

## Cloudera Data Engineering Top Tasks

Date published: 2020-07-30

Date modified: 2025-11-08

The Cloudera logo is displayed in a bold, orange, sans-serif font. The word "CLOUDERA" is written in all caps, with a stylized 'E' that has three horizontal bars.

# 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

<b>Creating sessions in Cloudera Data Engineering.....</b>	<b>4</b>
<b>Creating jobs in Cloudera Data Engineering.....</b>	<b>7</b>
<b>Automating data pipelines using Apache Airflow in Cloudera Data Engineering.....</b>	<b>10</b>
<b>Monitoring Cloudera Data Engineering service resources with Grafana dashboards.....</b>	<b>13</b>
Connecting to Grafana dashboards in Cloudera Data Engineering on premises.....	14
Accessing Grafana dashboards.....	18

# Creating sessions in Cloudera Data Engineering

A Cloudera Data Engineering session is an interactive short-lived development environment for running Spark commands to help you iterate upon and build your Spark workloads.

## About this task

The commands that are run in a Cloudera Data Engineering session are called Statements. You can submit the Statements through the connect CLI command or the Interact tab in the Cloudera Data Engineering UI for a session. Python and Scala are the supported session types. Learn how to use Cloudera Data Engineering sessions using the user interface and CLI.

## Before you begin



**Important:** You must create the All-Purpose cluster, initialize each cluster, and configure each user who need to run sessions before creating sessions.

In Cloudera Data Engineering, sessions are associated with virtual clusters. Before you can create a session, you must create a virtual cluster that can run it. For more information, see [Creating virtual clusters](#).

## Procedure

1. In the Cloudera console, click the Data Engineering tile. The Home page displays.
2. Click Sessions in the left navigation menu and then click Create Session.

Status	Name	Description	Type	Timeout	Created On	Created By	Actions
Running	session-demo		PySpark	8 hour(s)	Dec 10, 2024, 1:23:26 PM		
Running	interactive-session		PySpark	8 hour(s)	Dec 10, 2024, 1:22:54 PM		
Running	interactive-session-demo		PySpark	8 hour(s)	Dec 10, 2024, 1:22:31 PM		
Running	pyspark-session-demo		PySpark	8 hour(s)	Dec 10, 2024, 1:21:45 PM		
Running	test-session		PySpark	8 hour(s)	Dec 10, 2024, 1:19:20 PM		

3. Enter a Name for the session.
4. Select a Type, for example, PySpark, Scala, or Spark Connect.
5. Select a Timeout value.

The session will stop after the indicated time has passed.

6. Enter a Description for the session.
7. Enter the Configurations.



**Note:** The Spark session is created during the job run or session creation. Most Spark configurations are not modifiable during runtime and has to be specified during job run or session creation. You can check if a configuration can be modified by using `spark.conf.isModifiable`. For example,

```
spark.conf.isModifiable("spark.executor.memory")
False
```

8. Click Data Connector drop-down list and select the name of the data connector from the list. The UI displays the storage information that is internally overwritten. For more information about how to add an Ozone data connector, see [Adding Ozone data connector for Cloudera Data Engineering service](#).

9. Set the Compute options.

- Optional: GPU Acceleration (Technical Preview): You can accelerate your session using GPUs. Click Enable GPU Accelerations checkbox to enable the GPU acceleration and configure selectors and tolerations if you want to run the job on specific GPU nodes. When you run this session, this particular session will request GPU resources.

10. In the **Files and Resources** section, you can upload Jar, Python, Egg, Zip, and other files. You can also add a resource, repositories, or a Python environment to be used in this session.

Files that are uploaded to a session are stored in the app/mount directory.

The screenshot shows the 'Files and Resources' configuration panel. At the top, it says 'Files and Resources' and 'Add files and mount resources to be used in this session'. Below this, there are several sections:

- Files & Resources**: A section header.
- Jar Files**: A dashed box containing the text 'Upload or Select from Resource'.
- Python, Egg, Zip files**: A dashed box containing the text 'Upload or Select from Resource'.
- Other Files**: A dashed box containing the text 'Upload or Select from Resource'.
- Resources**: A section header with a help icon. Below it is a button labeled 'Add Resource'.
- Repositories**: A section header with a help icon. Below it is a button labeled 'Add Repository'.
- Python Environment**: A section header with a help icon. Below it is a dashed box containing the text 'Select Python Environment'.

At the bottom of the panel, there are two buttons: 'Cancel' and 'Create'.

**11. Share the session with a user or group.**

- a) In the Sharing Settings section, click Add User or Group. The Add User or Group pop-up appears.

Sharing settings  
Add users or groups who can access this artifact.

[Add User or Group](#)

ⓘ Make sure the attached Resources or Repositories are shared with the user or group. If you do not have access to share them, contact their relevant owner.

Name	Id	Type	Access Level	Actions
All Users	-	User	Full	

- b) In the Search for a User or a Group field, type the user or group name and select the required user or group from the list.

## Add User or Group

Search for a user or group and select an Access level:-

**Full:** user will be able to view, update, delete and run the job and job runs.  
**Read only:** user will only be able to view the job and job runs.

ⓘ Please make sure attached resources and repositories are also shared.  
[Learn more](#)

Search for a User or a Group

Access Level

Full

Add Cancel

- c) Select Full or Read Only depending on the access you want to provide from the Access Level drop-down list.  
 d) Click Add.

**12. Click Create.**

The Connect tab displays a list of connectivity options available to interact with the session. The Interact tab allows you to interact with the session, and becomes available once the session is running.


**13. To delete a session, open the session and click Delete.**

**Note:** Deleting a session results in the termination of an active session and the loss of any attached logs and details.

# Creating jobs in Cloudera Data Engineering

A job in Cloudera Data Engineering consists of defined configurations and resources, including application code. Jobs can be run on demand or scheduled.

## Before you begin

 **Important:** You must create the cluster, initialize each cluster, and configure each user who need to submit jobs before creating jobs.

In Cloudera Data Engineering, jobs are associated with virtual clusters. Before you can create a job, you must create a virtual cluster that can run it. For more information, see [Creating virtual clusters](#).

## Procedure

- 1. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
- 2. In the left navigation menu click Jobs. The Jobs page is displayed.
- 3. Click Create Job. The Job Details page is displayed.

Job Details

\* Job Type

☒ Spark 3.5.4

☐ Airflow

Runtime Versions

Python 3.11, Iceberg 1.5.2, Java 17, Scala 2.12.18

\* Name

Job Name

Select Application Files

Resource

Upload or Select from Resource

Main Class

com.package.MainClass

Arguments (Optional)

Argument

Enter configurations as key value pairs, for example:-

spark.executor.instances=2

spark.speculation=true

Data Connector (Optional)

Data Connector

Advanced Options

Upload additional files, customize no. of executors, driver and executor cores and memory

Schedule

Turn on to schedule Job, enable catchup and jobs dependants

Sharing settings

Add users or groups who can access this artifact.

Add User or Group

Make sure the attached Resources or Repositories are shared with the user or group. If you do not have access to share them, contact their relevant owner.

Name	Id	Type	Access Level	Actions
All Users	-	User	Full	

Cancel

Create and Run

#### 4. Provide the Job Details.

- a) Select Spark for the job type. If you are creating the job from the Home page, select the virtual cluster where you want to create the job.
- b) Specify the Name.
- c) Select File or URL for your application file, and provide or specify the file. You can upload a new file or select a file from an existing resource.

If you select the URL option and specify an Amazon AWS S3 URL, add the following configuration to the job:

config\_key: spark.hadoop.fs.s3a.delegation.token.binding

config\_value: org.apache.knox.gateway.cloud.idbroker.s3a.IDBDelegationTokenBinding

- d) If your application code is a JAR file, specify the Main Class.
- e) Specify arguments if required. You can click the Add Argument button to add multiple command arguments as necessary.
- f) Enter configurations if needed. You can add multiple configuration parameters as key-value pairs in the text box as necessary.



#### Important:

- For Spark jobs, setting the `spark.app.id` property at the Spark job level configuration or within the Spark application code is not supported in Cloudera Data Engineering.
- The Spark session is created during the job run or session creation. Most Spark configurations are not modifiable during runtime, you must specify them during job run or session creation. To check if you can modify a configuration, use `spark.conf.isModifiable`. For example,

```
spark.conf.isModifiable("spark.executor.memory")
False
```

- g) Optional: Select the name of the data connector from the Data Connector drop-down list. The UI displays the storage information that is internally overwritten.



**Note:** Data connector option is only available in the Cloudera Control Plane UI and the job run which uses the data connector must be only created using the Cloudera Control Plane UI. Triggering the jobs with the data connector using the Cloudera Runtime UI is not supported and the job will fail.

- h) If your application code is a Python file, select the Python Version, and optionally select a Python Environment.
- #### 5. Click Advanced Configurations to display more customizations, such as additional files, initial executors, executor range, driver and executor cores, and memory.

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.

**GPU Acceleration (Technical Preview):** You can accelerate your Spark jobs using GPUs. Click Enable GPU Accelerations checkbox to enable the GPU acceleration and configure selectors and tolerations if you want to run the job on specific GPU nodes. When this job is created and run, this particular job will request GPU resources.



**Warning:** You must ensure this virtual cluster has been configured with GPU resource quota. Otherwise, the jobs will be in the Pending state as no GPU resource can be allocated to the pod.

#### 6. Click Schedule to display scheduling options.

You can schedule the application to run periodically using the Basic controls or by specifying a Cron Expression.

#### 7. Click Alerts and provide the email ID to receive alerts. Click + to add more email IDs. Optionally, you can select when you want email alerts for job failures, missed job service-level agreements, or both.



**Note:** You must configure the Configure Email Alerting option while creating a virtual cluster to send your email alerts. For more information about configuring email alerts, see [Creating virtual clusters](#).



8. Share the job with a user or group.

- a) In the Sharing Settings section, click Add User or Group. The Add User or Group pop-up appears.

Sharing settings  
Add users or groups who can access this artifact.

[Add User or Group](#)

ⓘ Make sure the attached Resources or Repositories are shared with the user or group. If you do not have access to share them, contact their relevant owner.

Name	Id	Type	Access Level	Actions
All Users	-	User	Full	

- b) In the Search for a User or a Group field, type the user or group name and select the required user or group from the list.

## Add User or Group ×

Search for a user or group and select an Access level:-

**Full:** user will be able to view, update, delete and run the job and job runs.  
**Read only:** user will only be able to view the job and job runs.

ⓘ Please make sure attached resources and repositories are also shared.  
[Learn more](#)

Search for a User or a Group

Access Level

Full ▼

Add Cancel

- c) Select Full or Read Only depending on the access you want to provide from the Access Level drop-down list.
- d) Click Add.
9. If you provided a schedule, click Schedule to create the job. If you did not specify a schedule, and you do not want the job to run immediately, click the drop-down arrow on Create and Run and select Create. Otherwise, click Create and Run to run the job immediately.

# Automating data pipelines using Apache Airflow in Cloudera Data Engineering

Cloudera Data Engineering enables you to automate a workflow or data pipeline using Apache Airflow Python DAG files. Each Cloudera Data Engineering virtual cluster includes an embedded instance of Apache Airflow. You can also use Cloudera Data Engineering with your own Airflow deployment. Cloudera Data Engineering on Cloudera on premises currently supports only the Cloudera Data Engineering job run operator.

## Before you begin



**Important:** Cloudera provides support for Airflow [core operators and hooks](#), but does not provide support for Airflow provider packages. Cloudera Support might require you to remove any installed provider packages during troubleshooting.

**About this task****Important:**

- The Cloudera Data Engineering API and the Airflow UI do not support clearing an Airflow task or a DAG run. You can perform this action only by using the Airflow admin command (for example, from the airflow-scheduler container). However, clearing an Airflow task or a DAG run causes a re-run of that task or DAG run and creates a new Cloudera Data Engineering Airflow job run for the re-run within Cloudera Data Engineering. Because this re-run occurs, the Airflow UI only updates the original job run's details in Airflow. Hence, there can be multiple Cloudera Data Engineering job runs pointing to the same Airflow job run.
- The default Airflow worker pod resource requests limits are as follows:
  - CPU requests = 1
  - CPU limits = No limit
  - Memory requests = 2 Gi
  - Memory limits = 2 Gi

Also, you can set the custom resource requests for an Airflow task through `executor_config`.

For example, sample DAG file is as follows:

```
from airflow import DAG
from airflow.operators.python import PythonOperator
from pendulum import datetime
from kubernetes.client import models as k8s

config_resource_requirements = {
    "pod_override": k8s.V1Pod(
        spec=k8s.V1PodSpec(
            containers=[
                k8s.V1Container(
                    name="base",
                    resources=k8s.V1ResourceRequirements(
                        requests={"cpu": 0.5, "memory": "1024Mi"},
                        limits={"cpu": 0.5, "memory": "1024Mi"}
                    )
                )
            ]
        )
    )
}

with DAG(
    dag_id="python_operator_custom_resources",
    start_date=datetime(2024, 1, 1),
    schedule=None,
    catchup=False,
):
    PythonOperator(
        task_id="hello_task",
        python_callable=lambda: print("Hello!"),
        executor_config=config_resource_requirements
    )
```

The following instructions are for using the Airflow service provided with each Cloudera Data Engineering virtual cluster. For instructions on using your own Airflow deployment, see [Using the Cloudera provider for Apache Airflow](#).

**Procedure**

1. Create an Airflow DAG file in Python. Import the Cloudera Data Engineering operator and define the tasks and dependencies.

Here is a complete DAG file:

```
from dateutil import parser
from datetime import datetime, timedelta
from datetime import timezone
from airflow import DAG
from cloudera.airflow.providers.operators.cde import CdeRunJobOperator

default_args = {
    'owner': 'psherman',
    'retry_delay': timedelta(seconds=5),
    'depends_on_past': False,
    'start_date': datetime(2024, 2, 10, tz="UTC"),
}

example_dag = DAG(
    'airflow-pipeline-demo',
    default_args=default_args,
    schedule_interval='@daily',
    catchup=False,
    is_paused_upon_creation=False
)

ingest_step1 = CdeRunJobOperator(
    connection_id='cde-vc01-dev',
    task_id='ingest',
    retries=3,
    dag=example_dag,
    job_name='etl-ingest-job'
)

prep_step2 = CdeRunJobOperator(
    task_id='data_prep',
    job_name='insurance-claims-job'
)

ingest_step1 >> prep_step2
```

Here are some examples of things you can define in the DAG file:

**CDE job run operator**

Use `CdeRunJobOperator` to specify a Cloudera Data Engineering job to run. This job must already exist in the virtual cluster specified by the `connection_id`. If no `connection_id` is specified, Cloudera Data Engineering looks for the job in the virtual cluster where the Airflow job runs.

```
from cloudera.cdp.airflow.operators.cde_operator import CdeRunJobOperator
...
ingest_step1 = CdeRunJobOperator(
    connection_id='cde-vc01-dev',
    task_id='ingest',
    retries=3,
    dag=example_dag,
    job_name='etl-ingest-job'
```

)

**Email Alerts**

Add the following parameters to the DAG `default_args` to send email alerts for job failures or missed service-level agreements or both.

```
'email_on_failure': True,
'email': 'abc@example.com',
'email_on_retry': True,
'sla': timedelta(seconds=30)
```

**Task dependencies**

After you have defined the tasks, specify the dependencies as follows:

```
ingest_step1 >> prep_step2
```

For more information on task dependencies, see [Task Dependencies](#) in the Apache Airflow documentation.

For a tutorial on creating Apache Airflow DAG files, see the [Apache Airflow documentation](#).

**2. Create a Cloudera Data Engineering job.**

- a) In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
- b) In the left navigation menu click Jobs. The Jobs page is displayed.
- c) Click Create Job. The Job Details page is displayed.
- d) Select the Airflow job type.
- e) Name: Provide a name for the job.
- f) DAG File: Use an existing file or add a DAG file to an existing resource or create a resource and upload it.

1. Select from Resource: Click Select from Resource to select a DAG file from an existing resource.
2. Upload: Click Upload to upload a DAG file to an existing resource or to a new resource that you can create by selecting Create a resource from the Select a Resource dropdown list. Specify the resource name and upload the DAG file to it.



**Note:** You must configure the Configure Email Alerting option while creating a virtual cluster to send your email alerts. For more information about configuring email alerts, see [Creating virtual clusters](#).

You can add the email alert parameters to the DAG `default_args` to get email alerts for job failures and missed service-level agreements. An example of email alert configurations is listed in *Step 1*.

3. Click Create and Run to create the job and run it immediately, or click the dropdown button and select Create to create the job.

## Monitoring Cloudera Data Engineering service resources with Grafana dashboards

Grafana is a visualisation and analytics software that enables the development of dashboards to monitor metrics data. You can access pre-built Grafana dashboards to monitor your jobs and virtual clusters in Cloudera Data Engineering.

The Cloudera metrics are stored centrally in the Prometheus database and monitored by Prometheus. Grafana uses these metrics for data visualization. Your workload databases are not involved in any way.

You can immediately view the following pre-built dashboards for viewing runtime metrics in Cloudera Data Engineering:

### Kubernetes Dashboard

This dashboard includes generalized visualizations of Cloudera Data Engineering job run statuses. It displays the following information:

- Number of succeeded, failed, and killed jobs for the given period
- Total number of jobs in the Starting phase
- Total number of jobs in the Running phase

### Virtual Cluster Metrics Dashboard

This dashboard includes visualizations of service requests, pod counts and job statuses for the selected Virtual Cluster. The available metrics are:

- Time series of CPU requests of running pods (includes virtual cluster service overhead)
- Time series of memory requests of running pods (includes virtual cluster service overhead)
- The response time of Livy's requests
- Time series for the number of pods in running and pending states
- Total number of running pods and pending pods
- Time series of starting and running jobs, and the total number of successful jobs

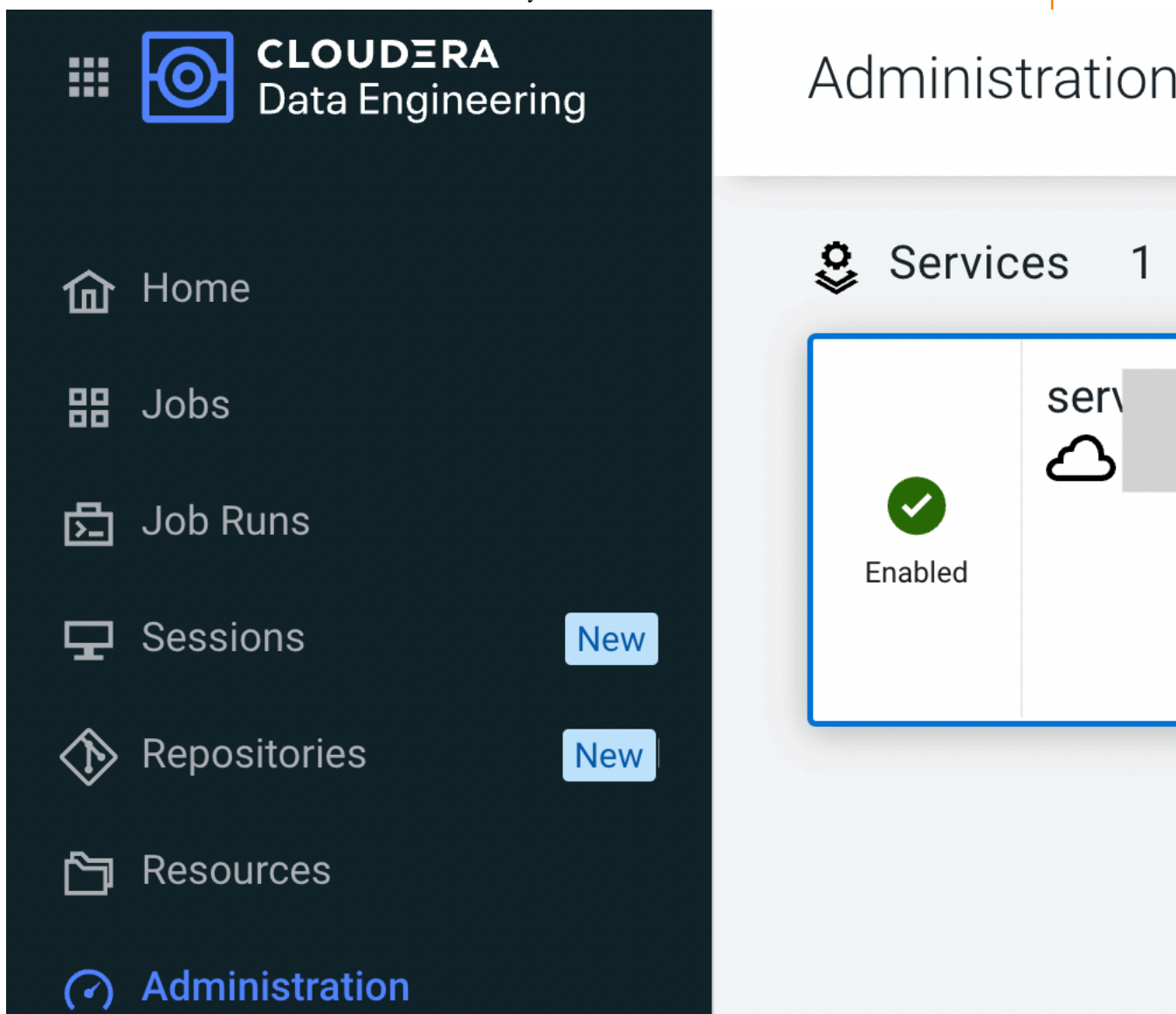
## Connecting to Grafana dashboards in Cloudera Data Engineering on premises


This topic describes how to access Grafana dashboards for advanced visualization of Virtual Cluster's metrics such as memory and CPU usage in Cloudera Data Engineering on premises.

#### For CDE Service

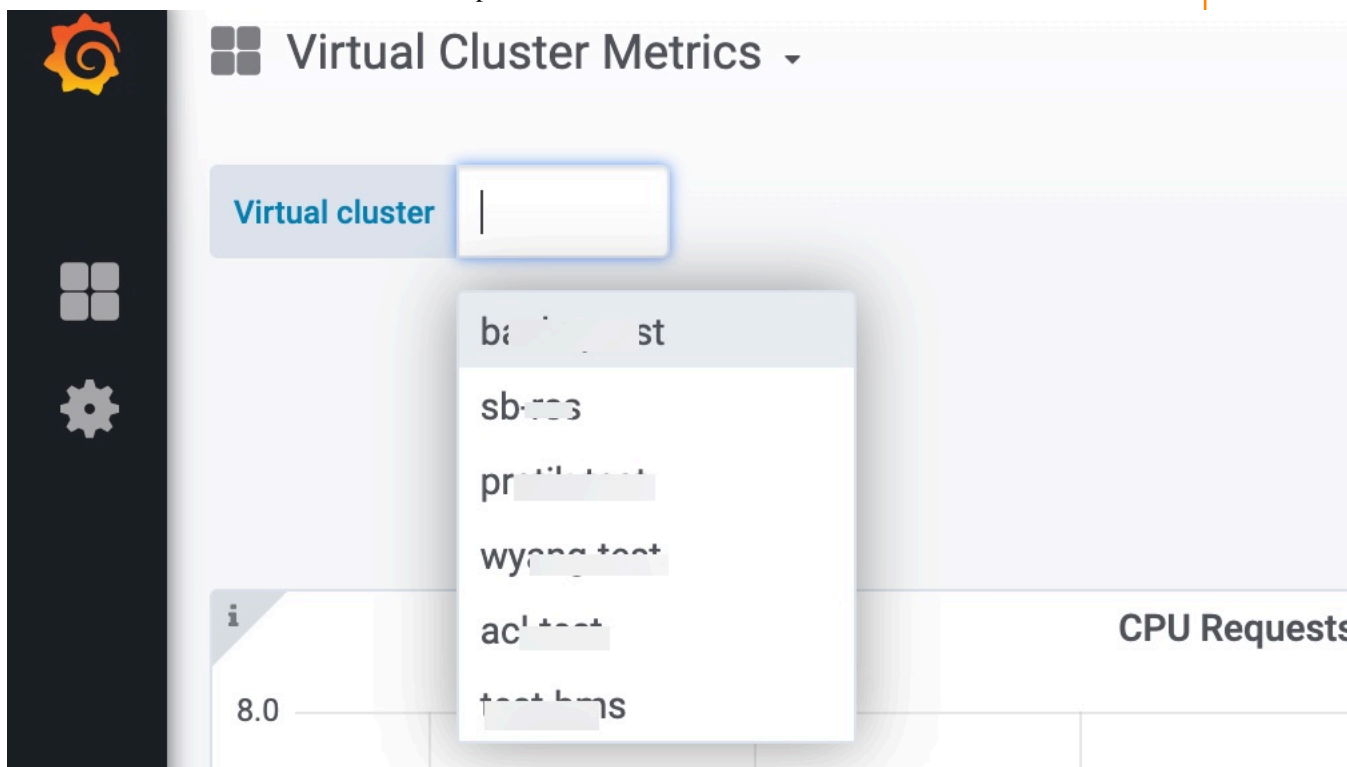
1. In the Cloudera console, click the Data Engineering tile. The Home page displays.

2. Click Administration in the left navigation menu and locate a Service in the Services column and click Service Details on the environment for which you want to see the Grafana dashboard.



3. In the Administration/Service page, click Grafana Charts. A read-only version of the Grafana interface opens in a new tab in your browser.
4.  In the Grafana dashboard, click the grid icon in the left navigation menu.
5. Select Virtual Cluster Metrics under the Dashboards pane.

6. Click on a virtual cluster name from the dropdown list to view the Grafana charts.



about CPU requests, memory requests, jobs, and other information related to the virtual cluster is displayed.

#### For Virtual Cluster

1. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
2. Click Administration in the left navigation menu and locate a Service in the Services column.
3. Navigate to the virtual cluster for which you want to see the Grafana dashboard.
4. Click Cluster Details for the virtual cluster.

The Administration/Virtual Cluster page is displayed.



**5.** Click Grafana Charts.

A read-only version of the Grafana interface opens in a new tab in your browser.

Overview / [Virtual Cluster](#)

**Running**

[Virtual Cluster](#) **23Mar**

VERSION	VC ID	CREATED BY	CPU	MEMORY	JOBS
1.1.0-b26	dev-ops-600+77va	<a href="#">S. [redacted]</a>	0	0 B	0 <a href="#">[external link]</a>

[CLI TOOL](#) : [API DOC](#) [JOBS API URL](#) **[GRAFANA CHARTS](#)**

[Configuration](#) [Charts](#) [Logs](#)

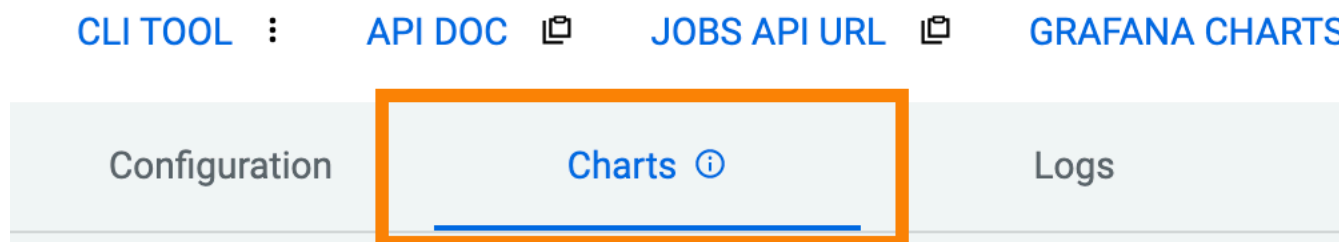
**CDE Service**

Information about CPU requests, memory requests, jobs, and other information related to the virtual cluster is displayed.

**6.** In the Virtual Cluster Metrics page, click on a virtual cluster name from the Virtual Cluster dropdown list to view the Grafana charts of that virtual cluster.



**Note:** You must first view the charts using the GRAFANA CHARTS link. Only then the charts in the Charts tab get loaded. Otherwise, it displays the The web page at <https://SERVICE.CDE-NRJCROWG7.APPS.APPS.SHARED-01.KCLOUD.CLDR.COM/GRAFANA/D/USZZ/KUBERNETES?KIOSK> might be temporarily down or it may have moved permanently to a new web address. error.




## Accessing Grafana dashboards

Cloudera provides pre-built Grafana dashboards comprising metrics data, charts, and other visuals. You can access pre-built Grafana dashboards to monitor your jobs and virtual clusters in Cloudera Data Engineering. You can immediately view the Kubernetes and Virtual Cluster Metrics pre-built dashboards in Cloudera Data Engineering.

### Before you begin

You must first connect to the Grafana dashboards in Cloudera Data Engineering on premises to view the Kubernetes and Virtual Cluster Metrics dashboards.

### Procedure

1. After you connect to the Grafana Dashboards from the Cloudera Data Engineering UI, click the  icon to view the left navigation pane.
2. Click Dashboards > Browse. The Dashboards screen is displayed.

3. In the Browse tab of Dashboards, click Kubernetes or Virtual Cluster Metrics to view the respective dashboard.

