syncsort

DMX-h ETL Use Case Accelerator

File Join Large

Table of Contents

1 Introduction

DMX-h ETL can efficiently join two large files in Hadoop. In this use case accelerator, DMX-h ETL performs an inner join of two large TPC Benchmark H (TPC-H) data sets: an orders file and a line item file. While an inner join is featured in this example, you can modify the DMExpress job and task to perform a left outer, right outer, or full outer join.

The use case accelerators are developed as standard DMExpress jobs with just minor accommodations that allow them to be run in or outside of Hadoop:

- When specified to run in the Hadoop cluster, DMX-h Intelligent Execution (IX) automatically converts them to be executed in Hadoop using the optimal Hadoop engine, such as MapReduce. In this case, some parts of the job may be run on the Linux edge node if not suitable for Hadoop execution.
- Otherwise, they are run on a Windows workstation or Linux edge node, which is useful for development and testing purposes before running in the cluster.

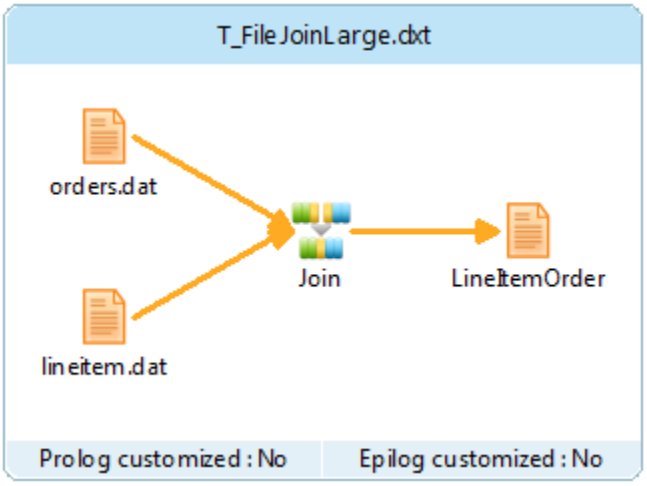
While the method presented in the next section is the simplest and most efficient way to develop this example, the more complex user-defined MapReduce solution is provided as a reference in Appendix A.

For guidance on setting up and running the examples both outside and within Hadoop, see Guide to DMExpress Hadoop Use Case Accelerators.

For details on the requirements for IX jobs, see “Developing Intelligent Execution Jobs” in the DMExpress Help.

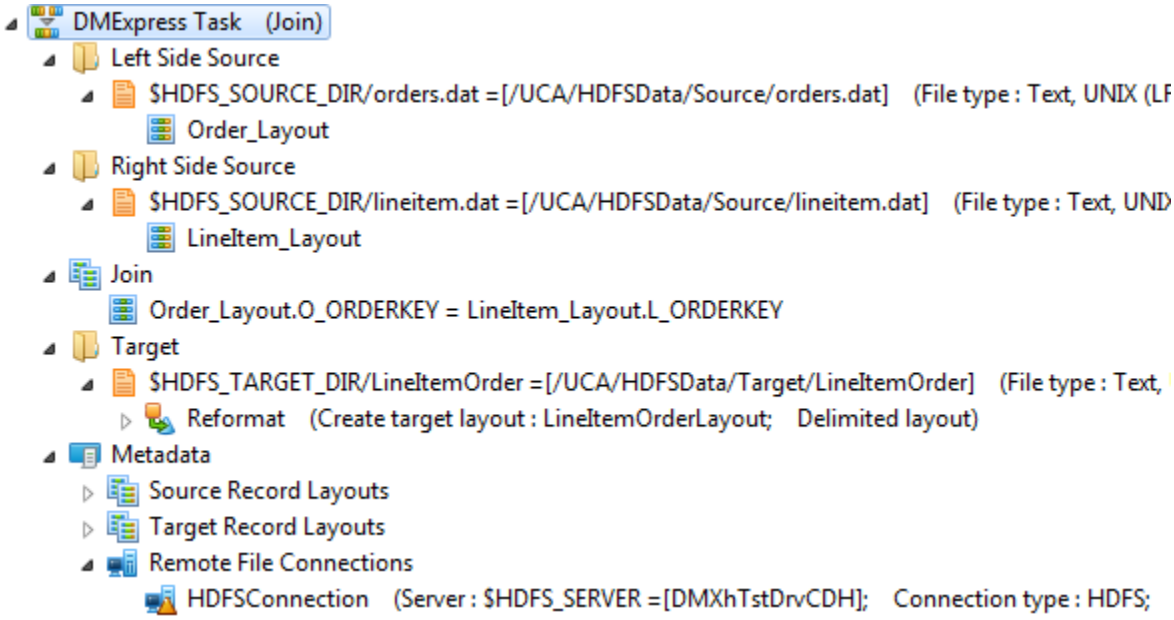
2 File Join Large with DMX-h IX

The File Join Large solution in DMX-h ETL consists of a job, J_FileJoinLarge.dxj, with a single join task.



2.1 T_FileJoinLarge Task

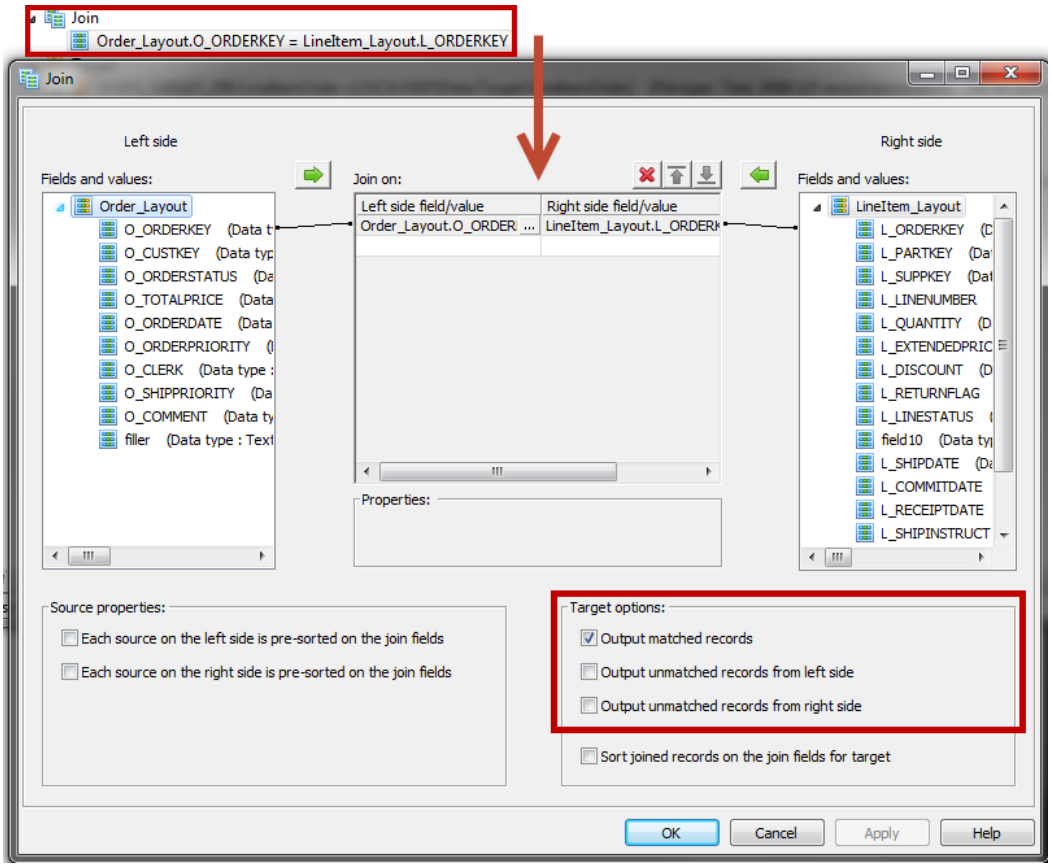
This task performs an inner join on the two HDFS file sources, *orders.dat* and *lineitem.dat*. Records that match on the key field, *ORDERKEY*, are joined and written to the *LineItemOrder* HDFS target. The HDFS source and target files are defined with a remote file connection to the Hadoop cluster.



DMX-h IX automatically partitions the data into groups based on the join keys. The join is divided into concurrently executed parts, each processed independently, such that all equal-keyed records on the input side will be matched and written to an output partition.

In the MapReduce framework, the bulk of the join processing occurs in the reduce phase. Dozens or even hundreds of concurrent reducers can be run to optimize the join performance. For more information on controlling concurrency, see the topic “Running DMX-h ETL Jobs” in the DMExpress Help.

If desired, you can modify the join behavior by double-clicking the join key definition. This example performs an inner join because Output matched records is selected. You can also enable Output unmatched records from left and/or right side. Note, however, that the join output may be the same unless a join side contains some records that do not have a match on the other side.



Appendix A File Join Large with DMX-h User-defined MapReduce

This user-defined MapReduce solution is provided as a reference in the event that particular knowledge of your application's data would benefit from manual control of the MapReduce process.

The File Join Large job in DMX-h contains a map step and a reduce step, separated by a MapReduce data flow connector between the final map task and the initial reduce task, as follows:

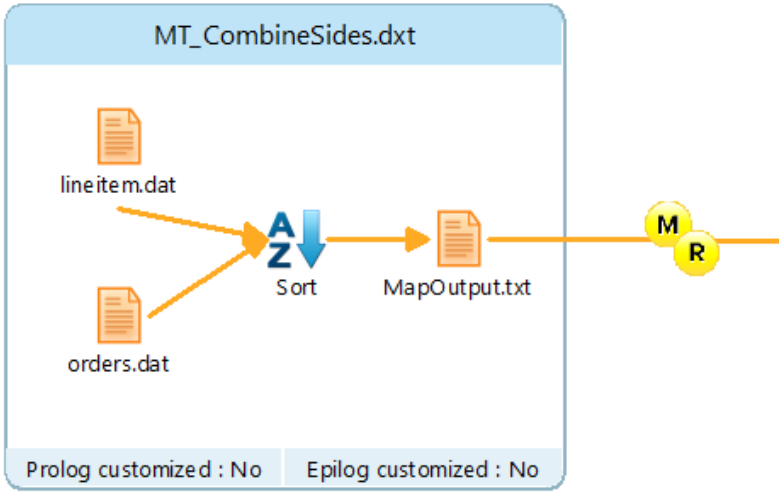
The map step reads two input files and produces a single output stream, marking each record to denote which source file it came from. It then assigns partition IDs to the data such that all records with the same sort key, regardless of source file, will go to the same reducer.

The reduce step first splits the single stream of data back out into two separate files in one task and then performs the join on the set of records it received, in another task.

While it may seem like wasted work to have the mapper merge the records together only to split them back apart in the reducer, this is necessary because in a MapReduce job, only a single stream of data can be passed between mappers and reducers. A map-only job would not work either because each mapper only receives a subset of the data, which may not include all the records with a given join key. However, since each mapper assigns partition IDs to the records in the same way, it guarantees that all records with the same join key will end up in the same reducer so that the join will capture all the changes.

A.1 Map Step

The File Join Large map step in DMX-h consists of one task which combines two large files. It marks each input record with its record type ID, assigns a partition ID to each record based on the key, sorts on that partition ID, and outputs the records in a single output stream.



While a large file join requires two inputs, only a single data path exists between the mappers and the reducers in the Hadoop environment. All records from both inputs are thus combined into a single data stream by the mappers, with a record type identifier (ID) added to each record to identify whether it originated from the orders file ('O') or the lineitems file ('L'). In addition to the record type ID, a partition ID is added to the beginning of each record to direct it to the correct reducer.

Sample source data records:

Orders.dat:

1|36901|O|173665.47|1996-01-02|5-LOW|Clerk#000000951| …

2|78002|O|46929.18|1996-12-01|1-URGENT|Clerk#000000880| …

LineItems.dat:

1|155190|7706|1|17|21168.23|0.04|0.02|N|O|1996-03-13| …

1|67310|7311|2|36|45983.16|0.09|0.06|N|O|1996-04-12| …

Sample single-stream output from map step, with partition ID and record type ID added at the beginning:

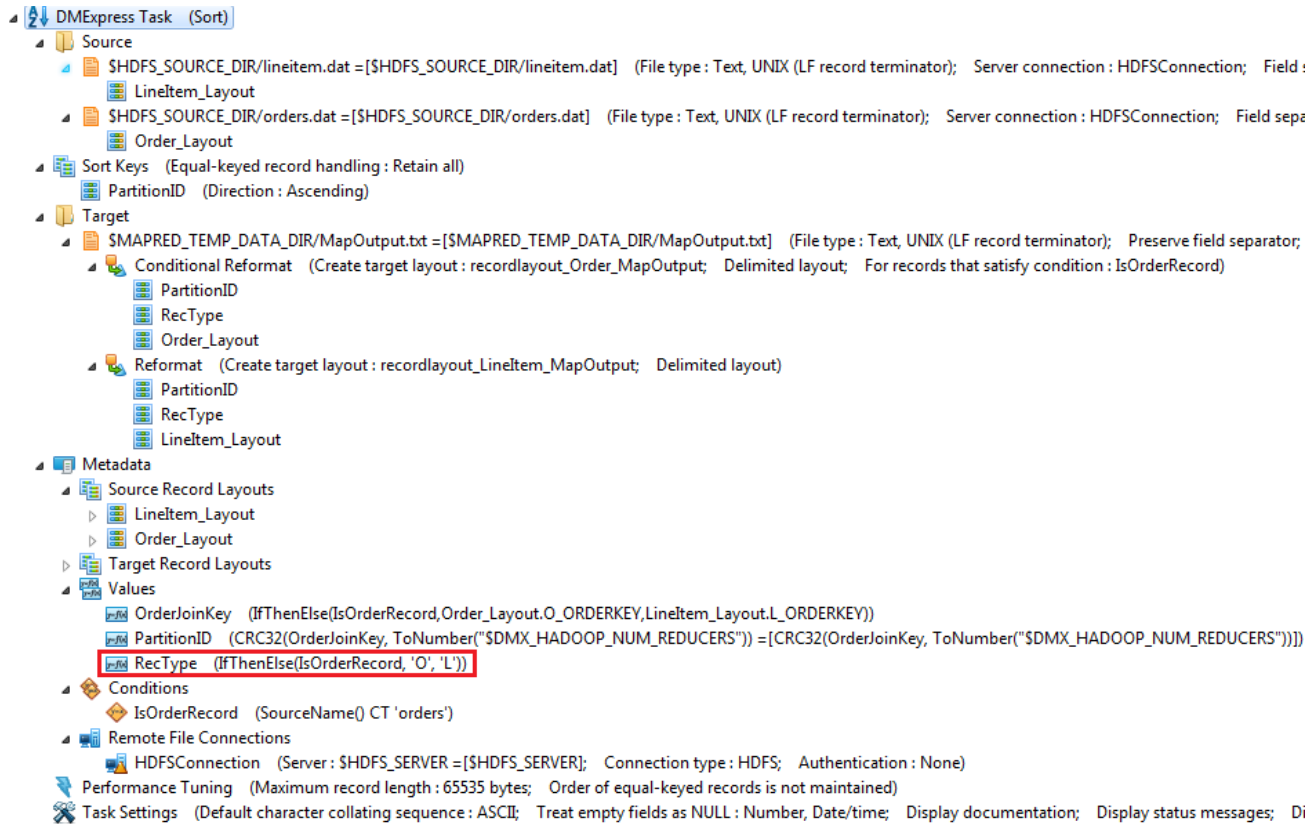
0|O|4854532|11743|O|295956.37|1997-02-15|2-HIGH|Clerk#000000957| …

0|L|180866|17419|4923|3|11|14700.51|0|0.08|A|F|1992-10-09| …

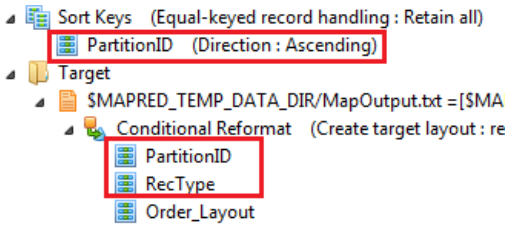
0|L|149056|158495|1011|3|3|4660.47|0.04|0.06|R|F|1995-04-04| …

A.1.1 MT_CombineSides.dxt

This task reads both source files and assigns the record type IDs to the records based on the source file names. The task uses the SourceName() function to retrieve the source file name. If the file name contains the text “orders”, the record is marked with an ‘O’, otherwise, it is marked with an ‘L’.



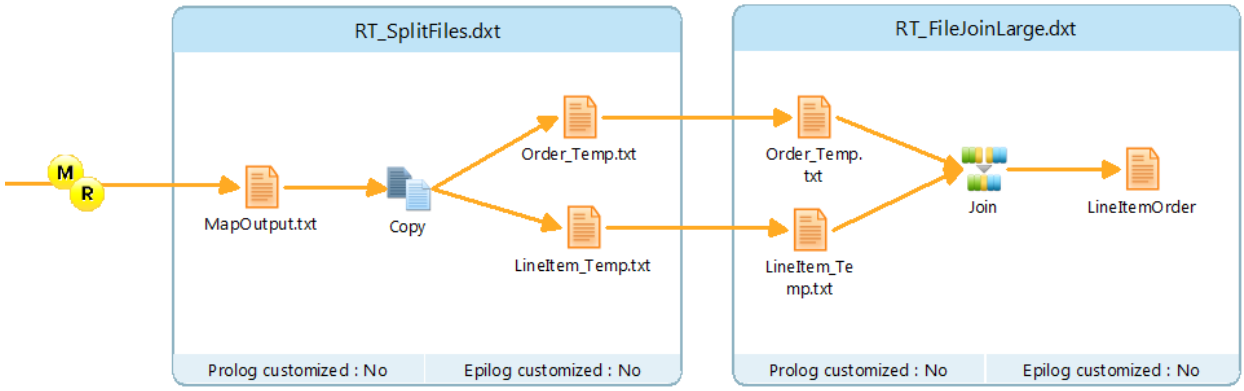
The task further assigns the partition ID (PartitionID) as the first field in each target record and sorts all records by that value so that the framework can properly distribute the data to the correct reducer.



Since all records with the same primary key value (L_ORDERKEY + L_PARTKEY + L_SUPPKEY) must go to the same reducer, the partition ID is determined by creating a CRC32() hash value based on this key. In addition, the environment variable DMX_HADOOP_NUM_REDUCERS (automatically set to the configured number of reducers when running in Hadoop) is passed to the function to limit the range of partition ID values to the number of reducers invoked for your MapReduce job.

A.2 Reduce Step

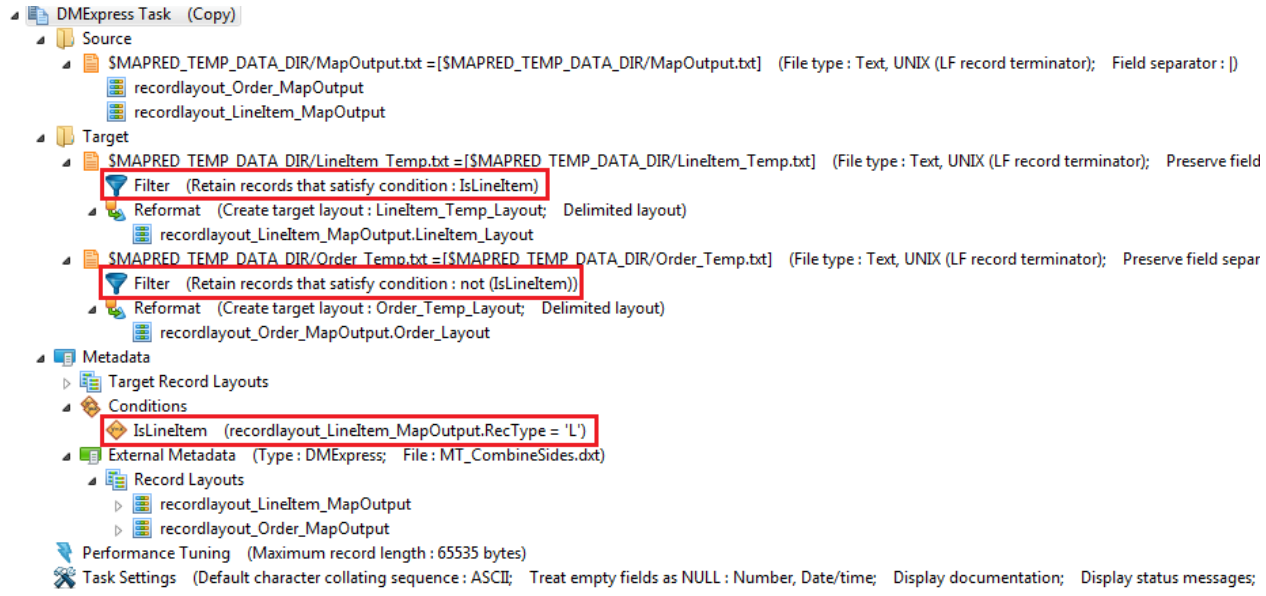
The File Join Large reduce step in DMX-h ETL consists of one task to split the data back out into separate files, and one task to perform the join and output the results.



As the partition IDs were assigned by the map job based on the join key values, records from either side that contain equal join key values are routed to the same reducer.

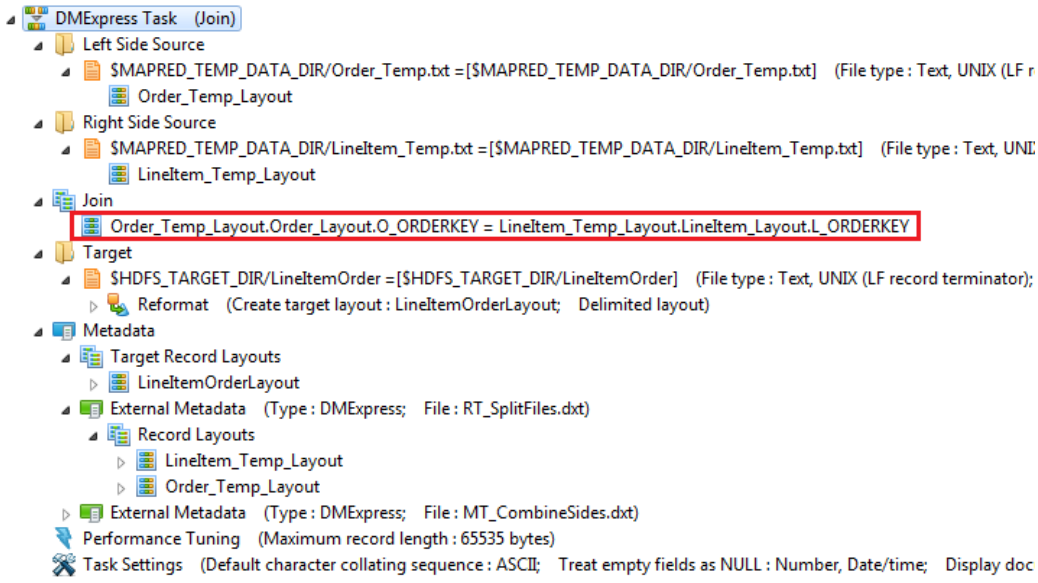
A.2.1 RT_SplitFiles.dxt

This task reads the single input stream containing one partition of data generated by the mappers. Based on the record type ID, this task filters the single input data stream into two temporary files: an order file and a line item file.



A.2.2 RT_FileJoinLarge.dxt

This task joins the temporary order file and temporary line item file into one large file and sends the results to an HDFS file.



The join is divided into concurrently executed parts, each processed by one reducer. Dozens or even hundreds of reducers can be run to optimize the join performance.

About Syncsort

Syncsort provides enterprise software that allows organizations to collect, integrate, sort, and distribute more data in less time, with fewer resources and lower costs. Thousands of customers in more than 85 countries, including 87 of the Fortune 100 companies, use our fast and secure software to optimize and offload data processing workloads. Powering over 50% of the world's mainframes, Syncsort software provides specialized solutions spanning "Big Iron to Big Data", including next gen analytical platforms such as Hadoop, cloud, and Splunk. For more than 40 years, customers have turned to Syncsort's software and expertise to dramatically improve performance of their data processing environments, while reducing hardware and labor costs. Experience Syncsort at www.syncsort.com.

Syncsort Inc. ***50 Tice Boulevard, Suite 250, Woodcliff Lake, NJ 07677*** ***201.930.8200***