

Cloudera Data Engineering 1.22.0

Upgrading Cloudera Data Engineering

Date published: 2020-07-30

Date modified: 2024-08-27

CLOUDERA

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2024. 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

CDE upgrade version compatibility.....	4
Upgrading Cloudera Data Engineering.....	4
In-place upgrade with Airflow Operators and Libraries.....	6
Airflow constraints file.....	9
Upgrading Airflow if DAGs and packages are compatible with new Airflow version.....	20
Handling upgrade failures for Cloudera Data Engineering.....	20
Handling ineligible upgrades for Cloudera Data Engineering.....	23

CDE upgrade version compatibility

Cloudera Data Engineering (CDE) provides the in-place upgrade option from the eligible versions to the latest version. If you are on a version that is ineligible for in-place upgrade, the in-place upgrade option is disabled and you can perform a backup and restore.

Checking your upgrade options

To check if your CDE version is eligible for the in-place upgrade option, see [Support lifecycle policy](#).

In-place upgrades

To perform an in-place upgrade, you must be on an a version eligible for the in-place upgrade option. The *CDE versions eligible for in-place upgrade* table includes the specific AWS and Azure versions from which you can perform an in-place upgrade to the latest CDE version.

Table 1: CDE versions eligible for in-place upgrade

Supported upgrade versions	Target version
Azure: 1.20.3 and higher AWS: 1.20.3 and higher	1.22.0

Backup and restore

If you are on a version that is not eligible for the in-place upgrade option, to upgrade to the latest version, create a new CDE service and transfer your jobs to the new service by following the instructions in *Backing up and restoring CDE jobs*.

Upgrading Cloudera Data Engineering

Cloudera Data Engineering (CDE) supports in-place upgrades on both AWS and Azure. CDE provides an in-place upgrade option from the eligible versions to the latest version. If you are on a CDE version that is not eligible for the in-place upgrade, the in-place upgrade option is disabled and you can perform a backup and restore. The upgrades can be triggered by an Admin from the CDE user interface.

About this task

To check whether your CDE version is eligible for the in-place upgrade option, see [Support lifecycle policy](#).

When you upgrade to the latest version in Cloudera Data Engineering, the upgrade process prepares your service for the upgrade with a click of a button. Your information is also backed up in this process. Learn how to complete the upgrade process here.



Important:

- Virtual clusters for this service are unavailable during the upgrade process and the upgrade puts your service into maintenance. During this time, scheduled jobs are paused and any active jobs are killed. The upgrade can take some time so be advised and alert your team of the downtime.
- Before the upgrade, delete any existing Airflow operators and libraries. If you have configured and built any Airflow libraries and operators, activate and then delete them before the upgrade. For more information, see [In-place upgrade with Airflow Operators and Libraries](#) and [Upgrading Airflow if DAGs and packages are compatible with new Airflow version](#).

The following are not included in the backup:

- Python-venv resources
- Airflow custom operators and libraries
- Airflow connections
- Airflow variables
- Logs
 - Job Run logs (Driver, Executor, API)
 - Service logs
 - Virtual Cluster event logs
 - Spark Session logs (Including statement history)
 - Airflow DAG logs
- Job Runs
 - Job Run history
- Virtual Cluster end-points
- Resource: Docker runtimes

Before you begin

Ensure that the catchup option is not enabled for any user's Airflow jobs. Before the backup starts, if the Airflow DAG catchup options are enabled, disable them manually.

Procedure

1. In the Cloudera Data Platform (CDP) console, click the Data Engineering tile.
The CDE **Home** page displays.
2. In the left navigation menu, click Administration, select a Service, and click Upgrade.
3. Click the Maintenance tab and click Upgrade Service.
4. Click Start Preparation.

CDE prepares the service for you. If the service cannot be prepared, click Retry or Cancel Upgrade. Canceling the upgrade returns your service to the original state. Optionally, you can click Download Logs to view the Diagnostics page where you can generate a diagnostic bundle. If the preparation is successful, proceed to the next step. If the preparation and cancellation fails, you must contact Cloudera support.

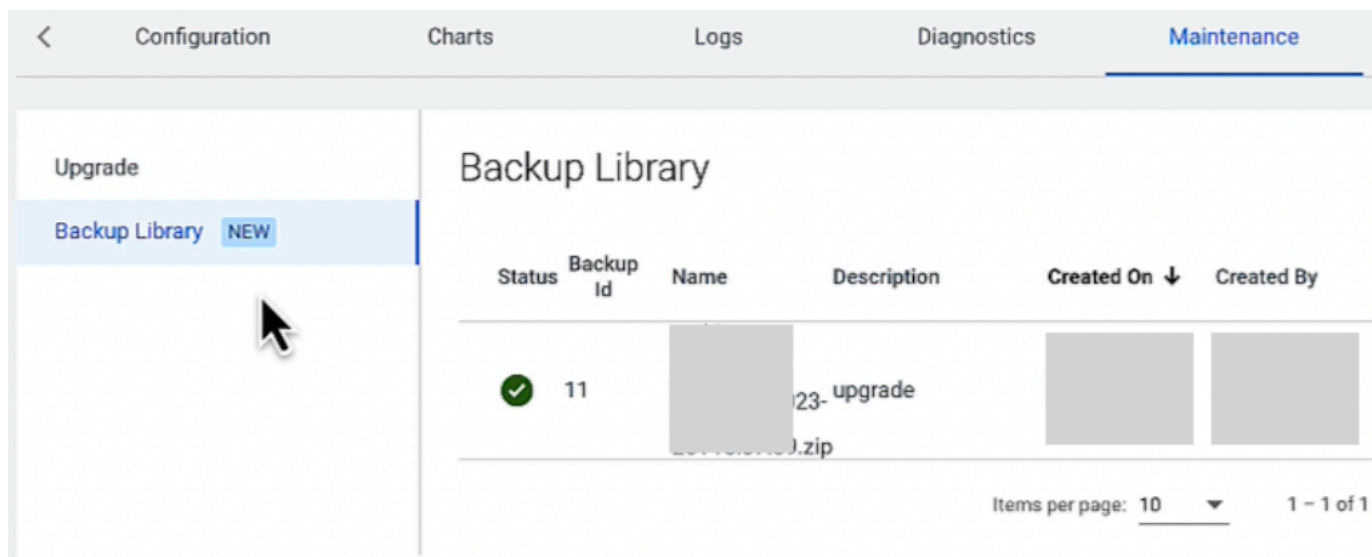
5. Click Start Backup.

CDE automatically creates a backup for you. The backup is used for restoring if an irrecoverable failure happens during the upgrade.



Note: If the backup fails, you have the option to click Retry Upgrade or Contact Support. If the upgrade fails a second time, you have the option to contact support or perform a manual upgrade with jobs backup and restore. If the backup is successful, you can proceed with the upgrade.

6. When the backup is complete, to view the backup that was created, select Backup Library.



7. Click Start Upgrade.
8. After the upgrade is complete, to enable all paused jobs and sessions, click Resume Service.
The service restarts.

Results

After the upgrade is complete and you resumed the service, the service operates on the upgraded version of CDE. If you cancel the upgrade, your service resumes the previous state, which enables all paused jobs and resumes all sessions.

Related Information

[Handling upgrade failures for Cloudera Data Engineering](#)

In-place upgrade with Airflow Operators and Libraries

This section details Airflow considerations to avoid issues during and after the upgrade. Before performing an in-place upgrade of the Airflow operators and libraries, check whether the environment is compatible. This allows you to determine if the DAGs or the python package dependencies require changes.

About this task



Note:

Check the compatibility for all used Airflow operators and libraries. If you do not check the compatibility, the recreation of the Airflow operators and libraries, or the DAG execution can fail.

The following is the recommended procedure. However, if you are confident that the DAGs and the packages in the Airflow Custom Libraries and Operators are compatible with the new Airflow version, instead of following this procedure, you can perform the upgrade as described in [Upgrading Airflow if DAGs and packages are compatible with new Airflow version](#).

Before you begin

Ensure that the catchup option is not enabled for any user's Airflow jobs. Before the backup starts, if the Airflow DAG catchup options are enabled, disable them manually.

Verify Airflow operators and libraries and DAGs in the newer CDE service:

Procedure

1. Create a new test CDE service with the latest CDE version.
2. Obtain the requirements.txt for the existing Airflow environment. The requirements.txt is required after the upgrade to restore the environments.

To obtain the Airflow environment requirements.txt for a virtual cluster, perform the following steps.

For In CDE 1.21 and higher

- a. Obtain the environment information json from the Airflow environment API endpoint.
 1. Copy the JOBS API URL. For more information, check the Determine the virtual cluster endpoint URL step in *Configuring the CLI client*.
 2. Use the JOBS API URL as a base and add /admin/airflow/env.

JOBS API URL example with /admin/airflow/env:

```
https://t5kh5fjp.cde-hfrsp8ww.dex-priv.xcu2-8y8x.dev.cldr.work/dex/api/v1/admin/airflow/env
```

Example for the json:

```
cat pyenv.json
{
  "status": "activated",
  "packages": [
    {
      "type": "python-module",
      "name": "mypy",
      "version": "1.2.0",
      "created": "2024-04-03T16:58:07Z"
    },
    {
      "type": "python-module",
      "name": "mypy-extensions",
      "version": "1.0.0",
      "created": "2024-04-03T16:58:07Z"
    },
    {
      "type": "python-module",
      "name": "tomli",
      "version": "2.0.1",
      "created": "2024-04-03T16:58:07Z"
    }
  ],
  "requirements": "mypy\n"
}
```

- b. From the returned `pyenv.json`, generate a `requirements.txt` file with the `jq` tool:

```
cat pyenv.json | jq -r '.requirements'> requirements.txt
```

Example for the generated `requirements.txt`:

```
mypy
```

For In CDE 1.20

Use the CDE CLI to find and download the `requirements.txt` file which belongs to the Airflow environment in a virtual cluster:

- a. Find the resource name that belongs to the python environment. Optionally use `jq` to get the name:

```
cde resource list --filter "type[eq]airflow-python-env" --filter "status[eq]active" --show-hidden=true | jq '[0].name'
cde-airflow-pyenv-1712224823
```

- b. Download the `requirements.txt` file:

```
cde resource download --name cde-airflow-pyenv-1712224823 --resource-path requirements.txt
```

3. Verify the `requirements.txt` file against an [Airflow constraints file](#).

- Get the Python version used for Airflow from *Compatibility for Cloudera Data Engineering and Runtime components*.
- Check `requirements.txt` against the CDE constraint file by creating a temporary python virtual environment and use the dry-run feature of pip. If pip does not support dry-run, perform the check by installing the packages without the `--dry-run` flag:

```
# create a temporary local python virtual environment using the same python version used in Airflow in CDE.
python -m venv venv
# source the environment
source venv/bin/activate
# check if there are any conflicts between the packages in the requirements.txt # file and the packages needed by airflow
# At least pip 22.2 is needed for the dry-run feature.
(venv) python -m pip install -r requirements.txt -c "https://raw.githubusercontent.com/apache/airflow/constraints-2.7.3/constraints-3.8.txt"
--dry-run
```

4. Create a compatible python environment in a test virtual cluster in the test Service.

Python packages might need to be updated to work with the Airflow service in the new CDE version.

- Backup and restore your Airflow jobs or create all the necessary Airflow jobs in the test CDE Service that you created in step 1 on page 7.
- If the DAG parsing fails, go to the Airflow UI to identify the impacted DAGs.
- Fix the DAGs or update the dependencies in the `requirements.txt` to fix the issues.
- Conduct a test run of these DAGs if they call custom libraries at runtime to ensure that the code defined within the tasks is also valid.
- If the DAG execution fails, go to the Airflow job logs to identify the impacted DAGs.
- Fix the DAGs or update the dependencies in the `requirements.txt` to fix the issues.

Backport all DAG changes to the CDE service to be upgraded, so that DAGs are compatible with the previous and the new version of Airflow. This may require a retest on an older CDE service test virtual cluster, if the production

environment cannot be updated easily. If changes are needed for the older CDE service, test the changes again on the newer CDE service. Verify if both services work as expected.

11. Delete the test service.
12. Delete all Airflow operators and libraries in all virtual clusters in the CDE service before executing the upgrade.
13. Continue with the upgrade.
14. After a successful upgrade, build and activate the Airflow operators and libraries in the virtual clusters from the updated and fixed requirements.txt.
15. Verify that the Airflow jobs are properly running in the Airflow UI and CDE Jobs UI.
16. Remove and retest backward compatible changes made in the DAGs for the previous CDE version. Keep the DAGs updated to the latest Airflow version in CDE.

Related Information

[Configuring the CLI client](#)

[Compatibility for Cloudera Data Engineering and Runtime components](#)

[Upgrading Airflow if DAGs and packages are compatible with new Airflow version](#)

[Airflow constraints file](#)

Airflow constraints file

Before performing an in-place upgrade of the Airflow operators and libraries, check whether the environment is compatible. Use the Airflow constraints file to verify its content against the requirements.txt file that belongs to the Airflow environment in a virtual cluster.

Airflow version 2.7.3 constraints file

For more information about checking the Airflow constraints file against the requirements.txt file, see step 3 on page 8 in *In-place upgrade with Airflow Operators and Libraries*.

```
apache-airflow==2.7.3
waitress==2.1.1
argparse==1.4.0
Authlib==1.2.1
Babel==2.13.1
ConfigUpdater==3.1.1
Deprecated==1.2.14
Flask-AppBuilder==4.3.6
Flask-Babel==2.0.0
Flask-Bcrypt==1.0.1
Flask-Caching==2.1.0
Flask-JWT-Extended==4.5.3
Flask-Limiter==3.5.0
Flask-Login==0.6.3
Flask-SQLAlchemy==2.5.1
Flask-Session==0.5.0
Flask-WTF==1.2.1
Flask==2.2.5
GitPython==3.1.40
JPype1==1.4.1
JayDeBeApi==1.2.3
Jinja2==3.1.2
Js2Py==0.74
Mako==1.2.4
Markdown==3.5.1
MarkupSafe==2.1.3
PyGithub==2.1.1
PyHive==0.7.0
PyJWT==2.8.0
```

```
PyNaCl==1.5.0
PyYAML==6.0.1
Pygments==2.16.1
SQLAlchemy-JSONField==1.0.1.post0
SQLAlchemy-Utils==0.41.1
SQLAlchemy==1.4.50
SecretStorage==3.3.3
Sphinx==5.3.0
WTForms==3.0.1
Werkzeug==2.2.3
adal==1.2.7
adlfs==2023.10.0
aiobotocore==2.7.0
aiofiles==23.2.1
aiohttp==3.8.6
aioitertools==0.11.0
aioresponses==0.7.4
aiosignal==1.3.1
alabaster==0.7.13
alembic==1.12.1
alibabacloud-adb20211201==1.0.0
alibabacloud-credentials==0.3.2
alibabacloud-endpoint-util==0.0.3
alibabacloud-gateway-spi==0.0.1
alibabacloud-openapi-util==0.2.2
alibabacloud-tea-openapi==0.3.7
alibabacloud-tea-util==0.3.11
alibabacloud-tea-xml==0.0.2
alibabacloud-tea==0.3.3
aliyun-python-sdk-core==2.14.0
aliyun-python-sdk-kms==2.16.2
amqp==5.1.1
analytics-python==1.4.post1
annotated-types==0.6.0
ansiwrap==0.8.4
anyascii==0.3.2
anyio==4.0.0
apache-airflow-providers-airbyte==3.4.0
apache-airflow-providers-alibaba==2.6.0
apache-airflow-providers-amazon==8.10.0
apache-airflow-providers-apache-beam==5.3.0
apache-airflow-providers-apache-cassandra==3.3.0
apache-airflow-providers-apache-drill==2.5.0
apache-airflow-providers-apache-druid==3.6.0
apache-airflow-providers-apache-flink==1.2.0
apache-airflow-providers-apache-hdfs==4.2.0
apache-airflow-providers-apache-hive==6.2.0
apache-airflow-providers-apache-impala==1.2.0
apache-airflow-providers-apache-kafka==1.2.0
apache-airflow-providers-apache-kylin==3.3.0
apache-airflow-providers-apache-livy==3.6.0
apache-airflow-providers-apache-pig==4.2.0
apache-airflow-providers-apache-pinot==4.2.0
apache-airflow-providers-apache-spark==4.3.0
apache-airflow-providers-apache-sqoop==4.1.0
apache-airflow-providers-apprise==1.1.0
apache-airflow-providers-arangodb==2.3.0
apache-airflow-providers-asana==2.3.0
apache-airflow-providers-atlassian-jira==2.2.0
apache-airflow-providers-celery==3.4.1
apache-airflow-providers-cloudant==3.3.0
apache-airflow-providers-cncf-kubernetes==7.8.0
apache-airflow-providers-common-sql==1.8.0
apache-airflow-providers-daskexecutor==1.1.0
```

```
apache-airflow-providers-databricks==4.7.0
apache-airflow-providers-datadog==3.4.0
apache-airflow-providers-dbt-cloud==3.4.0
apache-airflow-providers-dingding==3.3.0
apache-airflow-providers-discord==3.4.0
apache-airflow-providers-docker==3.8.0
apache-airflow-providers-elasticsearch==5.1.0
apache-airflow-providers-exasol==4.3.0
apache-airflow-providers-facebook==3.3.0
apache-airflow-providers-ftp==3.6.0
apache-airflow-providers-github==2.4.0
apache-airflow-providers-google==10.11.0
apache-airflow-providers-grpc==3.3.0
apache-airflow-providers-hashicorp==3.5.0
apache-airflow-providers-http==4.6.0
apache-airflow-providers-imap==3.4.0
apache-airflow-providers-influxdb==2.3.0
apache-airflow-providers-jdbc==4.1.0
apache-airflow-providers-jenkins==3.4.0
apache-airflow-providers-microsoft-azure==8.1.0
apache-airflow-providers-microsoft-mssql==3.5.0
apache-airflow-providers-microsoft-psrp==2.4.0
apache-airflow-providers-microsoft-winrm==3.3.0
apache-airflow-providers-mongo==3.4.0
apache-airflow-providers-mysql==5.4.0
apache-airflow-providers-neo4j==3.4.0
apache-airflow-providers-odbc==4.1.0
apache-airflow-providers-openfaas==3.3.0
apache-airflow-providers-openlineage==1.2.0
apache-airflow-providers-opensearch==1.0.0
apache-airflow-providers-opsgenie==5.2.0
apache-airflow-providers-oracle==3.8.0
apache-airflow-providers-pagerduty==3.4.0
apache-airflow-providers-papermill==3.4.0
apache-airflow-providers-plexus==3.3.0
apache-airflow-providers-postgres==5.7.1
apache-airflow-providers-presto==5.2.1
apache-airflow-providers-redis==3.4.0
apache-airflow-providers-salesforce==5.5.0
apache-airflow-providers-samba==4.3.0
apache-airflow-providers-segment==3.3.0
apache-airflow-providers-sendgrid==3.3.0
apache-airflow-providers-sftp==4.7.0
apache-airflow-providers-singularity==3.3.0
apache-airflow-providers-slack==8.3.0
apache-airflow-providers-smtp==1.4.1
apache-airflow-providers-snowflake==5.1.0
apache-airflow-providers-sqlite==3.5.0
apache-airflow-providers-ssh==3.8.1
apache-airflow-providers-tableau==4.3.0
apache-airflow-providers-tabular==1.3.0
apache-airflow-providers-telegram==4.2.0
apache-airflow-providers-trino==5.4.0
apache-airflow-providers-vertica==3.6.0
apache-airflow-providers-zendesk==4.4.0
apache-beam==2.51.0
apispec==6.3.0
apprise==1.6.0
argcomplete==3.1.3
arrow==1.3.0
asana==3.2.2
asgiref==3.7.2
asn1crypto==1.5.1
astroid==2.15.8
```

```
asttokens==2.4.1
async-timeout==4.0.3
atlasclient==1.0.0
atlassian-python-api==3.41.3
attrs==23.1.0
aws-sam-translator==1.79.0
aws-xray-sdk==2.12.1
azure-batch==14.0.0
azure-common==1.1.28
azure-core==1.29.5
azure-cosmos==4.5.1
azure-datalake-store==0.0.53
azure-identity==1.15.0
azure-keyvault-secrets==4.7.0
azure-kusto-data==4.2.0
azure-mgmt-containerinstance==10.1.0
azure-mgmt-containerregistry==10.2.0
azure-mgmt-core==1.4.0
azure-mgmt-cosmosdb==9.3.0
azure-mgmt-datafactory==3.1.0
azure-mgmt-datalake-nspkg==3.0.1
azure-mgmt-datalake-store==0.5.0
azure-mgmt-nspkg==3.0.2
azure-mgmt-resource==23.0.1
azure-mgmt-storage==21.1.0
azure-nspkg==3.0.2
azure-servicebus==7.11.3
azure-storage-blob==12.18.3
azure-storage-common==2.1.0
azure-storage-file-datalake==12.13.2
azure-storage-file-share==12.14.2
azure-storage-file==2.1.0
azure-synapse-spark==0.7.0
backcall==0.2.0
backoff==1.10.0
backports.zoneinfo==0.2.1
bcrypt==4.0.1
beautifulsoup4==4.12.2
billiard==4.1.0
bitarray==2.8.2
black==23.10.1
blinker==1.6.3
boto3==1.28.64
botocore==1.31.64
cachelib==0.9.0
cachetools==5.3.2
cassandra-driver==3.28.0
cattr==23.1.2
celery==5.3.4
certifi==2023.7.22
cffi==1.16.0
cfgv==3.4.0
cfn-lint==0.83.1
cgroupspy==0.2.2
chardet==5.2.0
charset-normalizer==3.3.2
checksumdir==1.2.0
ciso8601==2.3.1
click-didyoumean==0.3.0
click-plugins==1.1.1
click-repl==0.3.0
click==8.1.7
clickclick==20.10.2
cloudant==2.15.0
```

```
cloudpickle==2.2.1
colorama==0.4.6
colorlog==4.8.0
confluent-kafka==2.3.0
connexion==2.14.2
coverage==7.3.2
crcmod==1.7
cron-descriptor==1.4.0
croniter==2.0.1
cryptography==41.0.5
curlify==2.2.1
dask==2023.4.1
databricks-sql-connector==2.9.3
datadog==0.47.0
db-dtypes==1.1.1
decorator==5.1.1
defusedxml==0.7.1
deprecation==2.1.0
dill==0.3.1.1
distlib==0.3.7
distributed==2023.4.1
dnspython==2.4.2
docker==6.1.3
docopt==0.6.2
docutils==0.20.1
duckdb==0.9.1
ecdsa==0.18.0
elastic-transport==8.10.0
elasticsearch==8.10.1
email-validator==1.3.1
entrypoints==0.4
eralchemy2==1.3.8
et-xmlfile==1.1.0
eventlet==0.33.3
exceptiongroup==1.1.3
execnet==2.0.2
executing==2.0.1
facebook-business==18.0.3
fastavro==1.9.0
fasteners==0.19
fastjsonschema==2.18.1
filelock==3.13.1
flower==2.0.1
frozenlist==1.4.0
fsspec==2023.10.0
future==0.18.3
gcloud-aio-auth==4.2.3
gcloud-aio-bigquery==7.0.0
gcloud-aio-storage==9.0.0
gcsfs==2023.10.0
geomet==0.2.1.post1
gevent==23.9.1
gitdb==4.0.11
google-ads==22.1.0
google-api-core==2.12.0
google-api-python-client==2.106.0
google-auth-http2==0.1.1
google-auth-oauthlib==1.1.0
google-auth==2.23.4
google-cloud-aiplatform==1.36.0
google-cloud-appengine-logging==1.3.2
google-cloud-audit-log==0.2.5
google-cloud-automl==2.11.3
google-cloud-batch==0.17.2
```

```
google-cloud-bigquery-datatransfer==3.12.1
google-cloud-bigquery-storage==2.22.0
google-cloud-bigquery==3.13.0
google-cloud-bigtable==2.21.0
google-cloud-build==3.20.1
google-cloud-compute==1.14.1
google-cloud-container==2.32.0
google-cloud-core==2.3.3
google-cloud-datacatalog==3.16.0
google-cloud-dataflow-client==0.8.5
google-cloud-dataform==0.5.3
google-cloud-dataplex==1.7.0
google-cloud-dataproc-metastore==1.13.0
google-cloud-dataproc==5.6.0
google-cloud-dlp==3.12.3
google-cloud-kms==2.19.2
google-cloud-language==2.11.1
google-cloud-logging==3.8.0
google-cloud-memcache==1.7.3
google-cloud-monitoring==2.16.0
google-cloud-orchestration-airflow==1.9.2
google-cloud-os-login==2.11.0
google-cloud-pubsub==2.18.4
google-cloud-redis==2.13.2
google-cloud-resource-manager==1.10.4
google-cloud-run==0.10.0
google-cloud-secret-manager==2.16.4
google-cloud-spanner==3.40.1
google-cloud-speech==2.21.1
google-cloud-storage-transfer==1.9.2
google-cloud-storage==2.13.0
google-cloud-tasks==2.14.2
google-cloud-texttospeech==2.14.2
google-cloud-translate==3.12.1
google-cloud-videointelligence==2.11.4
google-cloud-vision==3.4.5
google-cloud-workflows==1.12.1
google-crc32c==1.5.0
google-re2==1.1
google-resumable-media==2.6.0
googleapis-common-protos==1.61.0
graphql-core==3.2.3
graphviz==0.20.1
greenlet==3.0.1
grpc-google-iam-v1==0.12.6
grpcio-gcp==0.2.2
grpcio-status==1.59.2
grpcio==1.59.2
gssapi==1.8.3
unicorn==21.2.0
h11==0.14.0
hdfs==2.7.3
hmsclient==0.1.1
httpcore==0.16.3
httplib2==0.22.0
httpx==0.23.3
humanize==4.8.0
hvac==2.0.0
identify==2.5.31
idna==3.4
ijson==3.2.3
imagesize==1.4.1
importlib-metadata==6.8.0
importlib-resources==6.1.0
```

```
impyla==0.18.0
incremental==22.10.0
inflection==0.5.1
influxdb-client==1.38.0
iniconfig==2.0.0
ipdb==0.13.13
ipython==8.12.3
isodate==0.6.1
itsdangerous==2.1.2
jaraco.classes==3.3.0
jedi==0.19.1
jeepney==0.8.0
jmespath==0.10.0
jschema-to-python==1.2.3
json-merge-patch==0.2
jsondiff==2.0.0
jsonpatch==1.33
jsonpath-ng==1.6.0
jsonpickle==3.0.2
jsonpointer==2.4
jsonschema-path==0.3.1
jsonschema-specifications==2023.7.1
jsonschema==4.19.2
junit-xml==1.9
jupyter_client==8.5.0
jupyter_core==5.5.0
keyring==24.2.0
kombu==5.3.2
krb5==0.5.1
kubernetes-asyncio==24.2.3
kubernetes==23.6.0
kylinpy==2.8.4
lazy-object-proxy==1.9.0
ldap3==2.9.1
limits==3.6.0
linkify-it-py==2.0.2
loket==1.0.0
lockfile==0.12.2
looker-sdk==23.16.0
lxml==4.9.3
lz4==4.3.2
markdown-it-py==3.0.0
marshmallow-oneofschema==3.0.1
marshmallow-sqlalchemy==0.26.1
marshmallow==3.20.1
matplotlib-inline==0.1.6
mdit-py-plugins==0.4.0
mdurl==0.1.2
mongomock==4.1.2
monotonic==1.6
more-itertools==10.1.0
moto==4.2.7
mpmath==1.3.0
msal-extensions==1.0.0
msal==1.24.1
msgpack==1.0.7
msrest==0.7.1
msrestazure==0.6.4
multi-key-dict==2.0.3
multidict==6.0.4
mypy-boto3-appflow==1.28.42
mypy-boto3-rds==1.28.76
mypy-boto3-redshift-data==1.28.36
mypy-boto3-s3==1.28.55
```

```
mypy-extensions==1.0.0
mypy==1.2.0
mysql-connector-python==8.0.29
mysqlclient==2.2.0
nbclient==0.8.0
nbformat==5.9.2
neo4j==5.14.0
networkx==3.1
nh3==0.2.14
nodeenv==1.8.0
numpy==1.24.4
oauthlib==3.2.2
objsize==0.6.1
openapi-schema-validator==0.6.2
openapi-spec-validator==0.7.1
openlineage-integration-common==1.4.1
openlineage-python==1.4.1
openlineage_sql==1.4.1
openpyxl==3.1.2
opensearch-py==2.3.2
opentelemetry-api==1.20.0
opentelemetry-exporter-otlp-proto-common==1.20.0
opentelemetry-exporter-otlp-proto-grpc==1.20.0
opentelemetry-exporter-otlp-proto-http==1.20.0
opentelemetry-exporter-otlp==1.20.0
opentelemetry-exporter-prometheus==1.12.0rc1
opentelemetry-proto==1.20.0
opentelemetry-sdk==1.20.0
opentelemetry-semantic-conventions==0.41b0
opsgenie-sdk==2.1.5
oracledb==1.4.2
ordered-set==4.1.0
orjson==3.9.10
oscrypto==1.3.0
oss2==2.18.3
packaging==23.2
pandas-gbq==0.19.2
pandas==2.0.3
papermill==2.4.0
paramiko==3.3.1
parso==0.8.3
partd==1.4.1
pathable==0.4.3
pathspec==0.11.2
pbr==5.11.1
pdpyras==5.1.2
pendulum==2.1.2
pexpect==4.8.0
pickleshare==0.7.5
pinotdb==5.1.0
pipdeptree==2.13.0
pipx==1.2.1
pkginfo==1.9.6
pkgutil_resolve_name==1.3.10
platformdirs==3.11.0
pluggy==1.3.0
ply==3.11
plyvel==1.5.0
portalocker==2.8.2
pre-commit==3.5.0
presto-python-client==0.8.4
prison==0.2.1
prometheus-client==0.18.0
prompt-toolkit==3.0.39
```



```
proto-plus==1.22.3
protobuf==4.24.4
psutil==5.9.6
psycpg2-binary==2.9.9
ptyprocess==0.7.0
pure-eval==0.2.2
pure-sasl==0.6.2
py-partiql-parser==0.4.1
py4j==0.10.9.7
pyOpenSSL==23.3.0
pyarrow==11.0.0
pyasn1-modules==0.3.0
pyasn1==0.5.0
pycountry==22.3.5
pycparser==2.21
pycryptodome==3.19.0
pycryptodomex==3.19.0
pydantic==2.4.2
pydantic_core==2.10.1
pydata-google-auth==1.8.2
pydot==1.4.2
pydruid==0.6.5
pyenchanted==3.2.2
pyexasol==0.25.2
pygraphviz==1.11
pyjspark==2.7.1
pykerberos==1.2.4
pymongo==4.5.0
pymssql==2.2.10
pyodbc==5.0.1
pyparsing==3.1.1
pypsrp==0.8.1
pyspark==3.5.0
pyspnego==0.10.2
pytest-asyncio==0.21.1
pytest-cov==4.1.0
pytest-httpx==0.21.3
pytest-instafail==0.5.0
pytest-mock==3.12.0
pytest-rerunfailures==12.0
pytest-timeouts==1.2.1
pytest-xdist==3.3.1
pytest==7.4.3
python-arango==7.7.0
python-daemon==3.0.1
python-dateutil==2.8.2
python-dotenv==1.0.0
python-http-client==3.3.7
python-jenkins==1.7.0
python-jose==3.3.0
python-ldap==3.4.3
python-nvd3==0.15.0
python-slugify==8.0.1
python-telegram-bot==20.2
pytz==2023.3.post1
pytzdata==2020.1
pywinrm==0.4.3
pyzmq==25.1.1
reactivex==4.0.4
readme-renderer==42.0
redis==4.6.0
redshift-connector==2.0.915
referencing==0.30.2
regex==2023.10.3
```

```
requests-file==1.5.1
requests-kerberos==0.14.0
requests-mock==1.11.0
requests-ntlm==1.2.0
requests-oauthlib==1.3.1
requests-toolbelt==1.0.0
requests==2.31.0
responses==0.23.3
rfc3339-validator==0.1.4
rfc3986==1.5.0
rich-argparse==1.4.0
rich-click==1.7.1
rich==13.6.0
rpds-py==0.10.6
rsa==4.9
ruff==0.1.3
s3transfer==0.7.0
sarif-om==1.0.4
scrap==1.4.4
scrapbook==0.5.0
semver==3.0.2
sendgrid==6.10.0
sentinels==1.0.0
sentry-sdk==1.33.1
setproctitle==1.3.3
shapely==2.0.2
simple-salesforce==1.12.5
six==1.16.0
slack-sdk==3.23.0
smbprotocol==1.11.0
smap==5.0.1
sniffio==1.3.0
snowballstemmer==2.2.0
snowflake-connector-python==3.3.1
snowflake-sqlalchemy==1.5.0
sortedcontainers==2.4.0
soupsieve==2.5
sphinx-airflow-theme==0.0.12
sphinx-argparse==0.4.0
sphinx-autoapi==2.1.1
sphinx-copybutton==0.5.2
sphinx-jinja==2.0.2
sphinx-rtd-theme==1.3.0
sphinxcontrib-applehelp==1.0.4
sphinxcontrib-devhelp==1.0.2
sphinxcontrib-htmlhelp==2.0.1
sphinxcontrib-httpdomain==1.8.1
sphinxcontrib-jquery==4.1
sphinxcontrib-jsmath==1.0.1
sphinxcontrib-qthelp==1.0.3
sphinxcontrib-redoc==1.6.0
sphinxcontrib-serializinghtml==1.1.5
sphinxcontrib-spelling==8.0.0
spython==0.3.1
sqlalchemy-bigquery==1.8.0
sqlalchemy-drill==1.1.4
sqlalchemy-redshift==0.8.14
sqlalchemy-spanner==1.6.2
sqlparse==0.4.4
sshpkeys==3.3.1
sshtunnel==0.4.0
stack-data==0.6.3
starkbank-ecdsa==2.2.0
statsd==4.0.1
```

```
sympy==1.12
tableauserverclient==0.25
tabulate==0.9.0
tblib==3.0.0
tenacity==8.2.3
termcolor==2.3.0
text-unidecode==1.3
textwrap3==0.9.2
thrift-sasl==0.4.3
thrift==0.16.0
time-machine==2.13.0
tomli==2.0.1
tomlkit==0.12.1
toolz==0.12.0
tornado==6.3.3
towncrier==23.10.0
tqdm==4.66.1
traitlets==5.13.0
trino==0.327.0
twine==4.0.2
types-Deprecated==1.2.9.3
types-Markdown==3.5.0.0
types-PyMySQL==1.1.0.1
types-PyYAML==6.0.12.12
types-certifi==2021.10.8.3
types-croniter==2.0.0.0
types-docutils==0.20.0.3
types-paramiko==3.3.0.0
types-protobuf==4.24.0.4
types-pyOpenSSL==23.3.0.0
types-python-dateutil==2.8.19.14
types-python-slugify==8.0.0.3
types-pytz==2023.3.1.1
types-redis==4.6.0.8
types-requests==2.31.0.6
types-setuptools==68.2.0.0
types-tabulate==0.9.0.3
types-termcolor==1.1.6.2
types-toml==0.10.8.7
types-urllib3==1.26.25.14
typing_extensions==4.8.0
tzdata==2023.3
tzlocal==5.2
uc-micro-py==1.0.2
unicodecsv==0.14.1
uritemplate==4.1.1
urllib3==1.26.18
userpath==1.9.1
vertica-python==1.3.6
vine==5.0.0
virtualenv==20.24.6
watchtower==3.0.1
wcwidth==0.2.9
websocket-client==1.6.4
wrapt==1.15.0
xmldict==0.13.0
yamllint==1.32.0
yarl==1.9.2
zeep==4.2.1
zenpy==2.0.41
zict==3.0.0
zipp==3.17.0
zope.event==5.0
zope.interface==6.1
```

```
zstandard==0.22.0
```

Related Information

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

Upgrading Airflow if DAGs and packages are compatible with new Airflow version

You can upgrade Airflow after ensuring the DAGs and the packages in the Airflow Custom Libraries and Operators are compatible with the new Airflow version by using the requirements.txt file.

Before you begin



Note:

To avoid issues after the upgrade, ensure that the DAGs and the packages in the Airflow Custom Libraries and Operators are compatible with the new Airflow version. Fixing the issues after the upgrade process is complete might cause further downtime in performing the job.

Ensure that the catchup option is not enabled for any user's Airflow jobs. Before the backup starts, if the Airflow DAG catchup options are enabled, disable them manually.

Procedure

1. Get the requirements.txt file for the current Airflow Custom Libraries and Operators.
After the upgrade, you need the requirements.txt file to restore Python environments. For information on how to get the requirements.txt file, see *In-place upgrade with Airflow Operators and Libraries*.
2. Verify the requirements.txt file against an Airflow constraints file.
For more information, see *In-place upgrade with Airflow Operators and Libraries*.
 - a) If you run into errors during the verification, update the dependencies in the requirements.txt file.
3. Disable all Airflow Custom Libraries and Operators in all virtual clusters in the CDE service.
4. Perform the upgrade process.
5. Using the saved requirements.txt file from step 2 on page 20, build and activate the Airflow Python environment in the virtual clusters.
If the build fails because of non-python related dependency issues, you may need to make version adjustments in the requirements.txt file.
6. On the Airflow UI and CDE jobs UI, verify that the Airflow jobs are running.
In the case of DAG-related errors:
 - If the DAG parsing fails, look into the Airflow UI to identify the impacted DAGs.
 - If the DAG execution fails, look into the Airflow job logs to identify the impacted DAGs.
7. Fix the impacted DAGs or update the dependencies in the requirements.txt file.

Related Information

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

Handling upgrade failures for Cloudera Data Engineering

If your upgrade of Cloudera Data Engineering (CDE) fails, you have the option to clone the service with the latest version of CDE. Learn how to handle an upgrade failure.

About this task

During a CDE upgrade, a backup is created as part of the upgrade preparation process. This procedure uses that backup to be restored in a new cluster.

The list of service backups is available in the Backup Library. To locate the Backup Library, in the left navigation menu of CDE select Administration, select Service Details, select the Maintenance tab, and select Backup Library.

To obtain the list of all available backups, in the CDP CLI, run:

```
cdp de list-backups
```

To obtain the list of service backups associated with a specific CDP environment, run "cdp de list-backups --filter "environment(eq)[****CDP ENVIRONMENT NAME****]"

The CDE backup includes the following:

- CDE Service configurations
- Virtual cluster names
- Virtual cluster configurations
- Virtual cluster file-based resources
- Spark job definitions
- Airflow job definitions
- Spark Python-env resources

The following are not yet included in the backup:

- Non file-based resources, for example, Python-venv resources and custom runtimes
- Airflow custom operators & libraries
- Logs
- Job run history
- Endpoints

Before you begin



Important: This procedure changes the virtual cluster endpoints.

1. Ensure that the catchup option is not enabled for any user's Airflow jobs.

Before the backup starts, if the Airflow DAG catchup options are enabled, disable them manually.

2. By default, the restored service receives the name and ID of the original backed-up service. To ensure that the backup does not fail due to name and ID conflicts, perform either of these options:
 - a. Delete the original service, which failed to upgrade during upgrading CDE.
 - b. Rename the service and assign a new ID to it using the --service-id and --service-name options.



Note:

Assigning a new ID results in new Fully Qualified Domain Names (FQDNs) for the service and all its virtual clusters, which affects the endpoint stability.

A valid service ID is:

- An 8 character-long alphanumeric string
- Does not contain vowels
- Unique

Procedure

1. Restore the service from the backup.

```
cdp de restore-service --backup-id <backup-id> --environment-crn
<environment-crn>
```

Where:

backup-id

The ID of the backup that you are restoring from.

environment-crn

The Customer Resource Number (CRN) of the Cloudera Data Platform (CDP) environment with which a restored CDE service is associated. Currently, you can restore the CDE service only to the same CDP environment to which the backed-up service is associated.

For example:

```
cdp de restore-service --backup-id 2 --environment-crn crn:cdp:environme
nts:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:environment:c67b9089-
2d3b-4579-861d-c0df12a105b1
```

2. To obtain a list of backups, run:

```
cdp de list-backups
```

3. To describe a particular backup, run:

```
cdp de describe-backup --backup-id <backup-id>
```

For example:

```
$ cdp de describe-backup --backup-id 2 --profile priv
{
  "backup": {
    "id": 2,
    "serviceID": "cluster-cf6h74lq",
    "serviceName": "dex-priv-default-azure-env-1689008683873",
    "environmentName": "dex-priv-default-azure-env",
    "environmentCrn": "crn:cdp:environments:us-west-1:9d74eee4-1cad-45
d7-b645-7ccf9edbb73d:environment:c67b9089-2d3b-4579-861d-c0df12a105b1",
    "creator": "crn:altus:iam:us-west-1:9d74eee4-1cad-45d7-b645-7ccf
9edbb73d:user:0f9a97a7-23a7-43bd-bc71-ecdb2aa34ed5",
    "cloudPlatform": "AZURE",
    "status": "completed",
    "created": "2023-07-17T18:02:58.385455Z"
  }
}
```

4. In the case of an Airflow DAG failure, identify the impacted DAG on the Airflow UI and fix it.

For more information, see the DAG-related steps in *In-place upgrade with Airflow Operators and Libraries*.

Related Information

[Upgrading CDE](#)

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

Handling ineligible upgrades for Cloudera Data Engineering

In certain scenarios, your Cloudera Data Engineering (CDE) service is not eligible for an in-place upgrade to the latest CDE version. In such a scenario, follow a manual upgrade process where administrators create a new service and import all jobs and resources. Learn how to conduct a workaround for each scenario.

Older CDE versions

If the version of your CDE service is older than six months, it is outside of the upgrade window. Older CDE versions do not support in-place upgrade and you cannot use the process listed in *Handling upgrade failures for CDE*. To check whether your CDE version is eligible for the in-place upgrade option, see [Support lifecycle policy](#).

CDE provides the in-place upgrade option from the eligible versions to the latest version. If you are on a release that is ineligible for the in-place upgrade, the in-place upgrade option is disabled and you can perform a backup and restore. For more information, see *CDE upgrade version compatibility*.

Solution: A manual upgrade option is available. To upgrade, create a new CDE Service and transfer your jobs to the new service by following the instructions in *Backing up and restoring CDE jobs*.

Older Data Lake versions

If your CDE service uses an old Data Lake version, and therefore it is not compatible with the latest CDE version, your CDE service is not eligible to perform either an in-place upgrade or a manual upgrade. For information about CDE and its compatible Data Lake version, see *Compatibility for Cloudera Data Engineering and Runtime components*.

Solution: Upgrade your Data Lake to a compatible version associated with the CDE version. For more information about upgrading Data Lake, see *Upgrading a Data Lake*.

Airflow operators and libraries

If you use Airflow operators and libraries, before you can use the in-place upgrade, follow the procedure described in *In-place upgrade with Airflow Operators and Libraries*.

Solution: See *In-place upgrade with Airflow Operators and Libraries*.

Hotfix version of CDE

There is no upgrade available to CDE hotfix versions.

Related Information

[Handling upgrade failures for CDE](#)

[Backing up CDE jobs on remote storage](#)

[Compatibility for Cloudera Data Engineering and Runtime components](#)

[Upgrading a Data Lake](#)

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