深入理解 Gluten 中Table转换的三大 Transformer

在大数据处理领域，数据的高效读取和处理至关重要。Gluten作为一个强大的大数据处理优化框架，在Table转换方面有着精妙的设计。其中，主要有三个关键的Transformer，分别是BatchScanExecTransformer、FileSourceScanExecTransformer和HiveTableScanExecTransformer，它们各自对应着不同的应用场景和实现逻辑，下面我们将深入探讨它们的特性和工作原理。

一、BatchScanExecTransformer：高效读取的现代方案

BatchScanExecTransformer对应于Spark的BatchScanExec，主要应用于Spark datasource V2的Table scan场景。在实际应用中，诸如Iceberg数据源、Delta Lake数据源以及使用datasource v2读取的csv、parquet、orc文件等，都可以通过BatchScanExecTransformer来实现高效的表扫描。

1. 适用场景

对于一些新型的数据湖格式（如Iceberg、Delta Lake）以及常见的二进制格式文件（csv、parquet、orc），使用BatchScanExecTransformer能够充分发挥其优势。以Parquet文件为例，它是一种列式存储格式，支持高效的列裁剪和向量化读取。BatchScanExecTransformer可以针对这些特性进行优化，只扫描需要的列，同时利用向量化技术提高数据读取的效率。

2. 关键实现

Spark的datasource V2引入了统一的编程接口，使得不同的数据源能够以标准化的方式进行集成。BatchScanExecTransformer在这个框架下，负责协调和优化Table scan的过程。它与数据源的底层实现紧密结合，通过解析查询计划，确定需要读取的数据范围和列，并以高效的方式进行数据读取。

此外，可以通过 `spark.sql.sources.useV1SourceList` 参数来控制是否使用V1的数据源实现。例如，当我们将parquet格式设置为不在V1源列表中时，Spark在读取parquet文件时就更倾向于使用BatchScanExecTransformer来实现更高效的扫描。

二、FileSourceScanExecTransformer：传统读取机制的延续

FileSourceScanExecTransformer对应于Spark的FileSourceScanExec，是Spark datasource V1的Table scan实现方式。在默认情况下，我们使用Spark创建的表都会通过它来读取数据。

1. 适用场景

FileSourceScanExecTransformer适用于传统的、基于Spark原生机制创建和管理的数据表。这些表通常遵循Spark的表结构和数据存储格式规范，适用于大多数常见的数据存储和分析场景。例如，当我们使用Spark SQL创建一个简单的CSV文件表，或者从一个已有的HDFS文件目录读取数

据创建DataFrame时，FileSourceScanExecTransformer就会被调用。

2. 工作原理

FileSourceScanExecTransformer的核心职责是从文件系统中读取数据并将其转换为Spark内部的DataFrame格式。它通过解析查询语句和表元数据，定位需要读取的文件，并根据文件的格式和存储结构进行数据解析和加载。在这个过程中，它会处理文件的解压缩、数据格式解析、分区裁剪等操作，以提高数据读取的效率。

三、HiveTableScanExecTransformer：适配Hive生态的专用Transformer

HiveTableScanExecTransformer则主要用于读取那些用非Spark创建的表，特别是当表的提供者（provider）为Hive时。它针对Hive表的存储和查询特性进行了优化，能够高效地读取和解析Hive表的数据。

1. 应用场景

在实际的大数据生态中，Hive是一个广泛使用的数据仓库工具。许多企业可能已经使用Hive构建了自己的数据仓库和ETL流程，而不想因为引入Spark而改变现有的数据存储和查询逻辑。在这种情况下，当需要使用Spark来查询这些非Spark创建的Hive表时，HiveTableScanExecTransformer就会发挥作用。

2. 配置与优化

需要注意的是，当使用非Spark创建的parquet、orc格式的表，并且配置了`spark.sql.hive.convertMetastoreParquet`和`spark.sql.hive.convertMetastoreOrc`参数时，将会使用FileSourceScanExecTransformer而非HiveTableScanExecTransformer去读取数据。这是因为Spark在处理parquet和orc格式的Hive表时，能够进行一些优化和转换，提高读取效率。例如，Spark可以将Hive的元数据信息转换为Spark内部的格式，减少元数据的解析时间，同时利用Spark的向量化读取等技术提高数据读取性能。

四、总结

Gluten中的这三个Transformer——BatchScanExecTransformer、FileSourceScanExecTransformer和HiveTableScanExecTransformer，分别针对不同的数据源和场景提供了优化的数据读取解决方案。BatchScanExecTransformer适用于新型的数据湖格式和支持datasource V2的数据源，能够提供高效的数据读取性能；FileSourceScanExecTransformer则延续了Spark的传统读取机制，适用于默认创建的Spark表；而HiveTableScanExecTransformer则是为了更好地适配Hive生态系统，能够处理非Spark创建的Hive表。通过合理地选择和配置这些Transformer，我们可以充分发挥Spark在大数据处理中的优势，提高数据读取和查询的效率。

然而，在实际应用中，我们还需要根据具体的业务需求和数据特点，灵活选择和调整这些Transformer，以达到最佳的性能表现。例如，在构建数据仓库时，如果数据主要是存储在Hive中，并且

需要频繁使用Spark进行查询分析，那么可以适当调整Hive的元数据配置和Spark的相关参数，以实现更高效的查询性能。