How-To Library

Informatica

Performance Tuning Guidelines for PowerExchange for Google Cloud Storage for Spark

Abstract

When you use Informatica PowerExchange for Google Cloud Storage to read data from or write data to Google Cloud Storage, multiple factors such as hardware parameters, database parameters, and application server parameters impact the performance of PowerExchange for Google Cloud Storage. You can optimize the performance by tuning these parameters appropriately. This article lists general reference guidelines to help you tune the performance of PowerExchange for Google Cloud Storage on Spark.

Supported Versions

- PowerExchange for Google Cloud Storage 10.2.2 Service Pack 1

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, and Hadoop cluster parameters that you can tune to optimize the performance of PowerExchange for Google Cloud Storage for Spark. The document also includes case studies that involve loading data into Google Cloud Storage from HDFS and reading data from Google Cloud Storage to HDFS using Amazon EMR 5.16.

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 PowerExchange for Google Cloud Storage might vary based on individual environments and other parameters even when you use the same data.

General Best Practices

To optimize the performance, consider the following best practices for Google Cloud Storage and Amazon EMR:

- Host Informatica domain on an Amazon EC2 virtual machine for better MRS monitoring.
- Launch the Amazon EMR cluster in the region where Google Cloud Storage is configured.
- Make sure that all the Spark level tuning is completed. For more information on tuning the Spark engine, see "Tune the Spark Engine" on page 4.
- If the Google Cloud Storage file contains a high precision decimal data type, change the precision for decimal data type to low precision to reduce the size of decimal data.
- Informatica recommends to use Avro as the file format when you read data from or write data to Google Cloud Storage for better performance.

Performance Tuning Parameters

You can optimize the performance of Google Cloud Storage mappings by tuning the following components:

- Tune the Hardware
- Tune the Spark Engine

Tune the Hardware

To optimize the performance, acquire the right type of hardware for your Big Data Management environment.

Cluster Configuration based on Dataset

You can tune the hardware by configuring the Hadoop cluster based on different data sets when you run mappings on the Spark engine.

Small Data Set

Small data set is defined as a system with one master node and two worker nodes that loads 7.5 GB of data from a Google Cloud Storage data source to a HDFS target in an Amazon EC2 instance and from an HDFS target in an Amazon EC2 instance to a Google Cloud Storage target in 00:12:30 (hh:mm:ss).

The following table shows the sizing outcome for default settings:

	Infrastructure	**Hadoop Services**
Services	Domain, MRS for Monitoring, DIS	Yarn Resource Manager, Node Manager
vCPUs	2	(4*1) + (4*2) = 12
Memory	8 GB	(4*1) + (12 *2) = 28 GB
Storage	14 GB	7.5 * 1 = 7.5 GB

Note: Total vCPUs = vCPUs of 1 Master + (vCPUs in core Node * No of Core Nodes).

Total Memory = (Master Node Process Memory + (Node Process Memory * Number of core Nodes)).

Total Storage = (I/P Data * Replication Factor)

Medium Data Set

Medium data set is defined as a system, with one master node and nine worker nodes, that loads 75 GB of data from a Google Cloud Storage data source to a HDFS target in an Amazon EC2 instance and from an HDFS target in an Amazon EC2 instance to a Google Cloud Storage target in 00:12:30 (hh:mm:ss).

The following table shows the sizing recommendations for a medium data set:

	Infrastructure	**Hadoop Services**
Services	Domain, MRS for Monitoring, DIS	Yarn Resource Manager, Node Manager
vCPUs	2	(4*1) + (4*9) = 40

	Infrastructure	Hadoop Services
Memory	8 GB	(4*1) + (12*9) = 112 GB
Storage	14 GB	75*1 = 75 GB

Large Data Set

Large data set is defined as a system, with one master node and 45 worker nodes, that loads 375 GB of data from a Google Cloud Storage data source to a HDFS target in an Amazon EC2 instance and from an HDFS target in an Amazon EC2 instance to a Google Cloud Storage target in 00:12:30 (hh:mm:ss).

The following table shows the sizing recommendation for a large data set:

	Infrastructure	Hadoop Services
Services	Domain, MRS for Monitoring, DIS	Yarn Resource Manager, Node Manager
vCPUs	2	(4*1) + (4*45) = 184
Memory	8 GB	(4*1) + (12*45) = 544 GB
Storage	14 GB	375*1 = 375 GB

Tune the Spark Engine

The following table lists the tuning recommendations for the Spark parameters:

Parameter	Default Value	Sandbox Dataset Recommendation	Small Dataset Recommendation	Medium Dataset Recommendation	Large Dataset Recommendation
spark.executor.memory	6 GB	1 GB	3 GB	3 GB	3 GB
spark.executor.cores	2	1	1	1	1
spark.executor.instances	10	10	8	30	150
spark.driver.memory	4 GB	1 GB	2 GB	2 GB	2 GB

The following list describes the Spark tuning parameters:

- spark.executor.memory. The memory to use per executor process.
- spark.executor.cores. The number of cores to use on each executor.
- spark.executor.instances. The initial number of executors to run if dynamic allocation is enabled.
- spark.driver.memory. The memory to use for the driver process where SparkContext is initialized.

Note: Any changes to the Spark parameters have to be done in the cluster configuration object. You need not restart Data Integration Service post the changes. The changes are expected to be picked automatically for the subsequent runs.

In a small, medium, or large data set, you might get the following error:

```
Container is running beyond physical memory limits.
```

To resolve this error, increase the value of `spark.executor.memory` parameter from 2GB to 4GB.

Authors

Krishna Prabhakar Devarakonda

Akanksha Gauniyal

Anupam Nayak