. Data types
2. Metric types
3. Operators
4. Functions/queries

Expression language data types

1. Scalar - a simple numeric floating point value
2. String - a simple string value
3. Instant vector
4. Range vector

Instant vector

A set of time series containing a single sample for each time series, all sharing the same timestamp.

$$\text{CPU}_{\text{load}} = \begin{bmatrix} \text{CPU}_1 \\ \text{CPU}_2 \\ \text{CPU}_3 \\ \text{CPU}_4 \end{bmatrix} = \begin{bmatrix} 9.99 \\ 12.13 \\ 14.31 \\ 4.22 \end{bmatrix} \text{ for } T_{\text{now}}$$

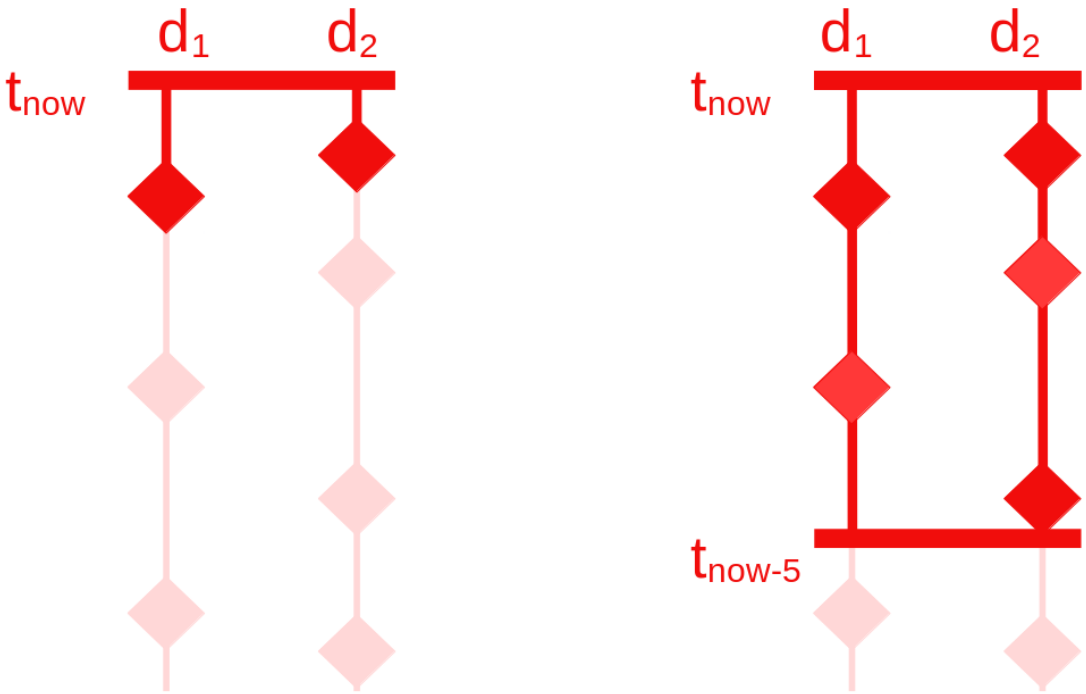
Range vector

A set of time series containing a range of data points over time for each time series.

$$\text{CPU}_{\text{load}} = \underbrace{\begin{bmatrix} 4.23 \\ 6.44 \\ 12.32 \\ 11.23 \end{bmatrix}}_{\text{CPU}_1} \underbrace{\begin{bmatrix} 3.13 \\ 5.74 \\ 11.92 \\ 2.43 \\ 32.43 \end{bmatrix}}_{\text{CPU}_2} \underbrace{\begin{bmatrix} 4.99 \\ 21.14 \\ 1.12 \end{bmatrix}}_{\text{CPU}_3} \underbrace{\begin{bmatrix} 5.15 \\ 19.24 \\ 9.42 \\ 21.87 \end{bmatrix}}_{\text{CPU}_4}$$

For the time range of last 5 minutes.

Instant vectors and range vectors



Instant vector guarantees at least one value per time series while range vector guarantees any number of values between two timestamps.

Red Hat

Available metrics

1. Counter
2. Gague
3. Histogram
4. Summary

Counter

- monotonically increasing counter
- value can reset on restart

Gague

- a metric that represents a single numerical value that can arbitrarily go up and down
- used for temperature, load, memory usage, concurrent requests

Histogram & Summary

Histogram can be used for any calculated value which is counted based on bucket values. Bucket boundaries can be configured by the developer.

```
var cacheEntryAge = prometheus.NewHistogramVec(
    prometheus.HistogramOpts{
        Name:    "ghcache_cache_entry_age_seconds",
        Help:    "The age of cache entries by API path.",
        Buckets: []float64{5, 900, 1800, 3600, 7200, 14400},
    },
    []string{"token_hash", "path", "user_agent"},
)
```

Summary measures events and are an alternative to histograms. It is used when the buckets of a metric is not known beforehand, but it is highly recommended to use histogram over summary whenever possible.

Red Hat

Binary operators

- arithmetic: +, -, /...
- comparison: !=, ==, <...
- set: and, or, unless

Binary operators accept scalars and instant vectors. **Labeling must be correct when using them with instatnt vectors.**

Red Hat

Aggregation operators

- sum() calculate sum over dimensions
- min() select minimum over dimensions
- max() select maximum over dimensions
- avg() calculate the average over dimensions
- count() count number of elements in the vector
- ...

Aggregation operators take an instant vector as input and output also an instant vector.

They can be used together with without or by clause.

Red Hat

Functions

Functions take an instant vector or a range vector as an input. The result always will be an instant vector.

rate()

- Input: range vector
- Output: instant vector
- Description: calculates the per-second average rate of increase in the counter

rate(http_requests_total[5m])

rate() function is special as breaks in monotonicity (like counter resets) are automatically adjusted for.

rate(), increase() and irate() are dedicated to use only with counters.

Red Hat

Demo

Demo!

Sources

- Prometheus documentation
- robustperception Blog
- PromCon EU 2019: PromQL for Mere Mortals
- How to build a PromQL (Prometheus Query Language)