How-To Library

Informatica

Performance tuning and best practices for Google BigQuery V2 Connector

Abstract

When you use Google BigQuery V2 Connector to read data from or write data to Google BigQuery, multiple factors such as hardware, Google Compute Engine, Secure Agent, and Informatica mapping parameters impact the connector performance. You can optimize the performance by tuning these parameters appropriately. This article describes general reference guidelines to help you tune the performance of Google BigQuery V2 Connector in Cloud Data Integration.

Supported Versions

- Informatica® Cloud Data Integration

Table of Contents

Overview

Performance tuning is an iterative process in which you analyze the performance, use guidelines to estimate and define parameters that impact the performance, and monitor and adjust the results as required.

This document describes the key hardware, database, Google Compute Engine, Secure Agent, and Informatica mapping parameters that you can tune to optimize the performance of Google BigQuery V2 Connector. This document also includes case studies that involve the tuning specifications required when you configure a mapping to write data to Google BigQuery from a flat file using Google Compute Engine and the Secure Agent. The graphs constructed from the case studies illustrate the impact that the tuning has on Google BigQuery V2 Connector performance.

Note: The performance testing results listed in this article are based on observations in an internal Informatica environment using data from real-world scenarios. The performance of Google BigQuery V2 Connector might vary based on individual environments and other parameters even when you use the same data.

Performance tuning areas

You can optimize the performance of a read or write operation when you use Google BigQuery V2 Connector by tuning the following parameters:

- Google Compute Engine
- Secure Agent
- Google BigQuery mapping

Google Compute Engine best practices

To optimize the performance, consider the following best practices for Google Compute Engine:

- Ensure that the Secure Agent hosted on the Google Compute Engine virtual machine is in the same region as the Google BigQuery dataset and the Google Cloud Storage bucket.
- Choose the Google BigQuery dataset in the region where the Google Compute Engine virtual machine is located.
- Choose the correct Google Compute Engine virtual machine instance type based on your requirements.

Tune the Secure Agent

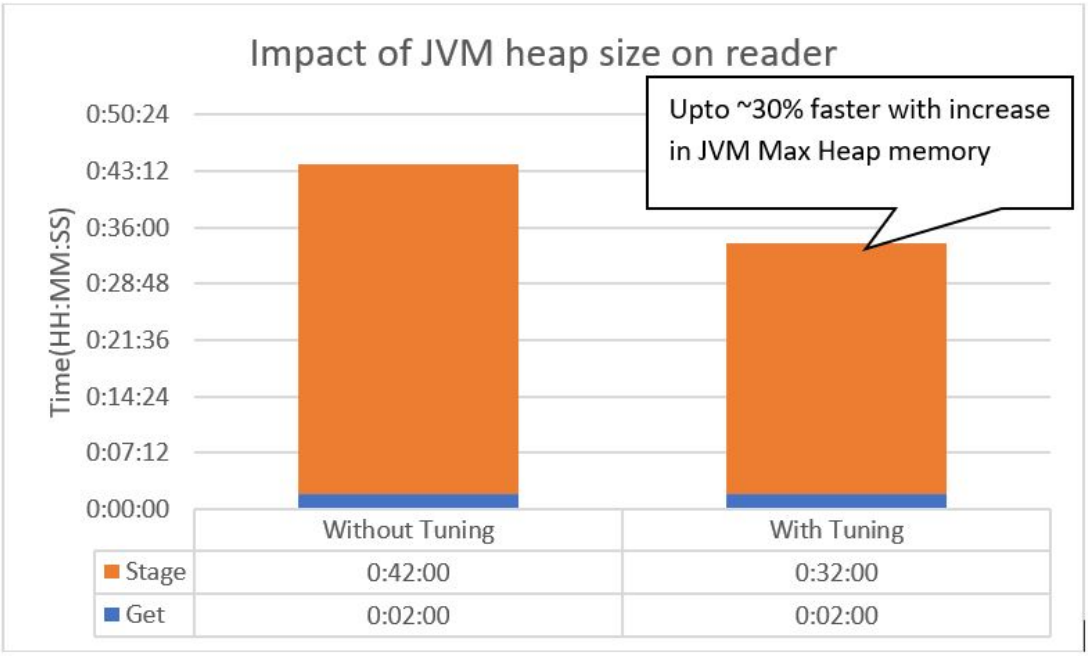
You can tune the JVM heap size and the DTM process limit for the Secure Agent to optimize the performance of a Google BigQuery read or write operation.

JVM heap size

You can specify the amount of memory that the Secure Agent uses for each pmdtm process. Default is 64 MB.

If you use the default JVM heap size, the mapping fails with an out-of-memory error. You can increase the JVM heap size to improve the performance of the read and write operations.

The following graph shows the results and the recommended values for setting the JVM maximum heap size:



For Google BigQuery V2 mappings and mapping tasks that read data from Google BigQuery, specify a JVM memory of -Xmx256m.

If you already configured a heap size value that is higher than 256 MB in the JVM options, do not change it to -Xmx256m.

Perform the following steps to configure the JVM memory:

1. In Administrator, select the Secure Agent listed on the **Runtime Environments** tab.
2. Click **Edit**.
3. In the **System Configuration Details** section, select **Data Integration Service** as the service and **DTM** as the type.
4. Edit the **JVMOption1** property, and enter *-Xmx256m*.
5. Click **Save**.

Bulk processing for write operations

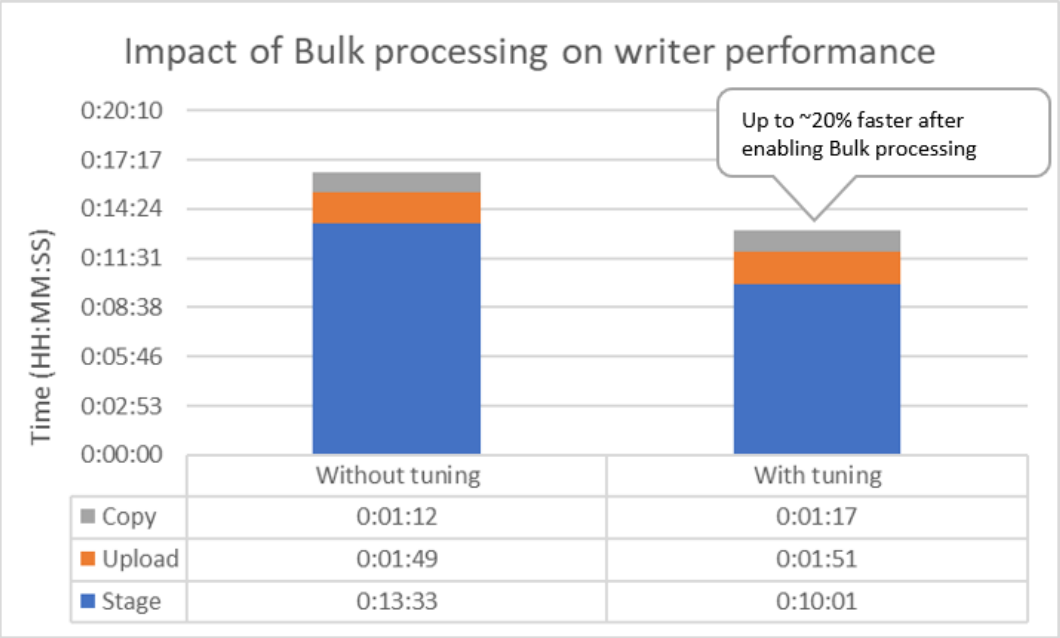
You can enable bulk processing to write large amounts of data to Google BigQuery. Bulk processing utilizes minimal number of API calls and the performance of the write operation is optimized.

To enable bulk processing, specify the property *-DENABLE_WRITER_BULK_PROCESSING=true* in the Secure Agent properties:

Perform the following steps to configure bulk processing before you run a mapping:

1. In Administrator, select the Secure Agent listed on the **Runtime Environments** tab.
2. Click **Edit**.
3. In the **System Configuration Details** section, select **Data Integration Server** as the service and **DTM** as the type.
4. Edit the JVM option, and enter *-DENABLE_WRITER_BULK_PROCESSING=true*.
5. Click **Save**.

The following graph illustrates the impact of partitioning on the writer performance:



Note: This feature does not apply for mapping tasks configured with pushdown optimization.

Configure Secure Agent concurrency

By default, the Secure Agent processes two concurrent mapping tasks.

If there are more than two concurrent tasks, additional tasks are queued and then scheduled for execution when a slot becomes available. This can cause the capacity of the Secure Agent machine to be underutilized.

To achieve better utilization of the CPU capacity of the Secure Agent machine and achieve a higher degree of concurrency, you can set the **maxDTMProcesses** custom property for the Data Integration Server to the number of parallel tasks.

To configure the Secure Agent concurrency, perform the following steps:

1. In Administrator, open the **Runtime Environments** page.
2. Select the Secure Agent to view its details, and then click Edit on the Secure Agent page.
3. Specify the following details:
   - Service. Choose **Data_Integration_Server**.
   - Type. Choose **Tomcat**.
   - Name. Enter **maxDTMProcesses**.
   - Value. Enter the Secure Agent concurrency value.
4. Click Save.

For example, if you set the maxDTMProcesses property to 16, the Secure Agent allows 16 mapping tasks to run simultaneously.

Note: Each mapping task spawns its own JVM and reserves the specified JVM heap from the physical memory.

Tune the Google BigQuery mapping

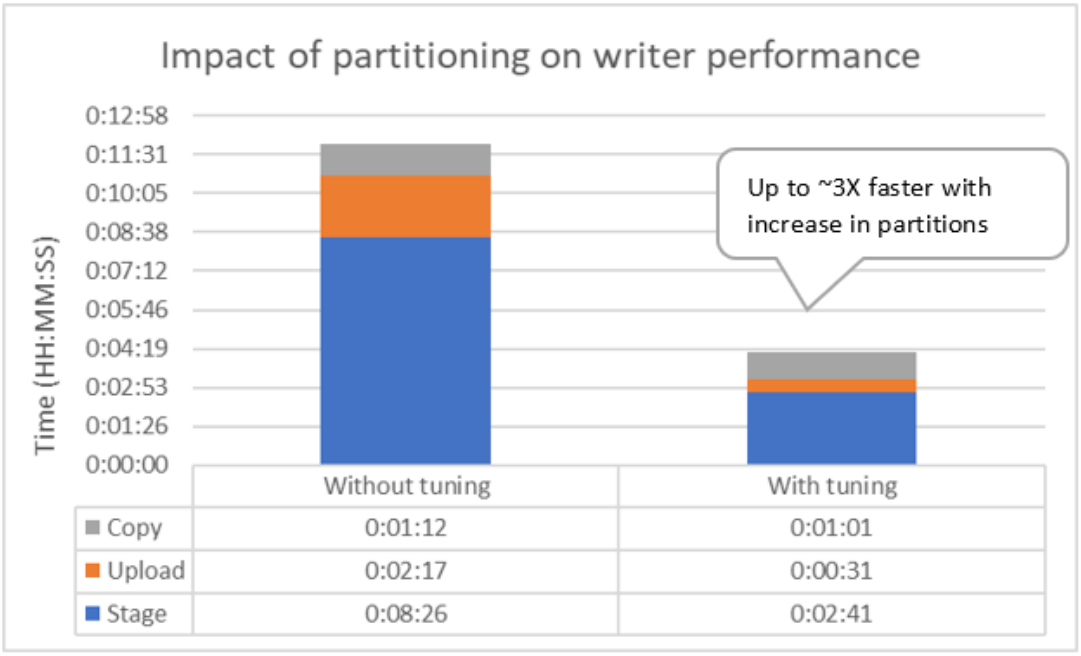
You can configure the following properties to optimize the Google BigQuery mapping performance:

- Partitions
- File Compression
- Data format of the staging file
- Read and write modes
- Poll interval

Partitions

You can configure partitions to optimize the mapping performance. Specify the number of partitions configured for the read or write operation in each mapping.

The following graph illustrates the impact of partitioning on the writer performance:



Compress files

If the Secure Agent and Google BigQuery are configured in different regions and there is a network latency, you can enable compression to compress the staged file to reduce the transfer time when you read data from or write data to Google BigQuery.

Number of threads for uploading staging file

You can configure the number of threads to optimize the performance of uploading the staging file when you write data to a Google BigQuery target in Bulk mode. Default value is 1.

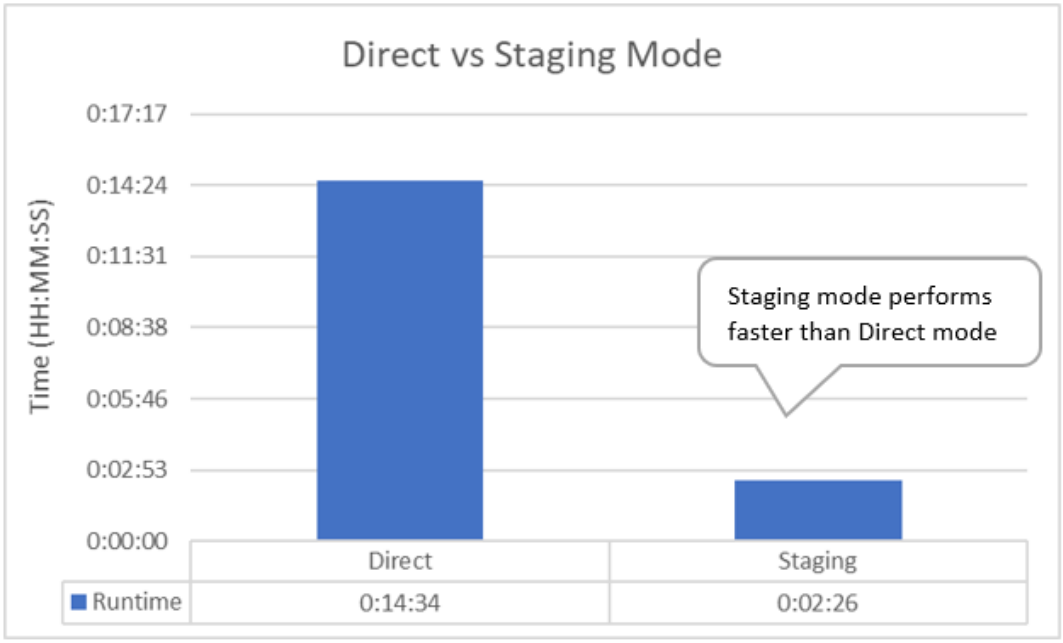
Informatica recommends that you set the value of the **Number of Threads for Uploading Staging file** target advanced property to 4 to optimize the performance to upload the staging file.

This property is only honored when the number of partitions configured for the source is set to 1.

Read modes for Google BigQuery sources

You can configure different modes to optimize the performance when you read data from a Google BigQuery source. If you want to read large volumes of data from a Google BigQuery source, you can specify the **Read mode** as **Staging** in the advanced source properties.

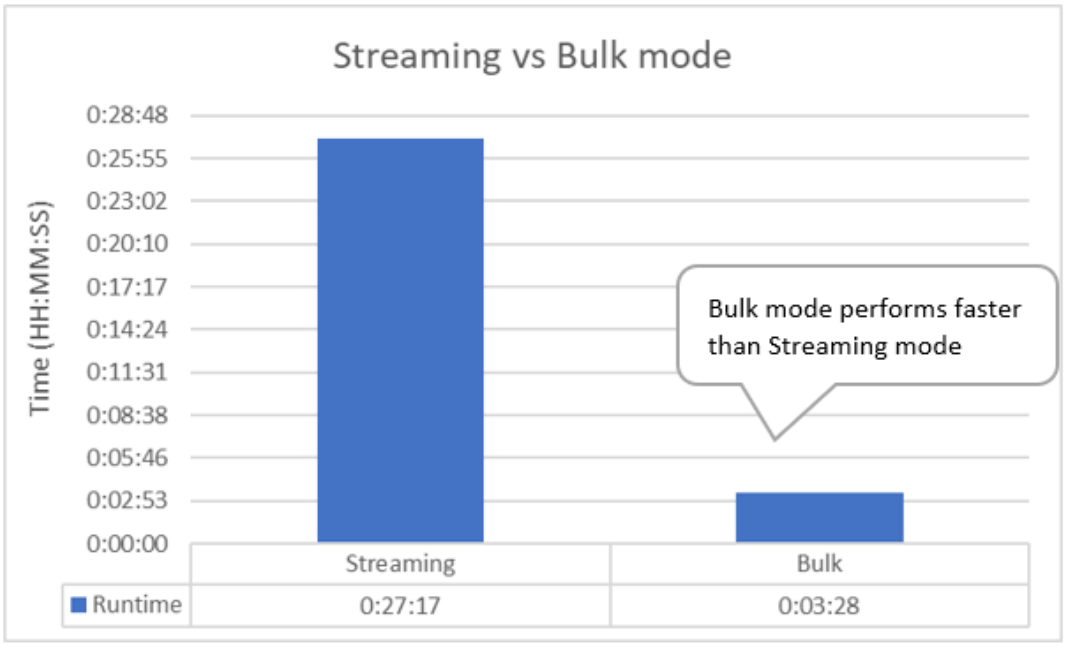
The following graph illustrates the impact of Read modes on the reader performance:



Write modes for Google BigQuery targets

You can configure different modes to optimize the performance when you write data to a Google BigQuery target. If you want to write large volumes of data to a Google BigQuery target, specify the **Write mode** as **Bulk** in the advanced target properties.

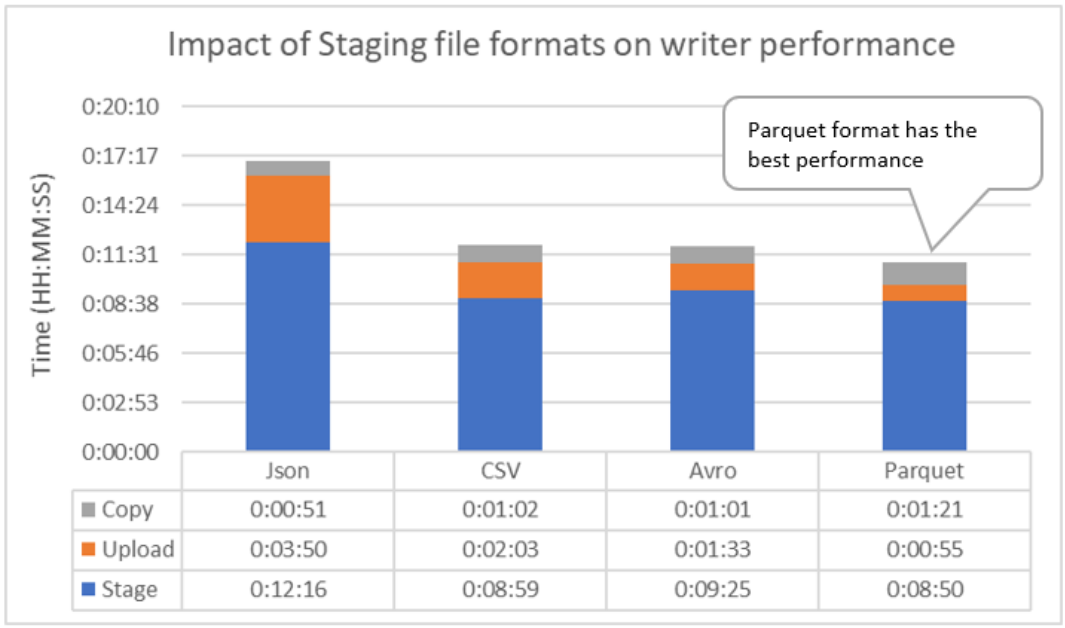
The following graph illustrates the impact of Write modes on the writer performance:



Data format of the staging file

You can optimize the performance of a Google BigQuery mapping by specifying the data format of local staging file in the Google BigQuery target transformation.

The following graph illustrates how different data formats for the staging files impact the writer performance:



Local flat file staging

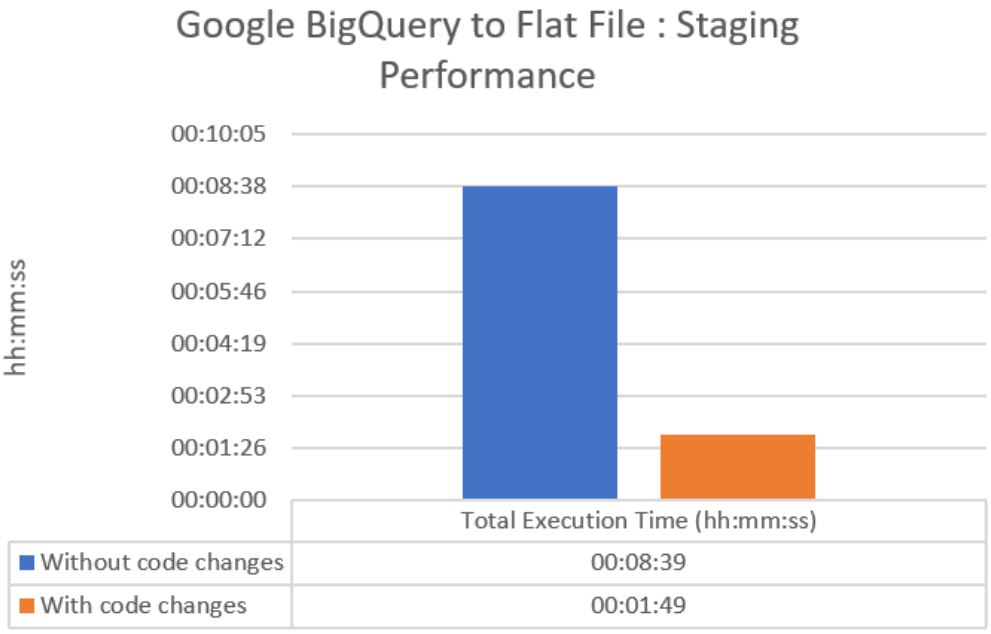
Informatica has redesigned the way Data Integration stages the data when you read from and write to a Google BigQuery target. This has significantly improved the performance in the local staging execution time.

You can optimize the mapping performance by setting the staging property in the Secure Agent to create a flat file in a temporary folder to stage the data locally before reading from or writing to Google BigQuery.

Reading from an Google BigQuery source

When you set the staging property `INFA_DTM_RDR_STAGING_ENABLED_CONNECTORS` for the Tomcat to the plugin ID of the Google BigQuery V2 Connector in the Secure Agent properties, Data Integration, by default, creates a flat file locally to stage the data and then reads the data from the staging file from the Google BigQuery source. For more information on how to set the property, see the help for Google BigQuery V2 Connector.

The following graph shows the performance comparison of the execution time before and after setting the staging property for a mapping that reads from a Google BigQuery source and writes to a flat file target:



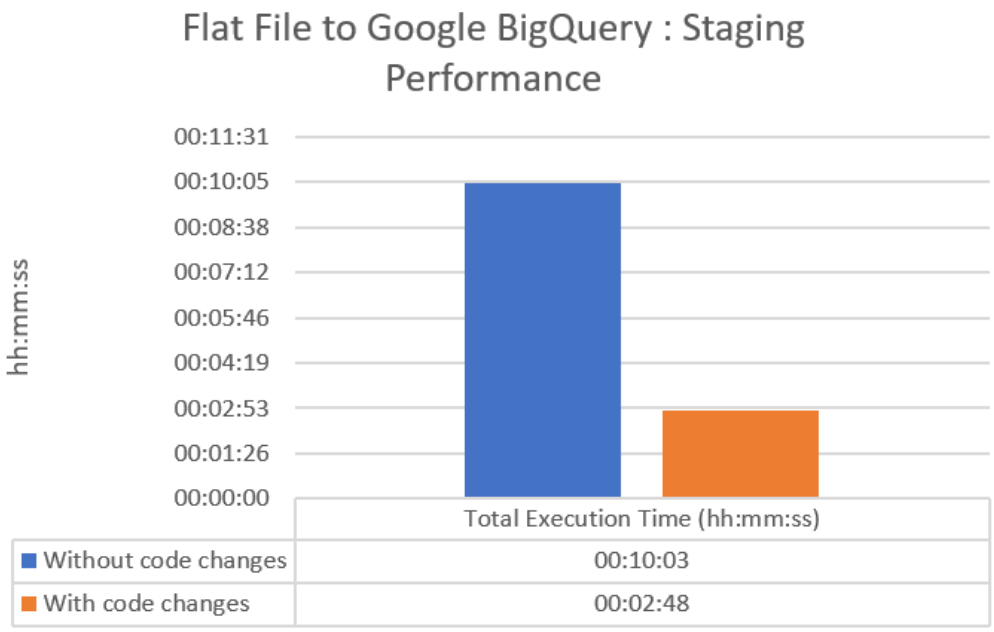
Writing to a Google BigQuery target

When you set the staging property `INFA_DTM_STAGING_ENABLED_CONNECTORS` for the Tomcat to the plugin ID of the Google BigQuery V2 Connector in the Secure Agent properties, Data Integration, by default, creates a flat file locally to stage the data and then loads the data from the staging file to the Google BigQuery target.

You can find the plugin ID in the manifest file located in the following directory: `<Secure Agent installation directory>/downloads/<Google BigQuery V2 package>/CCIManifest`.

For more information on how to set the property, see the help for Google BigQuery V2 Connector.

The following graph shows the performance comparison of the execution time before and after setting the staging property for a mapping that reads from a flat file source and writes to a Google BigQuery target:



Informatica recommends that you add three processor cores to the agent machine to increase the processing power of the mapping.

The following table shows the resource sizing in the old and new environment for a mapping that reads from a flat file source and writes to a Google BigQuery target:

Staging	CPU Requirement (cores)	Memory Requirement (GB)
Without code changes	3	2 GB (JVMHeap - 1 GB and DTM Buffer Size 1 GB)
With code changes	3	2 GB (JVMHeap - 1 GB and DTM Buffer Size 1 GB)

When you use three cores, the performance is optimized and the local staging execution time is far lesser. Though Informatica recommends that you include three cores for a Google BigQuery mapping execution, it is not mandatory. However, when you include lesser cores, the mapping performance is impacted but it does not result in execution failures.

Poll interval

When you use a custom query to read from or write to Google BigQuery, you can increase or decrease the poll interval size to improve the mapping performance.

You can specify the poll interval value in the source and target advanced attributes in a mapping based on the time that the query job takes. If a query job takes less than 7 to 8 seconds, you can reduce the poll interval. For queries that take longer than 7 seconds, increase the poll interval. If you reduce the poll interval, it causes additional overhead and impacts the mapping performance.

Authors

Akanksha Gauniyal

Manav Khurana