Presto 全新的 Parquet Writer 介绍

随着越来越多的公司广泛部署 Presto，Presto 不仅用于查询，还用于数据摄取和 ETL 作业。所有很有必要提高 Presto 文件写入的性能，尤其是流行的列文件格式，如 Parquet 和 ORC。本文我们将介绍 Presto 的全新原生的 Parquet writer ，它可以直接将 Presto 的列式数据结构写到 Parquet 的列式格式，最高可提高6倍的吞吐量，并减少 CPU 和内存开销。新的 Native Parquet Writer 是从 Presto 0.237 版本开始引入的，可以参见 #14411 ；Trino 是在 337 版本默认打开这个功能的 #3400。

Presto 旧的 Parquet Writer 实现

如下图所示，Presto 使用 Hive 旧的 Parquet writer 来写文件：这种旧方法首先遍历 page 中的每个 columnar block 并重建每条记录。 然后 Presto 将调用 Hive record writer 将每条记录写到到 Parquet 的 pages 中。 写入过程中如果超过缓冲区大小时，旧的 writer 会将内存数据刷新到底层文件系统。

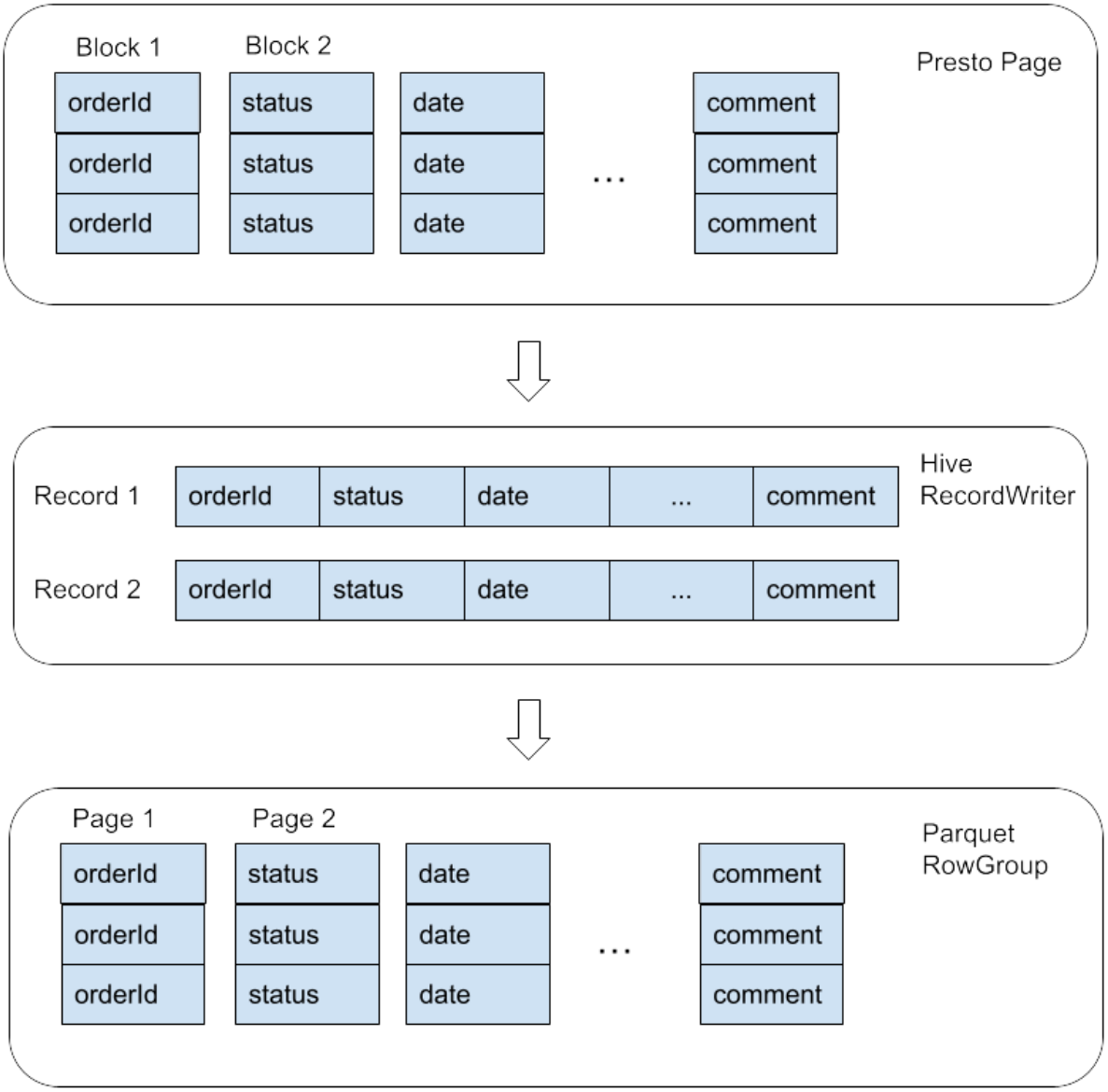


Figure 1

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：过往记忆大数据

Presto 对内存中的列式数据进行了向量化执行，而 Parquet 是一种列式文件格式。 旧的 Parquet writer 增加了不必要的开销，将 Presto
的列式内存数据转换为基于行的记录，然后再进行一次转换以将基于行的记录写入 Parquet
的列式磁盘文件格式。
不必要的数据转换会显着增加性能开销，尤其是在涉及复杂数据类型（例如嵌套结构）时。

全新的内置 Parquet Writer

为了提高文件写入效率，克服旧 Parquet writer 的弊端，Presto 社区引入了全新的原生 Parquet writer，它直接将 Presto 的内存数据结构写入 Parquet

的列式文件格式，包括数据值、重复值和定义值。 原生 Parquet writer 显着降低了 Presto 的 CPU 和内存开销。

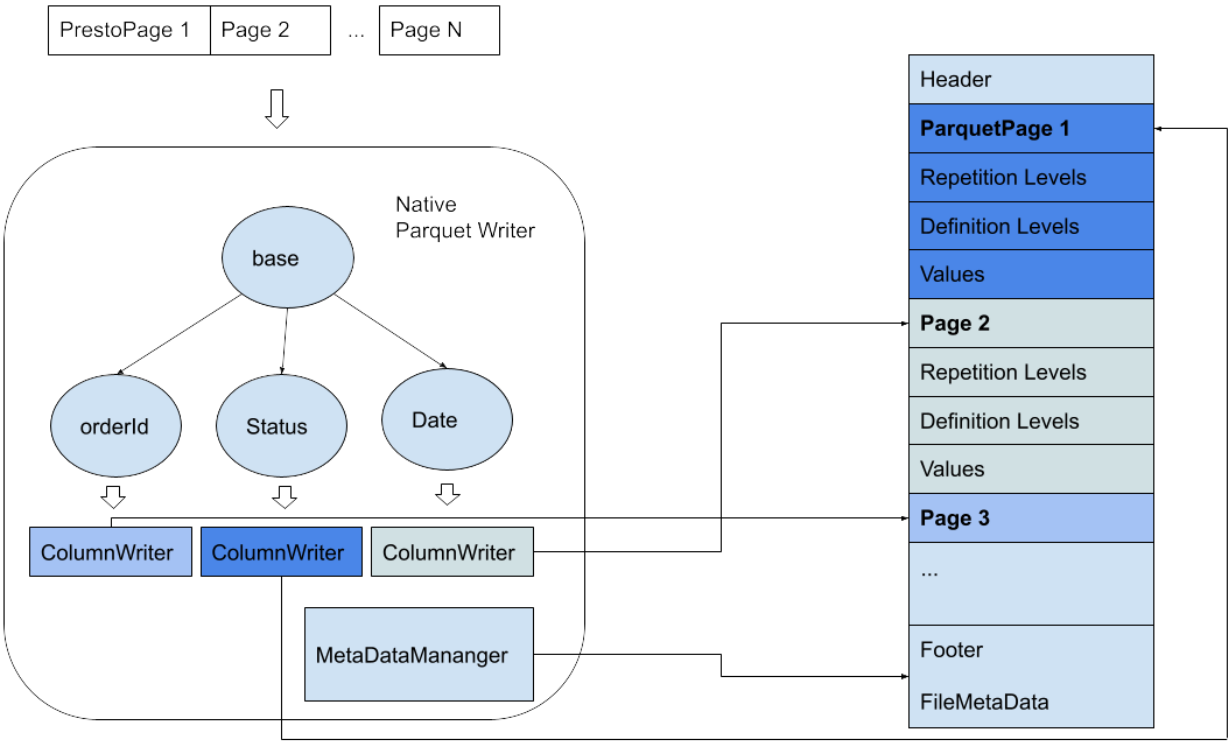


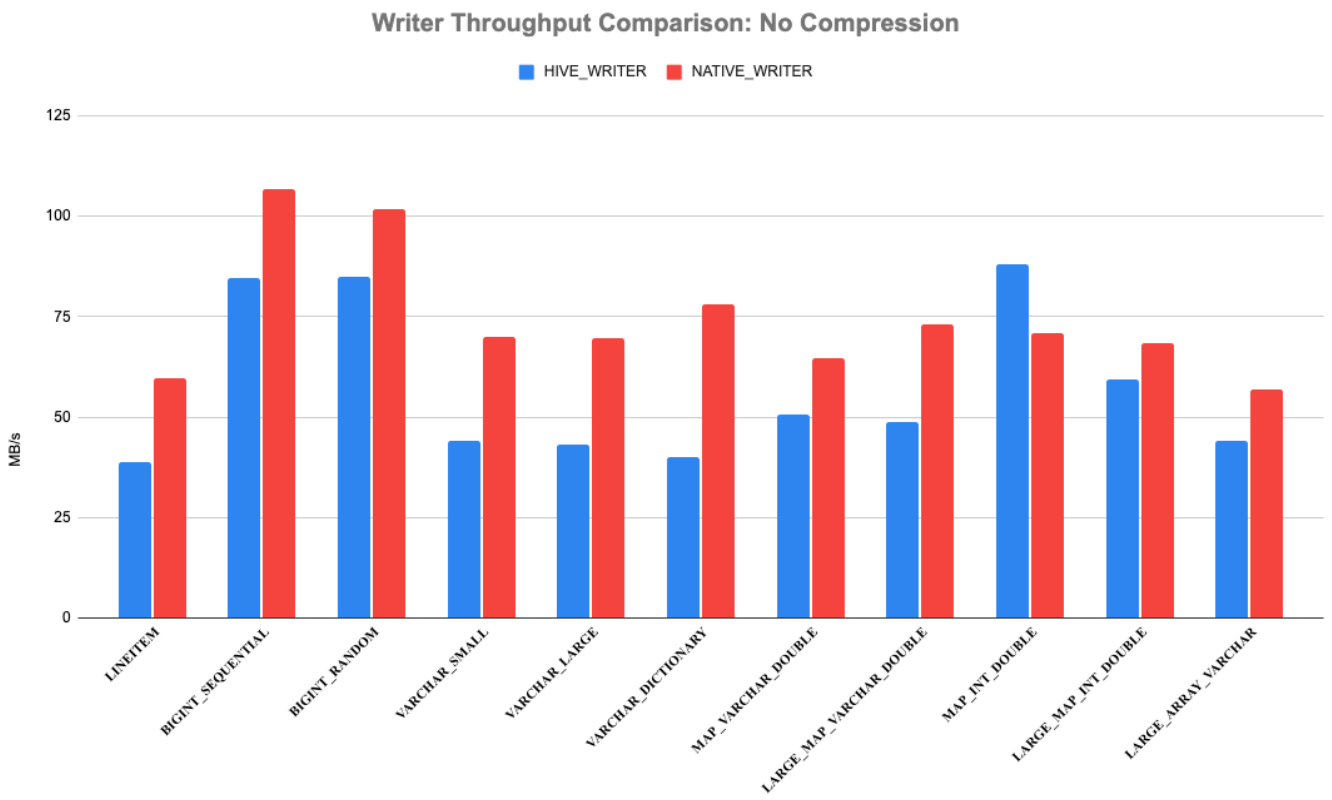
Figure 2

如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：过往记忆大数据

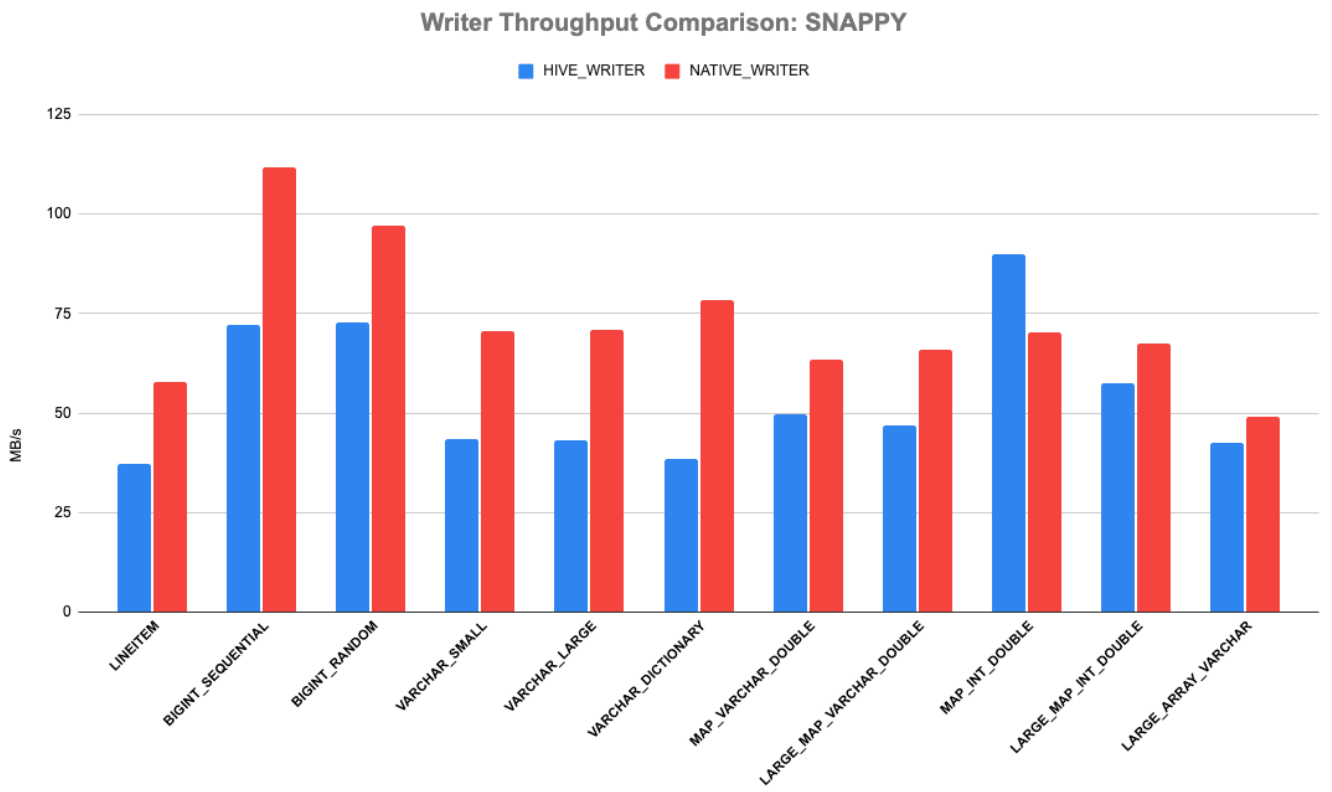
如上图所示，原生 Parquet writer 根据列名和类型构造 Parquet 模式。基本类型（Primitive types）和复杂类型（Struct、Array、Map）转换为相应的 Parquet 类型。使用 schema 信息来创建 Column writers。 对于每个 Presto page，内置的 writer 迭代每个 block，将 Presto block 转换为 Parquet 值。 相应的 column writer 将字节流写入 Parquet 文件。

性能测试

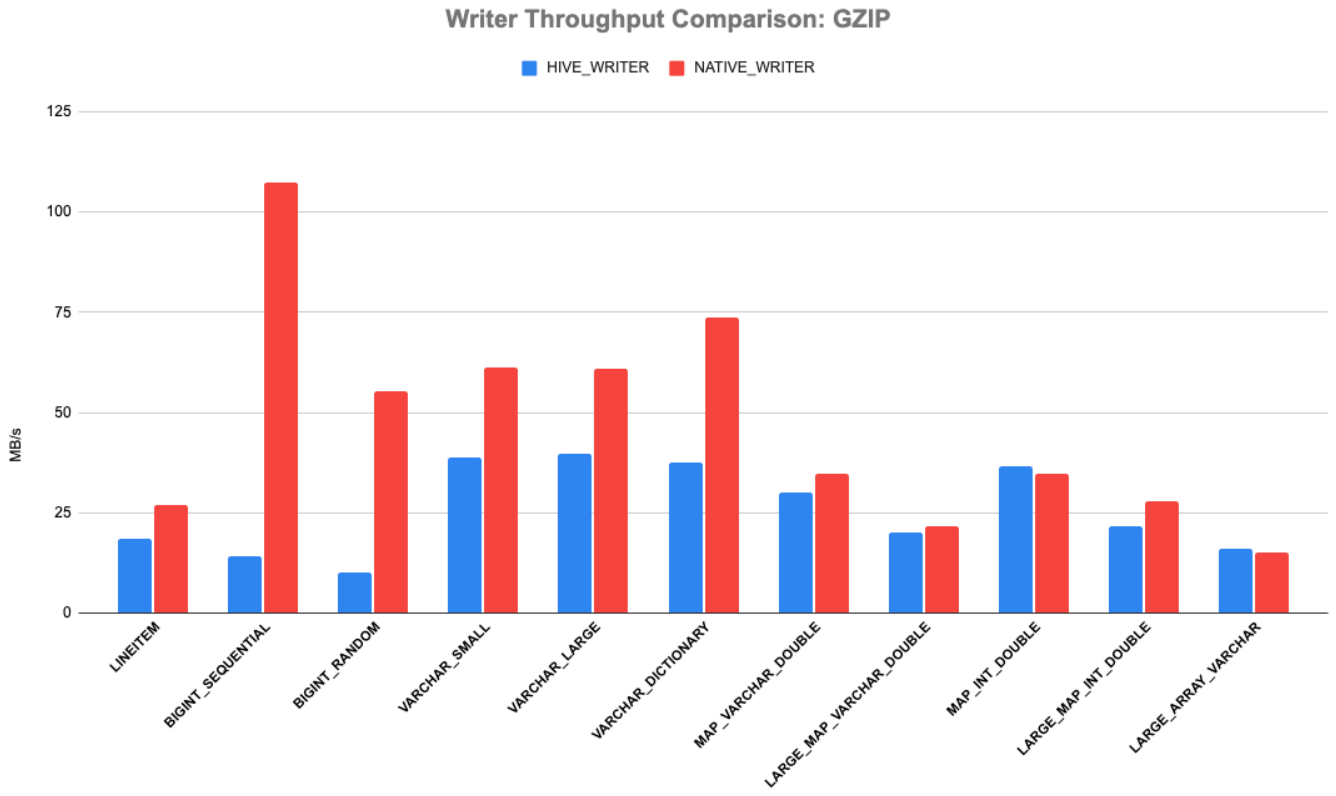
Presto 有一个 Hive 文件格式基准测试来测试 reader 和 writer 的性能。
该测试创建一个包含数百万行的 pages 列表，使用新的 native writer 或旧的 hive record writer 将它们写入到临时文件，然后比较性能。 下图显示了三种压缩方案的结果：gzip、snappy 和不压缩。 X 轴是各种类型的数据； Y 轴是写入吞吐量。 显然，我们可以看到全新的原生 Parquet writer 优于旧的 writer。 它可以持续实现 > 20% 的吞吐量改进。 原生 Parquet writer 在使用 GZIP 压缩的 BIGINT_SEQUENTIAL 和 BIGINT_RANDOM 方面表现最佳，吞吐量提高高达 650%。 写入 TPCH LINEITEM 的所有列时，吞吐量增益约为 50%。



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：过往记忆大数据



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：过往记忆大数据



如果想及时了解Spark、Hadoop或者HBase相关的文章，欢迎关注微信公众号：过往记忆大数据

在生产环境中使用 Native Parquet Writer

我们可以在生产环境中使用 Native Parquet Writer，可以使用以下配置打开这个功能：

```
# in /catalog/hive.properties
hive.parquet.optimized-writer.enabled=true
```

本文翻译自：Native Parquet Writer for Presto

本博客文章除特别声明，全部都是原创！
原创文章版权归过往记忆大数据（过往记忆）所有，未经许可不得转载。
本文链接: 【】（）