

Backing up and restoring Cloudera Data Engineering jobs

Date published: 2020-07-30

Date modified: 2025-07-30



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

Backing up Cloudera Data Engineering jobs on remote storage.....	4
Backing up Cloudera Data Engineering jobs on local storage.....	6
Restoring Cloudera Data Engineering jobs from backup.....	10
Restoring jobs from a 1.20.3 VC backup.....	13

Backing up Cloudera Data Engineering jobs on remote storage

You can back up Cloudera Data Engineering jobs and associated resources. Backups are saved as ZIP files that can be used to restore jobs and their associated resources. Backups and restores are also supported for remote storage (also known as object store).



Note: Currently, the user who runs the cde backup restore command has permissions, by default, to run the jobs. This may cause Cloudera Data Engineering jobs to fail if the workload user differs from the user who runs the jobs on Source Cloudera Data Engineering Service where the backup was performed. This failure is due to the Workload User having different privileges as the user who is expected to run the job. Additionally, backup and restore functionality does not include Airflow metadata related to connections and variable definitions. Those must be manually recreated in the new service.

Table 1: Artifacts included in the backup, comparison of CLI and UI methods

Artifacts	Using CDE CLI	Using Backup Jobs link on
Job: Spark	Y	Y
Job: Airflow	Y	Y
Resource: files	Y	Y
Resource: docker runtimes	Y	Y
Resource: python-venv (for Spark)	Y	Y
Resource: python-venv (for Airflow)	Y (Since 1.21.0)	Y (Since 1.21.0)
Git Repository	Y	N
Credentials	Y	N
Spark Session	N	N
Spark Session logs (Including statement history)	N	N
Job Runs	Y (Since 1.22.0)	N
Job Run logs (Driver, Executor, API)	N	N
Airflow DAG logs	N	N
Airflow connections	N	N
Airflow variables	N	N



Note: By default, backups created using the CDE CLI include only jobs and resources. To include additional artifacts in the backup, use the appropriate --include-xxx option in the CLI command.

For CDE CLI

Before you begin

- Download and configure the CDE CLI.

Steps for backing up on remote storage

1. Run the `cde backup create` command to create a backup of the jobs in the virtual cluster your CDE CLI is configured to interact with. By default, all job configurations in the cluster are backed up, but the resources are not. You can use command flags to change this behavior as follows:

--continue-on-error

If set to true, the archive generation continues even if there are errors for some items. It works only for remote storage.

--credential-filter

Selects credentials to be backed up using the `fieldname[operator]` argument syntax. This command flag can be repeated. The name of the job and resource API field is 'fieldname' and the 'operator' is one of the following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using `and`. For example:

```
cde backup create --credential-filter "name[noteq]test" --credential-filter "created[gte]2020-01-01"
```

--include-active-airflow-pyenv

If set to true, it backs up the active Airflow Libraries and Operators with the associated credentials and secret. The default value is false.

--include-credential-secrets

Backs up credential secrets.



Warning: In the archive, the backed up secrets are not encrypted.

--include-credentials

Backs up all virtual cluster credentials. By default, the credential secrets are not included in the backup.

--include-job-resources

Backs up all resources associated with the selected jobs. These resources cannot be filtered out by the `--resource-filter` parameter.

--include-jobs

Backs up all jobs in the virtual cluster. This is the default behavior.

--include-resource-credentials

Backs up credentials for the resources. You cannot filter out selected credentials using the `credentialfilter` parameter. This is the default behavior.

--include-resources

Backs up all resources in the virtual cluster, including those not associated with jobs.

--job-filter <FILTER_STRING>

Selects jobs to back up using the `fieldname[operator]` argument syntax. This command flag can be repeated. The name of the job and resource API field is 'fieldname' and the 'operator' is one of the following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using `and`. For example:

```
cde backup create --job-filter "name[noteq]test" --job-filter "created[gte]2020-01-01"
```

--resource-filter <FILTER_STRING>

Selects resources to back up using the `fieldname[operator]` argument syntax. This command flag can be repeated. Filter by adding detail to the filter syntax, for example, filter syntax 'fieldname[operator]argument'. The name of the job and resource API field is 'fieldname' and the

'operator is one of the following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using and. For example:

```
'name[noteq]my-resource', 'created[gte]2020-01-01'
```

--output

The output in text or JSON format. The default is text format.

--remote-storage

Backs up to the remote storage. The default value is false.

--remote-path

Use the remote backup file relative path together with the remote-storage parameter. Do not include the filename in the path. The path is relative to /dex/backup/.

--remote-name

Use the remote backup file name together with the remote-storage parameter. If the file name is not specified, then a default generated value is used.

--validate

After a remote backup archive is created, it validates the archive.

Example for backing up to remote storage all jobs and its related resources, plus all resources whose name contains "data".

```
cde backup create --remote-storage --include-resources --resource-filter
"name[like]%data%"
```

Example for creating a backup on remote storage:

```
./cde backup create --remote-storage --remote-path test --remote-name ar
chive.zip

# output
{"archiveRelativePath":"test/archive.zip", "archiveURL":"s3a://dex-dev-d
efault-aws-storage/datalake/logs/dex/backup/test/archive.zip", "code":201}
```

For Web UI

Result

Depending on your browser settings, you are either prompted for a location to save the file, or the file is downloaded to your default download location. The file is a ZIP file named archive-*<TIMESTAMP>*.zip.

To restore a backup file, see [Restoring Cloudera Data Engineering jobs from backup](#).

Backing up Cloudera Data Engineering jobs on local storage

You can back up Cloudera Data Engineering jobs and their associated resources and/or repositories. Backups are saved as ZIP files that can be used to restore jobs and their associated resources and/or repositories. Backups and restores is also supported for remote storage (also known as object store).



Note: Currently, the user who runs the cde backup restore command has permissions, by default, to run the Jobs. This may cause Cloudera Data Engineering jobs to fail if the workload user differs from the user who runs the jobs on Source Cloudera Data Engineering Service where the backup was performed. This failure is due to the Workload User having different privileges as the user who is expected to run the job. Additionally, backup and restore functionality does not include Airflow metadata related to connections and variable definitions. Those must be manually recreated in the new service.

For CDE CLI

Before you begin

- Download and configure the CDE CLI.

Steps for backing up on local storage

1. Run the cde backup create command to create a backup of the jobs in the virtual cluster your CDE CLI is configured to interact with. By default, all job configurations in the cluster are backed up, but the resources are not. You can use command flags to change this behavior as follows:

--credential-filter

Selects credentials to be backed up using the fieldname[operator]argument syntax. This command flag can be repeated. The name of the job and resource API field is 'fieldname' and the 'operator' is one of the following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using and. For example:

```
cde backup create --credential-filter "name[noteq]test" --credential-filter "created[gte]2020-01-01"
```

--include-active-airflow-pyenv

If set to true, it backs up the active Airflow Libraries and Operators with the associated credentials and secret. The default value is false.

--include-credential-secrets

Backs up credential secrets.



Warning: In the archive, the backed up secrets are not encrypted.

--include-credentials

Backs up all virtual cluster credentials. By default, the credential secrets are not included in the backup.

--include-job-resources

Backs up all resources associated with the selected jobs. These resources cannot be filtered out by the --resource-filter parameter.

--include-jobs

Backs up all jobs in the virtual cluster. This is the default behavior.

--include-resource-credentials

Backs up credentials for the resources. You cannot filter out selected credentials using the credentialfilter parameter. This is the default behavior.

--include-resources

Backs up all resources and repositories (internally these are also resources only) in the virtual cluster, including those not associated with jobs.

--job-filter <FILTER_STRING>

Selects jobs to back up using the fieldname[operator]argument syntax. This command flag can be repeated. The name of the job and resource API field is 'fieldname' and the 'operator' is one of the

following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using and.

For example: The following command backs up locally all jobs whose name is not test, and which is created later than or equal to 2020-01-01, and all their related resources:

```
cde backup create --job-filter "name[noteq]test" --job-filter "created[gte]2020-01-01"
```

--local-path <FILENAME>

Specifies the local file path and name to store the backup. By default, the backup filename is archive-**<timestamp>**.zip.

--output

The output in text or JSON format. The default is text format.

--resource-filter <FILTER_STRING>

Selects resources to back up using the fieldname[operator]argument syntax. This command flag can be repeated. The name of the job and resource API field is 'fieldname' and the 'operator' is one of the following: 'eq', 'noteq', 'lte', 'lt', 'gte', 'gt', 'in', 'notin', 'like', 'rlike'. You can add multiple filters using and. For example:

```
cde backup create --resource-filter "name[eq]test" --resource-filter "created[gte]2020-01-01"
```

For example, to backup all jobs containing the string etl and include all resources associated with those jobs:

```
cde backup create --job-filter "name[like]%etl%" --include-job-resources
```

2. Validate the backup using the cde backup list-archive command. For example:

```
cde backup list-archive --local-path archive-2021-11-10T01:24:06.zip
```

Confirm that all jobs and resources that you expected to be backed up are included.

Result

The output of cde backup list-archive is similar to the following:

```
{
  "backup_set": [
    {
      "id": "v1/backupset/v1/f332bb06-7612-4345-8f3b-da4f27f315b3/",
      "cdeVersion": "1.18.0-b59",
      "clusterID": "cluster-2bqdpfrs",
      "appID": "dex-app-82wlp6d",
      "app_name": "yjtest",
      "user": "csso_yjlu",
      "backupStarted": "2022-10-07T06:39:46.082837691Z"
    }
  ],
  "entries": [
    {
      "backup_set": "v1/backupset/v1/f332bb06-7612-4345-8f3b-da4f27f315b3/",
      "entityType": "Job",
      "name": "example-a",
      "adjustedName": "example-a",
      "archiveDirectoryPath": "v1/jobs/v1/d7826797-4985-455f-a9c8-2ab1cc624d9b/",
      "user": "csso_yjlu"
    }
  ],
}
```



```
{
  "backup_set": "v1/backupset/v1/f332bb06-7612-4345-8f3b-da4f27f315
b3/",
  "entityType": "Resource",
  "name": "example-data",
  "adjustedName": "example-data",
  "archiveDirectoryPath": "v1/resources/v1/41979747-5ad1-40c3-b301-cd5
7111411f9/",
  "user": "csso_yjlu"
}
]
```

For Cloudera Data Engineering API



Note: Some of the command examples provided use [jq](#) to parse the JSON output and make it more readable.

Before you begin

Request an access token and save it as an environment variable to use in API calls. For instructions, see [Getting a Cloudera Data Engineering API access token](#).

Steps

1. Determine the API URL for the virtual cluster containing the job you want to back up:
 - a. Navigate to the Cloudera Data Engineering Overview page.
 - b. In the Cloudera Data Engineering Services column, select the service containing the virtual cluster with the jobs you want to back up.
 - c. In the Virtual Clusters column on the right, click the Cluster Details icon for the virtual cluster containing the jobs you want to back up.
 - d. Copy the URL under JOBS API URL, and set it as an environment variable. For example:

```
export CDE_JOBS_API="https://pmjkrn5.cde-czlmkz4y.na-01.xvp2-7p8o.c
loudera.site/dex/api/v1"
```

2. Back up jobs using a URL-encoded filter with the syntax `name[like]<QUERY>`, modeled after the SQL like operator. For example, to back up jobs containing the string `etl`, set `jobfilter` to `name[like]%etl%` (URL-encoded as `name%5Blike%5D%25etl%25`):

```
curl -k \
-H "Authorization: Bearer ${CDE_TOKEN}" \
-X GET "${CDE_JOBS_API}/admin/export?exportjobs=true&jobfilter=name%5Bli
ke%5D%25etl%25&exportjobresources=true&exportresources=false" \
-H "accept: application/zip" \
--output cde-etl-jobs-backup.zip
```

To back up all jobs and associated resources, omit the `jobfilter` parameter:

```
curl -k \
-H "Authorization: Bearer ${CDE_TOKEN}" \
-X GET "${CDE_JOBS_API}/admin/export?exportjobs=true&exportjobresources=
true&exportresources=false" \
-H "accept: application/zip" \
--output cde-all-jobs-backup.zip
```

To back up all jobs and associated resources, and the active Airflow Libraries and Operators with the associated credentials and secrets, run:

```
curl -k \
-H "Authorization: Bearer ${CDE_TOKEN}" \
```

```
-X GET "${CDE_JOBS_API}/admin/export?exportjobs=true&exportjobresources=
true&exportresources=false&exportactiveairflowpyenv=true" \
-H "accept: application/zip" \
```

3. (Optional) You can validate a backup file by uploading it to the /admin/list-archive endpoint. For example, for a backup file named cde-all-jobs-backup.zip:

```
curl -k \
-H "Authorization: Bearer ${CDE_TOKEN}" \
-X POST "${CDE_JOBS_API}/admin/list-archive" \
-H "accept: application/json" \
-H "Content-Type: multipart/form-data" \
-F "file=@/path/to/cde-all-jobs-backup.zip;type=application/zip" \
| jq
```


For Web UI

Before you begin



Important: The user interface for Cloudera Data Engineering 1.17 and higher has been updated. The left-hand menu provides easy access to commonly used pages. The steps below vary slightly, for example, the Overview page has been replaced with the Home page. You can also manage a job by clicking Jobs on the left-hand menu, then selecting a Virtual Cluster from a drop-down at the top of the Jobs page. The new home page still displays Virtual Clusters, but now includes quick-access links located at the top for the following categories: Jobs, Resources, and Download & Docs.

Steps

1. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
2. Click Jobs in the left navigation menu. The Jobs page displays.
3. From the drop-down in the upper left-hand corner, select the Virtual Cluster with the jobs that you want to back up.
4. Click  at the top right, and then click Backup Jobs.

Result

Depending on your browser settings, you are either prompted for a location to save the file, or the file is downloaded to your default download location. The file is a ZIP file named archive-*<TIMESTAMP>*.zip.

To restore a backup file, see [Restoring Cloudera Data Engineering jobs from backup](#).

Restoring Cloudera Data Engineering jobs from backup

You can restore Cloudera Data Engineering jobs and associated resources from a backup ZIP file.



Note: The backup and restore functionality does not include Airflow metadata related to connections and variable definitions. Recreate those manually in the new service. If the backup contains Airflow Libraries and Operators, consider the following:

- Restoring a backup with active Airflow Libraries and Operators can take 5 to 60 minutes, depending on the packages.
- You can restore the backup only to a virtual cluster that does not have active or maintenance mode Airflow Libraries and Operators.
- Restore the backup to the same Cloudera Data Engineering version. If you try to restore to a newer Cloudera Data Engineering and Airflow version, the restore can fail due to Python version or PIP constraint changes. In such a case, remove the Airflow Libraries and Operators from the archive.

For CDE CLI

Before you begin

- You must have a valid backup file to restore from. For instructions on backing up Cloudera Data Engineering jobs, see [Backing up Cloudera Data Engineering jobs](#) .
- Download and configure the CDE CLI.

Steps

1. Run the `cde backup restore` command to restore a backup file to the virtual cluster your CDE CLI is configured to interact with. Use the `--duplicate-handling` flag to select the policy for handling duplicate jobs. The possible values are:

error

Return an error if there are duplicate job names, and abort the restore operation. This is the default behavior.

rename

If a job name in the backup conflicts with an existing job, keep the existing job and rename the restored job by appending a numerical identifier to the job name.

keep-original

If a job name in the backup conflicts with an existing job, keep the existing job and do not restore the backed up job.

For example:

```
cde backup restore --local-path archive-2021-11-10T01:24:06.zip --duplicate-handling rename
```

--remote-path

If you use a remote storage, use this flag for the relative remote storage path of the backup to be restored. This restores the archive in the remote object store. Include a filename in the path. The path is relative to `/dex/backup/`.

Restore example for remote storage:

```
./cde backup restore --remote-path test/archive.zip  
# the output is similar to list-archive
```

Result

Validate that the jobs and resources were restored by running `cde job list` and `cde resource list`.

For Cloudera Data Engineering API



Note: Some of the command examples provided use `jq` to parse the JSON output and make it more readable.

Before you begin

- You must have a valid backup file to restore from. For instructions on backing up Cloudera Data Engineering jobs, see [Backing up Cloudera Data Engineering jobs](#) .
- Request an access token and save it as an environment variable to use in API calls. For instructions, see [Getting a Cloudera Data Engineering API access token](#) .

Steps

1. Determine the API URL for the virtual cluster that you want to restore the jobs and resources to:
 - a. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
 - b. Click Administration on the left navigation menu. The Administration page displays.
 - c. In the Services column, select the service containing the virtual cluster where you want to restore the job. Then, in the Virtual Clusters column, click the Cluster Details icon.
 - d. Click JOBS API URL. The Jobs API URL is copied to the clipboard.
 - e. Paste the URL into a text editor to set the Jobs API URL as an environment variable. For example:

```
export CDE_JOBS_API="https://pmjkrn5.cde-czlmkz4y.na-01.xvp2-7p8o.c
loudera.site/dex/api/v1"
```

2. Restore jobs from the backup file by uploading the backup file to the /admin/import endpoint. You can choose how to handle duplicate job names using the duplicatehandling=<POLICY> parameter. The options are:

error

Return an error if there are duplicate job names, and abort the restore operation.

rename

If a job name in the backup conflicts with an existing job, keep the existing job and rename the restored job by appending a numerical identifier to the job name.

keep-original

If a job name in the backup conflicts with an existing job, keep the existing job and do not restore the backed up job.

For example, to restore a backup named cde-etl-jobs-backup.zip using the rename duplicate handling policy:

```
curl -k \
-H "Authorization: Bearer ${CDE_TOKEN}" \
-X POST "${CDE_JOBS_API}/admin/import" \
-H "accept: application/json" \
-H "Content-Type: multipart/form-data" \
-F "file=@/path/to/cde-etl-jobs-backup.zip;type=application/zip" \
-F duplicatehandling=rename \
| jq
```

For Web UI


Before you begin



Important: The user interface for Cloudera Data Engineering 1.17 and higher has been updated. The left-hand menu provides easy access to commonly used pages. The steps below vary slightly, for example, the Overview page has been replaced with the Home page. You can also manage a job by clicking Jobs on the left-hand menu, then selecting a Virtual Cluster from a drop-down list at the top of the Jobs page. The new home page still displays Virtual Clusters, but now includes quick-access links located at the top for the following categories: Jobs, Resources, and Download & Docs.

- You must have a valid backup file to restore from. For instructions on backing up Cloudera Data Engineering jobs, see [Backing up Cloudera Data Engineering jobs](#).

Steps

1. In the Cloudera console, click the Data Engineering tile. The Cloudera Data Engineering Home page displays.
2. Click Jobs in the left navigation menu. The Jobs page displays.
3. From the drop-down list in the upper left-hand corner, select the Virtual Cluster that you want to restore jobs to.
4. Click  at the top right, and then click Restore Jobs.
5. Click Choose a zip file.
6. Browse to the ZIP file containing the backup of jobs and resources you want to restore, and then click Open.

7. Click Select to restore the backup.

Result

The jobs and resources from the backup file are restored using the rename duplicate handling policy. If a job name in the backup conflicts with an existing job, the restore operation keeps the existing job and renames the restored job by appending a numerical identifier to the job name. If the backup contains Airflow Libraries and Operators, the Airflow Libraries and Operators are restored as well.

Restoring jobs and Airflow Operators and Libraries from a Cloudera Data Engineering version 1.20.3 VC backup

In Cloudera Data Engineering version 1.20.3, the backup and restore function was not implemented yet for Airflow Operators and Libraries. To restore a Cloudera Data Engineering version 1.20.3 Virtual Cluster (VC) from a backup, first create the VC manually, after that create and activate the Airflow Operators and Libraries with the same requirements.txt file that the original VC used, and restore the jobs, resources, and so on from the archive.

Before you begin

Make sure that you have created a backup on a local storage from a Cloudera Data Engineering version 1.20.3 VC with active state Airflow Operators and Libraries with the following command:

```
cde backup create --include-jobs --include-resources --include-credential-secrets --include-credentials --local-path backup.zip
```

Optionally, you can validate the backup.

```
cde backup list-archive --local-path backup.zip
```

Procedure

1. Create a VC manually with the same configuration (Spark version, CPU, memory, and so on) as the original VC. For more information, see *Creating virtual clusters*.

2. Create and activate the Airflow Operators and Libraries manually. Use the same requirements.txt file that the backed up VC used.

For more information about obtaining the requirements.txt file, see *In-place upgrade with Airflow Operators and Libraries*.

3. Restore the jobs, resources, and so on from the archive.

```
cde backup restore --local-path backup.zip
```



Important: If there is a `cde-airflow-pyenv-[***ID***]` resource listed in the output in the Failed items section, ignore it.

Example:

```
$ cde backup restore --local-path backup.zip
 8.2KB/8.2KB 100% [=====]
backup.zip
All items restored successfully! Details:
Successfully restored items:
  Job:helloworld ( Created )
  Job:sparkhello ( Created )
  Resource:files ( Created )
  Resource:cde-airflow-pyenv-1711096070 ( Created )
```

Related Information

[Creating virtual clusters](#)

[In-place Upgrade with Airflow Operators and Libraries](#)