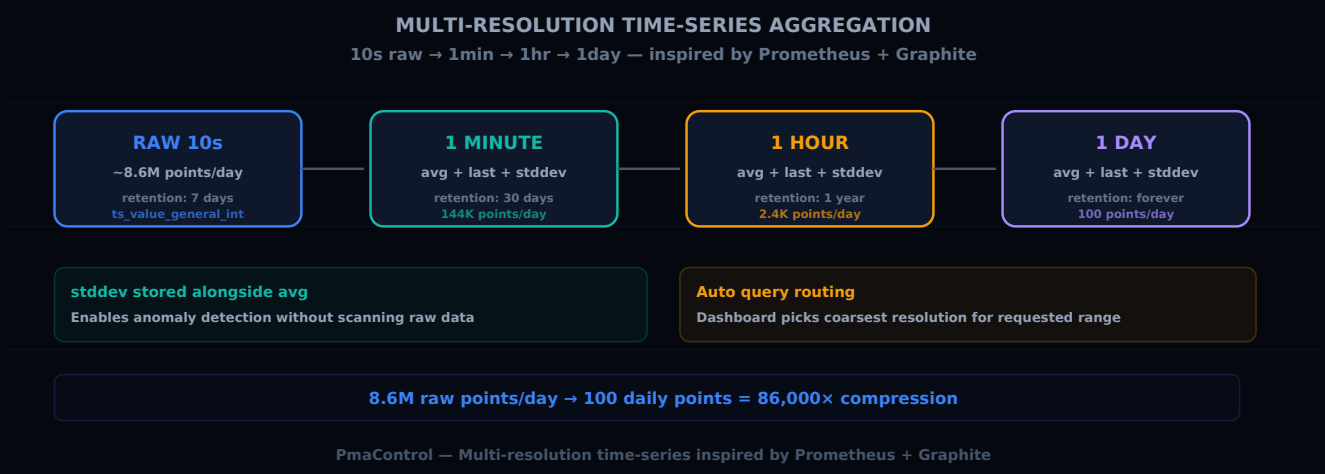
时间序列聚合：从数百万原始数据点到快速查询

Aurélien LEQUOY · 2026年3月21日

PMACONTROL **TIME-SERIES** **AGGREGATION** **MONITORING** **ARCHITECTURE**



PmaControl — Multi-resolution time-series inspired by Prometheus + Graphite

原始数据量

PmaControl 每10 秒从每个受监控的 MariaDB / MySQL 实例采集指标。对于每台服务器，这意味着：

- 每分钟 6 个数据点
- 每小时 360 个数据点
- 每天 8,640 个数据点
- 每周 **60,480** 个数据点

以 100 台服务器、每台 50 个指标计算：

```
100 台服务器 × 50 个指标 × 8,640 数据点/天 = 43,200,000 数据点/天
```

每天 4300 万个数据点。每周 3.02 亿。每月超过十亿。

以原始分辨率（10 秒）无限期存储所有这些数据，在技术上是可行的，但在实际中毫无意义。没有人会去查看 6 个月前 10 秒分辨率的图表。而为了显示一年期图表而扫描数百万行数据，既缓慢又昂贵。

灵感来源：**Prometheus** 和 **Graphite**

这个问题并不新鲜。两个系统已经优雅地解决了它：

- **Prometheus recording rules**: 预计算的 PromQL 查询，以固定间隔将原始数据聚合为派生指标
- **Graphite 及 Whisper** 格式: 多分辨率留存系统，数据随着时间推移自动聚合

PmaControl 借鉴了这两种方法来设计自己的聚合系统。

多分辨率方案

四个分辨率层级:

层级	间隔	保留时间	预估数据量 (100 台服务器)
原始数据	10 秒	7 天	3.02 亿数据点/周
1 分钟	1 分钟	30 天	2.16 亿数据点/月
1 小时	1 小时	1 年	4380 万数据点/年
1 天	1 天	无限期	180 万数据点/年

在任意时刻 (100 台服务器) 存储的总数据量:

```
原始数据 (7 天):      3.02 亿数据点
1 分钟 (30 天):       2.16 亿数据点
1 小时 (1 年):        4380 万数据点
1 天 (全部):          约 200 万数据点
总计:                 约 5.64 亿数据点
```

如[illegible]聚合数据[illegible]聚合[illegible]数[illegible]轴。

每个聚合层级存储什么

对于每个聚合数据点，存储三个值:

```
CREATE TABLE ts_aggregated_1min (
    server_id     INT,
    metric_id     INT,
    timestamp     DATETIME,
    last_value    DOUBLE,    -- 区间内的最后一个值
    avg_value     DOUBLE,    -- 区间内的平均值
    stddev_value  DOUBLE,    -- 区间内的标准差
    PRIMARY KEY (server_id, metric_id, timestamp)
```

```
);
```

`last_value`

`avg_value`

`stddev_value`

- A CPU 42% 48% `avg=45%, stddev=2%`
- B CPU 5% 85% `avg=45%, stddev=28%`

```
IF current_stddev > 3 × average_stddev_last_30_days
THEN alert:
```


cron

1 1

6

```
INSERT INTO ts_aggregated_1min (server_id, metric_id, timestamp, last_value, avg_value,
stddev_value)
SELECT
    server_id,
    metric_id,
    DATE_FORMAT(timestamp, '%Y-%m-%d %H:%i:00') AS minute,
    -- last_value:
    (SELECT value FROM ts_raw r2
```

```
     WHERE r2.server_id = ts_raw.server_id
       AND r2.metric_id = ts_raw.metric_id
       AND r2.timestamp >= DATE_FORMAT(ts_raw.timestamp, '%Y-%m-%d %H:%i:00')
       AND r2.timestamp < DATE_FORMAT(ts_raw.timestamp, '%Y-%m-%d %H:%i:00') + INTERVAL 1
MINUTE
     ORDER BY r2.timestamp DESC LIMIT 1),
    AVG(value),
    STDDEV(value)
FROM ts_raw
WHERE timestamp >= NOW() - INTERVAL 1 MINUTE
GROUP BY server_id, metric_id, minute;
```

步骤 2：1 分钟到 1 小时

每小时，一个工作进程将 60 个一分钟数据点聚合为一个一小时数据点。组合标准差的计算使用合并方差公式：

```
σ_combined = sqrt( mean(σ²_i) + var(μ_i) )
```

其中 `σ_i` 和 `μ_i` 是每个区间的标准差和均值。这个公式在数学上是精确的，不需要原始数据。

步骤 3：1 小时到 1 天

同样的原理，每天一次，24 个一小时数据点合并为一个一天数据点。

步骤 4：清理旧数据

每次聚合之后，超出保留窗口的数据将被删除：

```
DELETE FROM ts_raw WHERE timestamp < NOW() - INTERVAL 7 DAY;
DELETE FROM ts_aggregated_1min WHERE timestamp < NOW() - INTERVAL 30 DAY;
DELETE FROM ts_aggregated_1hr WHERE timestamp < NOW() - INTERVAL 1 YEAR;
-- ts_aggregated_1day: 永不清理
```

查询路由

当 PmaControl 仪表盘显示图表时，它必须选择正确的分辨率。原则很简单：使用能覆盖请求范围的最粗分辨率。

```
function selectResolution(int $timeRangeSeconds): string {
    if ($timeRangeSeconds <= 3600) {       // <= 1 小时
        return 'ts_raw';                   // 10 秒分辨率
    } elseif ($timeRangeSeconds <= 86400 * 2) {  // <= 2 天
```

```
        return 'ts_aggregated_1min';       // 1 分钟分辨率
    } elseif ($timeRangeSeconds <= 86400 * 90) { // <= 90 天
        return 'ts_aggregated_1hr';        // 1 小时分辨率
    } else {
        return 'ts_aggregated_1day';       // 1 天分辨率
    }
}
```

结[illegible]期图表所需的数据点（10 秒分辨率），而不是 310 万个（10 秒分辨率）。查询时间从数秒降至几毫秒。

对查询的影响

请求范围	分辨率	加载的数据点	查询时间
1 小时	10 秒（原始）	360	< 10 ms
24 小时	1 分钟	1,440	< 20 ms
30 天	1 小时	720	< 15 ms
1 年	1 天	365	< 10 ms

查询时间变得与时间范围无关。一年期图表和一小时图表一样快。

利用存储的标准差进行异常检测

得益于预计算的标准差，PmaControl 可以在聚合数据上检测异常，无需回溯原始数据：

1. 基线计算：计算过去 30 天每个指标的标准差的均值和标准差
2. 比较：将当前小时的标准差与基线进行比较
3. 告警：如果标准差超过基线的 3 倍，则为异常行为

具体示例：

- threads_running [illegible] `stddev = 2.1, stddev_stddev = 0.8`
- 当[illegible] `ev = 14.3`
- 得分 `4.3 - 2.1) / 0.8 = 15.25` 个西格玛[illegible]常

这种机制可以检测简单的均值监控无法发现的异常：一台 CPU 剧烈波动但始终回到正常均值的服务器。

[illegible]

[illegible]

[illegible]PmaControl [illegible] 100 [illegible] MariaDB / MySQL [illegible] 20 [illegible]