链滴

ClickHouse 的 Buffer 引擎

作者：flowaters

原文链接：https://ld246.com/article/1517214401442

来源网站：链滴

背景

Buffer，指把数据先写入内存Buffer表，再周期性的刷入磁盘表中。

读取数据时，会同时从Buffer表和磁盘表读取。

示例

先给例子

```
CREATE TABLE merge.hits_buffer AS merge.hits ENGINE = Buffer(merge, hits, 16, 10, 100, 100
0, 1000000, 10000000, 100000000)
```

创建一张 merge.hits_buffer表，结构与表 merge.hits相同，使用了Buffer引擎。在数据写入这张表，会先写入内存缓冲，随后再写入 merge.hits表了。

例子中给出了16个缓冲区。每一个缓冲区中的数据的触发条件为：

- 缓存达到了100秒
- 写入了100万条数据
- 写入了100MB数据
- 同时达到了 缓存10秒，写入1万条数据，并且写入了10MB数据

在服务器stop时，或者执行 DROP TABLE和DETACH TABLE时，Buffer表内容也会写入目标表中。

说明

再给说明

```
Buffer(database, table, num_layers, min_time, max_time, min_rows, max_rows, min_bytes, max
bytes)
```

- database: 数据库
- table: 数据要写入的磁盘表
- num_layers: buffer的个数，推荐为16

数据在所有的min条件均满足时，或者有一个max条件满足时，则会被刷新到磁盘中。

- min_time, max_time: 秒数
- min_rows, max_rows: 行数
- min_bytes, max_bytes: 字节数

写操作时，会随机写入num_layers中的一个。如果数据过大时，即超过了max_rows和max_bytes时会直接写入磁盘中。

每一个buffer layers的操作都是独立进行的。

当使用默认值时，即 num_layers = 16 和 max_bytes = 100000000时，使用的总内存为1.6GB.

注意事项

● 如果数据库和目标表留空，数据则不会写入目标表。在flush时，buffer将被清空。这个特点可以用实现内存窗口。

● buffer表是没有索引的，查询时会进行全表扫描。buffer表很大时，会变的慢。

● 如果buffer表和目标表的列不一致，则两个表公共的列将写入目标表中。

● 如果需要改为表结构，推荐先删除Buffer表，再改变目标表结构，再重建Buffer表。

● 如果机器异常重启，则Buffer表内容会丢失。

● PREWHERE, FINAL and SAMPLE语句，不支持Buffer表，这些语句将直接在目标表中操作，不会作Buffer表中的数据。

● 在向Buffer表写数据时，这个Buffer区将会加锁，这时读请求会有延迟。

● 写入Buffer表的顺序，和刷新到磁盘的顺序，不一定是一致的。如果要同时使用Buffer表和CollapsigMergeTree表，可以将num_layers设置为1，来避免这个问题。

● 如果目标表是replicated，Buffer表不能保证一条数据只写入一次？？

If the destination table is replicated, some expected characteristics of replicated tables are lost when writing to a Buffer table. The random changes to the order of rows and sizes of data par s cause data deduplication to quit working, which means it is not possible to have a reliable ' xactly once' write to replicated tables.

结论：只在少数有限的情况下，推荐使用Buffer表。

性能

● 每秒可以发起几千个请求。如果每个请求只有一条数，则QPS只有几千；如果每个请求的日志数大则QPS可以达到百万级。

应用场景

经过测试，使用少量线程(1-3)，大包发送(2000-4000)的情况下，使用Buffer引擎和直接使用MergeTee引擎的性能是无差异的。

所以据推测，Buffer引擎适用于多线程，小包发送的场景。未实际测试。