Expectations on Remote Data

Supporting the Prometheus Remote Storage API

Alfred Landrum

Engineer @ Sysdig

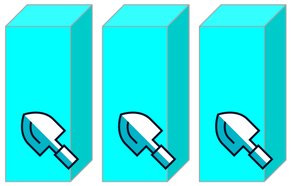
Twitter: @alfred-landrum
Github: alfred-landrum

Sysdig
Dashboards/Alerts/Topology

Sysdig Backend

Sysdig Data Engine
and Store

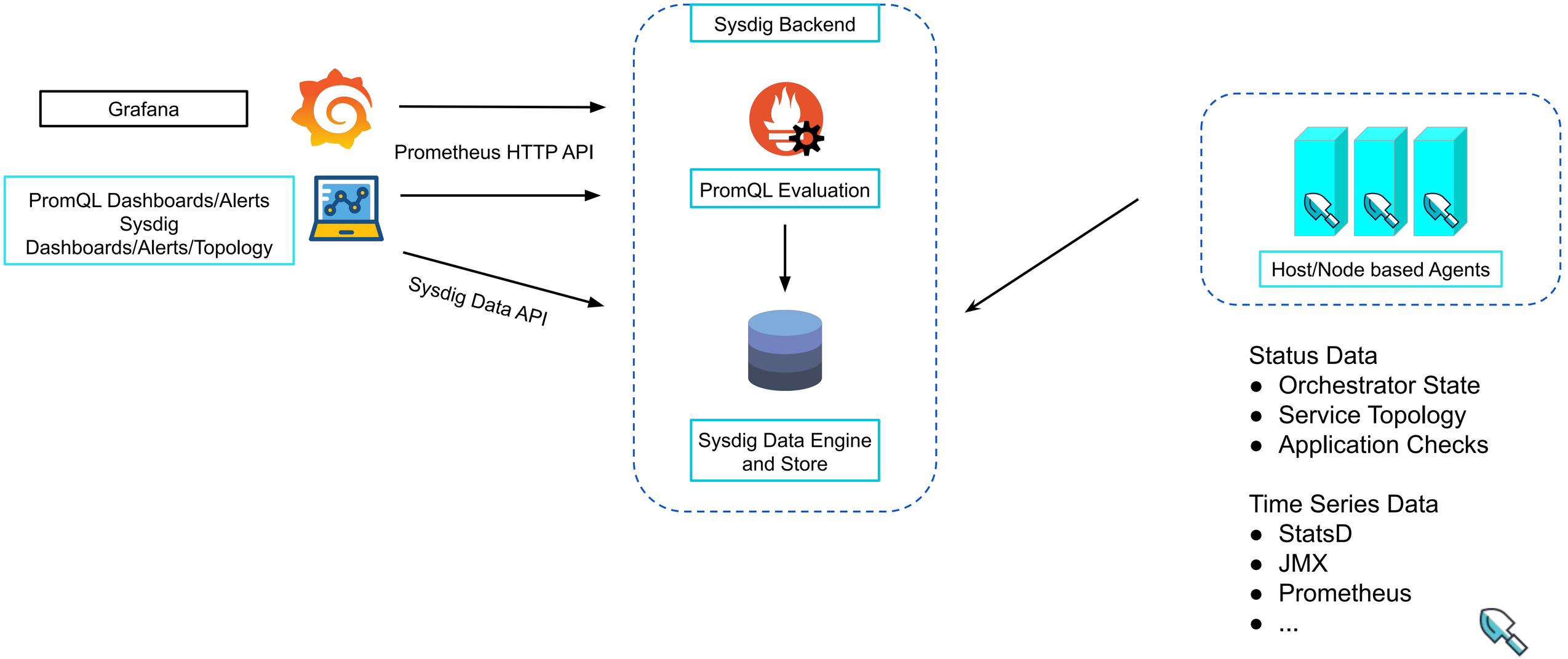
Distributed Datastore
- Time/Group Aggregation
- RBAC
- Downsampling



Host/Node based Agents

Status Data
- Orchestrator State
- Service Topology
- Application Checks

Time Series Data
- StatsD
- JMX
- Prometheus
- ...

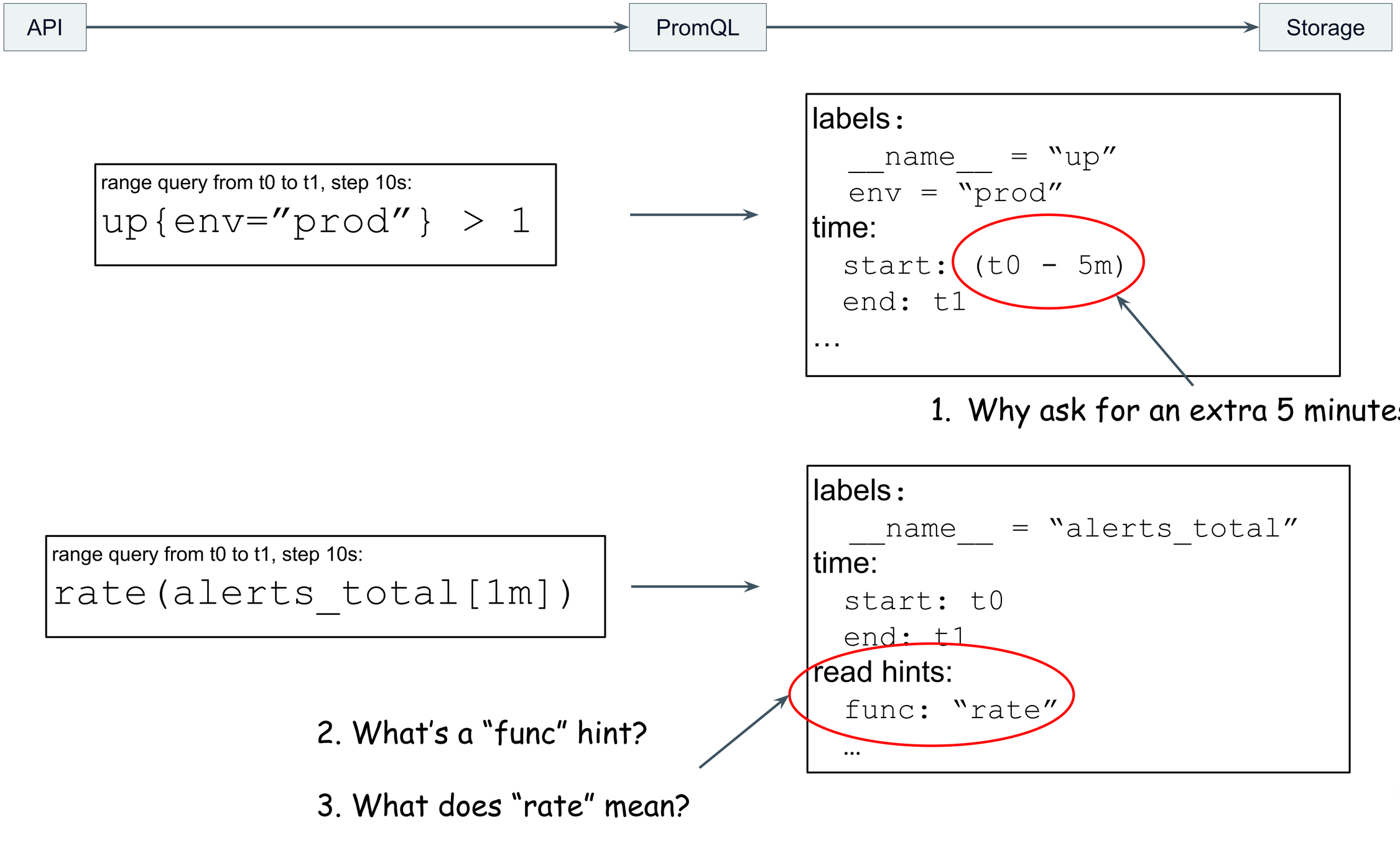


Status Data

- Orchestrator State
- Service Topology
- Application Checks

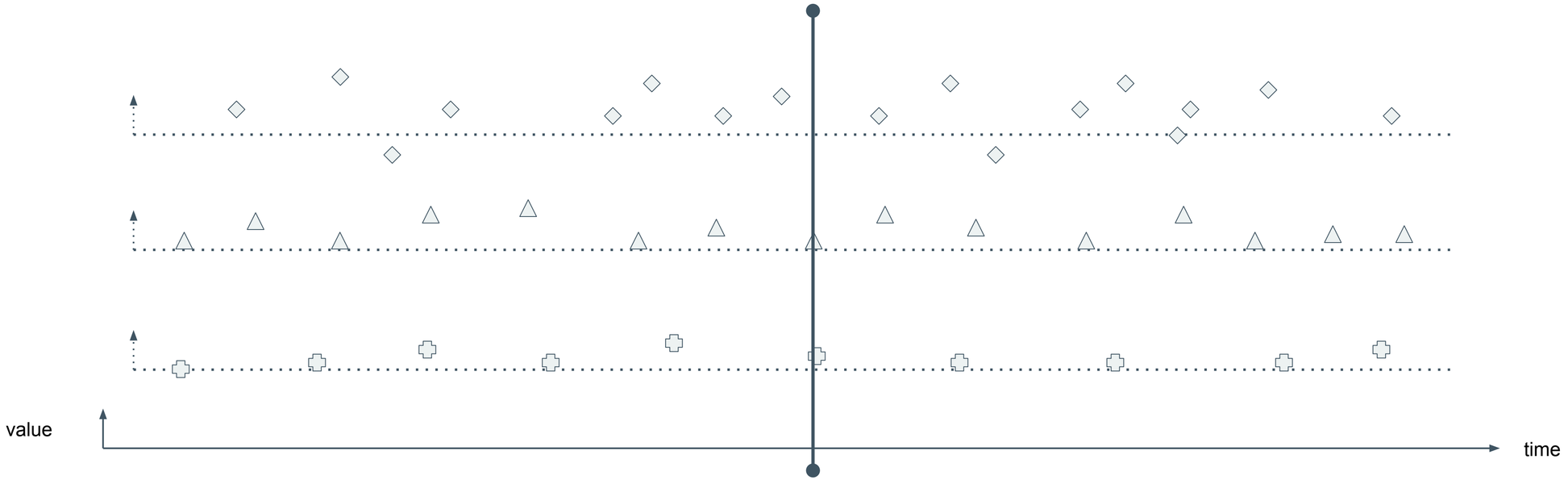
Time Series Data

- StatsD
- JMX
- Prometheus
- ...

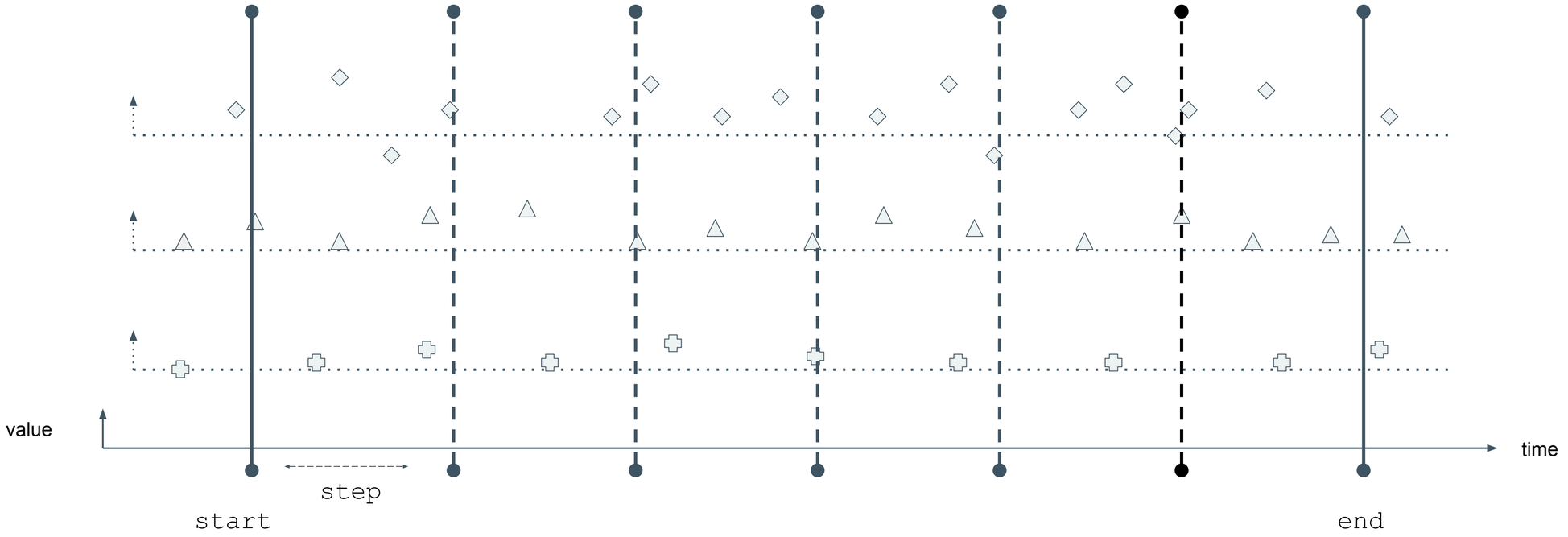

API
PromQL
Storage
range query from t0 to t1, step 10s:
up{env="prod"} > 1
labels:
__name__ = "up"
env = "prod"
time:
start: (t0 - 5m)
end: t1
...
1. Why ask for an extra 5 minutes?
range query from t0 to t1, step 10s:
rate(alerts_total[1m])
labels:
__name__ = "alerts_total"
time:
start: t0
end: t1
read hints:
func: "rate"
...
2. What's a "func" hint?
3. What does "rate" mean?

Storage data model: a set of time series, identified by metric name and labels.

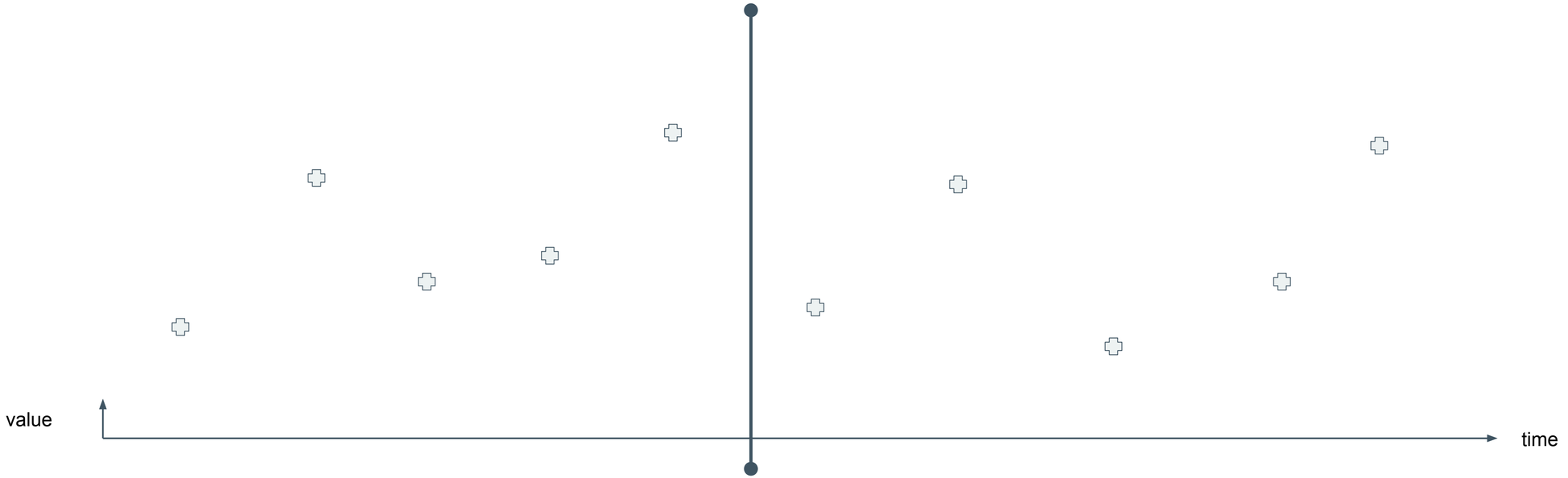
No alignment guarantees.



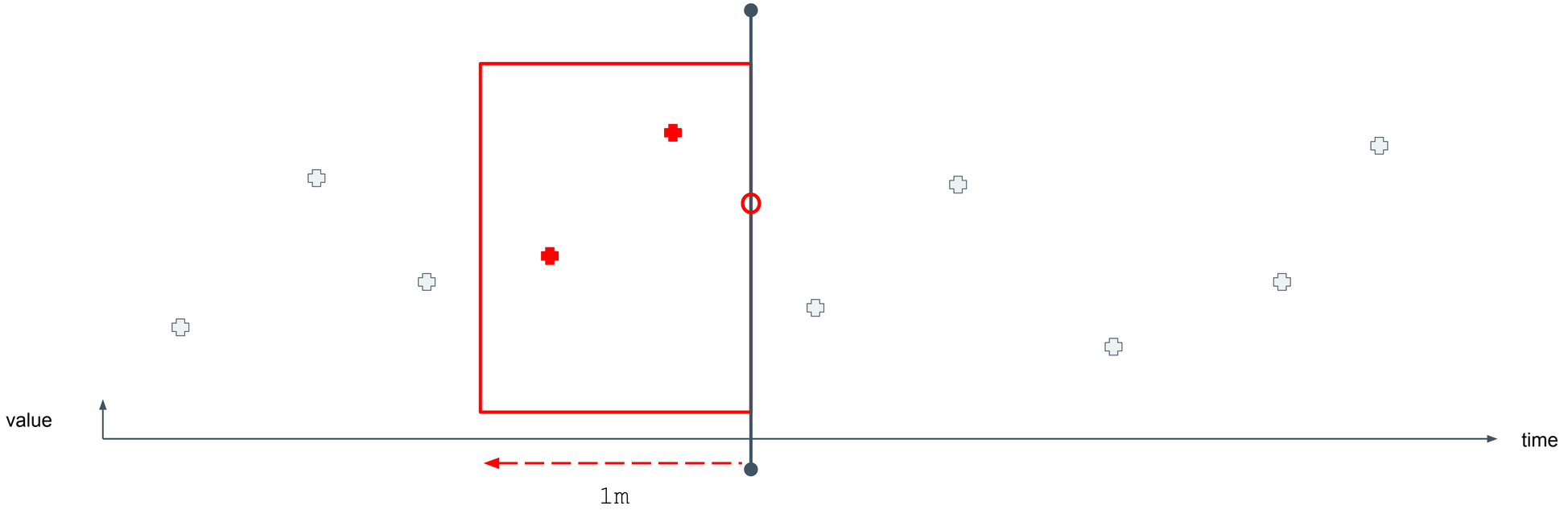
Instant Query: evaluate an expression at a particular time.



Range Query: logically, a repeated instant query on `[start,end]` every `step`.

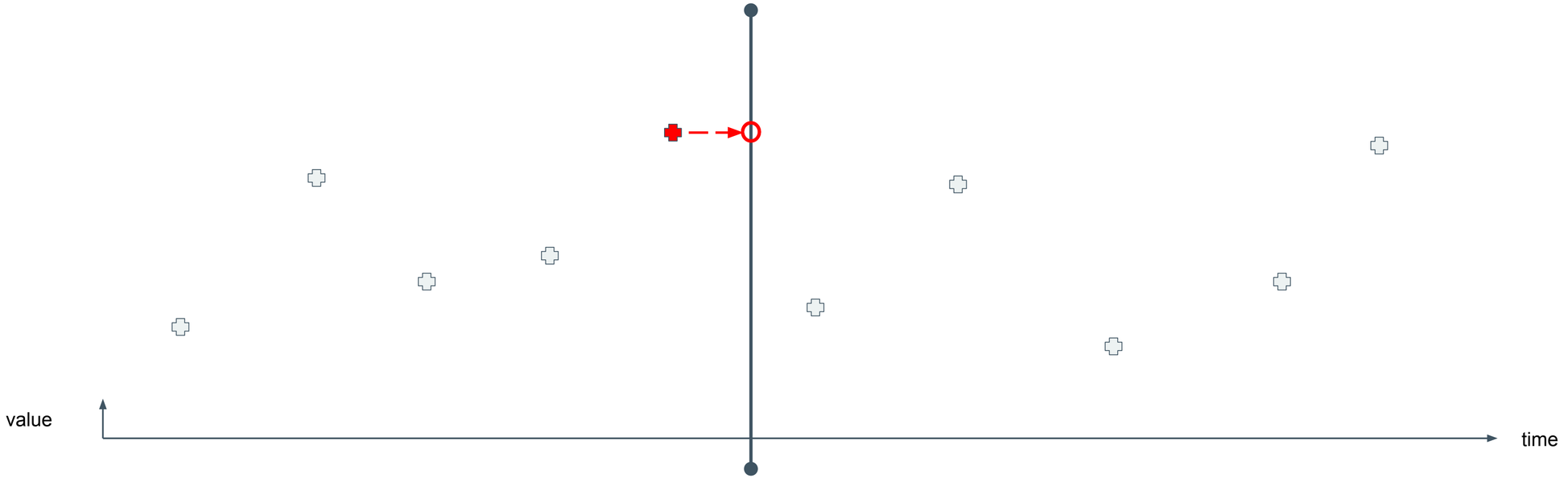


What if there’s no sample at the evaluation time?



Range Vector Selector

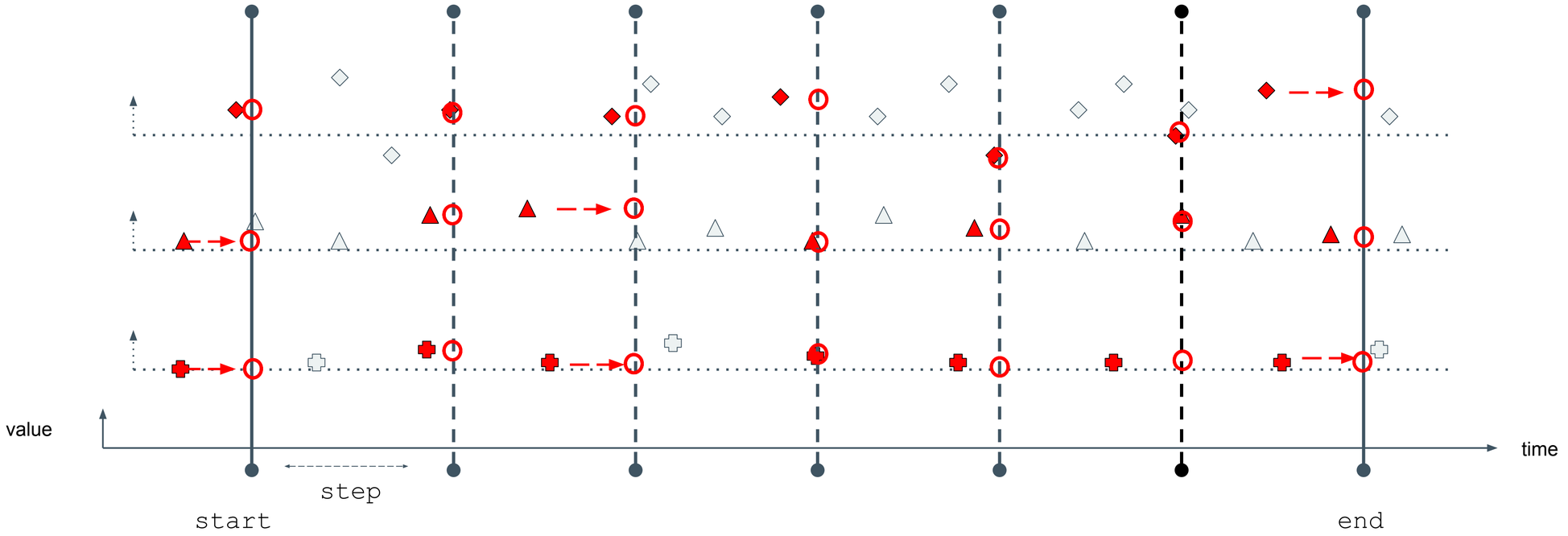
PromQL: `avg_over_time(queue_depth[1m])`



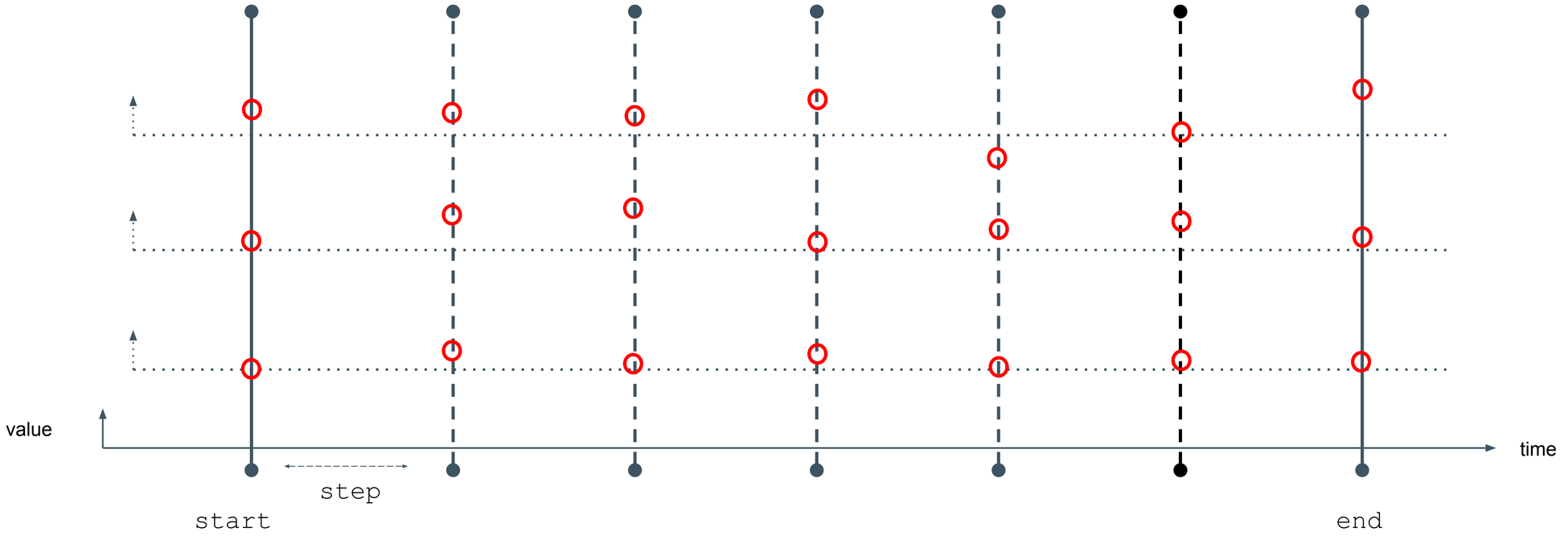
Instant vector selector

PromQL: `queue_depth`

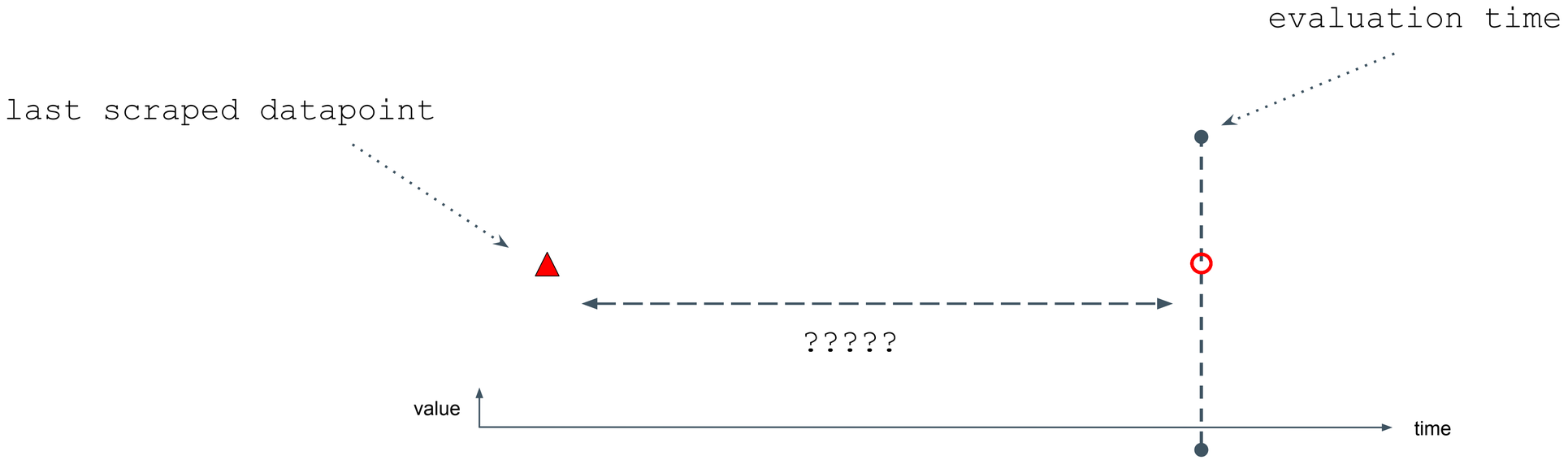
The most recent value found at **or before** the evaluation time.



Same for range queries, applied at each step.

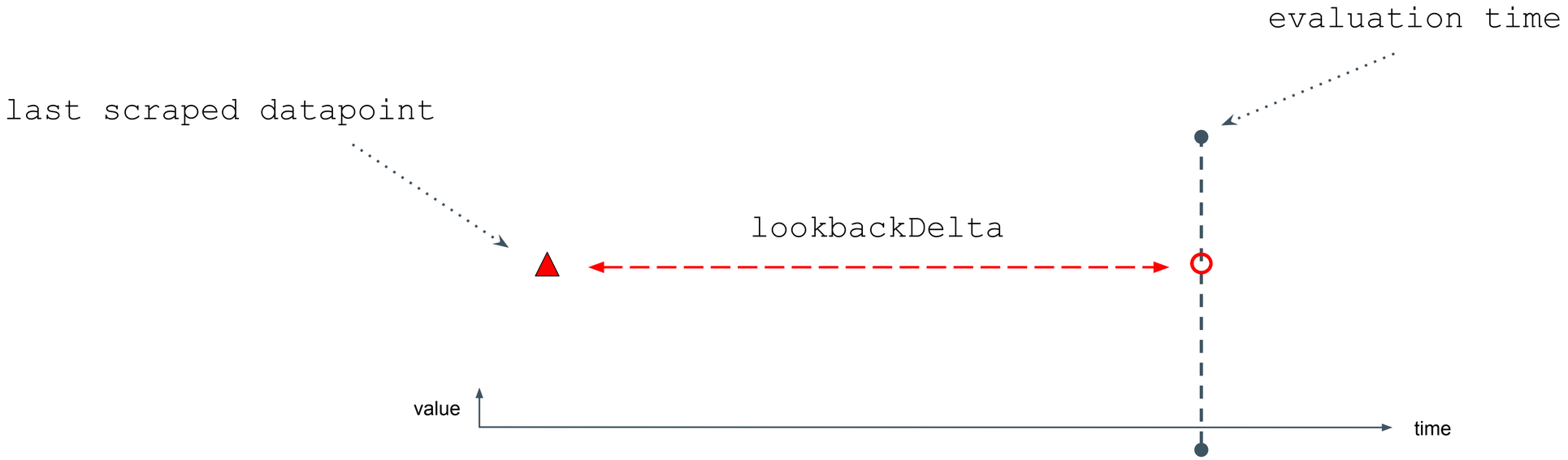


PromQL now has aligned values, for calculations, comparisons, etc.



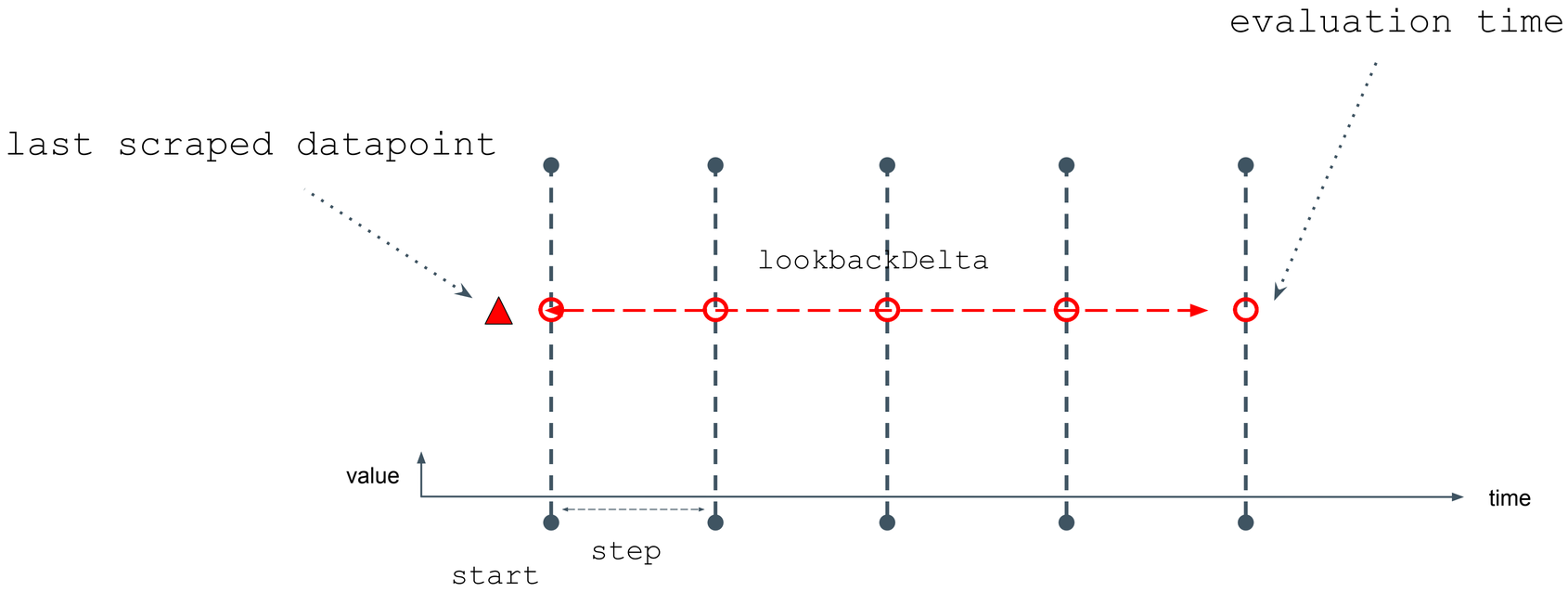
How long do you see the value of the last sample?

Controlled in 2 different ways:

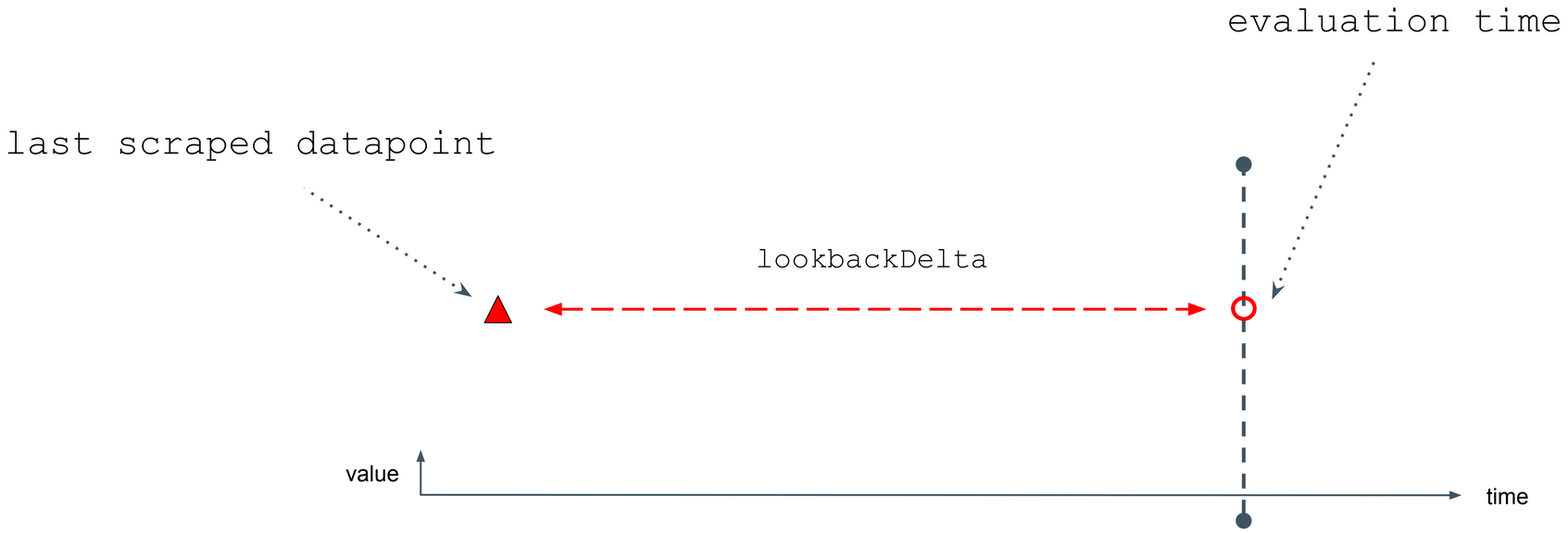


First way is via a configuration setting: `lookbackDelta`

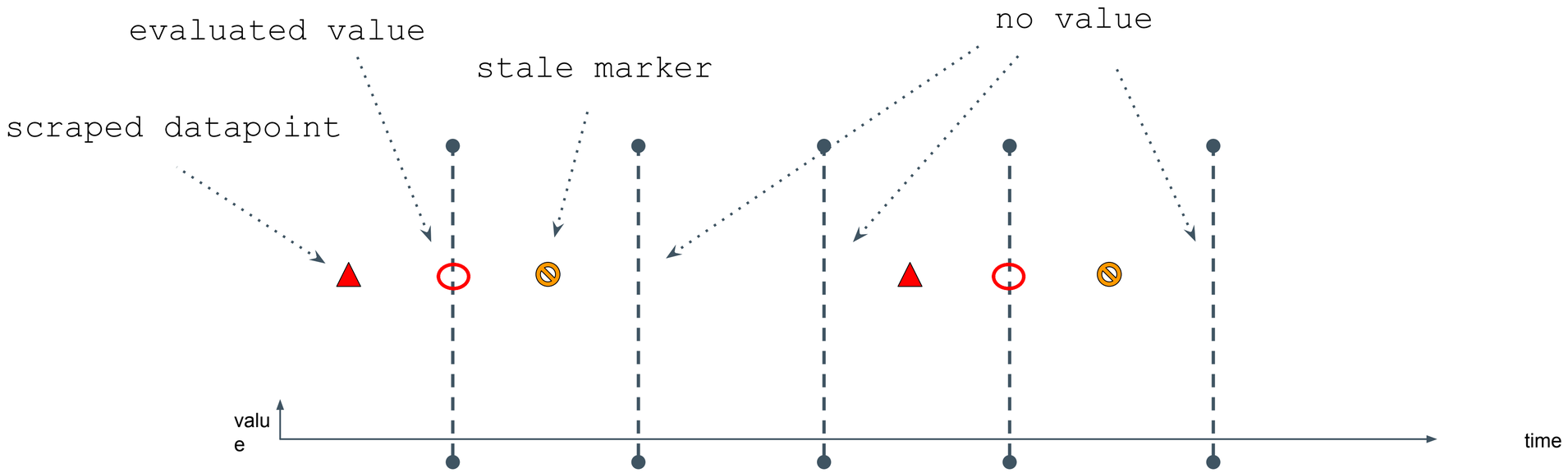
Default is **5 minutes.**



Same for range queries, applied at each step.

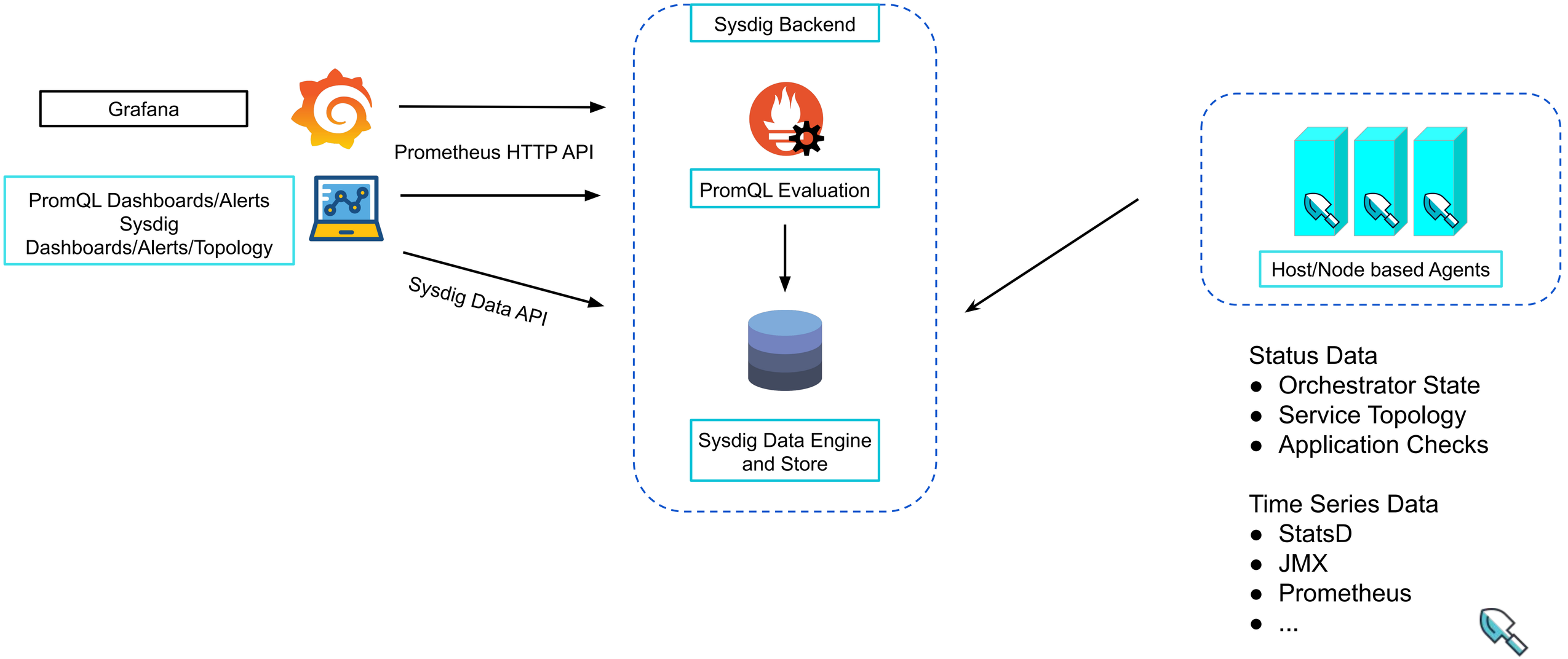


Consider an alert that should fire if there’s no value.



The second way: *stale markers*

Scraping logic adds them 1-2 intervals after the last sample.

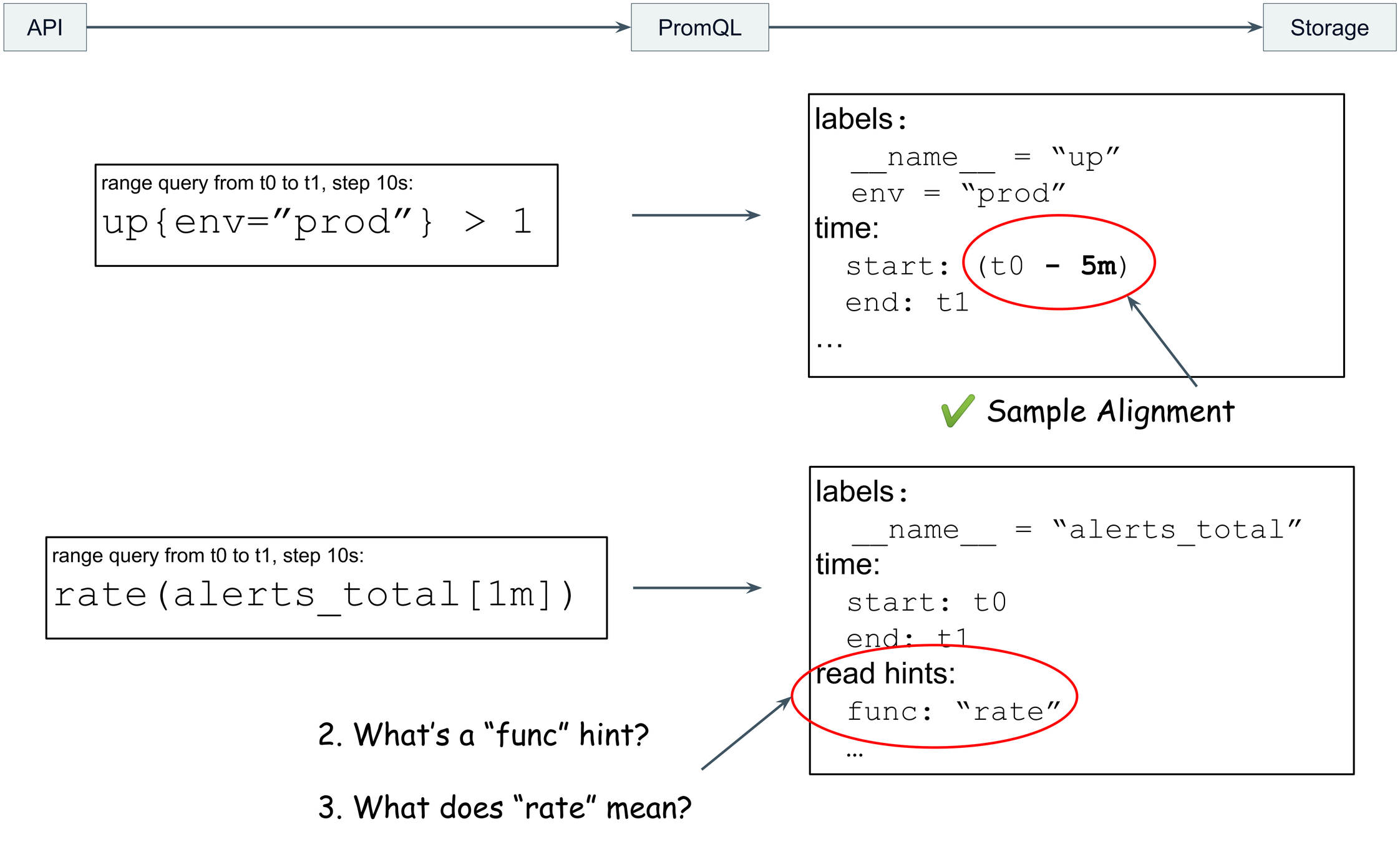


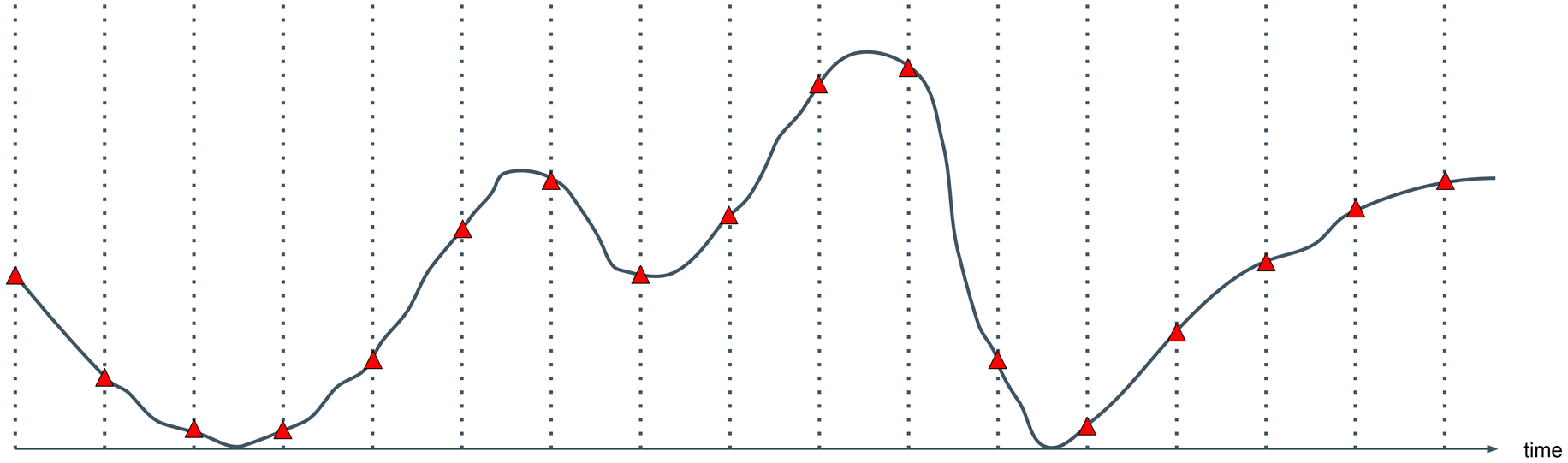
Status Data

- Orchestrator State
- Service Topology
- Application Checks

Time Series Data

- StatsD
- JMX
- Prometheus
- ...

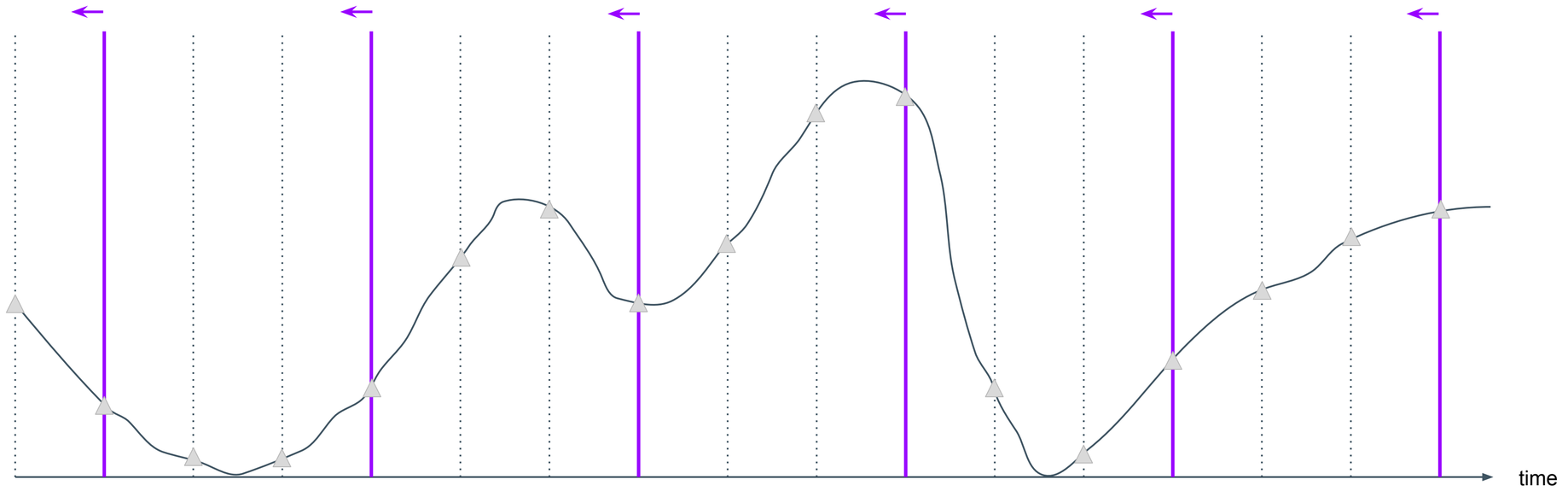

API
PromQL
Storage
range query from t0 to t1, step 10s:
up{env="prod"} > 1
labels:
__name__ = "up"
env = "prod"
time:
start: (t0 - 5m)
end: t1
...
✔ Sample Alignment
range query from t0 to t1, step 10s:
rate(alerts_total[1m])
labels:
__name__ = "alerts_total"
time:
start: t0
end: t1
read hints:
func: "rate"
...
2. What's a "func" hint?
3. What does "rate" mean?



Scraping intervals typically on the order of 1 minute.

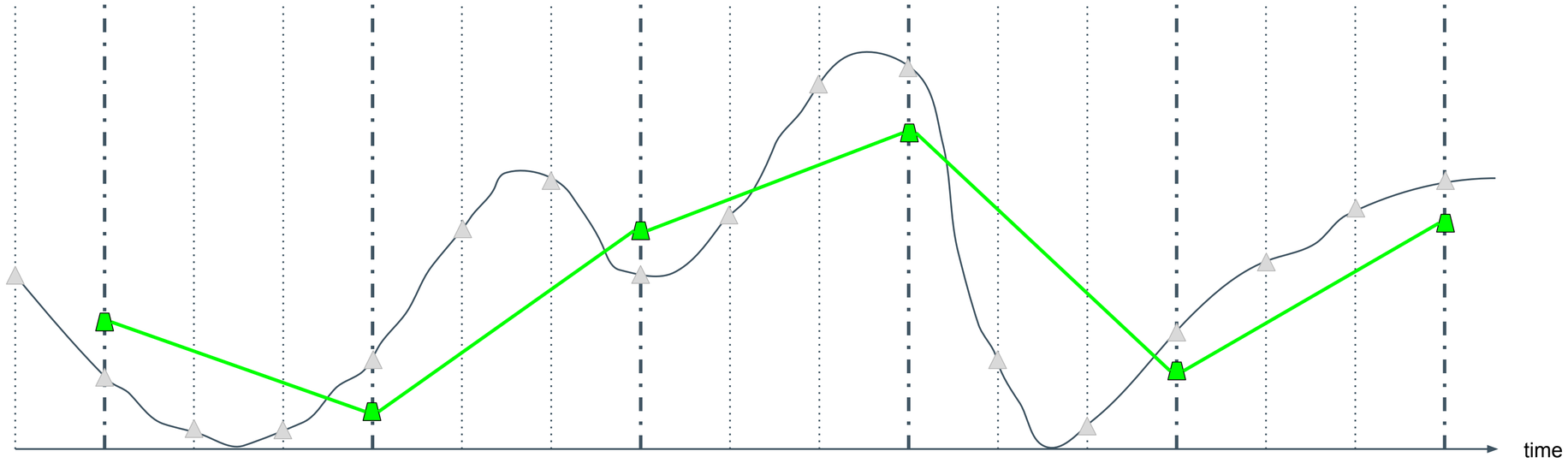
A query for a month’s data would take ~45k samples.

That’s likely more than the pixel width of your display.

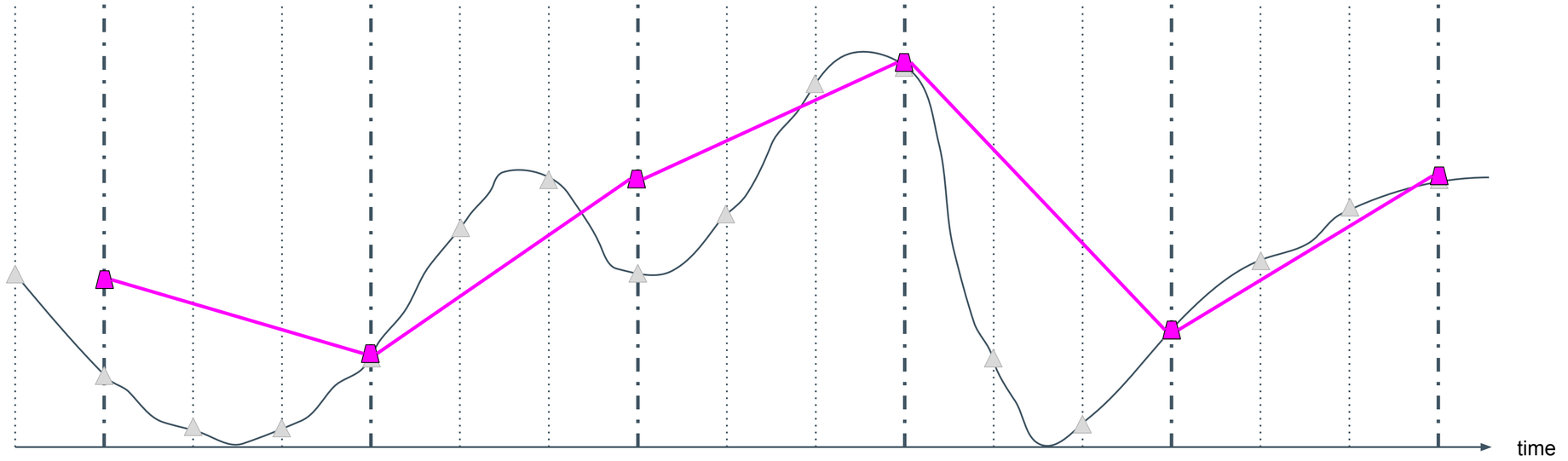


Store an *aggregation* of many samples within some fixed resolution.

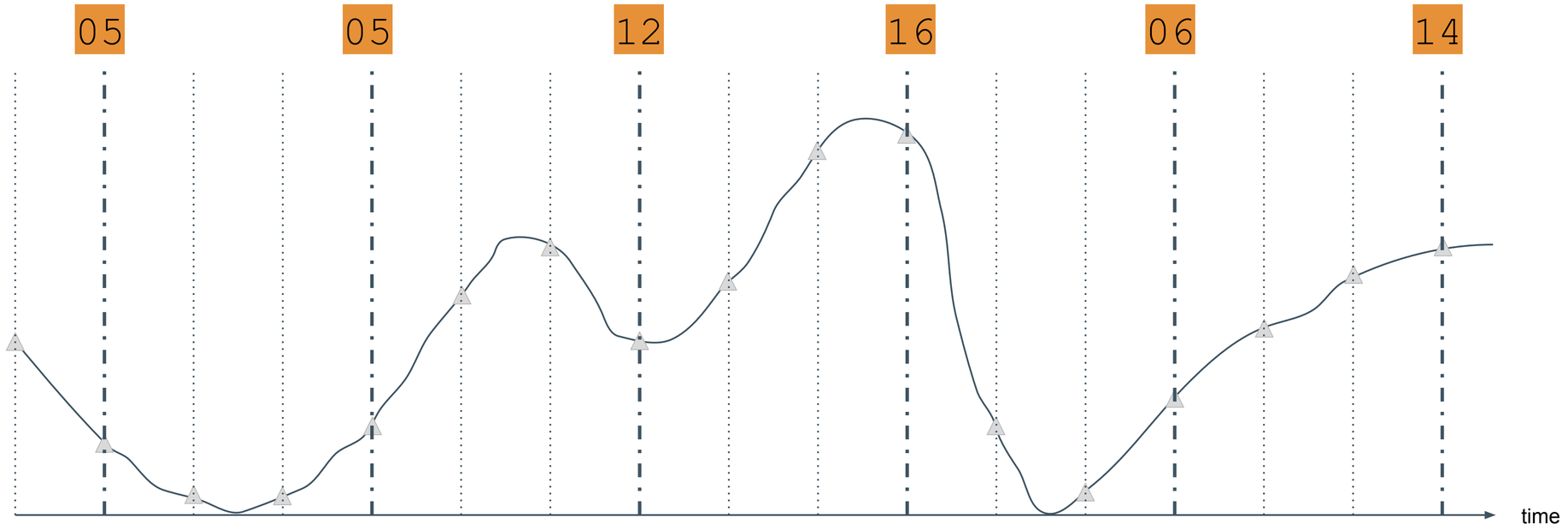
What representative value should you store?



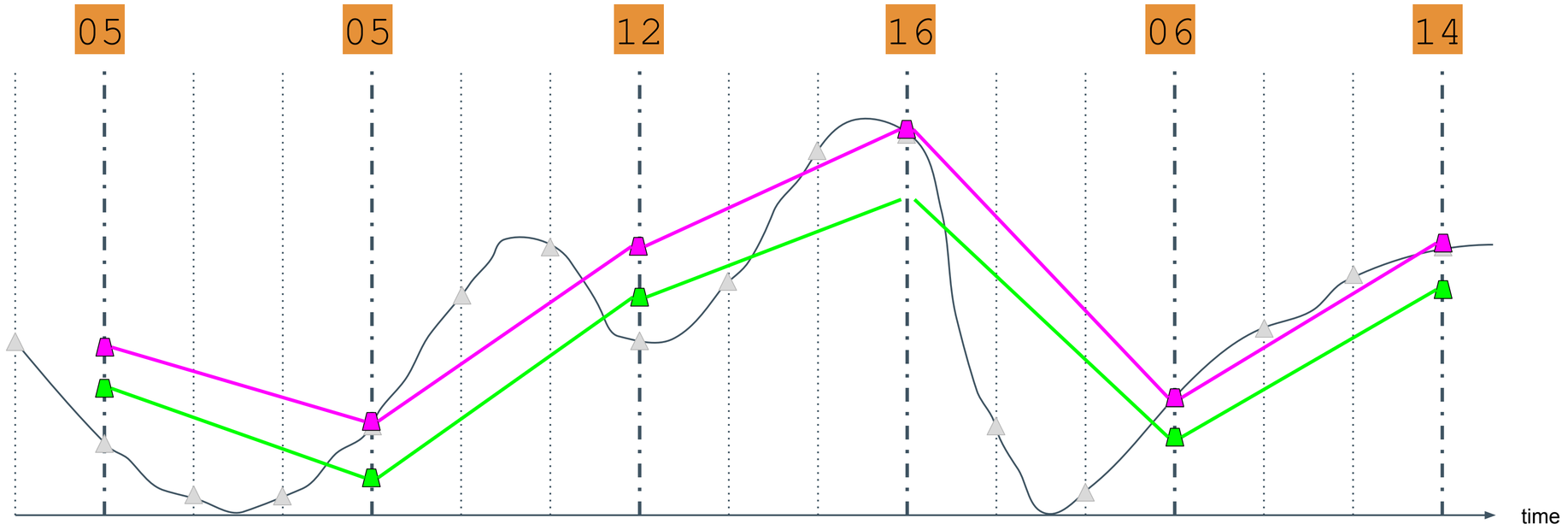
Average



Maximum



Sum



Not limited to a single aggregation - store several.

How to select the best one for a query?

range query from t0 to t1, step 10s:

```
up{env="prod"} > 1
```

```
labels:
  __name__ = "up"
  env = "prod"
time:
  start: (t0 - 5m)
  end: t1
...
```

✔ Sample Alignment

range query from t0 to t1, step 10s:

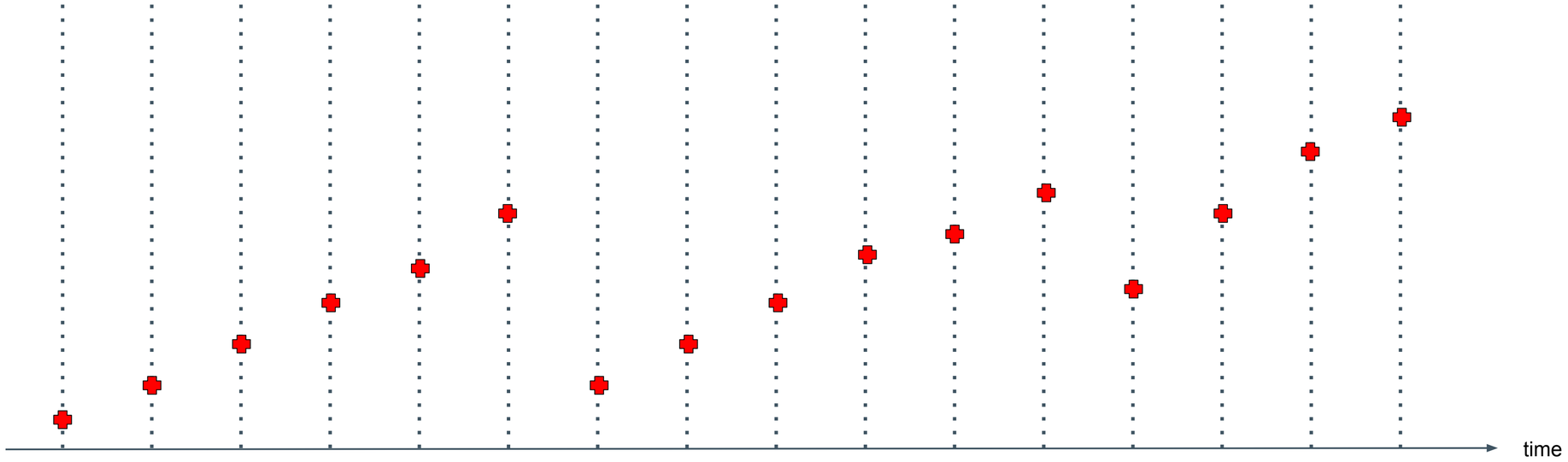
```
rate(alerts_total[1m])
```

```
labels:
  __name__ = "alerts_total"
time:
  start: t0
  end: t1
read hints:
  func: "rate"
  ...
```

✔ Aggregation Selection

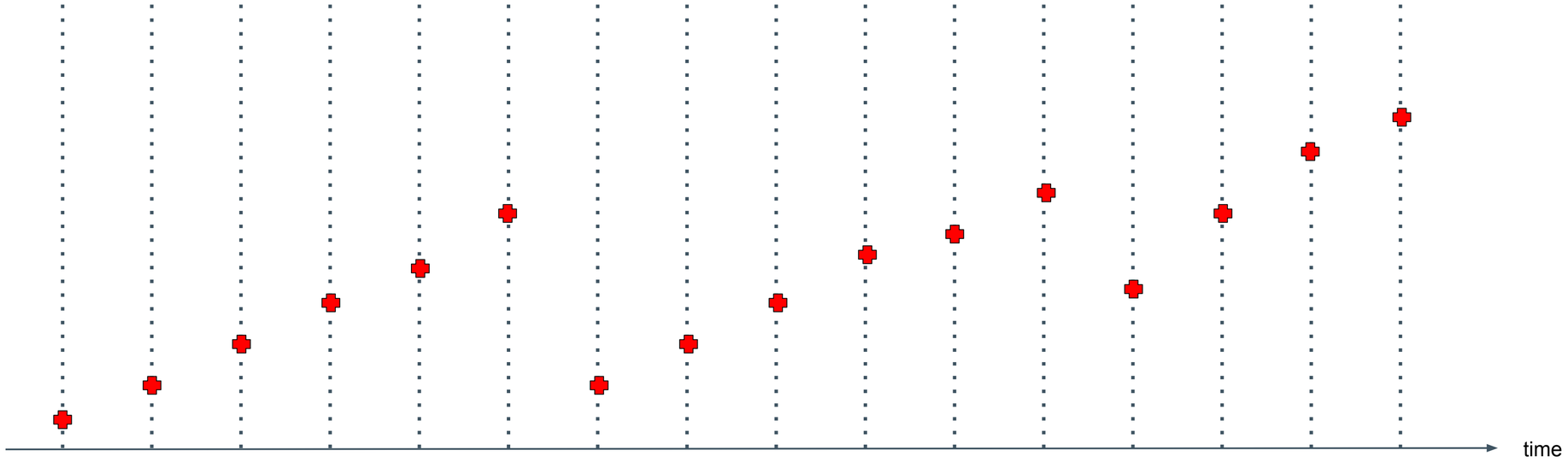
3. What does "rate" mean?


0
DAYS
SINCE LAST
ACCIDENT
1
DAYS
SINCE LAST
ACCIDENT
0
DAYS
SINCE LAST
ACCIDENT

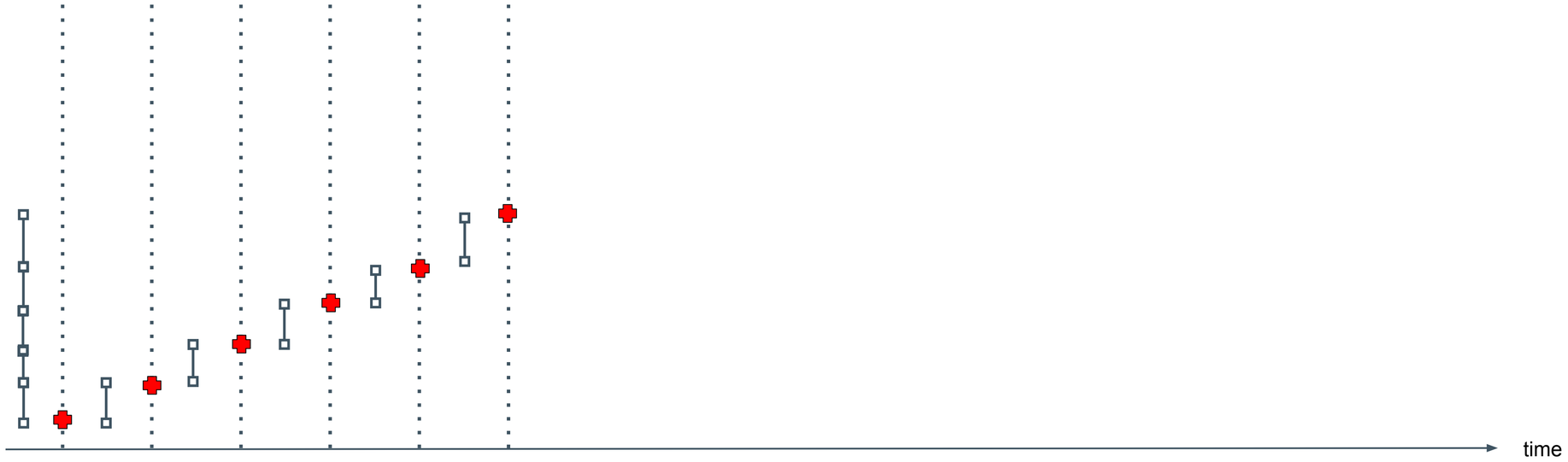


A decrease in value indicates a reset occurred.

A common reason for a reset is a restarted instance.

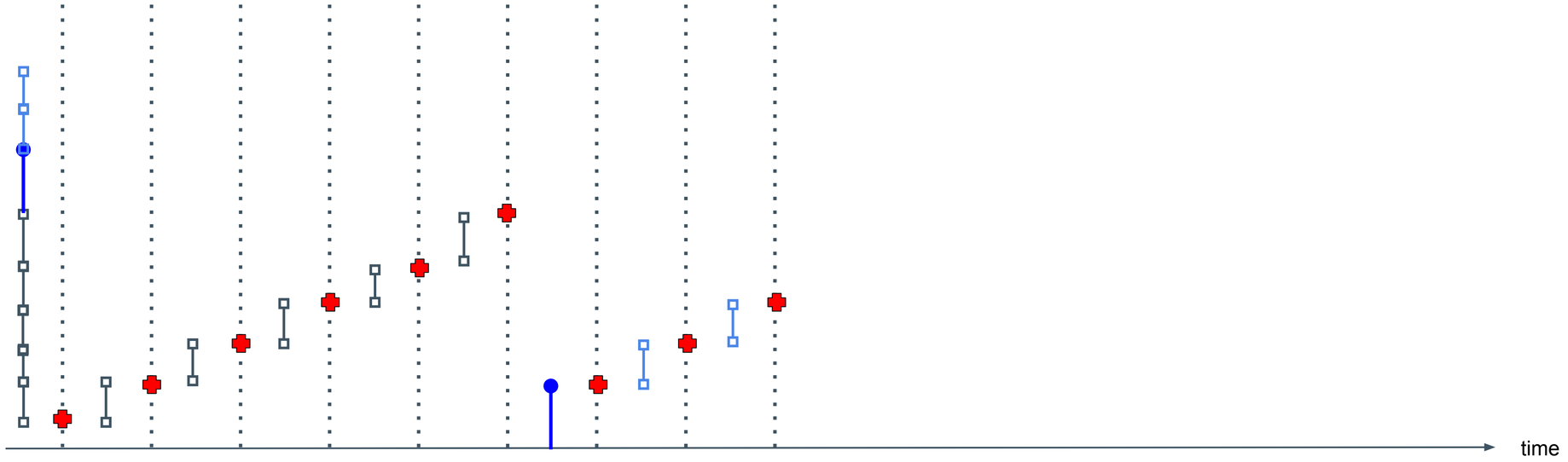


`rate()`: divide the difference in events by a time duration.

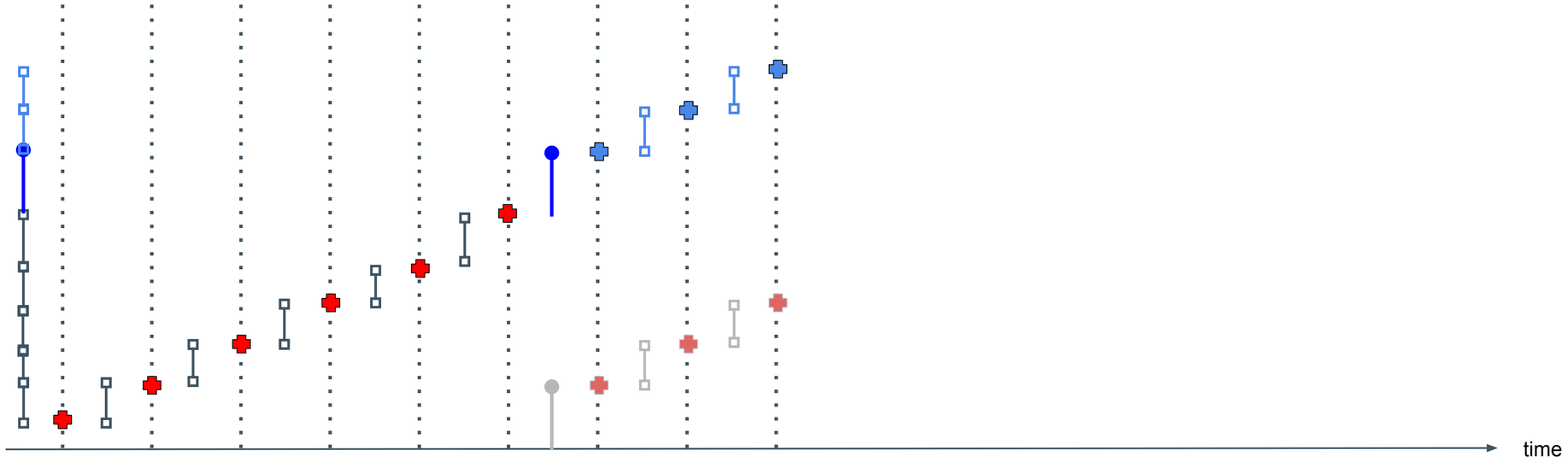


No resets?

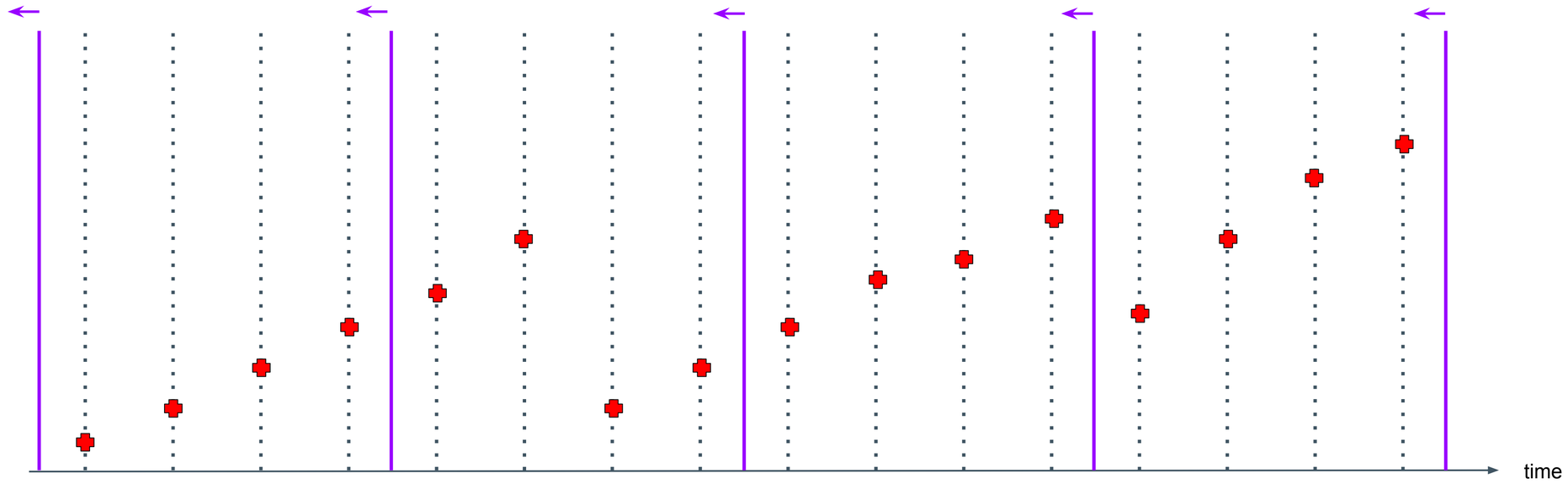
Sum the deltas between samples.



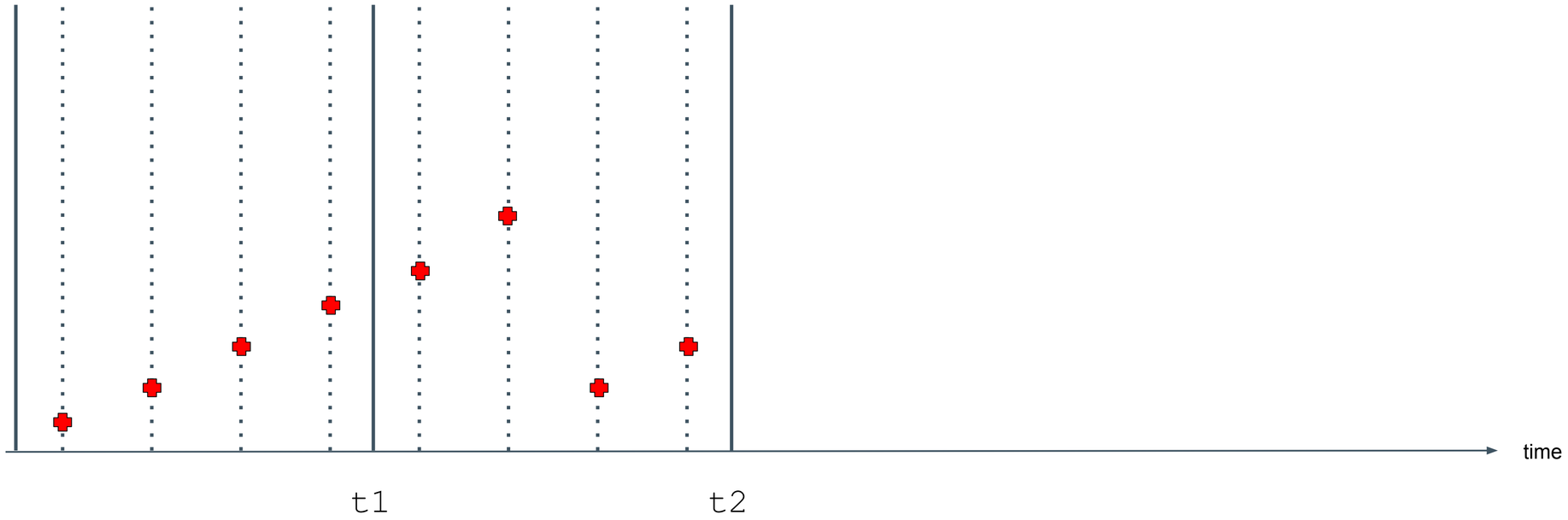
Reset? Add post-reset value.



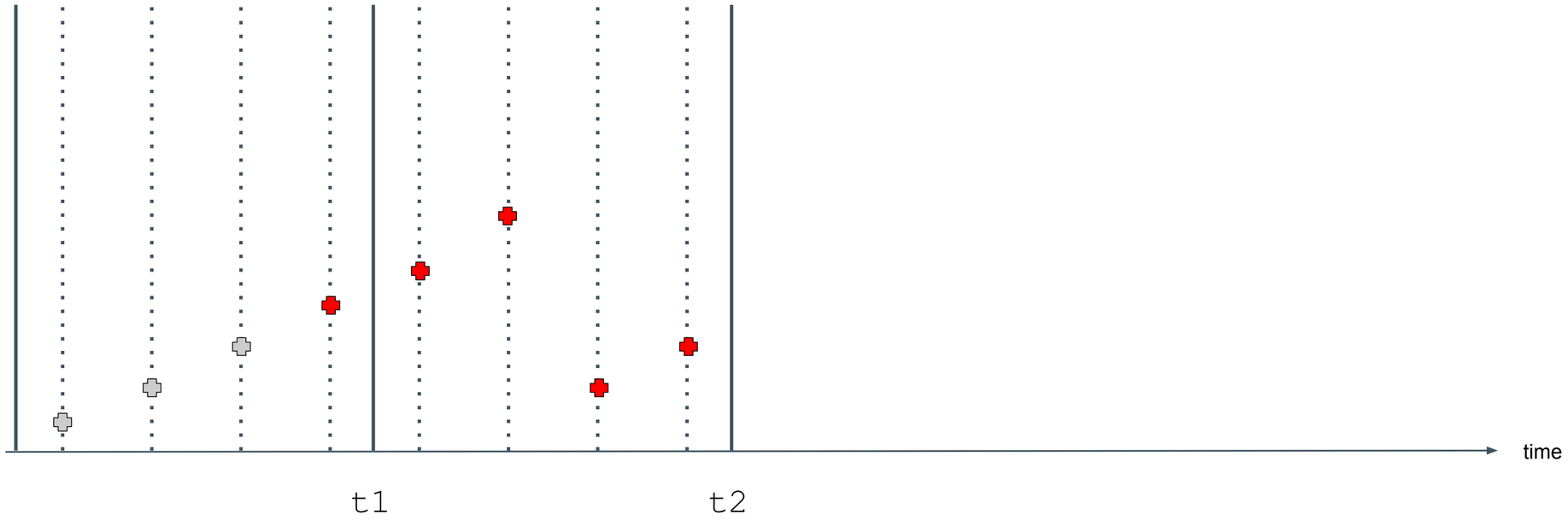
effectively: slide everything up after each reset.



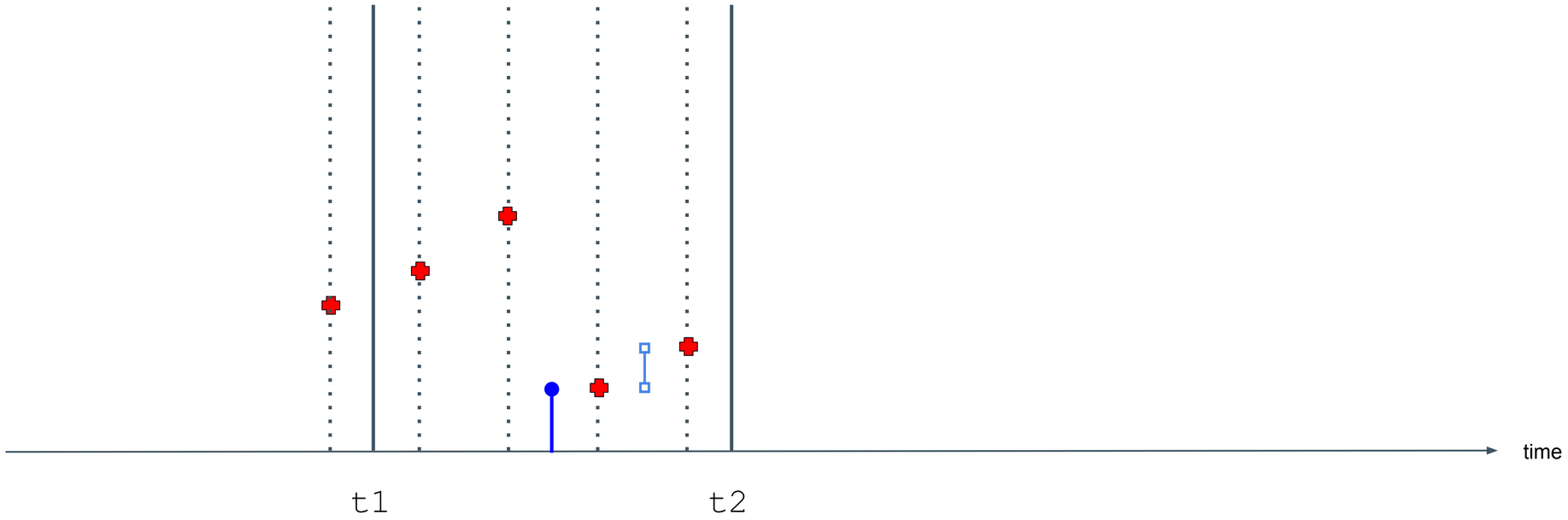
What kind of aggregation would you need for rate?



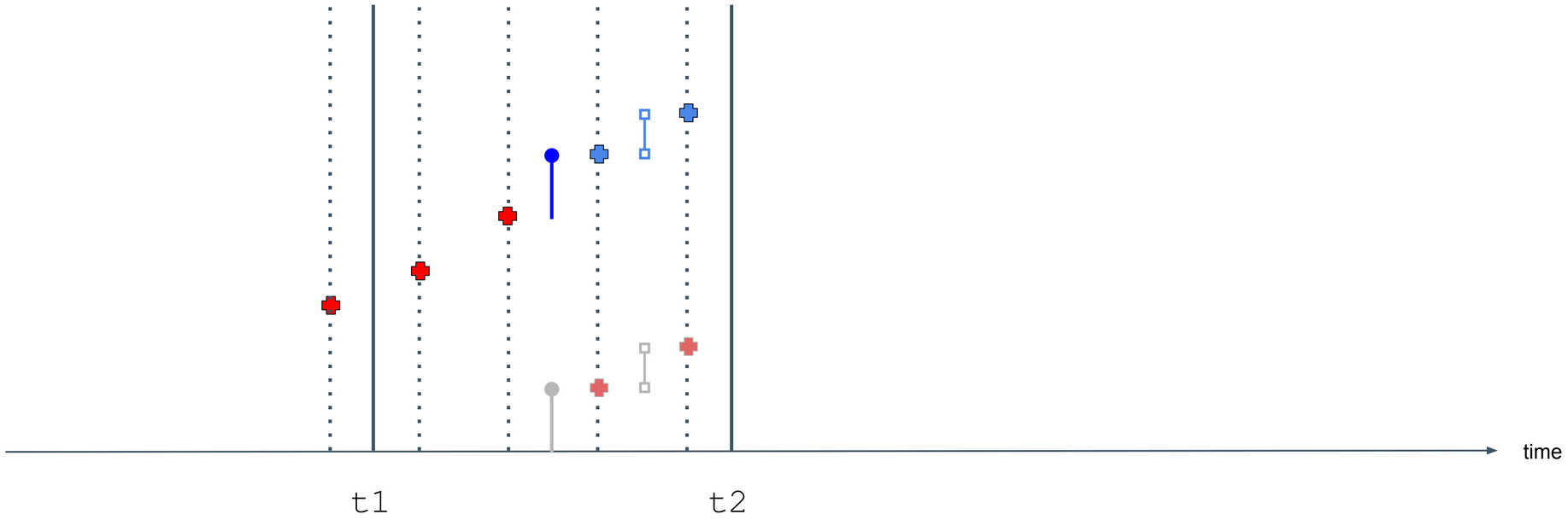
How many events occurred between *t1* and *t2*?



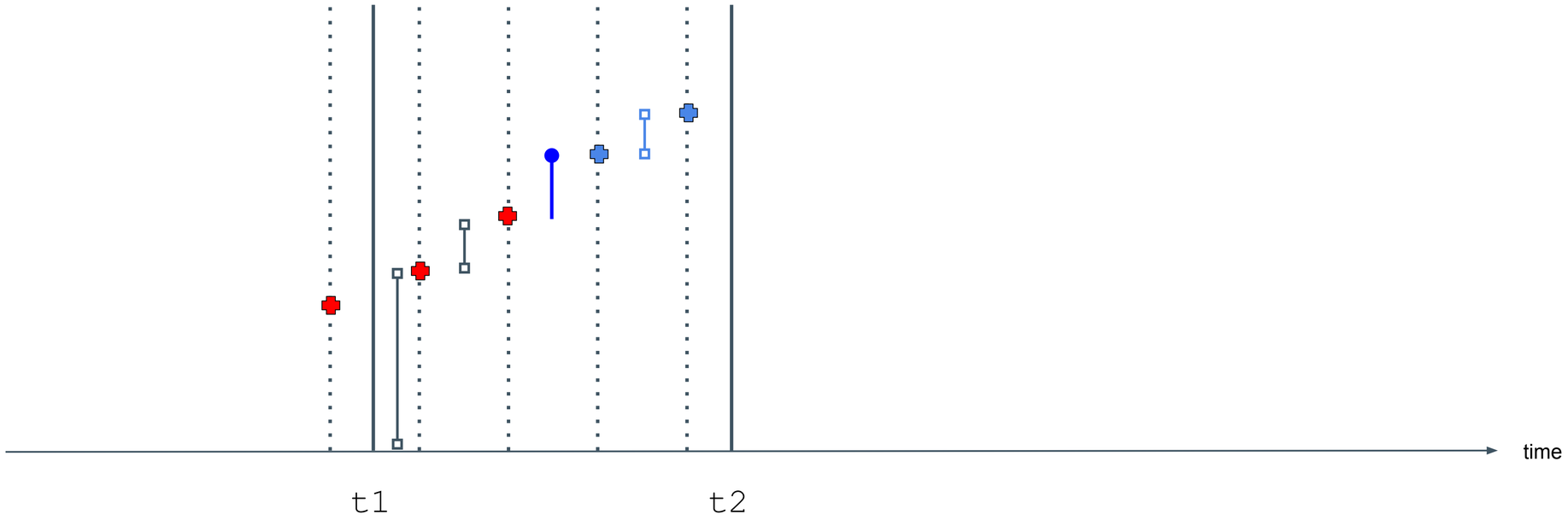
How many events occurred between *t1* and *t2*?



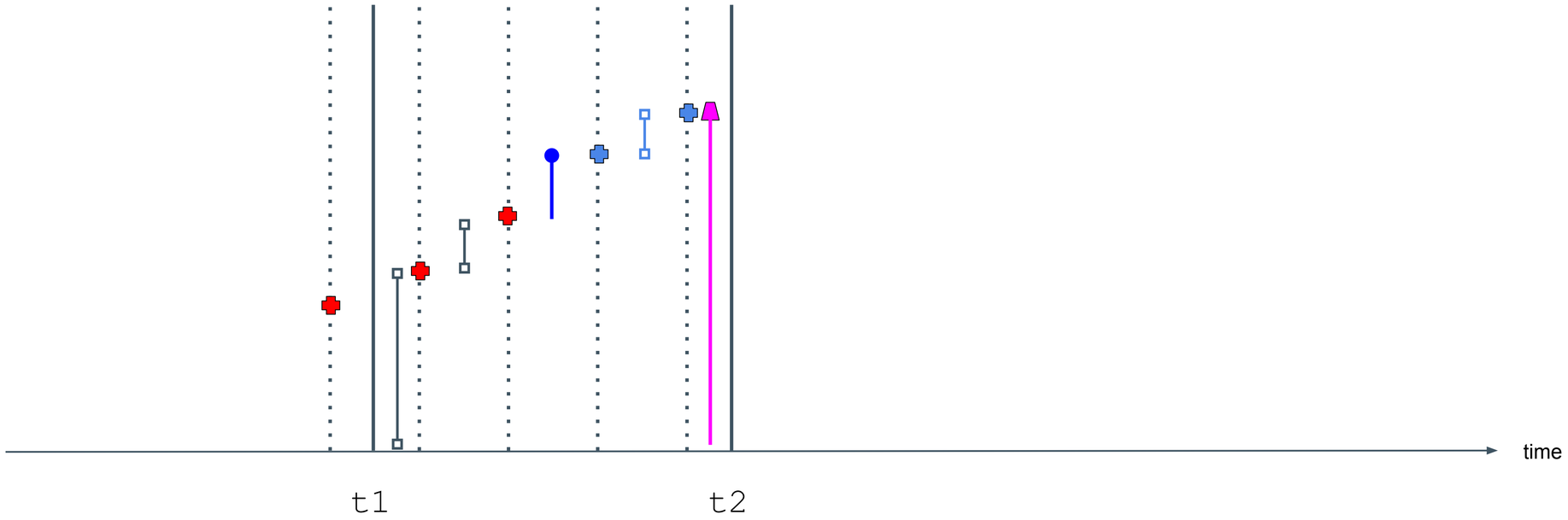
How many events occurred between *t1* and *t2*?



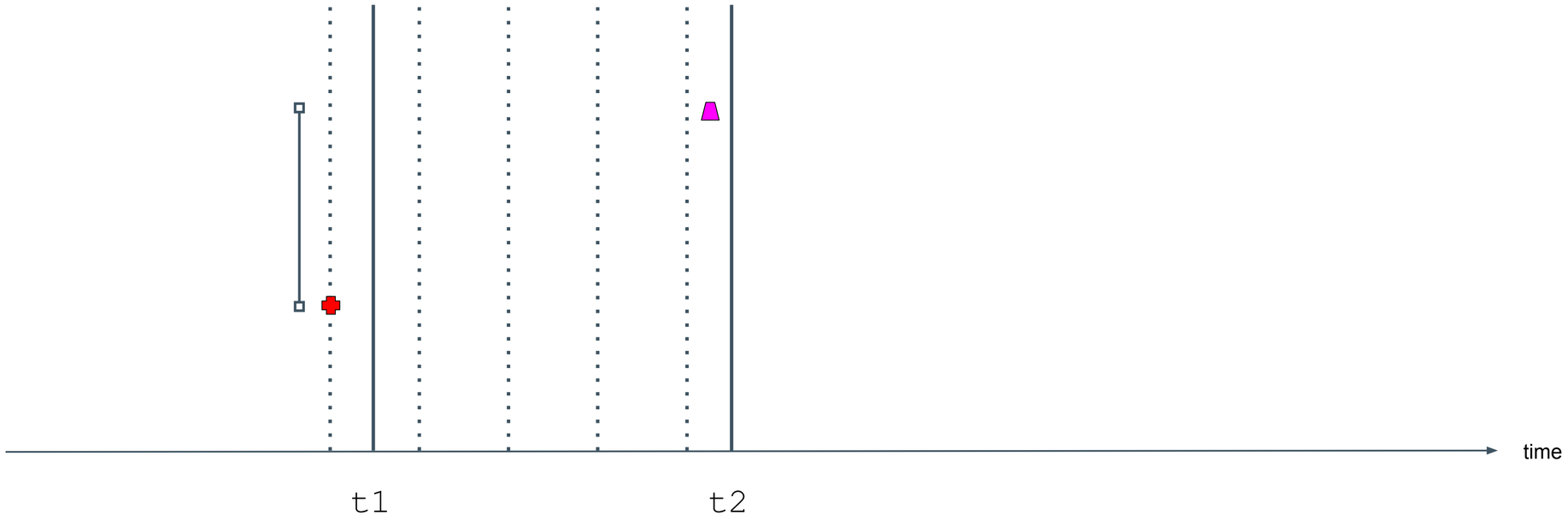
How many events occurred between *t1* and *t2*?



How many events occurred between *t1* and *t2*?



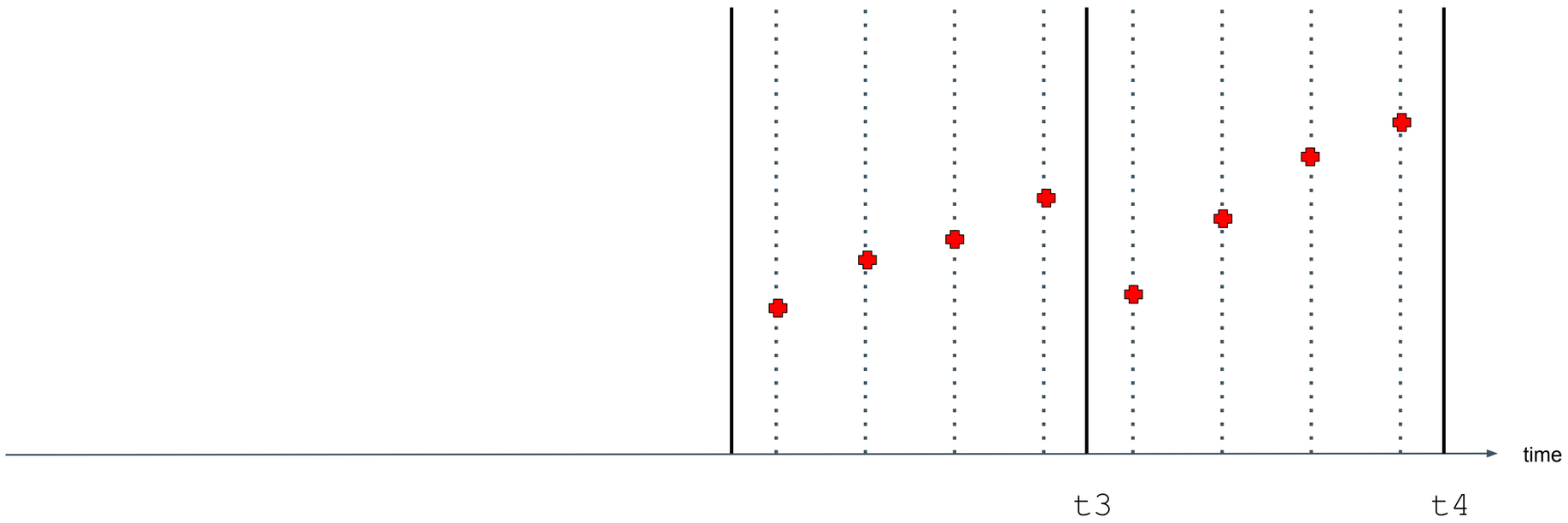
How many events occurred between *t1* and *t2*?



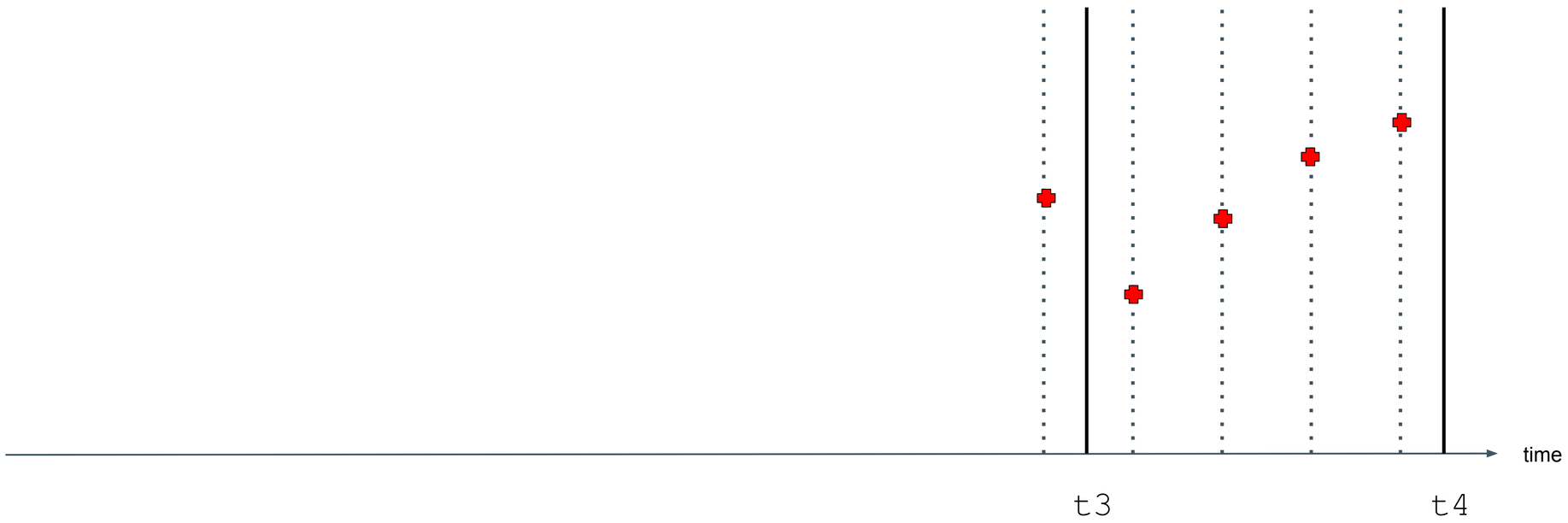
How many events occurred between *t1* and *t2*?

In this case, the difference between:

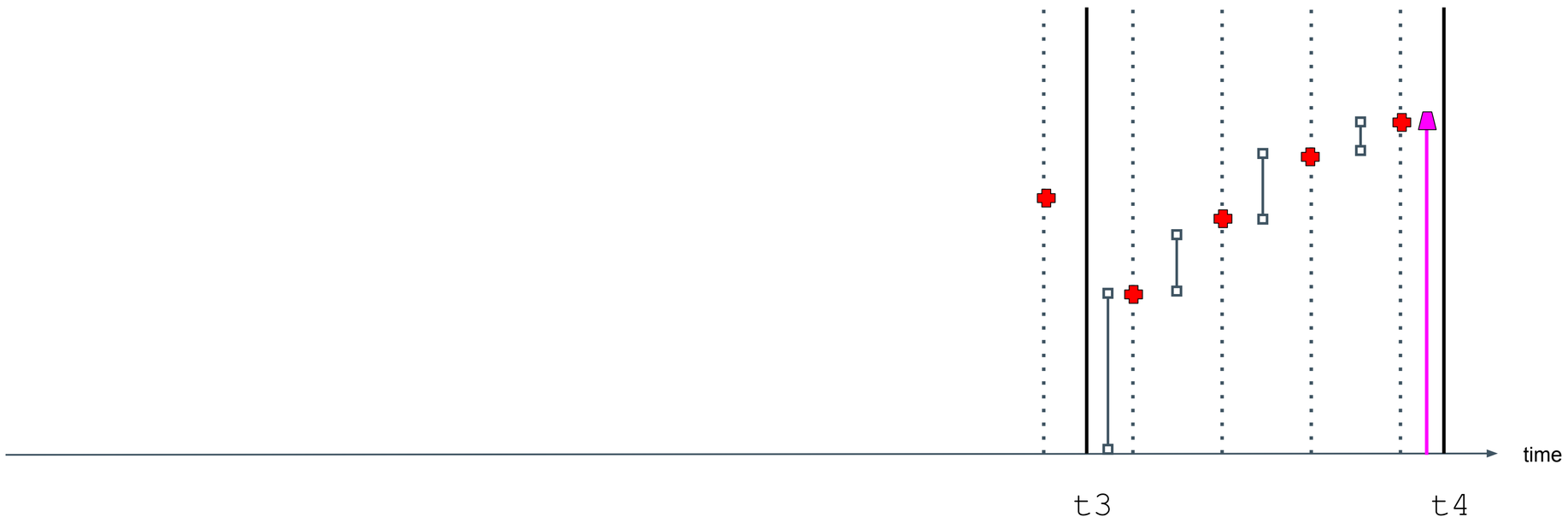
- the sum of events in t2 window
- the last value before t1



How many events occurred between *t3* and *t4*?

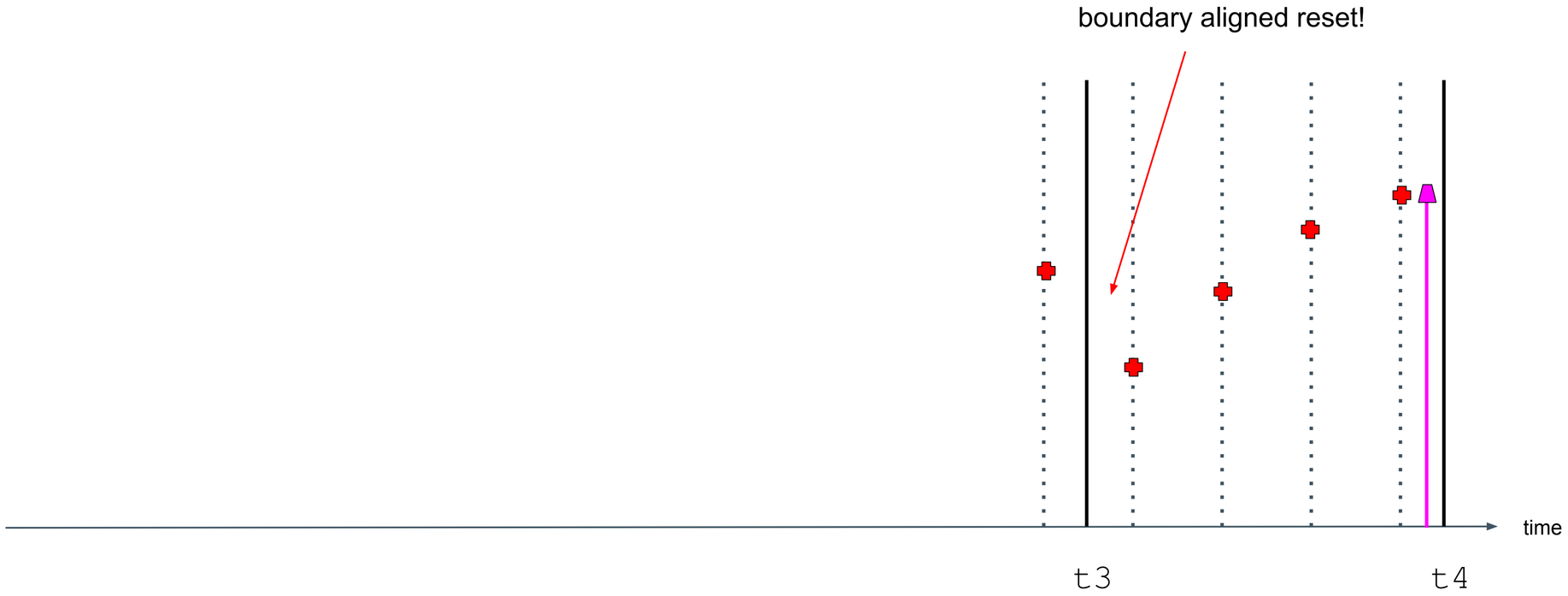


How many events occurred between *t3* and *t4*?



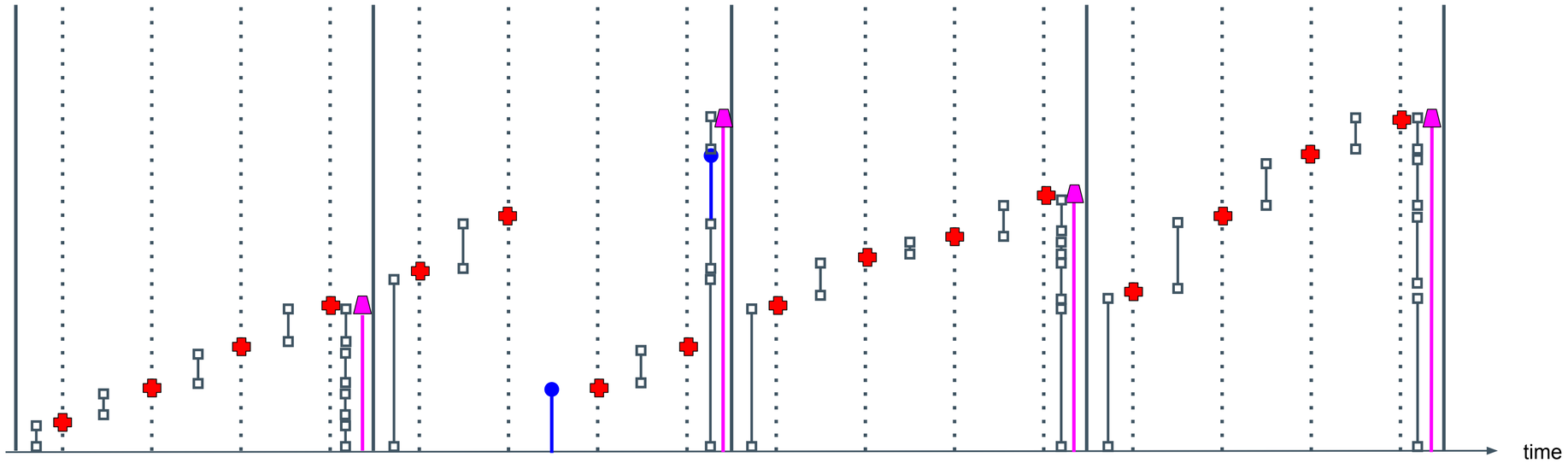
How many events occurred between *t3* and *t4*?

Can we just take the difference between the last raw & the sum?

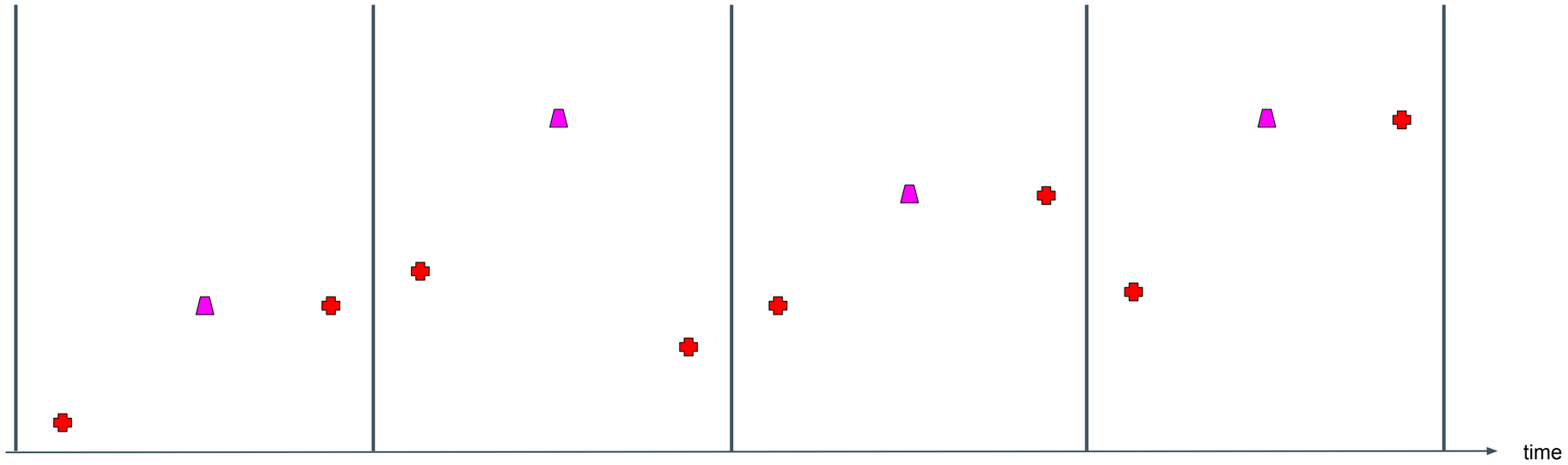


No: a border reset means values in t3 don’t matter: just the t4 event sum.

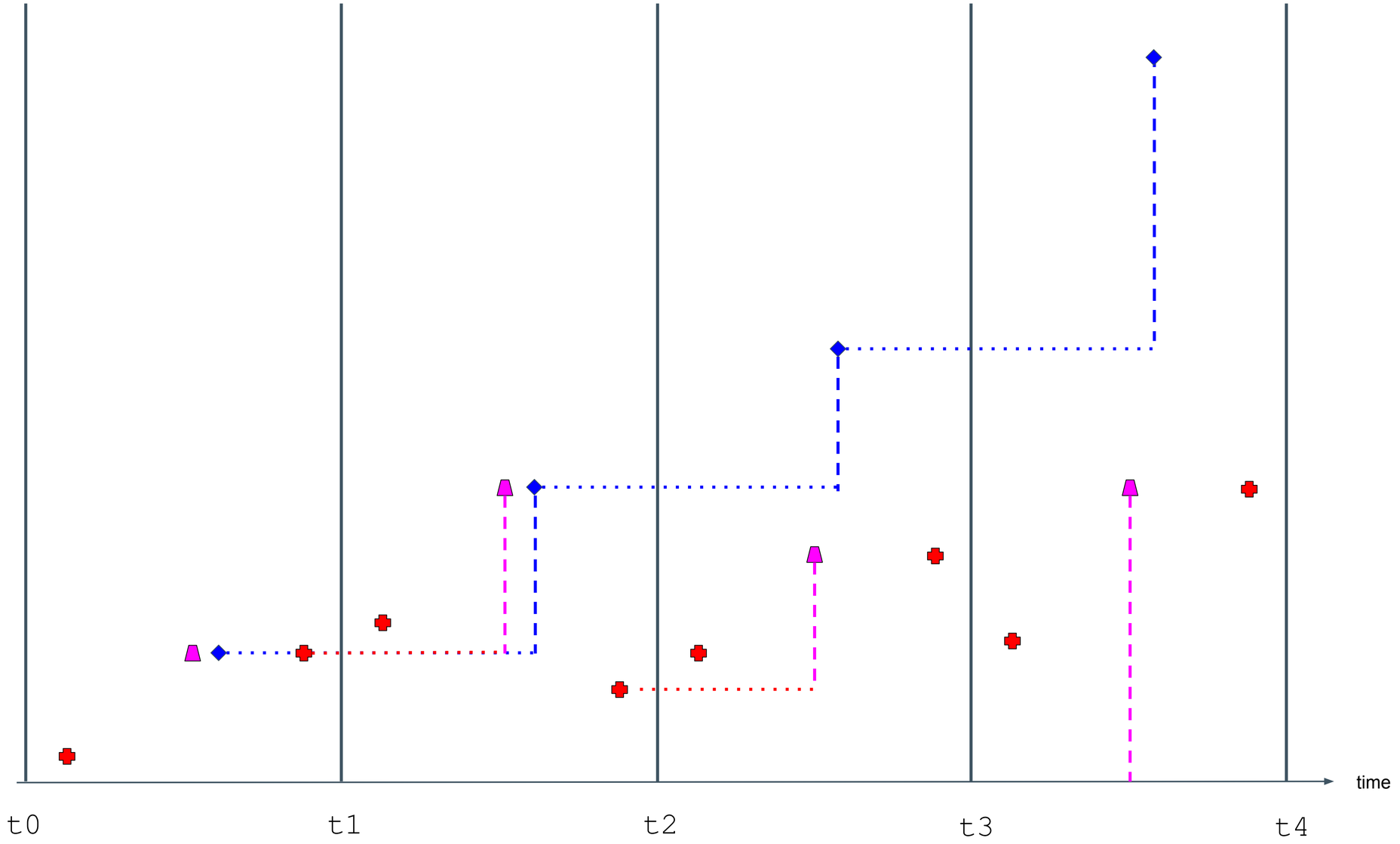
Need to store the **first** and **last** raw values to detect border resets.



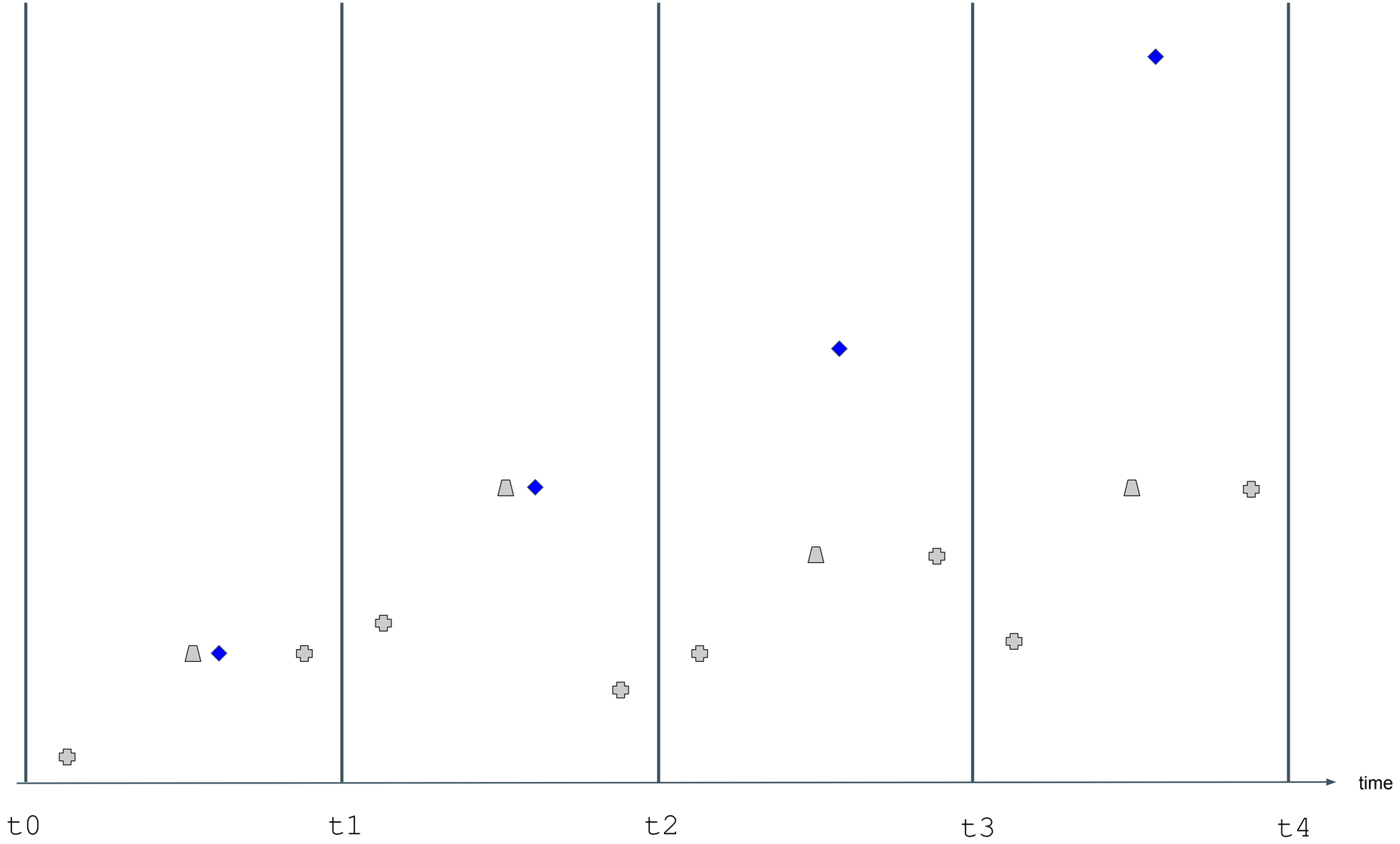
Store first, last raw values, and sum in events.



How to turn this into a response for PromQL remote read?



Generate a response sequence for each query.



PromQL sees a monotonically increasing value, with no resets.

range query from t0 to t1, step 10s:

```
up{env="prod"} > 1
```

```
labels:
  __name__ = "up"
  env = "prod"
time:
  start: (t0 - 5m)
  end: t1
...
```

✔ Sample Alignment

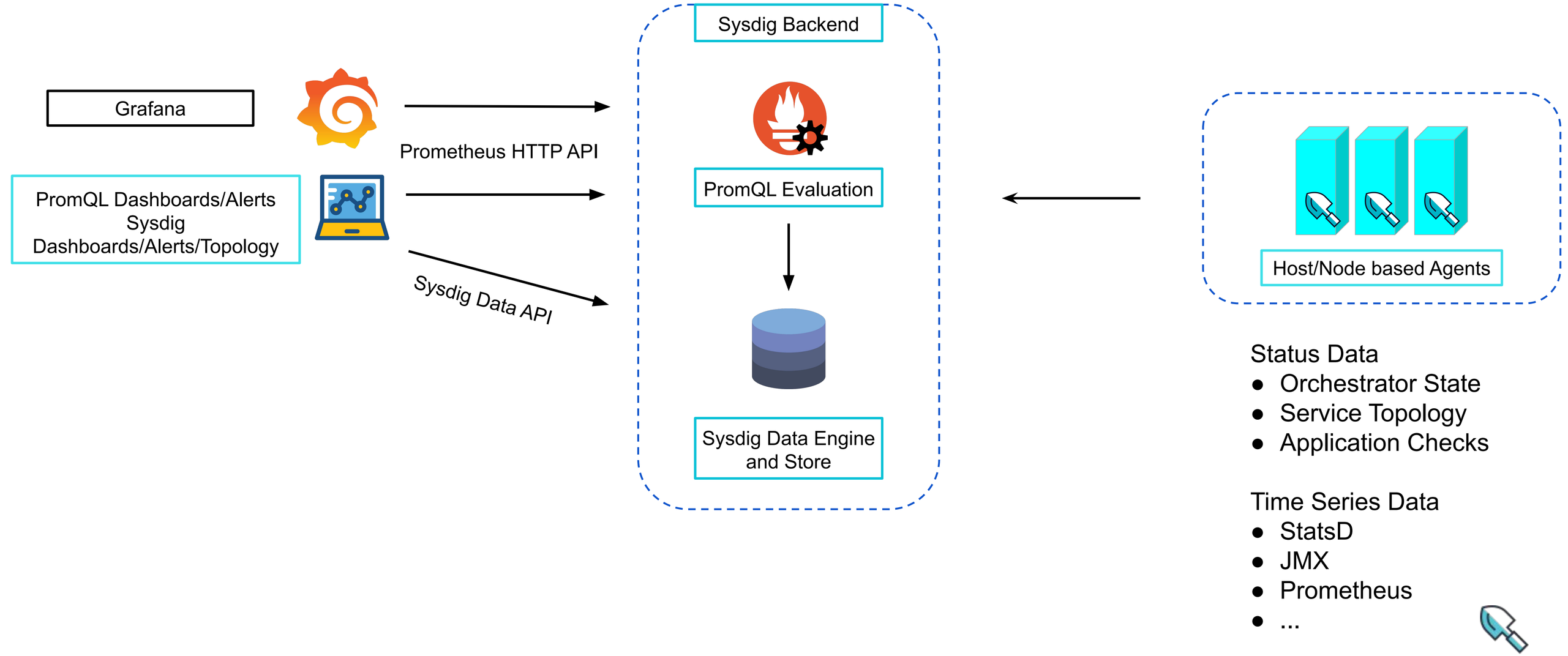
range query from t0 to t1, step 10s:

```
rate(alerts_total[1m])
```

```
labels:
  __name__ = "alerts_total"
time:
  start: t0
  end: t1
read hints:
  func: "rate"
  ...
```

✔ Aggregation Selection

✔ Counter Downsampling



Status Data

- Orchestrator State
- Service Topology
- Application Checks

Time Series Data

- StatsD
- JMX
- Prometheus
- ...

Thanks!