Cloudera Data Engineering 1.25.0

Data Engineering deployment architecture

Date published: 2020-07-30
Date modified: 2025-11-20

CLOUDERA

https://docs.cloudera.com/

Legal Notice

Contents

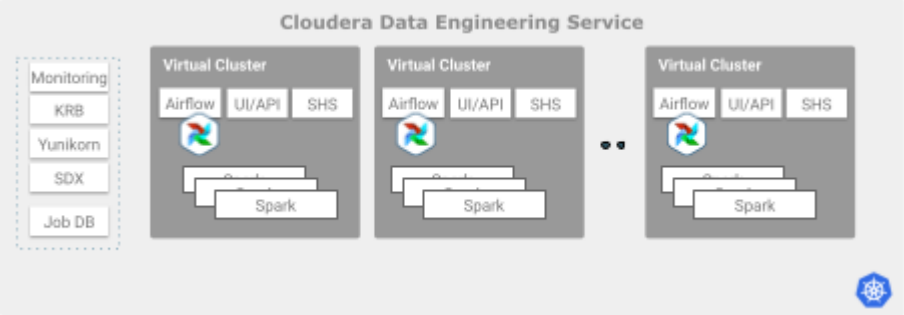
Recommendations for scaling Cloudera Data Engineering deployments

Your business might experience a sudden increase or drop in demand due to which your Cloudera Data Engineering deployment needs to autoscale. You can scale your Cloudera Data Engineering deployment by either adding new instances of a Cloudera Data Engineering service or Virtual Cluster, or by adding additional resources to the existing ones.

You can scale your Cloudera Data Engineering deployment horizontally - new instances of Cloudera Data Engineering service or virtual cluster are provisioned.

Note: Vertically scaling the Cloudera Data Engineering service by changing the instance or machine type is currently not supported. To modify the instance or machine type, you must redeploy the Cloudera Data Engineering service.

The key components of a typical Cloudera Data Engineering service deployment are:



Virtual Clusters provide an isolated autoscaling compute capacity to run Spark and Airflow jobs. You can use Virtual Clusters to isolate individual teams or lines of business by using user-based Access Control Lists (ACLs).

Guidelines for scaling Virtual Clusters

- Each Virtual Cluster requires infrastructure capacity to run various services such as Airflow, API server, and Spark-History-Server (SHS).

  Recommendation: Do not scale horizontally beyond 50 Virtual Clusters within the same Cloudera Data Engineering service.
- Virtual Clusters can actively run hundreds of parallel jobs. In certain scenarios, it might be required to simultaneously submit multiple jobs as per the schedule or due to a burst in demand. In these scenarios, the API server enforces guardrails and limits the number of simultaneous Spark job submissions to 60. Once the jobs move from submission to running state, you can submit more jobs.

  Note: The Airflow jobs are not impacted by the submission limit of 60 simultaneous jobs.

  Recommendation: Distribute simultaneous submission of jobs over time or horizontally scale across multiple Virtual Clusters.

Limit on resource files in job runs

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

Job submission rate guardrails

When jobs are submitted to the Cloudera Data Engineering API server of a particular Virtual Cluster, it takes time and resources, known as preparing and starting states, before they begin running. This process is called the job submission intake process. To ensure proper resourcing and handling of incoming jobs, guardrails have been set up. By default, the guardrail, or limit is set to 60 simultaneous job submissions. Simultaneous incoming Spark job submissions that exceed 60 return a 429 error message to the client. The example error message is: Failed to submit. Too many requests.

Recommendation:

- Incorporate Cloudera Data Engineering Clients error handling for this error. The CDE CLI exit code is 77. The CDE CLI receives an HTTP 429 response code for the request to the Runtime API Server's REST API. For more information on exit codes, see Cloudera Data Engineering CLI exit codes.
- If necessary, you can decrease the default job submission limit value of 60 simultaneous jobs during the Virtual Cluster creation by adjusting the limiting.simultaneousJobSubmissions configuration.
- Cloudera discourages you from increasing the job submission limit value beyond the default 60 simultaneous jobs.

Related Information

Creating virtual clusters

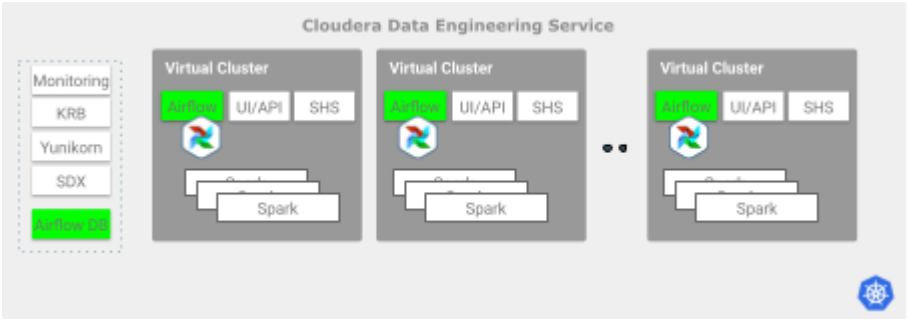
Cloudera Data Engineering CLI exit codes

Apache Airflow scaling and tuning considerations

There are certain limitations related to the deployment architecture, and guidelines for scaling and tuning of the deployment, that you must consider while creating or running Airflow jobs (DAGs).

Cloudera Data Engineering deployment architecture

If you use Airflow to schedule and/or develop multi-step pipelines, you must consider the following deployment limitations to scale your environment.



These guidelines describe when to scale horizontally, given total loaded DAGs and/or number of parallel (concurrent) tasks. These apply to all cloud providers.

Cloudera Data Engineering service guidelines

The number of Airflow jobs that can run at the same time is limited by the number of parallel tasks that are triggered by the associated DAGs. The number of tasks can reach up to 250-300 when running in parallel per Cloudera Data Engineering service.

Recommendation: Create more Cloudera Data Engineering services to increase Airflow tasks concurrency beyond the limits noted above.

Virtual Cluster guidelines

- Each Virtual Cluster has a maximum parallel task execution limit of 250. This applies to both Amazon Web Services (AWS) and Azure.

  Note: It is recommended that you do not exceed 300 parallel tasks across all Virtual Clusters.

- For AWS: The recommended maximum number of Airflow jobs loaded per virtual cluster is 1500. As the number of jobs increases, the job page loading time experiences latency along with Airflow job creation time.
- For Azure: The recommended maximum number of Airflow jobs loaded per virtual cluster is 300 to 500. Higher capacity can be achieved but will require manual configuration changes.

  Recommendation: Create more Virtual Clusters to load additional Airflow DAGs.
- The suggested number of Airflow jobs created in parallel is three or less. For jobs that are created in parallel, specify a DAG and/or a task_group for each Airflow operator.

  Recommendation: Distribute the creation of Airflow jobs so that you do not create more than three jobs at the same time.

Guidelines for Virtual Cluster upkeep

There are upkeep guidelines for Cloudera Data Engineering Spark History Server (SHS) that you'll need to consider.

Lifecycle configuration of Cloudera Data Engineering Spark event logs

The number of Spark event logs (spark.eventLog.enabled), that are produced by Spark jobs that run via Cloudera Data Engineering Virtual Cluster, grows indefinitely with each new Cloudera Data Engineering Spark run. These event logs are not automatically deleted and are stored on the object store under <CDP env storage location>/dex/<Service ID>/<VC ID>/eventlog/.

Some examples of the event log location can look like the following:

- For Amazon Web Services (AWS): s3a://dex-storage-bucket/datalake/logs/dex/cluster-2xvl4pfp/rdw8q2sh/eventlog/
- For Azure: abfs://logs@dexstorageaccount.dfs.core.windows.net/dex/cluster-4p54mk8j/22bnm99g/eventlog/

Because the number of event logs continuously increases, the time from when the Cloudera Data Engineering job finishes and when the Spark UI is available for this run on the Virtual Cluster UI may increase. The delay is most apparent in Virtual Clusters with 6,000 or more completed Spark job runs.

To avoid delays in event log availability after Cloudera Data Engineering job runs, you can configure an object store lifecycle policy so that event logs are deleted automatically on the object store. For more information about an Amazon S3 lifecycle policy, see Setting lifecycle configuration on a bucket linked below. For more information about Azure lifecycle management policies, see Configure a lifecycle management policy linked below.

Note: The lifecycle policy rule prefix does not include a bucket or storage account name.

Spark History configuration to reduce I/O consumption

Cloudera recommends cleaning up the old Spark History Server (SHS) logs regularly, to reduce the Input/Output (I/O) consumption significantly. If the old SHS logs are not cleaned up, the Spark History Server generates a high volume of read operations, leading to excessive I/O usage, regardless of the number of running jobs.

To reduce the I/O consumption, follow this procedure:

1. Provide the Virtual Cluster (VC) ID in the command and run the command to open the spark-defaults configmap for editing.

```
kubectl edit configmap dex-app-[***VC-ID***]-spark-defaults -n dex-app-x
xxxxxxx
```

2. At the end of spark-defaults.conf section, add: spark.history.fs.update.interval: 300s

   You can adjust the time interval as desired. The default value is 10s.
3. Restart the dex-app-xxxxxxxx-shs-xxxx pod in the dex-app-xxxxxxxx namespace.

```
kubectl delete pod dex-app-xxxxxxxx-shs-xxxx -n dex-app-xxxxxxxx
```

Related Information

Recommendations for scaling Cloudera Data Engineering deployments
Setting lifecycle configuration on a bucket
Configure a lifecycle management policy

Guidelines for database upkeep

There are database upkeep guidelines for Cloudera Data Engineering that you'll need to consider.

Note: Despite the end of service for mySQL v8.0.28 in June 2023, you can continue using Cloudera Data Engineering with mySQL v8.0.28. Although mySQL will continue to work with your databases after end of service, troubleshooting support will not be available. Therefore, it is recommended that you upgrade to a supported version of mySQL.

Upgrade mySQL for AWS

Upgrade mySQL for AWS to ensure you are using a supported version to work with Cloudera Data Engineering.

Note: If you are using an in-place upgrade, your mySQL version will also be updated.

1. In the AWS console, navigate to RDS > cluster-id > Modify.
2. Select the version that you want to upgrade from the Database engine version drop-down list.
3. Click Continue.
4. Click Apply Immediately to apply changes.

Increasing database size on Azure

Azure databases can fill up with logs and entries with every job that runs in Cloudera Data Engineering Virtual Clusters. When the database fills up, increase the size of your database.

1. Ensure that there are no jobs running.
2. Go to the Azure Portal.
3. Locate the Azure SQL database named <cluster id>, for example, cluster-2cmmds8q.
4. Navigate to the Pricing Tier section.
5. Move the slider to the desired value.

   Note: The slider cannot be reduced.
6. Click OK and wait for the scaling to complete.
7. Resume running your jobs.

Increasing database size on Amazon Web Services

Amazon Web Services (AWS) databases can fill up with logs and entries with every job that runs in Cloudera Data Engineering Virtual Clusters. When the database fills up, increase the size of your database.

1. Go to the AWS Console.
2. Navigate to the RDS Service page.
3. Click Databases and use the filter to find your cluster id, for instance, cluster-w8d65nxp.
4. Select the target database.
5. In the Database homepage in the top right-hand corner, click Modify.
6. Scroll down to the database size settings and set the Allocated Storage Property to the desired value.
7. Click Continue.
8. Set the desired timeframe for maintenance.
9. Click Continue and wait for the database status to become available.
10. Resume running your jobs.