Cloudera Data Engineering 1.5.0

Orchestrating workflows and pipelines with Apache Airflow in Cloudera Data Engineering

Date published: 2020-07-30
Date modified: 2023-01-24

CLOUDƎRA

Legal Notice

Contents

Automating data pipelines using Apache Airflow in Cloudera Data Engineering

Cloudera Data Engineering (CDE) enables you to automate a workflow or data pipeline using Apache Airflow Python DAG files. Each CDE virtual cluster includes an embedded instance of Apache Airflow. You can also use CDE with your own Airflow deployment. CDE on CDP Private Cloud currently supports only the CDE 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 may require you to remove any installed provider packages during troubleshooting.

About this task

The following instructions are for using the Airflow service provided with each CDE 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 CDE operator and define the tasks and dependencies.
   For example, here is a complete DAG file:

```python
from dateutil import parser
from datetime import datetime, timedelta
from datetime import timezone
from airflow import DAG
from cloudera.cdp.airflow.operators.cde_operator import CDEJobRunOperator


default_args = {
    'owner': 'psherman',
    'retry_delay': timedelta(seconds=5),
    'depends_on_past': False,
    'start_date': parser.isoparse('2021-05-25T07:33:37.393Z').replace(tz
info=timezone.utc)
}

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

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

prep_step2 = CDEJobRunOperator(
    task_id='data_prep',
    dag=example_dag,
    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 `CDEJobRunOperator` to specify a CDE job to run. This job must already exist in the virtual cluster specified by the connection_id. If no connection_id is specified, CDE looks for the job in the virtual cluster where the Airflow job runs.

```
from cloudera.cdp.airflow.operators.cde_operator import CDEJobRu
nOperator
...
ingest_step1 = CDEJobRunOperator(
    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 CDE job.

   a) In the Cloudera Data Platform (CDP) console, click the Data Engineering tile. The CDE Home page displays.

   b) In the CDE Home page, in Jobs, click Create New under Airflow or click Jobs in the left navigation menu and then click Create Job.

   c) Select the Airflow job type.

   If you are creating the job from the Home page, select the virtual cluster where you want to create the job.

   d) Name: Provide a name for the job.

   e) 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.

Related Information

Provider packages

Managing Cloudera Data Engineering job resources using the CLI

Using CDE with an external Apache Airflow deployment

The Cloudera provider for Apache Airflow, available at the Cloudera GitHub repository, provides two Airflow operators for running Cloudera Data Engineering (CDE) and Cloudera Data Warehouse (CDW) jobs. You can install the provider on your existing Apache Airflow deployment to integrate.

Before you begin

⚠ **Important:** CDE on CDP Private Cloud currently supports only the CDE job run operator.

- The Cloudera provider for Apache Airflow is for use with existing Airflow deployments. If you want to use the embedded Airflow service provided by CDE, see Automating data pipelines with CDE and CDW using Apache Airflow.
- The provider requires Python 3.6 or higher.
- The provider requires the Python cryptography package version 3.3.2 or higher to address CVE-2020-36242. If an older version is installed, the plugin automatically updates the cryptography library.

About this task

This component provides two Airflow operators to be integrated in your DAGs:

- CDEJobRunOperator, for running Cloudera Data Engineering jobs.
- CDWOperator, for accessing Cloudera Data Warehouse

Procedure

Install Cloudera Airflow provider

1. Select one of the following installation methods:

   Note: Depending on your python installation policies, you may need to run pip install --user . instead of pip install .

   - Direct install

     Run the following command on your Airflow server:

     ```
     pip install <PACKAGE_URL>
     ```

     Replace *<PACKAGE_URL>* with the link to the desired wheel package at https://github.com/cloudera/cloudera-airflow-plugins/releases

     For example, to install version 1.0.0:

     ```
     pip install https://github.com/cloudera/cloudera-airflow-plugins/release
     s/download/v1.0.0/cloudera_airflow_provider-1.0.0-py3-none-any.whl
     ```

   - Local install

     Run the following commands on your Airflow server to install the latest version:

     ```
     git clone --depth 1 https://github.com/cloudera/cloudera-airflow-plugins
     .git
     cd cloudera-airflow-plugins/cloudera_airflow_provider
     pip install .
     ```

     Run the following commands on your Airflow server to install a specific version:

     ```
     git clone --depth 1 --branch <VERSION> https://github.com/cloudera/clo
     udera-airflow-plugins.git
     cd cloudera-airflow-plugins/cloudera_airflow_provider
     pip install .
     ```

     Replace *<VERSION>* with the provider version that you want to install. For example, to install version 1.0.0:

     ```
     git clone --depth 1 --branch v1.0.0 https://github.com/cloudera/cloudera
     -airflow-plugins.git
     cd cloudera-airflow-plugins/cloudera_airflow_provider
     pip install .
     ```

Create a connection using the Airflow UI

Before you can run a CDE job from your Airflow deployment, you must configure a connection using the Airflow UI.

2. In the Cloudera Data Platform (CDP) console, click the Data Engineering tile. The CDE Home page displays.
3. Click Administration in the left navigation menu. The Administration page displays.
4. In the Virtual Clusters column, click the Cluster Details icon.
5. Click JOBS API URL to copy the URL.
6. Go to your Airflow web console (where you installed the Cloudera provider).
7. Go to Admin Connection .
8. Click + Add a new record.

9. Fill in connection details:

Conn Id

> Create a unique connection identifier.

Conn Type

> The type of the connection. From the drop-down, select
>
> - HTTP (if you are using Apache Airflow version 1)
> - HTTP or Cloudera Data engineering (if you are using Apache Airflow version 2)

Host/Virtual API Endpoint

> URL of the host where you want the job to run. Paste here the JOBS API URL you copied in a previous step.

Login/CDP Access Key

> Provide the CDP access key of the account for running jobs on the CDE VC.

Password/CDP Private Key

> Provide the CDP private key of the account for running jobs on the CDE VC.

10. Click Save.

11. In the CDE Home page, click Jobs in the left navigation menu, and then click Create a Job.

12. Fill in the Job Details:

Job Type

> Select the option matching your use case.

Name

> Specify a name for the job.

DAG File

> Provide a DAG file.
>
> Use the `CDEJobRunOperator` to specify a CDE job to run. The job definition in the DAG file must contain:
>
> **connection_id**
>
> The Conn Id you specified on the Airflow UI when creating the connection.
>
> **task_id**
>
> The ID that identifies the job within the DAG.
>
> **dag**
>
> The variable containing the dag object
>
> **job_name**
>
> The name of the CDE job to run. This job must exist in the CDE virtual cluster you are connecting to.

For example:

```
from cloudera.cdp.airflow.operators.cde_operator import CDEJobRunOperator
...
t1 = CDEJobRunOperator(
    connection_id='cde-vc01-dev',
    task_id='ingest',
    dag=example_dag,
    job_name='etl-ingest-job'
)
```

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