Data flow performance considerations

Last updated by | Jackie Huang | Jan 4, 2022 at 12:24 AM PST

Contents

Important Note

Kindly ensure you've read the following article thoroughly before troubleshooting Data flow performance issues. It lists different ways you can optimize the performance and best practices & recommendations to follow while creating mapping data flow.

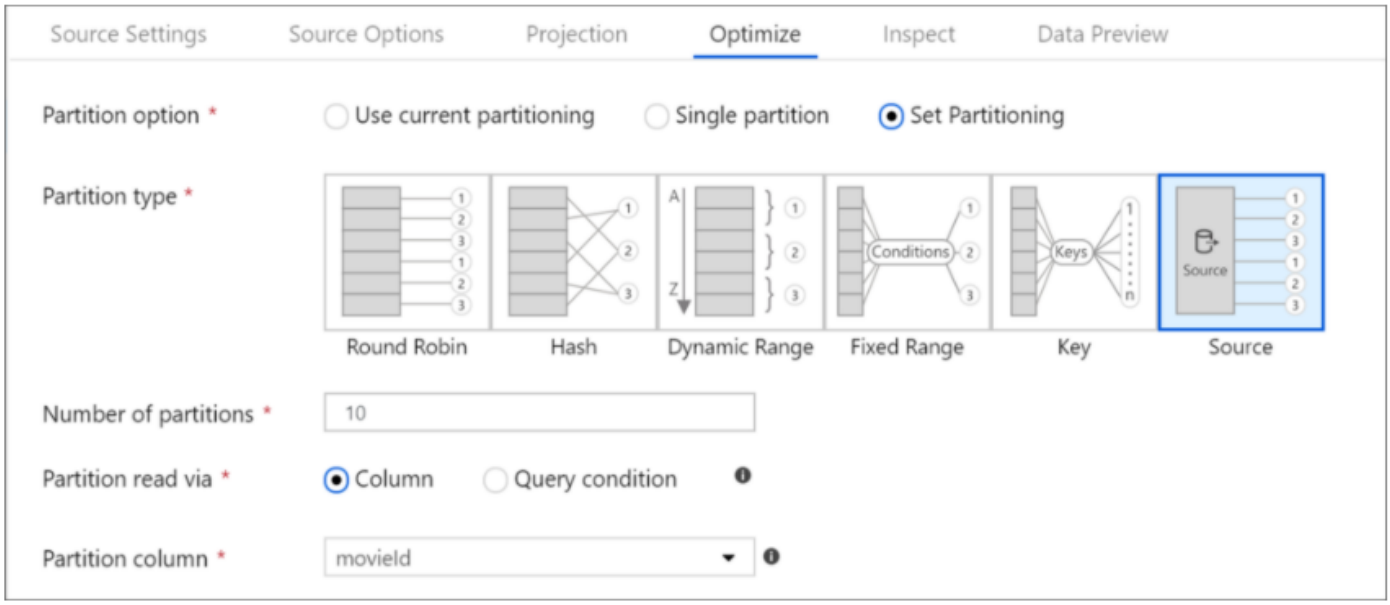
- Mapping data flows performance and tuning guide
- Use Azure IR to Tune ADF and Synapse Data Flows

Each Dataflow runs certain activities to read/write or transform the data. You can monitor the performance of each activity from the data flow graph. Refer the following article for monitoring performance of different data flow tasks:

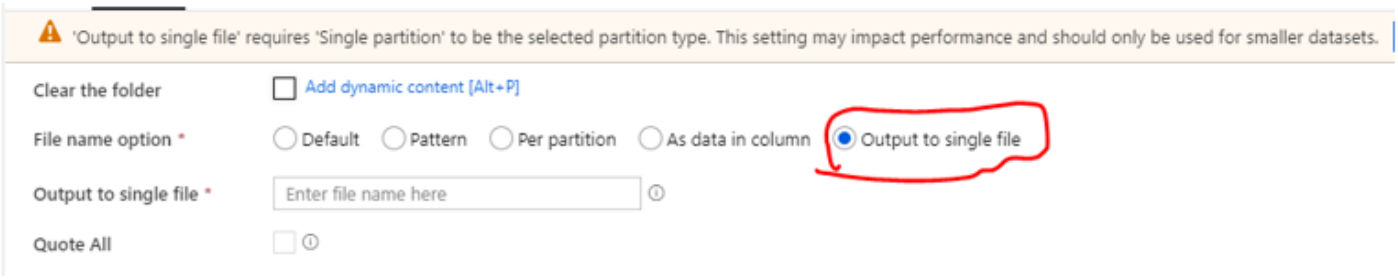
- Monitor Data Flows
- Data flow Execution Plans

Some of the performance scenarios are listed below and recommendations to resolve them:

1. If you're reading data from the Azure SQL DB source, use 'Source' partitioning. Enabling source partitioning can improve your read times from Azure SQL DB by enabling parallel connections on the source system. Specify the number of partitions and how to partition your data. Use a partition column with high cardinality. You can also enter a query that matches the partitioning scheme of your source table.



2. Using default partition will have better performance than single output file (if customer choose "partitionBy('hash', 1)" in the cluster log)



3. Use larger cluster if data size is huge on source:

Memory Optimized	
8	StandardDSv2Family
16 (8 + 8)	StandardDSv2Family
32 (16 + 16)	StandardDSv2Family

4. Try to reduce the amount for pivot, unpivot, window and aggregation transformation.

5. Similarly, using joins, partitioning etc. can also impact the job's performance.

How good have you found this content?