用一次 MariaDB 读取识别已记录的告警观测

Aurélien LEQUOY · 2026年11月21日

PMACONTROL MARIADB MYROCKS PERFORMANCE SQL BENCHMARK

PmaControl 两次收到相同的观测时，我们要识别重复数据，以免再次开启或关闭事件。原先，应用先向 MariaDB 查询最近一次观测的日期，再查刚收到的点。

在五项真实检查中，645 次取 363 毫秒，从 295824 包降到 0.91 毫秒。已测试，尚待部署。

以时区检查为例，逐条看查询

11 月 21 日 22:29:19，时区检查的指纹在两次查询中得到同一条观测，会发生什么？原实现先执行下面的查询。

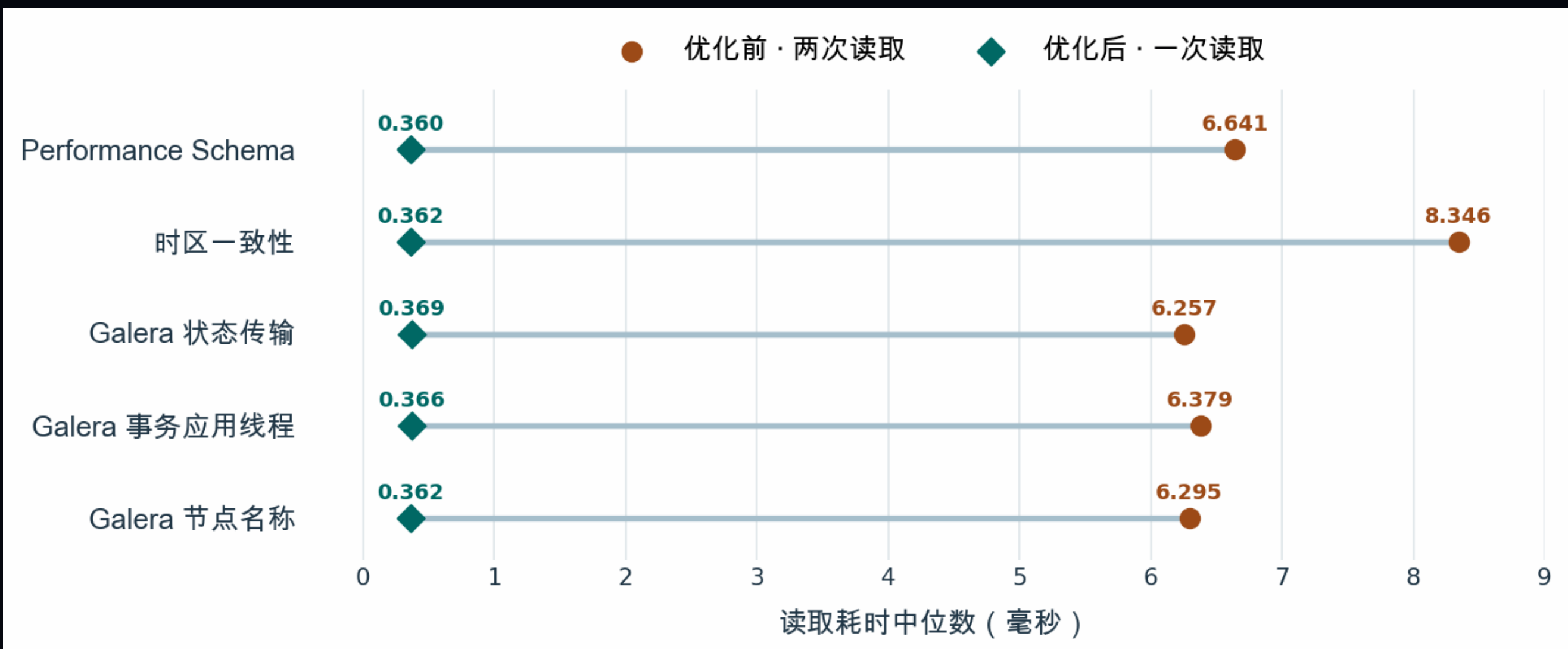
```
EXPLAIN PARTITIONS
SELECT observed_at
FROM pmc_alert_level_point
WHERE fingerprint = UNHEX(?)
ORDER BY observed_epoch DESC, observed_at DESC
LIMIT 1;
```

应用只根据查到一行来判断历史是否存在，随后还要查找精确的观测并读取状态。选定的实现直接从这一步开始。观测点和状态一致即可认定为重放，同一时刻的状态发生冲突则报错。

```
EXPLAIN PARTITIONS
SELECT is_open
FROM pmc_alert_level_point
WHERE fingerprint = UNHEX(?)
  AND observed_epoch = 1795296559
  AND observed_at = '2026-11-21 22:29:19.000000';
```

以下是服务器针对同一项检查生成的执行计划。第一列给果对应已删除的读取，第二列对应足以识别重放的读取。

EXPLAIN PARTITIONS	最近日期	精确观测点
访问类型（type）	`range`	`const`
索引（key）	`PRIMARY`	`PRIMARY`
键字节数（key_len）	32	48
引用（ref）	—	`const,const,const`
预估行数（rows）	8,617	1



EXPLAIN PARTITIONS	最近日期	精确观测点
计划中的分区	45	1（p740307）
Extra	`Using where; Using index`	—

两[illegible]。前者只限定 32 字节的指纹，计划中保留 45 个分区。后者还限定 epoch 和日期，覆盖复合键的全部 48 个字节。MariaDB 可以直接定位到当天分区中的唯一键。

`rows` 中的 8,617[illegible]一次 `FORMAT=JSON` 显示第[illegible]的[illegible]返回[illegible]（`ops = 1`），`LIMIT 1` 一致。这些计数器不衡量[illegible]，[illegible]21 日，该摘要[illegible]小时 08 分钟

查看五项检查的全部 20 份 MariaDB 执行计划（文本）

测试的是哪五项检查

摘要页显示了同一 SQL 的五次执行，分别使用五个 fingerprint 值。每个值标识一个监控条件。我们从对应观测中读取了规则名称。括号中的字母与 CSV 中的标记对应。

检查（CSV 标记）	规则代码	告警点数 / 总数
Performance Schema 启用状态 (A)	`AdvisorPerformanceSchema`	0 / 5,692
时区一致性 (B)	`AdvisorTimeZoneAlignment`	5,694 / 5,695
Galera 状态传输方式（SST）(C)	`AdvisorWsrepSstMethod`	0 / 5,692
Galera 事务应用线程 (D)	`AdvisorWsrepSlaveThreads`	0 / 5,692
Galera 节点名称 (E)	`AdvisorWsrepNodeName`	0 / 5,692

这五个重放点的时刻都是 11 月 21 日 22:29:19，当时只有时区检查处于告警状态。图中的每条线连接原读取路径的中位数和改为先查精确点后的中位数。

在观测服务器上，每项检查的每个版本运行 50 次。末行汇总该版本的全部 250 次测量。两张表的 SQL 耗时均以毫秒为单位。

重放已有观测点 · 从两次读取减为一次

检查	优化前平均值	优化后平均值	优化前中位数	优化后中位数
Performance Schema 启用状态	7.171	0.372	6.641	0.360
时区一致性	8.808	0.400	8.346	0.362
Galera 状态传输方式（SST）	7.372	0.575	6.257	0.369
Galera 事务应用线程	6.970	0.389	6.379	0.366
Galera 节点名称	7.132	0.374	6.295	0.362
五项检查合计	7.491	0.422	6.645	0.363

向已有历史添加观测点 · 从四次读取减为三次

检查	优化前平均值	优化后平均值	优化前中位数	优化后中位数
Performance Schema 启用状态	39.832	29.522	29.189	26.465
时区一致性	41.518	32.461	35.533	29.123
Galera 状态传输方式（SST）	39.431	31.138	31.509	28.157
Galera 事务应用线程	36.855	31.667	33.790	25.943
Galera 节点名称	39.696	31.043	35.195	28.511
五项检查合计	39.466	31.166	33.243	27.975

为什么新增点的收益较小

迟到的观测[illegible]在[illegible]存[illegible]之[illegible]，[illegible]而[illegible]件[illegible]。找不[illegible]精确键时查前取后继。只要找到任一相邻点，就能同时确认历史存在。

以下为时区检查在新增场景中的两条相邻点查找计划，参数时间为 11 月 22 日 22:29:19。即使省掉最近日期查询，这两次读取仍有成本。

相邻点	访问类型	索引	分区	预估行数
前驱	`range`	`PRIMARY`	45	8,653

相邻点	访问类型	索引	分区	预估行数
后继	`range`	`PRIMARY`	45	81

V1 原实现删除最近日期查询，利用精确点及其相邻点。三次都未命中时，程序检查旧版本是否保留了可导入的状态。第一组试验中，恢复操作的中位数从 9.665 升至 12.963 毫秒，因为 V2 在刚刚确认空白的历史里又选到的 直接处理这种情况。如果历史为空，且没有旧事件，就按原日期、epoch 和内容恢复唯一留存的正常观测点，然后在同一把锁下检查传入的点。

实验环境的完整事务包括加载读取器数据单位为毫秒。

操作	**V1** · 原实现	**V2** · 删除首次查询	**V3** · 直接恢复
重放已点	3.954 / 2.554	1.586 / 0.924	1.518 / 0.904
向已有史增	907 / 17.297	23.516 / 13.954	22.895 / 14.017
创建第个	28.373 / 10.477	20.031 / 8.689	22.708 / 8.660
恢复旧正	3.609 / 10.422	29.137 / 13.423	17.151 / 7.916

V2 与 V3 的主要区别在于恢复旧状态，常规新增的中位数接近。三个版本的决定、观测点、事件、区间和最新状态完全一致。

能否用 **fingerprint** 作为主键?

11 月 20 日有 **3,719** 个点，分属 **9,4** 其中 **8** 个指纹 1 天出 **83** 次。指纹标一项检，三元组 `epoch, observed_at)` 标识该检查的一次观测。

`SHOW INDEX` 对首列前缀的基数估算为 3,672,630，两个更长前缀的估算均为 2,448,420。增加一触发碰撞可能真实组合数因跨每个键前缀区间的副本上条相同指纹的观测触发重复键错误 106

SHOW INDEX · MariaDB · Partitioning

判断历史存在的其他写法

耗时汇总五项检查，展示的计划对应时区一致性检查。箭头代表未命中时回查完整历史，横线代表无分区表。每组实验均有自己的基线。

组别与查询	访问类型 / 索引	分区	平均 / 中位数（ms）
I · 最近日期	`range / PRIMARY`	45	7.577 / 6.660
I · 最近`FORCE INDEX (PRIMARY)`	`ref / PRIMARY`	45	4.927 / 4.283
I · `SELECT 1 … LIMIT 1`	`ref / level_neighbour`	45	6.361 / 5.434

[illegible]	[illegible] / [illegible]	[illegible]	[illegible] / [illegible]ms）
I · `SELECT EXISTS(…)`	`ref / level_neighbour`	45	6.870 / 5.625
I · [illegible]	`const / PRIMARY` + `const / PRIMARY` → `ref / level_neighbour`	— + 45 → 45	4.543 / 3.995
II · [illegible]	`range / PRIMARY`	45	6.401 / 5.623
II · [illegible]	`ref / level_neighbour_epoch` → `ref / level_neighbour`	1 → 45	3.739 / 3.411
II · [illegible]	`const / PRIMARY`	—	0.277 / 0.257

[illegible] 250 [illegible]

[illegible] SQL [illegible]

[illegible]

[illegible] MariaDB 11.8.10 [illegible] MyRocks [illegible] **621,865** [illegible] `when_observed(observed_epoch, observed_at)` [illegible]

[illegible]	[illegible] / [illegible]	[illegible]	[illegible] / [illegible]ms）
[illegible]	`range / PRIMARY`	45	612.119 / 549.937
[illegible]	`range / PRIMARY`	45	637.779 / 525.603
[illegible]	`index / when_observed`	45	64.429 / 48.306
[illegible]	`range / PRIMARY`	—	14.052 / 7.676
[illegible] 16 [illegible]	`ref / PRIMARY`	1	0.786 / 0.410

[illegible] `PRIMARY` `when_observed` [illegible] `index` [illegible] 16 [illegible]

[illegible] CPU、96 MiB MyRocks [illegible] 571,957 [illegible] 1.350 [illegible] 549.937 [illegible]

[illegible]

50 SHOW PROFILES

45 28,463 SQL

SQL ZIP

PmaControl

8.808 0.4008 346 0.362 V3