云数据仓库的性能与成本测试基准设计与实现报告

张博飞 2022201318

1 项目任务

1. 通过对 CAB (Cloud Analytics Benchmark, VLDB 2023) 论文的阅读，snowset、redset 数据集的分析得出云数据仓库的负载特征。
2. 分析 CAB 等现有测试基准的局限性，设计用于测试新型云数据仓库的测试基准。

2 项目实现

2.1 研究背景

尽管云与本地部署存在显著差异,学术界和从业者仍依赖旧有的基准如 TPC-H 或 TPC-DS 来比较云分析系统, Redset(https://github.com/amazon-science/redset) 和 Snowset(https://github.com/resource-disaggregation/snowset) 等主流云数据仓库的用户日志显示，真实环境中查询到达率和资源需求有明显的时间变化模式。由于 TPC-H 没有考虑云环境中工作负载的波动性和间歇性特征，其负载特特征也与实际工作环境有较大差异，所以不能够胜任云数据仓库性能的测试。

2.2 真实云工作负载与传统基准的差异

Redset 和 Snowset 等主流云数据仓库的用户日志揭示了真实环境中工作负载的复杂特性。Snowflake 发布的 Snowset 数据集包含了 2018 年 2-3 月两周期间约 6900 万个查询的性能统计信息，为提供了深入了解云数据仓库实际使用模式的机会。通过分析这些数据，CAB 研究人员发现了与传统基准测试显著不同的工作负载特征。

首先，查询类型分布差异明显。Snowset 工作负载中，读/写查询占比高达 59%，纯读取查询约 28%，纯写入查询约 13%。相比之下，TPC-H 主要侧重于分析查询，缺乏这种数据转换的场景。

其次，查询到达率呈现明显的时间模式。Snowset 显示不同用户使用的不同实例都有着较大的负载波动，尤为重要的是，某些用户展现” 尖峰” 工作负载，查询仅在短时间内集中运行。例如，Snowset 中的一些虚拟仓库每天仅使用一次，导致系统负载峰值。还有一些虚拟仓库始终活跃，但工作负载噪声大。

2.3 CAB 对负载的构建

对于不同数据库实例使用情况，CAB 作者将其模拟为 5 个模式：

2.3.1 正弦噪声模式 (Sinusoidal Noise)

这种模式创建了一个基础负载，并在其上添加了多个正弦波峰和随机噪声。它的特点是：

- 首先建立一个低强度的基础负载
- 添加多个随机位置的正弦波形高峰
- 在整体上添加随机噪声

这种模式可以模拟具有周期性峰值的工作负载，如每天特定时间出现的高流量，但带有一定的随机性。适合模拟典型的日常业务系统，如零售网站在一天中不同时段的访问模式。

2.3.2 随机峰值模式 (Random Spikes)

这种模式创建了随机出现的较短尖峰。特点是：

- 突发的随机峰值
- 峰值之间的强度变化大
- 峰值持续时间短

这种模式适合模拟不可预测的突发工作负载，如新闻网站在重大事件发生时的突发流量，或者社交媒体平台的病毒式内容传播。

2.3.3 突发模式 (Burst Pattern)

这种模式创建了一个随机位置的持续性高负载区域。特点是：

- 在随机位置产生一个持续的高流量区域
- 负载增长和下降较为平滑
- 突发区域持续时间长于随机峰值

这种模式适合模拟计划内的大规模批处理任务，如数据仓库的夜间 ETL 处理，或者定期报表生成过程中的数据库负载。

2.3.4 负载断路器模式 (Load Breaker)

这种模式首先建立一个基础负载，然后在某个区域大幅增加负载，同时在另一个区域完全抑制负载。特点是：

- 先建立一个中等强度的基础负载
- 随机区域负载突然增高
- 另一随机区域负载完全清零

这种模式可以模拟系统维护期间的负载转移，如数据库故障转移、负载均衡，或计划性停机期间的流量重定向。

2.3.5 小时级峰值模式 (Hourly Spikes)

这种模式创建了均匀分布的周期性峰值。特点是：

- 建立一个低强度的基础负载
- 添加 24 个均匀分布的峰值（模拟 24 小时）
- 峰值高度有随机变化

这种模式适合模拟具有固定周期性的工作负载，如每小时执行的调度任务，或者不同时区用户访问造成的周期性流量变化。

2.4 CAB 标准的局限性

CAB 标准能够基本反应云原生数据库的测试需求，但仍存在着较多局限，对于数据库开发者来说存在着较多不足。

1. 其测试过程中为模拟实际情况，按照云数据库同时具有多个用户和负载的特征，设置多个数据库实例，将每个实例赋予随机模式，进而了多条查询流。但由于其模式随机性，开发者难以判断云数据库在预测哪种模式时存在不足。即使生成多条查询流，实际应用过程中，开发者通常仍会使用单条查询流进行负载预测的开发。

2. 其模式的生成过程中将模式 1、4、5 的基础负载设置的过大，导致其得到的查询流负载波动有限，与 snowset 的实际情况不符。

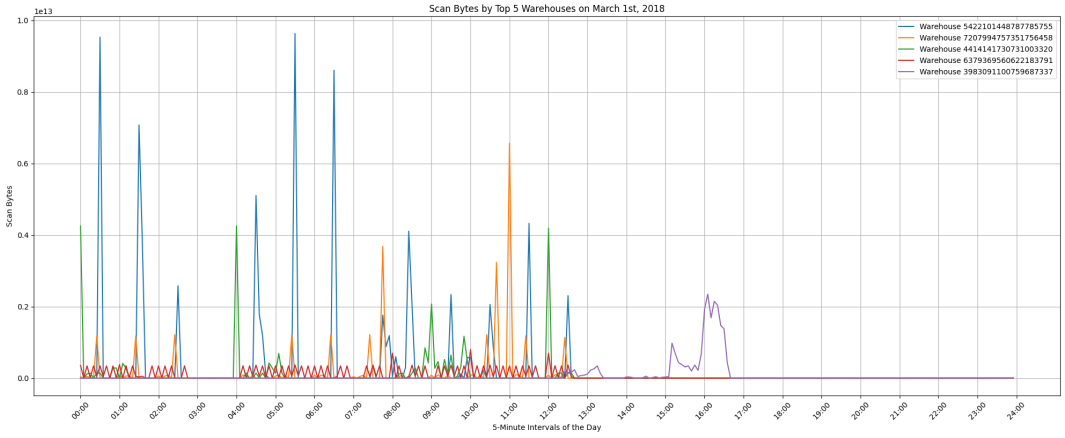

图 1: Snowset 中的实际负载波动示例

3. 其通过 tpch 查询来拟合对应模式的过程中，使用 tpch 的平均 cputime 和单个时间槽分配到的 cputime，来确定单个时间槽内分配几条查询。然后随机抽取对应数量的 tpch 查询放入该时间槽。由于 tpch 查询之间产生的负载差距较大，这种方式容易导致得到的查询流不饿能够匹配对应的模式。

4. 在对 snowset 数据集进行分析后，我发现实际负载情况具有复杂性，将某个数据库实例仅仅抽象为单一模式不足以反应其特征，如图，各数据库实例在不同日期的负载特征有较大差异，部分实例在一天内的负载情况也难以简单看作一种模式。简单的五种模式显然不足以反应实际负载情况。

5. CAB 由 cputime 的分布来生成模式，使用 tpch 查询进行拟合，然而 cputime 大致相同不能代表其他指标如内存消耗、系统 io 的吻合。开发者关注的重点也不一定时 cputime。

6. 在实际负载预测的开发中，开发者通常需要之前一段时间的负载进行预测，如果按照 CAB 生成测试查询时间跨度为 1 天，可能需要先运行其 12 小时的查询，才能对后 12 小时的负载进行预测，将消耗开发者很高的时间成本。

7. CAB 在参数处理上将参数写死，妨碍了开发者需要修改需求。

2.5 对 CAB 代码的改进

我的 benchamark 实现一定程度上使用了 CAB 代码中 Tpch 参数的生成方式以及部分基础模式的生成方式，并对其余功能进行了重写或改进。

1. 将多条随机模式查询流改成单一查询流指定模式或所有简单模式的查询流。由开发者指定模式、数据集大小和总负载量。

2. 将模式 1、4、5 的基础负载减少为原本的 1/20。

3. 为更好的拟合负载，我将 CAB 原本的代码逻辑改为随机从 22 条查询中抽取小于当前时间槽负载的查询，并不断累加比较，指导没有查询可以抽取。

4. 添加混合模式的选项，同一查询流可具有多种简单模式。将两种或多种简单模式直接叠加，得到更符合实际情况的负载。

5. 增加了用 scanned_bytes 拟合模式的选项，满足开发者的不同侧重点。所用 22 条查询和原本 CAB 的代码中一样，在 snowflake 平台上测得。

6. 为满足开发者对负载预测的需求，按开发者提供的参数反馈对应的前期数据库信息，比如，将某条查询流执行前，10 倍其时间跨度的 cputime、scanned_bytes 信息打包提供，避免使用者长时间训练，节约其时间成本。

7. 将重要参数提取至命令行输入，为开发者更改需求提供便利。

2.6 代码运行

最终代码在 https://github.com/pixelsdb/elastic-bench，参考其 readme 文档进行运行。

- .cpp 和.h 文件是生成查询流的文件，其他的是下载和分析 snowset 的文件
- query_stream_p1_s10_cpu24_t8.json 是查询流文件
- 其余 json 文件是其对应的前期数据库信息文件，对应前 10 倍时间跨度的 cputime 变化

命令行大致格式：

```
./benchmark -p <pattern_ids> -d <intensity> -s <scale> -cpu <hours/scanedbytes> -t <duration>
[-mode bytes]
```

例如：

```
./benchmark -p 1 -s 10 -cpu 24 -t 8
```

3 项目收获

通过本次云数据仓库性能与成本测试基准的设计与实现，我获得了以下几点收获：

1. **深入理解云数据仓库工作负载特征**：通过对 Snowset 和 Redset 等真实云数据仓库日志的分析，我对云环境中查询负载的时间变化模式、波动性和间歇性等特征有了更深入的理解。这些特征与传统数据库环境有显著不同，需要专门的测试基准来评估。

2. **测试基准的设计考量**：了解到一个好的云数据仓库测试基准需要考虑多个方面，包括查询类型分布、时间变化模式、资源需求波动等。设计测试基准时需要平衡真实性与易用性，既要反映真实工作负载特征，又要便于开发者使用。

3. **模式抽象的意义与局限**：虽然 CAB 提出的五种基本模式可以概括常见的负载模式，但实际工作负载往往更为复杂，需要组合多种模式或创建更精细的模型。模式抽象有助于理解和分析，但也存在简化真实情况的风险。

4. **开发者视角的重要性**：从开发者的实际需求出发，我认识到测试基准不仅要有学术价值，还要考虑实用性。例如，提供历史负载数据以支持预测研究，允许灵活配置参数，以及考虑除 CPU 时间外的其他资源指标。

5. **C++ 系统编程技能提升**：在改进 CAB 代码的过程中，我提高了 C++ 系统编程能力，特别是在处理复杂数据结构、算法设计和多种工作负载模式的生成方面。

6. **性能预测的挑战**：通过实验，我体会到云环境中的负载预测具有高度挑战性。多种波动模式的叠加、不同资源指标的相关性，以及云服务弹性特性都使得准确预测变得复杂。

7. **测试与评估方法**：设计测试基准时，需要兼顾测试数据的可复现性和真实性，同时要有明确的评估标准来判断系统性能。我学会了如何设计参数化的测试方案，以便进行公平比较和评估。

总的来说，这个项目不仅让我深入了解了云数据仓库的工作负载特征和测试需求，也提高了我的系统设计和编程能力。通过改进现有测试基准，我更好地理解了如何设计一个既反映真实工作负载特征又便于开发者使用的测试工具。这些经验对于未来在云数据系统领域的研究和开发工作将有很大帮助。

4 致谢

本项目在研究和实现过程中得到了多位老师和学长的悉心指导与帮助。在此特别感谢卞昊穹老师、阎世杰师兄和李文博师兄对我的指导与支持。他们的专业知识和宝贵建议对本项目的完成起到了至关重要的作用。