Cloudera Data Engineering Overview

Date published:
Date modified:

https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Cloudera Data Engineering service

Cloudera Data Engineering is a serverless service for Cloudera that allows you to submit batch jobs to auto-scaling virtual clusters. Cloudera Data Engineering enables you to spend more time on your applications, and less time on infrastructure.

Note: Cloudera Data Engineering is currently available only on Amazon AWS and Microsoft Azure.

Cloudera Data Engineering allows you to create, manage, and schedule Apache Spark jobs without the overhead of creating and maintaining Spark clusters. With Cloudera Data Engineering, you define virtual clusters with a range of CPU and memory resources, and the cluster scales up and down as needed to run your Spark workloads, helping to control your cloud costs.

The Cloudera Data Engineering service involves several components:

Environment

A logical subset of your cloud provider account including a specific virtual network. For more information, see Environments.

Cloudera Data Engineering Service

The long-running Kubernetes cluster and services that manage the virtual clusters. The Cloudera Data Engineering service must be enabled on an environment before you can create any virtual clusters.

Virtual Cluster

An individual auto-scaling cluster with defined CPU and memory ranges. Virtual Clusters in Cloudera Data Engineering can be created and deleted on demand. Jobs are associated with clusters.

Jobs

Application code along with defined configurations and resources. Jobs can be run on demand or scheduled. An individual job execution is called a job run.

Resource

A defined collection of files such as a Python file or application JAR, dependencies, and any other reference files required for a job.

Job run

An individual job run.

The Cloudera Data Engineering service differs from a Cloudera Data Engineering Data Hub cluster in several ways, including the following:

Table 1: Cloudera Data Engineering Service vs. Cloudera Data Engineering Data Hub

Feature	Cloudera Data Engineering	Data Hub DE Template
Cloud providers	Amazon AWS, Microsoft Azure	Amazon AWS, Microsoft Azure
Compute engines	Apache Spark	Apache Spark, Apache Hive
Deployment	Kubernetes	Virtual machines (cloud provider)
Resource management	Yunikorn, Kubernetes	YARN
Troubleshooting	Cloudera Data Engineering deep analysis, Spark History Server	Spark History Server
Portability	on cloud/on premises	on cloud
Job submission	Managed API	Apache Livy

Browser Requirements

Supported browsers:

- Chrome
- Safari

Unsupported browsers:

- Firefox

Cloudera Data Engineering auto-scaling

Cloudera Data Engineering auto-scales at the job level as well as the service and virtual cluster level. Service and virtual cluster autoscaling uses Apache Yunikorn (Incubating) for resource scheduling to improve efficiency and utilization, reducing cloud provider costs. Job auto-scaling is managed by Apache Spark dynamic allocation. Cloudera Data Engineering scales resources up and down as needed for running jobs.

Service and virtual cluster auto-scaling

Service and virtual cluster autoscaling uses Apache Yunikorn (Incubating) for resource scheduling to improve efficiency and utilization

When you create a Cloudera Data Engineering service, you specify an instance type (size) and auto-scale range of instances. Virtual clusters associated with the Cloudera Data Engineering service use CPU and memory resources as needed to run jobs. When more resources are required, virtual machines of the specified instance type are started. When resources are no longer required, instances are terminated.

Important: Before configuring the minimum value for the autoscaling ranges at Administration Enable a Service Capacity & Costs Autoscaling Range on the Cloudera Data Engineering UI, note that you are charged at least for the minimum number of instances that you set, even if you do not use any instances. For example, if you set the minimum value to 1, even if you do not use the instance, you are charged for that 1 instance.

Virtual clusters also have auto-scaling controls, specified as maximum CPU cores and memory (in gigabytes).

Cloudera Data Engineering takes advantage of YuniKorn resource scheduling and sorting policies, such as gang scheduling and bin packing, to optimize resource utilization and improve cost efficiency.

By default, Cloudera Data Engineering uses the bin packing policy to allocate resources. It sorts the list of nodes ascending by the amount of available resources and the node with the lowest amount of available resources is selected. Sorting is solely based on an array of available memory in MBs and available CPUs in milli CPUs (mCPU), considering absolute numbers only. It disregards resource types and percentage of total resources available. For example, if you have three nodes with the following amounts of available resources:

Node	Available memory in MB	Available CPU in mCPU
nodeA	100,000	4,000
nodeB	4,000	10,000
nodeC	10,000	10,000

YuniKorn makes the following sorting:

nodeB	4,000	10,000
nodeA	4,000	100,000
nodeC	10,000	10,000

If bin packing does not provide optimal results for the applications you run, consider selecting one of the other available sorting policies.

For more information on gang scheduling, see the Cloudera blog post Spark on Kubernetes – Gang Scheduling with YuniKorn.

Job auto-scaling

Job auto-scaling is controlled by Apache Spark dynamic allocation.

Dynamic allocation scales job executors up and down as needed for running jobs. This can provide large performance benefits by allocating as many resources as needed by the running job, and by returning resources when they are not needed so that concurrent jobs can potentially run faster.

Resources are limited by the job configuration (executor range) as well as the virtual cluster auto-scaling parameters. By default, the executor range is set to match the range of CPU cores configured for the virtual cluster. This improves resource utilization and efficiency by allowing jobs to scale up to the maximum virtual cluster resources available, without manually tuning and optimizing the number of executors per job.

Cloudera Data Engineering resources

A *resource* in Cloudera Data Engineering is a named collection of files used by a job or a session. Resources can include application code, configuration files, custom Docker images, and Python virtual environment specifications (requirements.txt).

Note: Custom Docker container images is a *Technical Preview* feature, requiring entitlement. Contact your Cloudera account representative to enable access to this feature. See Cloudera on cloud Preview Features below.

Resources are associated with virtual clusters. A resource can be used by multiple jobs, and jobs can use multiple resources.

Note: You can reference the same resource file at the maximum 177 times in job runs. Therefore, you can only run 177 jobs simultaneously that reference the same resource file.

The resource types supported by Cloudera Data Engineering are files, python-env, and custom-runtime-image.

files

An arbitrary collection of files that a job can reference. The application code for the job, including any necessary configuration files or supporting libraries, can be stored in a files resource. Files can be uploaded to and removed from a resource as needed.

python-env

A defined virtual Python environment that a job runs in. The only file that can be uploaded to a pyth on-env resource is a requirements.txt file. When you associate a python-env resource with a job, the job runs within a Python virtual environment built according to the requirements.txt specification.

custom-runtime-image

A Docker container image. When you run a job using a custom-runtime-image resource, the executors that are launched use your custom image.

Related Information

Cloudera on cloud Preview Features