

PromQL 查詢語言介紹



以繁體中文詳細說明:PromQL



當然可以！以下是 **PromQL (Prometheus Query Language)** 的繁體中文詳細說明，將從基本概念到常見用法循序漸進說明。

PromQL 是什麼？

PromQL (Prometheus Query Language) 是 Prometheus 所使用的查詢語言，用來從資料庫中提取、計算與聚合監控指標 (metrics)。
它支援強大的過濾、聚合與函數操作功能，可用於：

- 查詢即時監控資料
- 計算時間序列的統計數據（如平均、總和、最大、最小）
- 設定警示規則 (Alert Rules)
- 顯示在 Grafana 的圖表上

資料模型簡介

Prometheus 的資料是「**多維度的時間序列資料**」，以下列形式儲存：

- **metric 名稱**：例如 `http_requests_total`



- **label 標籤**：例如 ``instance="10.0.0.1:9100"`, `job="api-server"``

範例：

```
http_requests_total{job="api", instance="10.0.0.1:9090", status="200"}
```

這條表示：

- metric 名稱是 ``http_requests_total``
- 此資料點發生在 ``job=api``、``instance=10.0.0.1:9090``、``status=200`` 的情境下

PromQL 查詢語法基本結構

PromQL 中查詢可以分為兩大類型：

類型	說明	範例
瞬時查詢 (Instant Vector)	查詢某個時間點的所有指標數值	<code>`up`</code>
區間查詢 (Range Vector)	查詢一段時間內的數值變化	<code>`rate(http_requests_total[5m])`</code>

PromQL 常用查詢範例

1 查詢某個 Metric 的所有值

```
up
```

查詢 ``up`` 指標的所有實例（代表 target 是否正常）

2 篩選特定 Label 的值

```
http_requests_total{job="frontend", status="500"}
```

查詢 job 為 frontend 且 HTTP status 為 500 的請求總數

3 計算過去 5 分鐘的平均速率 (Rate)

```
rate(http_requests_total[5m])
```

計算每秒的 HTTP 請求速率 (基於過去 5 分鐘)

4 聚合：統計多個實例的總和

```
sum(rate(http_requests_total[5m])) by (job)
```

對每個 job，計算 HTTP 請求的總速率

5 聚合：計算每個 job 的最大 CPU 使用率

```
max(rate(process_cpu_seconds_total[1m])) by (job)
```

6 計算百分比 / 失敗率

```
100 * (1 - avg(rate(up[5m])) by (job))
```

計算每個 job 的 target 失敗率百分比

PromQL 函數分類（常見）

分類	常見函數	說明
速率函數	<code>`rate()`</code> 、 <code>`irate()`</code> 、 <code>`increase()`</code>	用於計數型 metric 的速率計算
統計函數	<code>`sum()`</code> 、 <code>`avg()`</code> 、 <code>`min()`</code> 、 <code>`max()`</code> 、 <code>`count()`</code> 、 <code>`stddev()`</code>	對時間序列進行統計
比較與過濾	<code>`topk(k, metric)`</code> 、 <code>`bottomk(k, metric)`</code> 、 <code>`count_values()`</code>	排名前 N 名，統計特定值出現次數
時間處理	<code>`delta()`</code> 、 <code>`deriv()`</code> 、 <code>`predict_linear()`</code>	用於觀察變化趨勢或預測
邏輯判斷	<code>`abs()`</code> 、 <code>`clamp_max()`</code> 、 <code>`clamp_min()`</code>	數學處理，控制最大/最小值

PromQL 實戰範例

範例：顯示 CPU 使用率圖表（某主機）

```
100 - avg(irate(node_cpu_seconds_total{mode="idle", instance="host1:9100"}  
[5m])) * 100
```

說明：

- ``node_cpu_seconds_total{mode="idle"}``：表示 CPU 空閒時間
- ``irate(...)``：取即時速率
- ``100 - avg(...) * 100``：代表 CPU 使用率

警示使用 PromQL 範例

在 Prometheus 警示規則中你可以這樣使用 PromQL：

```
- alert: HighCPUUsage
  expr: avg(rate(process_cpu_seconds_total[1m])) by (instance) > 0.8
  for: 2m
  labels:
    severity: critical
  annotations:
    summary: "Instance {{ $labels.instance }} CPU 使用率過高"
```

小技巧與注意事項

1. 使用 ``[5m]`` 表示查詢過去 5 分鐘資料 (Range Vector)
2. ``by (label)`` 用於聚合 (保留 label)
3. 可搭配 ``offset`` 查詢過去特定時間點 (如 ``rate(metric[5m] offset 1h)``)
4. 支援正則：``job=~"api-.*"`` 表示 job 名稱符合 regex

如果你有特定的 PromQL 查詢需求、或想針對某個 dashboard 寫查詢語句，也可以提供範例，我可以幫你撰寫或解釋查詢語句！



以繁體中文詳細說明:使用 [5m] 表示查詢過去 5 分鐘資料 (Range Vector)，跟prometheus的time區間的差別



很棒的問題！`[5m]` 這種語法是 **PromQL** 中非常常見且重要的一部分，我會以下列順序為你詳細說明：

✓ `[5m]` 是什麼？(Range Vector)

在 PromQL 中，像 `rate(http_requests_total[5m])` 這樣的 `[5m]` 表示 **Range Vector** (區間向量)，意思是：

“「對某個指標，從查詢時間點往前推 5 分鐘，取出這段時間內所有的資料點。」”

範例

```
rate(http_requests_total[5m])
```

這句話表示：

- 查詢 `http_requests_total` 這個計數型 metric，
- 在當下時間點「往前 5 分鐘」的資料，
- 並計算每秒的平均變化率 (rate)。



與 Prometheus 查詢「time」區間的差別？

Prometheus 有兩種查詢方式會牽涉到時間設定，容易混淆：

類型	使用情境	範例	意義
<code>`[5m]`</code> (Range Vector)	PromQL 語法內部	<code>`rate(metric[5m])`</code>	取「單一時間點」往前的資料，用來進行計算
<code>`start`, `end`, `step`</code> (HTTP API 或圖表工具)	外部查詢時間範圍	<code>`/api/v1/query_range?start=...&end=...&step=30s`</code>	指定「查詢整段時間」，每隔一步驟取一次資料

``[5m]`` 的關鍵特點：

項目	說明
資料量	只針對「當前時間點」往前 5 分鐘的資料
回傳型態	Range Vector：每個時間序列會包含一段時間的多個點
適用場景	用於 <code>`rate()`</code> 、 <code>`increase()`</code> 、 <code>`avg_over_time()`</code> 等函數計算
不會影響查詢時間線	是 相對時間範圍 ，不會查出多筆時間點資料，只針對單一查詢時間計算
例子	<code>`increase(metric[1h])`</code> 、 <code>`avg_over_time(metric[30m])`</code>

``query_range`` 時的 ``start`` ~ ``end`` 是什麼？

當你用 Grafana 或 HTTP API 查詢時（例如 `/api/v1/query_range``），你可以設定：

- ``start``：查詢起始時間
- ``end``：查詢結束時間
- ``step``：每隔幾秒取一筆（以瞬時查詢為主）

這個查詢會回傳 **多個時間點的值**，每筆都會執行一樣的 PromQL 查詢。

✓ 整理：`[5m]` VS `query_range``

項目	<code>[5m]</code> (Range Vector)	<code>`start ~ end`</code> (query_range)
查詢位置	PromQL 語法內部	HTTP API / Grafana 查詢參數
用途	提供函數計算用的「一段區間」資料	用來「畫圖」或查詢「一段時間」的多個資料點
回傳資料型態	Range Vector（時間序列＋時間段）	多筆瞬時向量（每筆是 Instant Vector）
資料點數	每條序列回傳一段資料	每隔 <code>`step`</code> 秒回傳一筆資料，共多筆
典型用途	<code>`rate(metric[5m])`</code> 、 <code>`increase(...)`</code>	Grafana、API 查詢圖表
查詢時間點	「一個時間點」（通常是現在）	「整段時間」

✓ 實務對照

✓ 使用 `[5m]`：

```
rate(http_requests_total[5m])
```


🔍 表示現在時間（例如 17:00）往前 5 分鐘（16:55 ~ 17:00）的資料拿來計算速率。

✅ 使用 `query_range` 時：

```
/api/v1/query_range?  
query=rate(http_requests_total[5m])&start=17:00&end=18:00&step=30s
```

🔍 表示從 17:00 到 18:00，每 30 秒計算一次 `rate(...)`，而每次都會用那個時間點往前 5 分鐘的資料進行計算。

💬 小結一句話

“✅ `[5m]` 是你「每一筆 PromQL 查詢」用來回顧過去資料的區間，而 `start~end` 是你「整個圖表查詢」的時間範圍。”

如果你有 PromQL 寫法或 Grafana 查圖疑問，也歡迎提供具體例子，我可以幫你調整查詢語句！



我想要理解`rate()`、`irate()`、`increase()`、`sum()`、`avg()`、`min()`、`max()`、`count()`、`stddev()`用法，為了用Grafana做VictoriaMetric- Cluster version的Capacity planning實驗，你能教我嗎，或是你能給更簡單的範例：



太好了！你提到的這些函數都是進行 **VictoriaMetrics Cluster 容量規劃（Capacity Planning）** 時很關鍵的 PromQL 工具。

下面我會用**簡化範例** + **圖表用途說明** + **Capacity Planning 實例**，幫你快速掌握它們的用途與差異。

資料前提

假設我們有一個 Prometheus/VictoriaMetrics 指標：

```
vm_ingest_rows_total{instance="vm1", job="vminsert"}
```

這是個「**計數型 (counter)**」指標，代表某個 vminsert 實例寫入的總行數 (Row 數)。

基礎計算函數（三個最常用）

1 `rate(metric[時間])`

計算 **平均速率**（每秒變化量），用於 `counter` 指標
適合看長期趨勢，平滑數據

```
rate(vm_ingest_rows_total[5m])
```

 解釋：5 分鐘內平均每秒 ingest 幾筆 row。

 Grafana 用途：

- 畫出每秒 ingest rate 的折線圖，觀察高峰負載。

2 `irate(metric[時間])`

計算「**最即時的一次變化率**」，用於 `counter`，比 `rate()` 靈敏
適合短期高波動變化偵測

```
irate(vm_ingest_rows_total[1m])
```

 解釋：只看最後兩筆樣本，算出目前每秒 ingest 速率。

 Grafana 用途：

- 快速警報（瞬間爆量 ingestion）

3 `increase(metric[時間])`

計算某段時間內 **總增加值**（不是速率！）

```
increase(vm_ingest_rows_total[10m])
```

 解釋：過去 10 分鐘總共 ingest 幾筆 row。

 Grafana 用途：

- 評估 10 分鐘內 ingestion 整體量（可搭配 CPU/Memory 做推估）

聚合與統計函數

這些函數可以與 `rate()`、`increase()` 搭配使用。

4 `sum(...)`

對多個時間序列 **加總**

```
sum(rate(vm_ingest_rows_total[5m])) by (job)
```

 Grafana 用途：

- 對所有 `vminsert` 節點的 row ingest rate 求總和 (Cluster ingest rate)

5 `avg(...)`

平均值

```
avg(rate(vm_ingest_rows_total[5m])) by (instance)
```

 Grafana 用途：

- 各個 instance 的平均寫入速率

6 `min(...)`

最小值 (可用來找表現最差節點)

```
min(rate(vm_ingest_rows_total[5m])) by (instance)
```

7 `max(...)`

最大值 (可用來偵測爆量寫入)

```
max(rate(vm_ingest_rows_total[5m])) by (instance)
```

8 `count(...)`

有幾條序列參與計算（可監控服務存活）

```
count(rate(vm_ingest_rows_total[5m])) by (job)
```

9 `stddev(...)`

計算標準差，了解 **分散程度**（常用於容量波動分析）

```
stddev(rate(vm_ingest_rows_total[5m])) by (job)
```

 Grafana 用途：

- 分析 ingest rate 是否穩定，越大代表波動越劇烈

Capacity Planning 實戰範例（用 Grafana + VictoriaMetrics）

 目的：預估 ingest rate、節點負載、儲存用量

◆ 1. 每秒總 ingest row 數

```
sum(rate(vm_ingest_rows_total[5m])) by (job)
```

◆ 2. 每個節點的 ingest load 分佈

```
rate(vm_ingest_rows_total[5m]) by (instance)
```

◆ 3. 節點寫入速率的標準差

```
stddev(rate(vm_ingest_rows_total[5m])) by (job)
```

◆ 4. 每 10 分鐘 ingestion 量

```
increase(vm_ingest_rows_total[10m]) by (job)
```

◆ 5. cluster 寫入爆點 (max)

```
max(rate(vm_ingest_rows_total[5m])) by (instance)
```

小技巧

情境	建議用法
長期趨勢	<code>`rate(...)`</code>
瞬間變化	<code>`irate(...)`</code>
累積量	<code>`increase(...)`</code>
cluster 統計	搭配 <code>`sum()`</code> / <code>`avg()`</code>
檢查負載平衡	<code>`max()`</code> / <code>`min()`</code> / <code>`stddev()`</code>

如果你有具體的 metric（像是 ``vmstorage_data_size_bytes``、``vm_cache_entries`` 等）或想設計某個容量預測 dashboard，我也可以直接幫你規劃查詢語句與圖表欄位喔！是否要幫你設計一個完整的「VictoriaMetrics Cluster 容量分析」面板範例？



我想要理解by (job)、by (instance)用法，為了用Grafana做 VictoriaMetric- Cluster version的Capacity planning實驗，你能教我嗎，或是你能給更簡單的範例：



當然可以！你提到的 ``by (job)``、``by (instance)`` 是 **PromQL** 中的 **label 分組操作符**，在做 Capacity Planning 時非常關鍵，尤其是用在 **VictoriaMetrics Cluster**（如 ``vminsert``、``vmselect``、``vmstorage``）時，能幫你比較不同節點或 job 的負載狀況。

``by (...)`` 是什麼？

在 PromQL 中，當你使用 ``sum()``、``avg()``、``max()`` 等聚合函數時：

“``by (label)`` 表示「依據某個 **Label**（標籤）群組來計算」”

簡單範例對比

假設有以下指標（表示每秒 ingest 行數）：

```
vm_ingest_rows_total{job="vminsert", instance="vm1:8480"}  
vm_ingest_rows_total{job="vminsert", instance="vm2:8480"}  
vm_ingest_rows_total{job="vminsert", instance="vm3:8480"}
```



✨ 範例 1：不加 `by`（整體計算）

```
sum(rate(vm_ingest_rows_total[5m]))
```

🔍 代表「**整個 cluster 的總 ingest rate**（不分節點）」

✨ 範例 2：加上 `by (instance)`

```
sum(rate(vm_ingest_rows_total[5m])) by (instance)
```

🔍 代表「**每個節點的 ingest rate**」，用來觀察負載是否均衡

📊 Grafana 用途：畫每個 instance 的時間序列折線圖

✨ 範例 3：加上 `by (job)`

```
sum(rate(vm_ingest_rows_total[5m])) by (job)
```

🔍 代表「**以 job 為單位聚合的 ingest rate**」

（例如你有 `vminsert`、`vmagent`、`vmstorage` 等 job）

📊 Grafana 用途：畫出各 job 的總處理量，對應模組層級容量分析

🧠 說明：`job` VS `instance`

標籤	含義	典型值	用途
`job`	任務名稱	`vminsert`, `vmselect`, `vmstorage`	分類整個服務
`instance`	節點位址	`10.0.0.1:8480`, `vm1:8480`	區分每個 pod 或實體主機

VictoriaMetrics Cluster 實用查詢語法

✓ 每個 `vminsert` 實例的寫入量

```
rate(vm_ingest_rows_total{job="vminsert"}[5m]) by (instance)
```

✓ 所有 `vmstorage` 實例的總資料大小（假設是 Gauge）

```
sum(vm_data_size_bytes{job="vmstorage"}) by (instance)
```

✓ 各 job 的節點數（計數用）

```
count(up) by (job)
```

✓ 每個 job 的負載分布（平均 CPU 使用）

```
avg(rate(process_cpu_seconds_total[1m])) by (job)
```

Grafana Capacity Planning 面板設計建議

圖表名稱	PromQL 查詢	解釋
Cluster 總 ingest rate	<code>`sum(rate(vm_ingest_rows_total[5m]))`</code>	查看總負載趨勢
每個節點 ingest rate	<code>`rate(vm_ingest_rows_total[5m]) by (instance)`</code>	節點負載均衡觀察
每個 job 寫入大小	<code>`sum(vm_data_size_bytes) by (job)`</code>	分 job 看儲存用量
CPU 使用率 by instance	<code>`avg(rate(process_cpu_seconds_total[1m])) by (instance)`</code>	預估資源瓶頸
標準差（穩定性）	<code>`stddev(rate(vm_ingest_rows_total[5m])) by (job)`</code>	波動性分析

小結

你想比較什麼？	用法
各節點負載	<code>`by (instance)`</code>
各服務模組	<code>`by (job)`</code>
統計整體	不加 <code>`by`</code>

如果你有具體的 ``metric name`` 或 Grafana panel 想設計，我可以幫你列出查詢語句、分組邏輯與建議圖表欄位，是否需要我幫你規劃出一整個 Capacity Dashboard 的面板設計清單？



我想要理解折線圖、長條圖、stat圖用法，為了用Grafana做VictoriaMetric- Cluster version的Capacity planning實驗，你能教我嗎，或是你能給更簡單的範例：



當然可以！以下是為你量身打造的「**Grafana 常用圖表元件（折線圖、長條圖、Stat 圖）用法教學**」，並搭配 **VictoriaMetrics Cluster 的 Capacity Planning 實驗** 實際應用，幫助你建立 Grafana 面板時知道選哪種圖、怎麼搭配查詢語法。

Grafana 常見圖表元件用途總覽

圖表類型	典型用途	適合的 PromQL	適合情境
折線圖（Time series / Line chart）	顯示數值隨時間變化	<code>`rate(...)`</code> , <code>`increase(...)`</code> , <code>`avg(...)`</code>	趨勢追蹤、容量變化
長條圖（Bar chart）	多個項目比較	<code>`sum(...) by (job)`</code> 、 <code>`topk(...)`</code>	節點比較、資源排行
Stat 圖（Stat）	顯示單一指標、即時值、彙總值	<code>`sum(...)`</code> , <code>`avg(...)`</code> , <code>`max(...)`</code>	顯示重點指標、概覽面板

折線圖（Line Chart / Time Series）

什麼時候用？

當你要看「數值如何隨時間變化」，例如：

- 每秒 ingest 速率

- 磁碟用量增長
- CPU 負載變化

範例 (VictoriaMetrics)

```
rate(vm_ingest_rows_total[5m]) by (instance)
```

 Grafana 設定：

- 選擇 **Time series** 圖
- `Legend`：使用 `{{instance}}`
- 單位設為 `rows/s`（可讀性高）

長條圖 (Bar Chart)

什麼時候用？

當你要比較「不同物件的數值大小」但不關心時間趨勢：

- 每個 `vmstorage` 的儲存量
- 各節點的 CPU 使用率
- top 5 ingest 節點

範例 (單一時間點的儲存空間)

```
vm_data_size_bytes{job="vmstorage"}
```

或顯示 top 5 使用率最高的節點：

```
topk(5, rate(vm_ingest_rows_total[5m])) by (instance))
```

 Grafana 設定：

- Panel 類型選擇 **Bar Chart**
- `Orientation`：橫式 or 直式
- 可設定 Y 軸單位，例如 bytes、rows

Stat 圖 (Stat)

什麼時候用？

當你要顯示「**某個重要數值的當前狀況或彙總結果**」，像是：

- Cluster 總寫入量
- 平均 ingest 速率
- 節點數

範例 (Cluster 寫入總量)

```
sum(rate(vm_ingest_rows_total[5m]))
```

或顯示節點總數：

```
count(up{job=~"vm.+"})
```

 Grafana 設定：

- Panel 類型選擇 **Stat**

- 設定文字格式、單位
- 可加入圖示或綠紅燈警示

Capacity Planning 實戰組合建議

目標	查詢語法	建議圖表類型
每個節點 Ingest Rate	<code>`rate(vm_ingest_rows_total[5m]) by (instance)`</code>	折線圖
Top 5 寫入最多節點	<code>`topk(5, rate(vm_ingest_rows_total[5m]) by (instance))`</code>	長條圖
Cluster 平均 Ingest Rate	<code>`avg(rate(vm_ingest_rows_total[5m]))`</code>	Stat 圖
Cluster 總 Ingest Rate	<code>`sum(rate(vm_ingest_rows_total[5m]))`</code>	Stat 圖
各節點儲存空間	<code>`vm_data_size_bytes`</code>	長條圖
Cluster 總儲存量	<code>`sum(vm_data_size_bytes)`</code>	Stat 圖

小總結

圖表類型	建議用途
折線圖	時間序列、資源變化趨勢
長條圖	節點/Job 間資源比較
Stat 圖	顯示總量、平均值、重要警示指標

如果你願意，我可以幫你設計一份完整的 **VictoriaMetrics Cluster Capacity Dashboard 模板**，列出：

- 每個圖表用途
- 對應 PromQL
- Grafana 設定建議（單位、顯示方式、範圍）

你需要這樣的設計表格嗎？還是有特定 metric 想設成圖表？我可以一步一步幫你規劃。