Cloudera Data Engineering 1.5.5

Accelerating Cloudera Data Engineering Jobs and Sessions using GPU

Date published: 2020-07-30
Date modified: 2025-11-08

CLOUDERA

https://docs.cloudera.com/

Legal Notice

Contents

Accelerating Cloudera Data Engineering Jobs and Sessions using GPUs (Technical Preview)

Cloudera Data Engineering supports accelerating the Spark jobs and sessions using GPUs. You can optionally choose GPU acceleration using Cloudera Data Engineering UI or CLI for a Spark job and session. The supported Spark version is Spark 3.

Note: This feature is in Technical Preview and not recommended for production deployments. Cloudera recommends that you try this feature in test or development environments.

You can leverage the power of GPUs to benefit from the faster execution times and reduced infrastructure costs without changing the existing Cloudera Data Engineering application code. By enabling GPU support, data engineers can make use of GPU resources available to the Cloudera Data Engineering service. You can configure GPU resource quota per virtual cluster which can be requested for running the Spark job or session.

Before you use GPUs to accelerate Cloudera Data Engineering jobs and sessions, you must ensure that the following are performed:

1. Have nodes with GPU and met the Software and Hardware requirements before installing Cloudera Data Services on premises.
2. Set up GPU nodes.
3. Test GPU node setup.
4. Set the GPU resource quota to allocate GPU resources effectively for Cloudera Data Engineering. GPU resources are limited in the cluster and usually shared among all data services.

Accelerating Cloudera Data Engineering Jobs and Sessions

You can accelerate your Spark jobs and Sessions using GPUs after you add GPU resources to the Cloudera Data Engineering service and virtual cluster.

Perform the following to use GPU resources:

1. Create a Cloudera Data Engineering service with GPU quota.
2. Create a virtual cluster in that Cloudera Data Engineering service with GPU quota.
3. Enable GPU acceleration when creating Spark jobs using Cloudera Data Engineering UI or when running jobs using CDE CLI.
4. Enable GPU acceleration when creating a session using Cloudera Data Engineering UI or CDE CLI.
5. Check whether the Spark job or session ran using the GPU resources.

Validating your Spark job or session run

After you run a job that utilizes the GPUs, you can verify if the GPUs were used. If you run any SQL query, you will observe GPU Filters in the SQL query plan in the "History Server UI" or in the Driver logs.

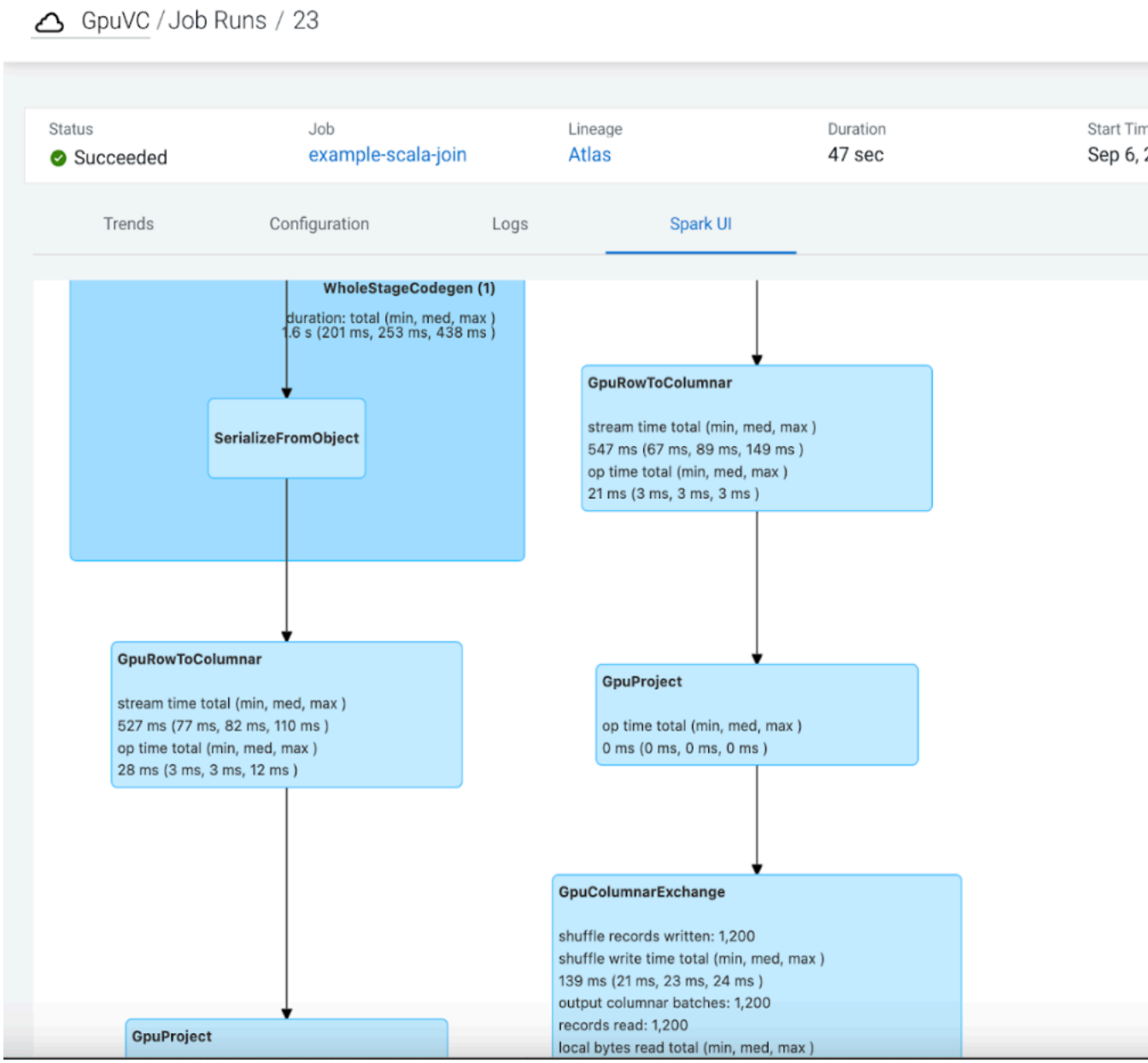
For Spark Job

You can check if the GPUs were utilized in the Job run using the SQL query.

1. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering **Home** page displays.
2. Click **Jobs Runs** on the left navigation menu. The **Job Runs** page displays.
3. Click on the job **Run Id** of the job which utilized GPU.

4. Click the **Spark UI** tab and click SQL. The SQL plan is displayed which shows that the job has utilized the GPUs.

   Note: This GPU details appears only if the job utilizes the GPU. Otherwise, job requests for GPU but never utilizes it.

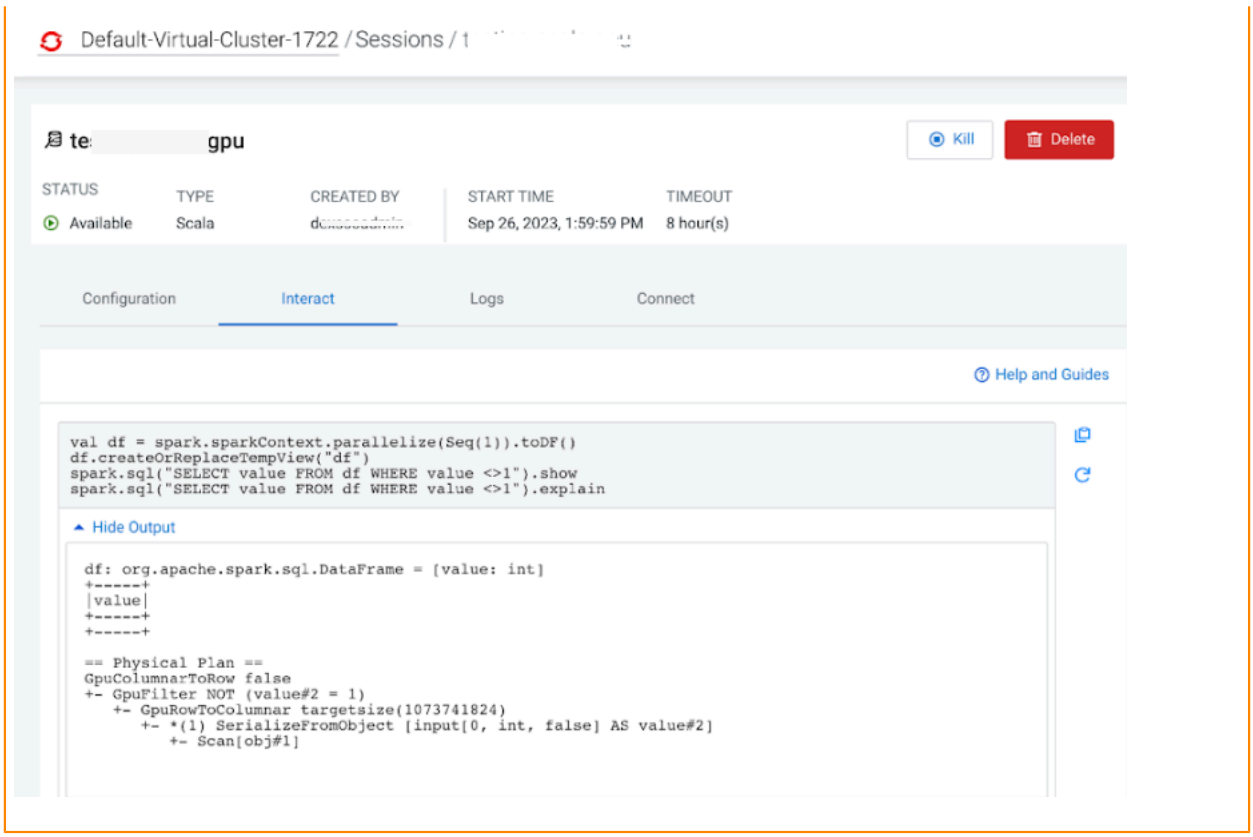


For Sessions

You can check if the GPUs were utilized by creating a session with Scala type. Open a session terminal and execute the below sample Scala code.

```
val df = spark.sparkContext.parallelize(Seq(1)).toDF()
df.createOrReplaceTempView("df")
spark.sql("SELECT value FROM df WHERE value <>1").show
spark.sql("SELECT value FROM df WHERE value <>1").explain
```

This should output a SQL plan with GPU Filter as below.


Default-Virtual-Cluster-1722 / Sessions / t
te gpu
Kill
Delete
STATUS
TYPE
CREATED BY
START TIME
TIMEOUT
Available
Scala
Sep 26, 2023, 1:59:59 PM
8 hour(s)
Configuration
Interact
Logs
Connect
Help and Guides
val df = spark.sparkContext.parallelize(Seq(1)).toDF()
df.createOrReplaceTempView("df")
spark.sql("SELECT value FROM df WHERE value <>1").show
spark.sql("SELECT value FROM df WHERE value <>1").explain
Hide Output
df: org.apache.spark.sql.DataFrame = [value: int]
+-----+
|value|
+-----+
+-----+
== Physical Plan ==
GpuColumnarToRow false
+- GpuFilter NOT (value#2 = 1)
   +- GpuRowToColumnar targetsize(1073741824)
      +- *(1) SerializeFromObject [input[0, int, false] AS value#2]
         +- Scan[obj#1]