Cloudera Runtime 7.1.7

CDS 3 Powered by Apache Spark

Date published: 2021-02-29
Date modified: 2022-01-12

https://docs.cloudera.com/

Legal Notice

Contents

CDS 3.2.3 Powered by Apache Spark Overview

Apache Spark is a general framework for distributed computing that offers high performance for both batch and interactive processing. It exposes APIs for Java, Python, and Scala.

For detailed API information, see the Apache Spark project site.

CDS 3.2.3 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, distributed as a parcel and custom service descriptor.

This document describes CDS 3.2.3 Powered by Apache Spark. It enables you to install and evaluate the features of Apache Spark 3 without upgrading your CDP Private Cloud Base cluster.

On CDP Private Cloud Base, a Spark 3 service can coexist with the existing Spark 2 service. The configurations of the two services do not conflict and both services use the same YARN service. The port of the Spark History Server is 18088 for Spark 2 and 18089 for Spark 3.

CDS 3 for GPUs

CDS 3.2.3 with GPU Support is an add-on service that enables you to take advantage of the RAPIDS Accelerator for Apache Spark to accelerate Apache Spark 3 performance on existing CDP Private Cloud Base clusters.

Unsupported connectors

This release does not support the following connectors:

- Phoenix
- SparkR
- Hive Warehouse
- Oozie
- Zeppelin

Unsupported Features

This release does not support the following feature:

- Hudi

Limitations of Spark in CDP

Limitations of Spark (in comparison to Apache Spark 3.2) in CDP are described below:

- spark.sql.orc.compression.codec config doesn't accept zsdt value.
- spark.sql.avro.compression.codec config doesn't accept zstandard value.
- Specifying avroSchemaUrl is not supported in datasource options.
- Spark3 parcel is not available on the IBM PowerPC platform.

CDS 3.2.3 Powered by Apache Spark Requirements

The following sections describe software requirements for CDS 3.2.3 Powered by Apache Spark.

CDP Versions

Important: CDS 3.2.3 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Supported versions of CDP are described below.

CDS Powered by Apache Spark Version	Supported CDP Versions
3.2.3.3.2.7172000.0-334	CDP Private Cloud Base with Cloudera Runtime 7.1.7

A Spark 2 service (included in CDP) can co-exist on the same cluster as Spark 3 (installed as a separate parcel). The two services are configured to not conflict, and both run on the same YARN service. Spark 3 installs and uses its own external shuffle service.

Although Spark 2 and Spark 3 can coexist in the same CDP Private Cloud Base cluster, you cannot use multiple Spark 3 versions simultaneously. All clusters managed by the same Cloudera Manager Server must use exactly the same version of CDS Powered by Apache Spark.

Software requirements

For CDS 3.2

Each cluster host must have the following software installed:

Java

JDK 8 or JDK 11. Cloudera recommends using JDK 8, as most testing has been done with JDK 8. Remove other JDK versions from all cluster and gateway hosts to ensure proper operation.

Python

Python 3.6 and higher

For CDS for GPUs

Each cluster host with a GPU must have the following software installed:

Java

JDK 8 or JDK 11. Cloudera recommends using JDK 8, as most testing has been done with JDK 8. Remove other JDK versions from all cluster and gateway hosts to ensure proper operation.

Python

Python 3.6 and higher

GPU drivers and CUDA toolkit

GPU driver v450.80.02 or higher

CUDA version 11.0 or higher

Download and install the CUDA Toolkit for your operating system. The toolkit installer also provides the option to install the GPU driver.

NVIDIA Library

NVIDIA RAPIDS version 22.02. For more information, see NVIDIA Release Notes

UCX (Optional)

Clusters with Infiniband or RoCE networking can leverage Unified Communication X (UCX) to enable the RAPIDS Shuffle Manager. For information on UCX native libraries support, see (Optional) Installing UCX native libraries.

Hardware requirements

For CDS 3.2

CDS 3.2.3 Powered by Apache Spark has no specific hardware requirements on top of what is required for Cloudera Runtime deployments.

For CDS for GPUs

CDS 3.2.3 with GPU Support requires cluster hosts with NVIDIA Pascal™ or better GPUs, with a compute capability rating of 6.0 or higher.

For more information, see Getting Started at the RAPIDS website.

Cloudera and NVIDIA recommend using NVIDIA-certified systems. For more information, see NVIDIA-Certified Systems in the NVIDIA GPU Cloud documentation.

Installing CDS 3.2.3 Powered by Apache Spark

CDS 3.2.3 Powered by Apache Spark (CDS 3.2.3) is distributed as two files: a custom service descriptor file and a parcel, both of which must be installed on the cluster.

Note: Due to the potential for confusion between CDS Powered by Apache Spark and the initialism CSD, references to the custom service descriptor (CSD) file in this documentation use the term service descriptor.

Install CDS 3.2.3 Powered by Apache Spark

Note:

Although Spark 2 and Spark 3 can coexist in the same CDP Private Cloud Base cluster, you cannot use multiple Spark 3 versions simultaneously. All clusters managed by the same Cloudera Manager Server must use exactly the same version of CDS 3.2.3 Powered by Apache Spark.

Follow these steps to install CDS 3.2.3:

1. Check that all the software prerequisites are satisfied. If not, you might need to upgrade or install other software components first.
2. Install the CDS 3.2.3 service descriptor into Cloudera Manager.
   a. To download the CDS 3.2.3 service descriptor, click the service descriptor link for the version you want to install.
   b. Log on to the Cloudera Manager Server host, and copy the CDS Powered by Apache Spark service descriptor in the location configured for service descriptor files.
   c. Set the file ownership of the service descriptor to cloudera-scm:cloudera-scm with permission 644.
   d. Restart the Cloudera Manager Server with the following command:

   ```
   systemctl restart cloudera-scm-server
   ```
3. In the Cloudera Manager Admin Console, add the CDS parcel repository to the Remote Parcel Repository URLs in Parcel Settings as described in Parcel Configuration Settings.

   Note: If your Cloudera Manager Server does not have Internet access, you can use the CDS Powered by Apache Spark parcel files: put them into a new parcel repository, and then configure the Cloudera Manager Server to target this newly created repository.
4. Download the CDS 3.2.3 parcel, distribute the parcel to the hosts in your cluster, and activate the parcel. For instructions, see Managing Parcels.
5. Add the Spark 3 service to your cluster.
   a. In step 1, select any optional dependencies, such as HBase and Hive, or select No Optional Dependencies.
   b. In step 2, when customizing the role assignments, add a gateway role to every host.
   c. On the Review Changes page, you can enable TLS for the Spark History Server.
   d. Note that the History Server port is 18089 instead of the usual 18088.
   e. Complete the remaining steps in the wizard.
6. Return to the Home page by clicking the Cloudera Manager logo in the upper left corner.
7. Click the stale configuration icon to launch the Stale Configuration wizard and restart the necessary services.

Install the Livy for Spark 3 Service Descriptor

CDS 3 supports Apache Livy, but it cannot use the included Livy service, which is compatible with only Spark 2. To add and manage a Livy service compatible with Spark 3, you must install a service descriptor for the Livy for Spark 3 service.

1. Install the Livy for Spark 3 service descriptor into Cloudera Manager.
   a. To download the service descriptor, click the service descriptor link for the version you want to install.
   b. Log on to the Cloudera Manager Server host, and copy the Livy service descriptor to the location configured for service descriptor files.
   c. Set the file ownership of the service descriptor to cloudera-scm:cloudera-scm with permissions set to 644.
   d. Restart the Cloudera Manager Server with the following command:

   ```
   systemctl restart cloudera-scm-server
   ```
2. In the Cloudera Manager Admin Console, add the CDS 3 parcel repository to the Remote Parcel Repository URLs in Parcel Settings as described in Parcel Configuration Settings.

   Note: If your Cloudera Manager Server does not have Internet access, you can use the Livy parcel files: put them into a new parcel repository, and then configure the Cloudera Manager Server to target this newly created repository.
3. Download the CDS 3.2.3 parcel, distribute the parcel to the hosts in your cluster, and activate the parcel. For instructions, see Managing Parcels.
4. Add the Livy for Spark 3 service to your cluster.
   a. Note that the Livy port is 28998 instead of the usual 8998.
   b. Complete the remaining steps in the wizard.
5. Return to the Home page by clicking the Cloudera Manager logo in the upper left corner.
6. Click the stale configuration icon to launch the Stale Configuration wizard and restart the necessary services.

If you want to activate the CDS 3.2.3 with GPU Support feature, Set up a Yarn role group to enable GPU usage on page 10 and optionally Configure NVIDIA RAPIDS Shuffle Manager on page 10

Enabling Spark rolling event log files in CDP

While running a long-running Spark application in CDP, (for example, a streaming application), the Spark Job generates a large single event log file until the Spark application is killed or stopped. A single event log file is not cost effective, requires high maintenance, and is resource-intensive. Therefore, to avoid creating a large event log file, you can use a rolling event log file.

About this task

If the Spark application is still running, only the spark.history.fs.eventLog.rolling.maxFilesToRetain / spark.history.fs.eventLog.rolling.compaction.score.threshold parameters are considered (the spark.history.fs.cleaner.maxAge parameter is not used in this case).

Procedure

1. Navigate to Cloudera Manager > Spark 3 > Configuration > Spark 3 Client Advanced Configuration Snippet (Safety Valve) for spark3-conf/spark-defaults.conf and set the following properties:

   spark.eventLog.rolling.enabled=true

   spark.eventLog.rolling.maxFileSize=128m

   The default spark.eventLog.rolling.maxFileSize value is 128 MB. The minimum value allowed is 10 MB.

2. Set the maximum number of rolling event log files to retain:
   a) Navigate to Cloudera Manager > Spark 3> Configuration > History Server Advanced Configuration Snippet (Safety Valve) for spark3-conf/spark-history-server.conf and set the following property:

      spark.history.fs.eventLog.rolling.maxFilesToRetain=2

      By default, spark.history.fs.eventLog.rolling.maxFilesToRetain value is infinity, meaning all event log files are retained. Therefore, this parameter needs to be specified to avoid retaining more files. The minimum value allowed is 1.
3. Verify the output from the Spark history server event log directory. An example output is displayed below:

Results

```
[root@c3543-node4 ~]# sudo -u spark hdfs dfs -ls -R /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002
-rw-rw---- 3 spark spark 0 2023-01-04 07:03 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
appstatus_application_1672813574470_0002.inprogress -rw-
rw---- 3 spark spark 10485458 2023-01-04 07:05 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_1_application_1672813574470_0002 -rw-rw---- 3 spark
spark 0 2023-01-04 07:05 /user/spark/spark3ApplicationHistory/
eventlog_v2_application_1672813574470_0002/
events_2_application_1672813574470_0002 [root@c3543-
node4 ~]# sudo -u spark hdfs dfs -ls -R /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002
-rw-rw---- 3 spark spark 0 2023-01-04 07:03 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
appstatus_application_1672813574470_0002.inprogress -rw-
rw---- 3 spark spark 492014 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_1_application_1672813574470_0002.compact -rw-
rw---- 3 spark spark 10489509 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_2_application_1672813574470_0002 -rw-rw----
3 spark spark 227068 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_3_application_1672813574470_0002 [root@c3543-
node4 ~]# sudo -u spark hdfs dfs -ls -R /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002
-rw-rw---- 3 spark spark 0 2023-01-04 07:03 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
appstatus_application_1672813574470_0002.inprogress -rw-
rw---- 3 spark spark 873356 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_2_application_1672813574470_0002.compact -rw-
rw---- 3 spark spark 10484816 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_3_application_1672813574470_0002 -rw-rw----
3 spark spark 339165 2023-01-04 07:06 /user/spark/
spark3ApplicationHistory/eventlog_v2_application_1672813574470_0002/
events_4_application_1672813574470_0002
```

Enabling CDS 3.2.3 with GPU Support

To activate the CDS 3.2.3 with GPU Support feature on suitable hardware, you need to create a Yarn role group and optionally make configuration changes to enable the NVIDIA RAPIDS Shuffle Manager.

Set up a Yarn role group to enable GPU usage

Create a Yarn role group so that you can selectively enable GPU usage for nodes with GPUs within your cluster.

Before you begin

GPU scheduling and isolation must be configured.

About this task

Enabling GPU on YARN through the Enable GPU Usage tickbox operates on cluster-level. Role groups in Yarn enable you to apply settings selectively, to a subset of nodes within your cluster.

Role groups are configured on the service level.

Procedure

1. In Cloudera Manager navigate to Yarn Instances .
2. Create a role group where you can add nodes with GPUs.

   For more information, see Creating a Role Group.
3. Move role instances with GPUs to the group you created.

   On the Configuration tab select the source role group with the hosts you want to move, then click Move Selected Instances To Group and select the role group you created.

   You may need to restart the cluster.
4. Enable GPU usage for the role group.
   a) On the Configuration tab select Categories GPU Management .
   b) Under GPU Usage click Edit Individual Values and select the role group you created.
   c) Click Save Changes.

Configure NVIDIA RAPIDS Shuffle Manager

The NVIDIA RAPIDS Shuffle Manager is a custom ShuffleManager for Apache Spark that allows fast shuffle block transfers between GPUs in the same host (over PCIe or NVLink) and over the network to remote hosts (over RoCE or Infiniband).

About this task

NVIDIA RAPIDS Shuffle Manager has been shown to accelerate workloads where shuffle is the bottleneck when using the RAPIDS accelerator for Apache Spark. It accomplishes this by using a GPU shuffle cache for fast shuffle writes when shuffle blocks fit in GPU memory, avoiding the cost of writes to host using the built-in Spark Shuffle, a spill framework that will spill to host memory and disk on demand, and Unified Communication X (UCX) as its transport for fast network and peer-to-peer (GPU-to-GPU) transfers.

CDS 3.2.3 with GPU Support has built in support for UCX, no separate installation is required.

Cloudera and NVIDIA recommend using the RAPIDS shuffle manager for clusters with Infiniband or RoCE networking.

Procedure

1. Validate your UCX environment following the instructions provided in the NVIDIA spark-rapids documentation.
2. Before running applications with the RAPIDS Shuffle Manager, make the following configuration changes:

```
--conf "spark.shuffle.service.enabled=false" \
--conf "spark.dynamicAllocation.enabled=false"
```

3. You are recommended to make the following UCX settings:

```
spark.executorEnv.UCX_TLS=cuda_copy,cuda_ipc,rc,tcp
spark.executorEnv.UCX_RNDV_SCHEME=put_zcopy
spark.executorEnv.UCX_MAX_RNDV_RAILS=1
spark.executorEnv.UCX_IB_RX_QUEUE_LEN=1024
```

For more information on environment variables, see the NVIDIA spark-rapids documentation.

Note: Running a job with the --rapids-shuffle=true flag does not affect these optional settings. You need to set them manually.

Updating Spark 2 applications for Spark 3

You must update your Apache Spark 2 applications to run on Spark 3. The Apache Spark documentation provides a migration guide.

For instructions on updating your Spark 2 applications for Spark 3, see the migration guide in the Apache Spark documentation.

Running Applications with CDS 3.2.3 Powered by Apache Spark

With CDS 3.2.3 Powered by Apache Spark (CDS 3.2.3), you can run Apache Spark 3 applications locally or distributed across a cluster, either by using an interactive shell or by submitting an application. Running Spark applications interactively is commonly performed during the data-exploration phase and for ad hoc analysis.

The Spark 3 job commands

With Spark 3, you use slightly different command names than with Spark 2, so that you can run both versions of Spark side-by-side without conflicts:

- spark3-submit instead of spark-submit.
- spark3-shell instead of spark-shell.
- pyspark3 instead of pyspark.

For development and test purposes, you can also configure each host so that invoking the Spark 2 command name runs the corresponding Spark 3 executable.

Canary test for pyspark3 command

The following example shows a simple pyspark3 session that refers to the SparkContext, calls the collect() function which runs a Spark 3 job, and writes data to HDFS. This sequence of operations helps to check if there are obvious

configuration issues that prevent Spark 3 jobs from working at all. For the HDFS path for the output directory, substitute a path that exists on your own system.

```
$ hdfs dfs -mkdir /user/jdoe/spark
$ pyspark3
...
SparkSession available as 'spark'.
>>> strings = ["one","two","three"]
>>> s2 = sc.parallelize(strings)
>>> s3 = s2.map(lambda word: word.upper())
>>> s3.collect()
['ONE', 'TWO', 'THREE']
>>> s3.saveAsTextFile('hdfs:///user/jdoe/spark/canary_test')
>>> quit()
$ hdfs dfs -ls /user/jdoe/spark
Found 1 items
drwxr-xr-x   - jdoe spark-users  0 2016-08-26 14:41 /user/jdoe/spark/canary_
test
$ hdfs dfs -ls /user/jdoe/spark/canary_test
Found 3 items
-rw-r--r--   3 jdoe spark-users  0 2016-08-26 14:41 /user/jdoe/spark/cana
ry_test/_SUCCESS
-rw-r--r--   3 jdoe spark-users  4 2016-08-26 14:41 /user/jdoe/spark/canary
_test/part-00000
-rw-r--r--   3 jdoe spark-users 10 2016-08-26 14:41 /user/jdoe/spark/canary
_test/part-00001
$ hdfs dfs -cat /user/jdoe/spark/canary_test/part-00000
ONE
$ hdfs dfs -cat /user/jdoe/spark/canary_test/part-00001
TWO
THREE
```

Fetching Spark 3 Maven Dependencies

The Maven coordinates are a combination of groupId, artifactId and version. The groupId and artifactId are the same as for the upstream Apache Spark project. For example, for spark-core, groupId is org.apache.spark, and artifactId is spark-core_2.12, both the same as the upstream project. The version is different for the Cloudera packaging: see Using the CDS 3 Powered by Apache Spark Maven Repository for the exact name depending on which release you are using.

Accessing the Spark 3 History Server

The Spark 3 history server is available on port 18089, rather than port 18088 as with the Spark 2 history server.

Running applications using CDS 3.2.3 with GPU SupportCDS 3.2.3 with GPU Support

1. Log on to the node where you want to run the job.
2. Run the following command to launch spark3-shell:

```
spark3-shell  --conf "spark.rapids.sql.enabled=true" \
```

```
                --conf "spark.executor.memoryOverhead=5g"
```

where

--conf spark.rapids.sql.enabled=true

enables the following environment variables for GPUs:

```
"spark.task.resource.gpu.amount" - sets GPU resource amount per
task

"spark.rapids.sql.concurrentGpuTasks" - sets the number of con
current tasks per GPU
"spark.sql.files.maxPartitionBytes" - sets the input partition
 size for DataSource API, The recommended value is "256m".
"spark.locality.wait" - controls how long Spark waits to obtain b
etter locality for tasks.
"spark.sql.adaptive.enabled" - enables Adaptive Query Execution.
"spark.rapids.memory.pinnedPool.size" - sets the amount of pinned
 memory allocated per host.
"spark.sql.adaptive.advisoryPartitionSizeInBytes" - sets the ad
visory size in bytes of the shuffle partition during adaptive op
timization.
```

For example,

```
$SPARK_HOME/bin/spark3-shell \
       --conf spark.task.resource.gpu.amount=2 \
       --conf spark.rapids.sql.concurrentGpuTasks=2 \
       --conf spark.sql.files.maxPartitionBytes=256m \
       --conf spark.locality.wait=0s \
       --conf spark.sql.adaptive.enabled=true \
       --conf spark.rapids.memory.pinnedPool.size=2G \
       --conf spark.sql.adaptive.advisoryPartitionSizeInBytes=1g
```

--conf "spark.executor.memoryOverhead=5g"

sets the amount of additional memory to be allocated per executor process

Note: cuDF uses a Just-In-Time (JIT) compilation approach for some kernels, and the JIT process can add a few seconds to query wall-clock time. You are recommended to set a JIT cache path to speed up subsequent invocations with: --conf spark.executorEnv.LIBCUDF_KERNEL_CACHE_PATH=[local path]. The path should be local to the executor (not HDFS) and not shared between different cluster users in a multi-tenant environment. For example, the path may be in /tmp: (/tmp/cudf-[***USER***]). If the/tmp directory is not writable, consult your system administrator to find a path that is.

You can override these configuration settings both from the command line and from code. For more information on environment variables, see the NVIDIA spark-rapids documentation and the Spark SQL Performance Tuning Guide.

3. Run a job in spark3-shell.

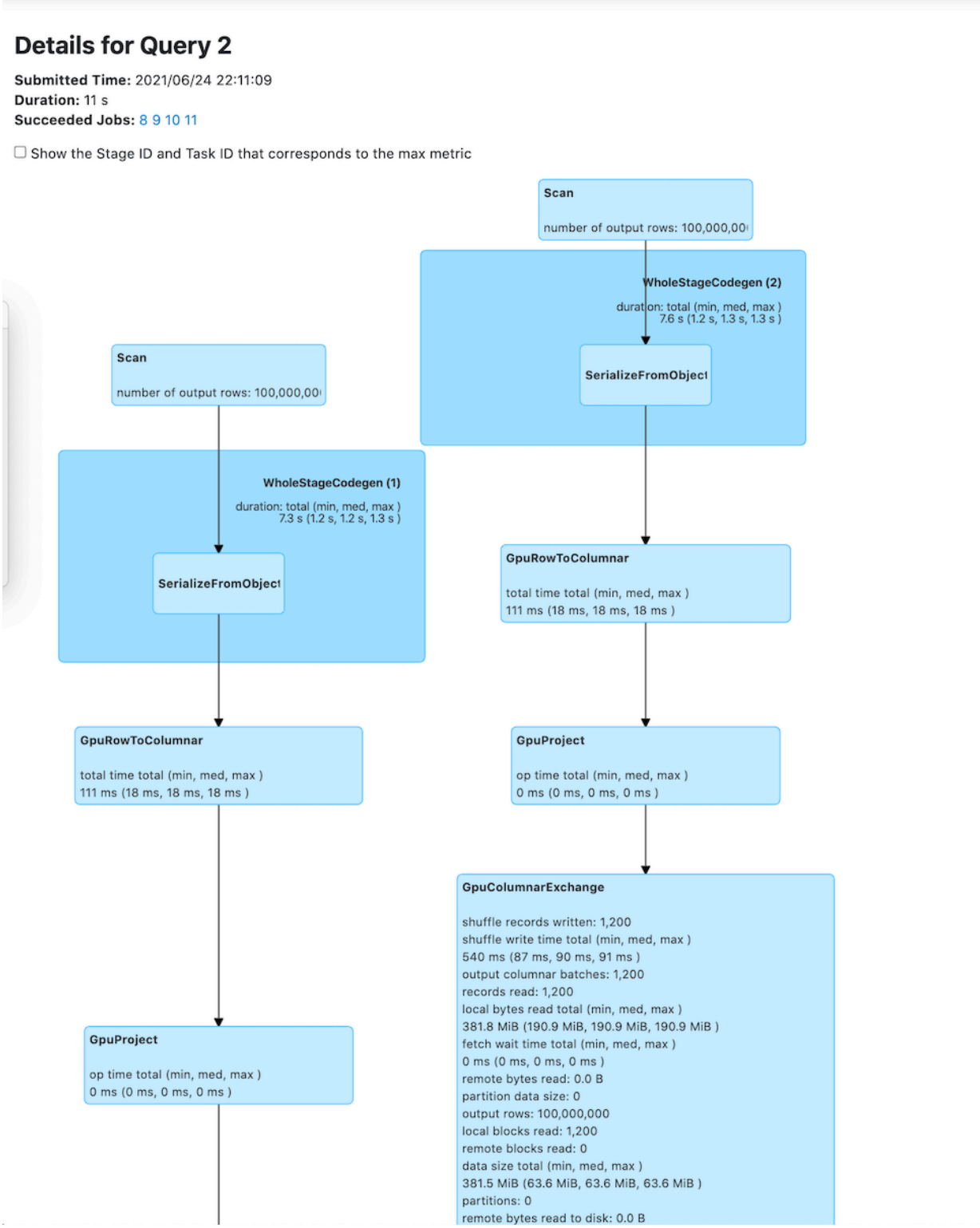
   For example:

```
scala> val df = sc.makeRDD(1 to 100000000, 6).toDF
df: org.apache.spark.sql.DataFrame = [value: int]

scala>val df2 = sc.makeRDD(1 to 100000000, 6).toDF
df2: org.apache.spark.sql.DataFrame = [value: int]
scala> df.select($"value" as "a").join(df2select($"value" as "b"), $"a"
 === $"b").count
res0: Long = 100000000
```

4. You can verify that the job run used GPUs, by logging on to the Yarn UI v2 to review the execution plan and the performance of your spark3-shell application:

   Select the Applications tab then select your [spark3-shell application]. Select ApplicationMaster SQL count at <console>:28 to see the execution plan.



Running a Spark job using CDS 3.2.3 with GPU Support with UCX enabled

1. Log on to the node where you want to run the job.
2. Run the following command to launch spark3-shell:

```
spark3-shell  --conf "spark.rapids.sql.enabled=true" \
              --conf "spark.executor.memoryOverhead=5g"
              --rapids-shuffle=true
```

where
--rapids-shuffle=true

makes the following configuration changes for UCX:

```
spark.shuffle.manager=com.nvidia.spark.rapids.spark320cdh.Rapid
sShuffleManager
spark.executor.extraClassPath=/opt/cloudera/parcels/SPARK3/lib/s
park3/rapids-plugin/*
spark.executorEnv.UCX_ERROR_SIGNALS=
spark.executorEnv.UCX_MEMTYPE_CACHE=n
```

Important: If you using Cloudera Runtime 7.1.7 Service Pack 1 or 2, you must provide the value of spark.shuffle.manager as com.nvidia.spark.rapids.spark321cdh.RapidsShuffleManager.

For more information on environment variables, see the NVIDIA spark-rapids documentation.

3. Run a job in spark3-shell.

CDS 3.2.3 Powered by Apache Spark version and download information

The following section provides links to the parcel and service descriptor files for both CDS 3.2.3 and CDS 3.2.3 with GPU Support.

Note:

The CDS version label is constructed in v.v.v.w.w.xxxx.y-z...z format and carries the following information:

v.v.v
Apache Spark upstream version, for example: 3.2.0

w.w
Cloudera internal version number, for example: 3.2

xxxx
CDP version number, for example: 7170 (referring to CDP Private Cloud Base 7.1.7)

y
maintenance version, for example, 1

z...z
build number, for example: 279

Table 1: Available CDS Versions

Version	Custom Service Descriptors	Parcel Repository
3.2.3.3.2.7172000.0-334	• SPARK3_ON_YARN-3.2.3.3.2.7172000.0-334.jar • LIVY_FOR_SPARK3-0.6.3000.3.2.7172000.0-334.jar	https://archive.cloudera.com/p/spark3/3.2.7172000.0/parcels/

Using the CDS 3.2.3 Powered by Apache Spark Maven Repository

Important: CDS 3.2.3 Powered by Apache Spark (CDS 3.2.3) does not include an assembly JAR. When you build an application JAR, do not include Cloudera Runtime or CDS JARs, because they are already provided. If you do, upgrading Cloudera Runtime or CDS can break your application. To avoid this situation, set the Maven dependency scope to provided. If you have already built applications which include the Cloudera Runtime or CDS JARs, update the dependency to set scope to provided and recompile.

If you want to build applications or tools for use with CDS 3.2.3, and you are using Maven or Ivy for dependency management, you can pull the CDS artifacts from the Cloudera Maven repository. The repository is available at https://repository.cloudera.com/artifactory/cloudera-repos/.

The following is a sample POM (pom.xml) file:

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.
org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache
.org/maven-v4_0_0.xsd">
  <repositories>
    <repository>
      <id>cloudera</id>
      <url>https://repository.cloudera.com/artifactory/cloudera-repos/</url>
    </repository>
  </repositories>
</project>
```

CDS 3.2.3 Powered by Apache Spark Maven Artifacts

The following table lists the groupId, artifactId, and version required to access the artifacts for CDS 3.2.3 Powered by Apache Spark:

POM fragment

The following pom fragment shows how to access a CDS 3.2.3 artifact from a Maven POM.

```
<dependency>
      <groupId>org.apache.spark</groupId>
      <artifactId>spark-core_2.12</artifactId>
      <version>3.2.3.3.2.7172000.0-334</version>
      <scope>provided</scope>
</dependency>
```

The complete artifact list for this release follows.

List of CDS 3.2.3 Maven artifacts

Project	groupId	artifactId	version
Apache Spark	org.apache.spark	spark-assembly_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-avro_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-catalyst_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-core_2.12	3.2.3.3.2.7172000.0-334

Project	groupId	artifactId	version
	org.apache.spark	spark-cypher_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-graph-api_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-graph_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-graphx_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-hadoop-cloud_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-hive-thriftserver_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-hive_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-kubernetes_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-kvstore_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-launcher_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-mllib-local_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-mllib_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-network-common_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-network-shuffle_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-network-yarn_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-parent_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-repl_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-sketch_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-sql-kafka-0-10_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-sql_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-streaming-kafka-0-10-assembly_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-streaming-kafka-0-10_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-streaming_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-tags_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-token-provider-kafka-0-10_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-unsafe_2.12	3.2.3.3.2.7172000.0-334
	org.apache.spark	spark-yarn_2.12	3.2.3.3.2.7172000.0-334

Cumulative hotfixes for CDS

You can review the list of cumulative hotfixes that were shipped for CDS.

Cumulative hotfix CDS 3.2.7172000.3-3 (CDS 3.2 CHF1 for 7.1.7)

Know more about CDS 3.2 CHF1 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on March 14, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.3-3-1.p0.38870303:

- CDPD-39575: Livy - Always close RSCClient's EventLoopGroup
- CDPD-21614: Spark SQL TRUNCATE table not permitted on external purge tables

Table 2: CDS cumulative hotfix 3.2.7172000.3-3 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.3/parcels/`

Table 3: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.3-3	Cloudera Runtime 7.1.7.2000 (7.1.7 SP2 GA)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.6-1 (CDS 3.2 CHF2 for 7.1.7)

Know more about CDS 3.2 CHF2 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on April 18, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.6-1-1.p0.40059346:

- CDPD-55150: Backport LIVY-763

Table 4: CDS cumulative hotfix 3.2.7172000.6-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.6/parcels/`

Table 5: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.6-1	Cloudera Runtime 7.1.7.2013 (7.1.7 SP2 CHF5)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.8-1 (CDS 3.2 CHF3 for 7.1.7)

Know more about CDS 3.2 CHF3 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on May 18, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.8-1-1.p0.41069953:

- CDPD-68752: [CDS 3.2] Backport SPARK-36163

Table 6: CDS cumulative hotfix 3.2.7172000.8-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.8/parcels/`

Table 7: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.8-1	Cloudera Runtime 7.1.7.2021 (7.1.7 SP2 CHF7)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.9-1 (CDS 3.2 CHF4 for 7.1.7)

Know more about CDS 3.2 CHF4 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on May 25, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.9-1-1.p0.41354351:

- CDPD-56187: Backport LIVY-783 to CDS 3.x
- CDPD-55116: Fix Spark and Livy vulnerability CVE-2023-22946

Table 8: CDS cumulative hotfix 3.2.7172000.9-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.9/parcels/`

Table 9: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.9-1	Cloudera Runtime 7.1.7.2021 (7.1.7 SP2 CHF7)	CDP Private Cloud Base with Cloudera Runtime 7.1.8

Cumulative hotfix CDS 3.2.7172000.10-1 (CDS 3.2 CHF5 for 7.1.7)

Know more about CDS 3.2 CHF5 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on August 30, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.10-1-1.p0.44657344:

- CDPD-59381: [CDS 3.x] The spark.kerberos.keytab (--keytab) parameter handling is missing from AtlasDelegationTokenProvider.scala
- CDPD-59379: [7.1.x, CDS 3.2] Spark - Upgrade kubernetes library to 5.7.4/5.8.1/5.10.2/5.11.2+ due to CVE-2021-4178
- CDPD-59293: [7.1.x, CDS 3.x] Spark Atlas Connector - Upgrade jettison to 1.5.4 due to CVE-2023-1436, CVE-2022-40149, CVE-2022-40150, CVE-2022-45685 and CVE-2022-45693
- CDPD-59247: [7.1.x, CDS 3.x] Spark - Upgrade snappy-java to 1.1.10.1 due to CVE-2023-34453, CVE-2023-34454 and CVE-2023-34455
- CDPD-58273: [7.1.x, CDS 3.x] Backport SPARK-43470
- CDPD-56193: [livy3] TypeError: required field "type_ignores" missing from Module
- CDPD-40910: Spark - Upgrade kubernetes library to 5.4.2 due to CVE-2021-4178, CVE-2021-20218

Table 10: CDS cumulative hotfix 3.2.7172000.10-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.10/parcels/`

Table 11: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.10-1	Cloudera Runtime 7.1.7.2032 (7.1.7 SP2 CHF13)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.12-1 (CDS 3.2 CHF6 for 7.1.7)

Know more about CDS 3.2 CHF6 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on September 27, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.12-1-1.p0.45461002:

- CDPD-61357: The Atlas lineage is missing in case of the following Spark CLI parameters: --principal test@ROOT.HWX.SITE --keytab test.keytab
- CDPD-61021: Spark3: add conf for disabling fallback to saving in Spark specific format when saving as Hive table results in error
- CDPD-60501: Spark3 - Update pre_commit_hook to use JDK8 version to u371
- CDPD-57831: Update the application tag generation logic in Livy

Table 12: CDS cumulative hotfix 3.2.7172000.12-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.12/parcels/`

Table 13: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.12-1	Cloudera Runtime 7.1.7.2035 (7.1.7 SP2 CHF14)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.13-4 (CDS 3.2 CHF7 for 7.1.7)

Know more about CDS 3.2 CHF7 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on December 8, 2023.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.13-4-1.p0.47975222:

- CDPD-64305: [Backport] Use centralized netty version for Livy-for-Spark3 in CDS 3.2
- CDPD-64135: [7.1.7 SP2, CDS 3.x] Backport HBASE-27624
- CDPD-63799: [7.1.9, CDS 3.x] Livy - Upgrade snakeyaml to 1.33 due to high CVEs
- CDPD-61904: [7.1.7 SP2, CDS 3.2] Spark - Upgrade commons-net to 3.9.0 due to CVE-2021-37533
- CDPD-61888: Livy3 - Kerberos ticket renewal fails with JDK-8186576
- CDPD-61742: Test failure: org.apache.spark.sql.hive.execution.HiveTableScanSuite.Spark-4077: timestamp query for null value
- CDPD-61564: Spark - Caused by: java.lang.NoClassDefFoundError: org/datanucleus/store/query/cache/QueryCompilationCache

Table 14: CDS cumulative hotfix 3.2.7172000.13-4 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.13/parcels/`

Table 15: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.13-4	Cloudera Runtime 7.1.7.2046 (7.1.7 SP2 CHF17)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.14-1 (CDS 3.2 CHF8 for 7.1.7)

Know more about CDS 3.2 CHF8 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on March 1, 2024.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.14-1-1.p0.50772531:

- CDPD-67041: [CDS 3.2] Backport SPARK-39144

- CDPD-66981: [CDS 3.x] Broken lineage on Atlas when calling .cache() or .persist() method on a Spark DataFrame
- CDPD-65588: [CDS 3.x] exclude log4j dependencies from spark-atlas-connector assembly
- CDPD-65584: [CDS 3.x] Spark - Upgrade Apache Derby to 10.17.1.0 due to CVE-2022-46337
- CDPD-60190: Backport SPARK-39441
- CDPD-55423: remove verbose output on Livy UI Error pages.
- CDPD-44220: Livy - Missing deploy mode param at Spark submit

Table 16: CDS cumulative hotfix 3.2.7172000.14-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.14/parcels/`

Table 17: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.14-1	Cloudera Runtime 7.1.7.2050 (7.1.7 SP2 CHF19)	CDP Private Cloud Base with Cloudera Runtime 7.1.7

Cumulative hotfix CDS 3.2.7172000.15-1 (CDS 3.2 CHF9 for 7.1.7)

Know more about CDS 3.2 CHF9 for Cloudera Runtime 7.1.7. This cumulative hotfix was released on March 22, 2024.

Important: CDS 3.2 for 7.1.7 Powered by Apache Spark is an add-on service for CDP Private Cloud Base, and is only supported with Cloudera Runtime 7.1.7. Spark 2 is included in CDP, and does not require a separate parcel.

Contact Cloudera Support for questions related to any specific hotfixes.

Following is the list of fixes that were shipped for CDS 3.2.3.3.2.7172000.15-1-1.p0.51304199:

- CDPD-67561: Backport [SPARK-47319][SQL] Improve missingInput calculation
- CDPD-66940: Timezone value not getting updated in Livy 3

Table 18: CDS cumulative hotfix 3.2.7172000.15-1 download URL

Parcel Repository Location
`https://[username]:[password]@archive.cloudera.com/p/spark3/3.2.7172000.15/parcels/`

Table 19: Supported Versions

CDS Powered by Apache Spark Version	Dependent Stack Version	Supported CDP Versions
3.2.3.3.2.7172000.15-1	7.1.7.2050 (7.1.7 SP2 CHF19)	CDP Private Cloud Base with Cloudera Runtime 7.1.7