链滴

about prometheus rules

作者：someone61489

原文链接：https://ld246.com/article/1609740001941

来源网站：链滴

1.Two types of rules

Prometheus supports two types of rules .whitch may be configured and then evaluated at reg lar intervals:

Recording rules and Alerting rules.

2.Recording rules

1.Recording rules allow you to precompute frequently needed or computationally expensive xpressions and save their result as a new set of time series.

2.Be mutch faster than executing the origin expression every time it's need.

3.Rules within a group run sequentially at a regular interval.

the syntax of a rule file following

```
groups:
  [ - <rule_group> ]
```

the rule group following

```
name: <string>
[ interval: <duration> | default = global.evaluation_interval ]
rules:
  [ - <rule> ... ]
```

the recording rule following

```
record: <string>
```

原文链接：about prometheus rules

```
expr: <string>
labels:
  [ <labelname>: <labelvalue> ]
```

the alerting rule following

```
alert: <string>
expr: <string>
[ for: <duration> | default = 0s ]
labels:
  [ <labelname>: <tmpl_string> ]
annotations:
  [ <labelname>: <tmpl_string> ]
```

3.Alerting rules

1.Alerting rules allow you to define some alert conditions that used Prometheus expression la guage and to send notifaction about firing alerts to an external service.

an example of alerting config would like this

```
groups:
- name: example
  rules:
  - alert: HighRequestLatency
    expr: job:request_latency_seconds:mean5m{job="myjob"} > 0.5
    for: 10m
    labels:
      severity: page
    annotations:
      summary: High request latency
```

and a template description and annotation would like this

```
groups:
- name: example
  rules:

  # Alert for any instance that is unreachable for >5 minutes.
  - alert: InstanceDown
    expr: up == 0
    for: 5m
    labels:
      severity: page
    annotations:
      summary: "Instance {{ $labels.instance }} down"
      description: "{{ $labels.instance }} of job {{ $labels.job }} has been down for more than 5 m
nutes."

  # Alert for any instance that has a median request latency >1s.
  - alert: APIHighRequestLatency
    expr: api_http_request_latencies_second{quantile="0.5"} > 1
    for: 10m
    annotations:
```

```
    summary: "High request latency on {{ $labels.instance }}"
    description: "{{ $labels.instance }} has a median request latency above 1s (current value: {{
value }}s)"
```

3.1 Template

1.Prometheus support templating in annotations and labels of alerts.

2.Some functions such as iterate over data,use conditionals,format data.

3.The prometheus template language is based on the Go Template.

3.1.1 Template examples

Iteration

```
{{ range query "up" }}
  {{ .Labels.instance }} {{ .Value }}
{{ end }}
```

Display one value

```
{{ with query "some_metric{instance='someinstance'}" }}
  {{ . | first | value | humanize }}
{{ end }}
```

Use console url param

```
{{ with printf "node_memory_MemTotal{job='node',instance='%s'}" .Params.instance | query }}
  {{ . | first | value | humanize1024 }}B
{{ end }}
```

That's the meaning of: http://xxx.html?instance=127.0.0.1 then the .Params.instance will evalu te to 127.0.0.1

Advanced iteration

```
<table>
{{ range printf "node_network_receive_bytes{job='node',instance='%s',device!='lo'}" .Params.i
stance | query | sortByLabel "device"}}
  <tr><th colspan=2>{{ .Labels.device }}</th></tr>
  <tr>
    <td>Received</td>
    <td>{{ with printf "rate(node_network_receive_bytes{job='node',instance='%s',device='%s'}
5m])" .Labels.instance .Labels.device | query }}{{ . | first | value | humanize }}B/s{{end}}</td>
  </tr>
  <tr>
    <td>Transmitted</td>
    <td>{{ with printf "rate(node_network_transmit_bytes{job='node',instance='%s',device='%s
}[5m])" .Labels.instance .Labels.device | query }}{{ . | first | value | humanize }}B/s{{end}}</td>
  </tr>{{ end }}
</table>
```

define reusable tempalte

```
{{define "myMultiArgTemplate"}}
  First argument: {{.arg0}}
  Second argument: {{.arg1}}
{{end}}
{{template "myMultiArgTemplate" (args 1 2)}}
```

3.1.2 Template reference

go template https://golang.org/pkg/text/template/#hdr-Functions

Queries

Name	Arguments	Returns	Notes
query	query string	[]sample	ueries the database, does not support returning range vectors.
first	[]sample	sample	E uivalent to index a 0
label	label, sample	string	quivalent to index sample.Labels label
value	sample	float64	E uivalent to sample.Value
sortByLabel	label, []samples	[]sample	orts the samples by the given label. Is stable.

first, label and value are intended to make query results easily usable in pipelines.

Numbers

Name	Arguments	Returns	Notes
humanize	number	string	onverts a number to a more readable format, using metric prefixes.
humanize1024	number	string	ike humanize, but uses 1024 as the base rather than 1000.
humanizeDuration	number	string	onverts a duration in seconds to a more readable format.
humanizePercentage	number	string	onverts a ratio value to a fraction of 100.
humanizeTimestamp	number	string	onverts a Unix timestamp in seconds to a more readable format.

Humanizing functions are intended to produce reasonable output for consumption by human , and are not guaranteed to return the same results between Prometheus versions.

Strings

Name	Arguments	Returns	Notes
title	string	string	strings.Title, capitalises first character of each word.
toUpper	string	string	strings.ToUpper, converts all characters to upper case.
toLower	string	string	strings.ToLower, converts all characters to lower case.
match	pattern, text	boolean	regexp.MatchString Tests for a unanchored regexp match.
reReplaceAll	pattern, replacement, text	string	regexp.ReplaceAllString Regexp substitution, unanchored.
graphLink	expr	string	Returns path to graph view in the expression browser for the expression.
tableLink	expr	string	Returns path to tabular ("Table") view in the expression browser for the expression.

Others

Name	Arguments	Returns	Notes
args	[]interface{}	map[string]interface{}	This converts a list of objects to a map with keys arg0, arg1 etc. This is intended to allow multiple arguments to be passed to templates.
tmpl	string, []interface{}	nothing	Like the built-in template, but allows non-literals as the template name. Note that the result is assumed to be safe, and will not be auto-escaped. Only available in consoles.
safeHtml	string	string	Marks string as HTML not requiring auto-escaping.

4.PromeQL

4.1 operators

complute

+ - * / % ^

compare binary

== != >= <= > <

Logical set binary

and , or ,unless

Vector Matching

one to one like this

method:http_requests:rate5m{method="get"}

<vector expr> <bin-op> ignoring(<label list>) <vector expr>
<vector expr> <bin-op> on(<label list>) <vector expr>

many to one and one to many like this

method_code:http_errors:rate5m / ignoring(code) group_left method:http_requests:rate5m

<vector expr> <bin-op> ignoring(<label list>) group_left(<label list>) <vector expr>
<vector expr> <bin-op> ignoring(<label list>) group_right(<label list>) <vector expr>
<vector expr> <bin-op> on(<label list>) group_left(<label list>) <vector expr>
<vector expr> <bin-op> on(<label list>) group_right(<label list>) <vector expr>

Aggreation operators

- sum (calculate sum over dimensions)
- min (select minimum over dimensions)
- max (select maximum over dimensions)
- avg (calculate the average over dimensions)
- group (all values in the resulting vector are 1)
- stddev (calculate population standard deviation over dimensions)
- stdvar (calculate population standard variance over dimensions)
- count (count number of elements in the vector)
- count_values (count number of elements with the same value)
- bottomk (smallest k elements by sample value)
- topk (largest k elements by sample value)
- quantile (calculate φ-quantile (0 ≤ φ ≤ 1) over dimensions)

parameter is only required for count_values, quantile, topk and bottomk.

Without remove the listed labels from the result vector

By does the opposite and drops labels that are not list in the by clause

<aggr-op> [without|by (<label list>)] ([parameter,] <vector expression>)

<aggr-op>([parameter,] <vector expression>) [without|by (<label list>)]

4.2 functions

- abs(): absolute value
- absent():

- ceil(): rounds the sample values of all elements in v up to the nearest integer.
- changes() :For each input time series, changes(v range-vector) returns the number of time its value has changed within the provided time range as an instant vector.
- clamp_max(): clamps the sample values of all elements in v to have an upper limit of max.
- clamp_min(): clamps the sample values of all elements in v to have a lower limit of min.
- day_of_month(): returns the day of the month for each of the given times in UTC. Returned alues are from 1 to 31.
- day_of_week(): returns the day of the week for each of the given times in UTC. Returned val es are from 0 to 6, where 0 means Sunday etc.
- day_in_month(): returns number of days in the month for each of the given times in UTC. Re urned values are from 28 to 31.
- delta():
- deriv(): calculates the per-second derivative of the time series in a range vector v, using s mple linear regression.
- exp(): calculates the exponential function for all elements in v
- floor(): rounds the sample values of all elements in v down to the nearest integer.
- histogram_quantile(): calculates the φ-quantile (0 ≤ φ ≤ 1) from the buckets b of a histog am.
- holt_winters():
- hour(): returns the hour of the day for each of the given times in UTC. Returned values are f om 0 to 23.
- idelta(): calculates the difference between the last two samples in the range vector v, retu ning an instant vector with the given deltas and equivalent labels.
- increase(): calculates the increase in the time series in the range vector. Breaks in monotonic ty (such as counter resets due to target restarts) are automatically adjusted for
- irate(): calculates the per-second instant rate of increase of the time series in the range vect r.
- label_join():
- label_replace():
- ln(): calculates the natural logarithm for all elements in v
- log2(): calculates the binary logarithm for all elements in v
- log10(): calculates the decimal logarithm for all elements in v
- minute(): returns the minute of the hour for each of the given times in UTC. Returned values are from 0 to 59.
- month(): returns the month of the year for each of the given times in UTC. Returned values re from 1 to 12, where 1 means January etc.
- predict_linear(): predicts the value of time series t seconds from now, based on the range ector v, using simple linear regression.
- rate(): calculates the per-second average rate of increase of the time series in the range vec or.
- resets(): returns the number of counter resets within the provided time range as an instant ector.

- round(): rounds the sample values of all elements in `v` to the nearest integer
- scalar(): returns the sample value of that single element as a scalar.
- sort(): returns vector elements sorted by their sample values, in ascending order.
- sort_desc():
- sqrt(): calculates the square root of all elements in `v`.
- time(): returns the number of seconds since January 1, 1970 UTC. Note that this does not actually return the current time, but the time at which the expression is to be evaluated.
- timestamp(): returns the timestamp of each of the samples of the given vector as the number of seconds since January 1, 1970 UTC.
- vector(): returns the scalar `s` as a vector with no labels.
- year(): returns the year for each of the given times in UTC.
- `<aggregation>_over_time()`:

4.3 demos

```
http_requests_total
http_requests_total{job="apiserver",handler="/api/comments"}
http_requests_total{job="apiserver",handler="/api/comments"}[5m]
http_request_total{job=~".*server"}
http_request_total{status!~"4.."}
rate(http_request_total[5m])[30m:1m]
max_over_time(deriv(rate(distance_covered_total[5s])[30s:5s])[10m:])
rate(http_request_total[5m])
sum by(job)(rate(http_request_total[5m]))
topk(3,sum by(app,proc)(rate(instance_cpu_time_ns[5m])))
count by (app) (instance_cpu_time_ns)
```

4.4 http api

expression queries

```
GET /api/v1/query
POST /api/v1/query

query=<string> prometheus exporession query string
time=<timestamp> evaluation timestamp
timeout=<duration> timeout
```

range query

```
GET /api/v1/query_range
POST /api/v1/query_range

query=<string> prometheus expression query string
start=<timestamp> start
end=<timestamp> end
step=<float> step
timeout=<duration> timeout
```

原文链接：about prometheus rules

metadata

GET /api/v1/series
POST /api/v1/series

match[]=<series_selector>: Repeated series selector argument that selects the series to return
At least one match[] argument must be provided.
start=<rfc3339 | unix_timestamp>: Start timestamp.
end=<rfc3339 | unix_timestamp>: End timestamp.

label names

GET /api/v1/labels
POST /api/v1/labels

start<timestamp>
end<timestamp>

label values

GET /api/v1/label/<label_name>/values
start
end

targets

GET /api/v1/targets

rules

GET /api/v1/rules
type: alert/record

alerts

GET /api/v1/alerts

target metadata

GET /api/v1/targets/metadata
match_target=<label_selectors>:
metric=<string>
limit=<number>

metric metadata

GET /api/v1/metadata
limit=<number>
metric=<string>

alertmanagers

GET /api/v1/alertmanagers

config

GET /api/v1/status/config

runtime information

GET /api/v1/status/runtimeinfo

build information

GET /api/v1/status/buildinfo

over