Cloudera Data Hub

Cluster Templates

Date published: 2019-12-17
Date modified: 2025-08-18

https://docs.cloudera.com/

Legal Notice

Contents

Cluster templates overview

A cluster template is a declarative definition of a cluster that defines cluster topology (cluster host groups and all cluster services and their components running on them).

A Cloudera template is a reusable cluster template in JSON format that can be used for creating multiple Cloudera Data Hub clusters with identical Cloudera Runtime settings. It primarily defines the list of host groups and how components of various Cloudera Runtime services are distributed on these host groups. A cluster template allows you to specify stack, component layout, and configurations to materialize a cluster instance via Cloudera Manager REST API, without having to use the Cloudera Manager install wizard. After you provide the cluster template to Cloudera Data Hub, the host groups in the JSON are mapped to a set of instances when starting the cluster, and the specified services and components are installed on the corresponding nodes.

Note:

A cluster template is not synonymous with a cluster definition, which primarily defines cloud provider settings. Each cluster definition must reference a specific cluster template.

Cloudera Data Hub includes a few default cluster templates and allows you to upload your own cluster templates.

Prior to creating your own cluster templates, we recommend that you review the default cluster templates to check if they meet your requirements. These default cluster templates can be accessed from Shared Resources Templates . To view details of a cluster template, click on its name. For each cluster template, you can access a graphical representation ("list view") and a raw JSON file ("raw view") of all cluster host groups and their components. If you want create a custom template to modify a service's configuration, for example to tune Yarn or Hive, refer to the Cloudera Manager configuration properties for the desired service. You can search these properties by their API Name, which is how they appear in a Cloudera Data Hub template.

If you require a custom cluster template, consider modifying a default template and registering it as a new cluster template.

Default cluster configurations

Cloudera Data Hub includes a set of prescriptive cluster configurations. Each of these default cluster configurations include a cloud-provider specific cluster definition, which primarily defines cloud provider settings. The cluster definition references a cluster template, which defines a number of Cloudera Runtime or Cloudera DataFlow components used for common data analytics and data engineering use cases.

Refer to the topic for each default cluster configuration to view the included services and compatible Runtime versions. These topics include links to documentation that will help you to understand the included components and use the workload cluster.

Many of the cluster components are included in the Cloudera Runtime software distribution. The Streams Messaging, Cloudera Flow Management, and Cloudera Streaming Analytics cluster configurations are part of Cloudera DataFlow for Data Hub and have distinct planning considerations and how-to information. See the Cloudera DataFlow for Data Hub documentation for more details.

You can access the default cluster definitions by clicking Environments, then selecting an environment and clicking the Cluster Definitions tab.

You can access the default cluster templates from Shared ResourcesCluster Templates.

To view details of a cluster definition or cluster template, click on its name. For each cluster definition, you can access a raw JSON file. For each cluster template, you can access a graphical representation ("list view") and a raw JSON file ("raw view") of all cluster host groups and their components.

Related Information

Cloudera DataFlow for Data Hub

Cloudera Runtime

Data Engineering clusters

Learn about the default Data Engineering clusters, including cluster definition and template names, included services, and compatible Cloudera Runtime version.

Data Engineering provides a complete data processing solution, powered by Apache Spark and Apache Hive. Spark and Hive enable fast, scalable, fault-tolerant data engineering and analytics over petabytes of data.

Data Engineering cluster definition

This Data Engineering template includes a standalone deployment of Spark and Hive, as well as Apache Oozie for job scheduling and orchestration, Apache Livy for remote job submission, and Hue and Apache Zeppelin for job authoring and interactive analysis.

Cluster definition names

- Data Engineering for AWS
- Data Engineering for Azure
- Data Engineering for Google Cloud
- Data Engineering HA - Spark3 for AWS
- Data Engineering HA - Spark3 for Azure
- Data Engineering HA - Spark3 for Google Cloud
- Data Engineering Spark3 for AWS
- Data Engineering Spark3 for Azure
- Data Engineering Spark3 for Google Cloud
- Data Engineering - Spark3 (ARM) for AWS
- Data Engineering HA - Spark3 (ARM) for AWS

Cluster template name

- Data Engineering: Apache Spark3, Apache Hive, Apache Oozie

  Note: This cluster template was formerly named "Data Engineering: Apache Spark, Apache Hive, Apache Oozie."

  The "Data Engineering: Apache Spark3" cluster template is deleted. Therefore, the "Data Engineering: Apache Spark3, Apache Hive, Apache Oozie" cluster template can be used instead.
- Data Engineering: HA: Apache Spark3, Apache Hive, Apache Oozie

  Note: This cluster template was formerly named "Data Engineering: HA: Apache Spark, Apache Hive, Apache Oozie."

Included services

- HDFS
- Hive
- Hue
- Livy
- Spark 3
- Yarn
- Zeppelin is no longer supported starting with Cloudera Runtime version 7.3.1.
- ZooKeeper
- Oozie is supported for Spark 3 as of Cloudera Runtime version 7.2.18.
- Hive Warehouse Connector is supported as of Cloudera Runtime version 7.2.16.

Compatible Cloudera Runtime version

- 7.2.15
- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

Topology of the Data Engineering cluster

Topology is a set of host groups that are defined in the cluster template and cluster definition used by Data Engineering. Data Engineering uses the following topology:

For AWS

Table 1: Data Engineering in AWS

Host group	Description	Node configuration
Master Node count: 1	The master host group runs the components for managing the cluster resources including Cloudera Manager, Name Node, Resource Manager, as well as other master components such HiveServer2, HMS, Hue etc.	For clusters created with Cloudera Runtime versions lower than 7.2.14: • m5.4xlarge; gp2 - 100 GB For clusters created with Cloudera Runtime versions 7.2.14 or higher versions: • m5.4xlarge; gp2 - 100 GB

Worker
Node count: 3
	The worker host group runs the components that are used for executing processing tasks (such as NodeManager) and handling storing data in HDFS such as DataNode.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• m5.2xlarge; gp2 - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • r5d.2xlarge - (gp2/EBS volumes)
		• DE HA:
		  • r5d.4xlarge - (gp2/EBS volumes)

Compute
Node count: 0+
	The compute host group can optionally be used for running data processing tasks (such as NodeManager). By default the number of compute nodes is set to 1 for proper configurations of YARN containers. This node group can be scaled down to 0 when there are no compute needs. Additionally, if load-based auto-scaling is enabled with minimum count set to 0, the compute nodegroup will be resized to 0 automatically.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• m5.2xlarge; gp2 - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • r5d.2xlarge - (ephemeral volumes)
		• DE HA:
		  • r5d.4xlarge - (ephemeral volumes)
		  **Note:** Compute nodes run YARN and require storage only for temporary data - this requirement is fulfilled by instance storage, so making the attached volumes count to 0 by default is more cost-efficient.

Host group	Description	Node configuration
Gateway Node count: 0+	The gateway host group can optionally be used for connecting to the cluster endpoints like Oozie, Beeline etc. This nodegroup does not run any critical services. This nodegroup resides in the same subnet as the rest of the nodegroups. If additional software binaries are required they could be installed using recipes.	m5.2xlarge; gp2 - 100 GB

For Azure

Table 2: Data Engineering in Azure

Host group	Description	Node configuration
Master Node count: 1	The master host group runs the components for managing the cluster resources including Cloudera Manager, Name Node, Resource Manager, as well as other master components such HiveServer2, HMS, Hue etc.	For clusters created with Cloudera Runtime versions lower than 7.2.14: • Standard_D16_v3; StandardSSD_LRS - 100 GB For clusters created with Cloudera Runtime versions 7.2.14 or higher versions: • Standard_D16_v3

Worker
Node count: 3
	The worker host group runs the components that are used for executing processing tasks (such as NodeManager) and handling storing data in HDFS such as DataNode.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• Standard_D8_v3; StandardSSD_LRS - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • Standard_D5_v2
		• DE HA:
		  • Standard_D5_v2

Compute
Node count: 0+
	The compute host group can optionally be used for running data processing tasks (such as NodeManager). By default the number of compute nodes is set to 1 for proper configurations of YARN containers. This node group can be scaled down to 0 when there are no compute needs. Additionally, if load-based auto-scaling is enabled with minimum count set to 0, the compute nodegroup will be resized to 0 automatically.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• Standard_D8_v3; StandardSSD_LRS - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • Standard_D5_v2
		• DE HA:
		  • Standard_D5_v2
		For Azure, the attached volume count for the compute host group is changed to 0. Only ephemeral/local volumes are used by default.
		Note: Compute nodes run YARN and require storage only for temporary data - this requirement is fulfilled by instance storage, so making the attached volumes count to 0 by default is more cost-efficient.

Host group	Description	Node configuration
Gateway Node count: 0+	The gateway host group can optionally be used for connecting to the cluster endpoints like Oozie, Beeline etc. This nodegroup does not run any critical services. This nodegroup resides in the same subnet as the rest of the nodegroups. If additional software binaries are required they could be installed using recipes.	Standard_D8_v3; StandardSSD_LRS - 100 GB

For GCP

Table 3: Data Engineering in GCP

Host group	Description	Node configuration
Master Node count: 1	The master host group runs the components for managing the cluster resources including Cloudera Manager, Name Node, Resource Manager, as well as other master components such HiveServer2, HMS, Hue etc.	For clusters created with Cloudera Runtime versions lower than 7.2.14: • e2-standard-16; pd-ssd - 100 GB For clusters created with Cloudera Runtime versions 7.2.14 or higher versions: • e2-standard-16; pd-ssd - 100 GB

Worker
Node count: 3
	The worker host group runs the components that are used for executing processing tasks (such as NodeManager) and handling storing data in HDFS such as DataNode.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• e2-standard-8; pd-ssd - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • e2-standard-8; pd-ssd - 100 GB
		• DE HA:
		  • e2-standard-8; pd-ssd - 100 GB

Compute
Node count: 0+
	The compute host group can optionally be used for running data processing tasks (such as NodeManager). By default the number of compute nodes is set to 1 for proper configurations of YARN containers. This node group can be scaled down to 0 when there are no compute needs. Additionally, if load-based auto-scaling is enabled with minimum count set to 0, the compute nodegroup will be resized to 0 automatically.	For clusters created with Cloudera Runtime versions lower than 7.2.14:
		• e2-standard-8; pd-ssd - 100 GB
		For clusters created with Cloudera Runtime versions 7.2.14 or higher versions:
		• DE and DE Spark3:
		  • e2-standard-8; pd-ssd - 100 GB
		• DE HA:
		  • e2-standard-8; pd-ssd - 100 GB
		Note: Compute nodes run YARN and require storage only for temporary data - this requirement is fulfilled by instance storage, so making the attached volumes count to 0 by default is more cost-efficient.

Gateway Node count: 0+	The gateway host group can optionally be used for connecting to the cluster endpoints like Oozie, Beeline etc. This nodegroup does not run any critical services. This nodegroup resides in the same subnet as the rest of the nodegroups. If additional software binaries are required they could be installed using recipes.	e2-standard-8; pd-ssd - 100 GB

Configurations

The following table summarizes the service configurations based on the host groups:

Table 4:

Host group	Service configuration
Master	Cloudera Manager, HDFS, Hive (on Tez), HMS, Yarn RM, Oozie, Hue, DAS, Zookeeper, Livy, Zeppelin and Sqoop
Gateway	Configurations for the services on the master node
Worker	Data Node and YARN NodeManager
Compute	YARN NodeManager

Note the following:

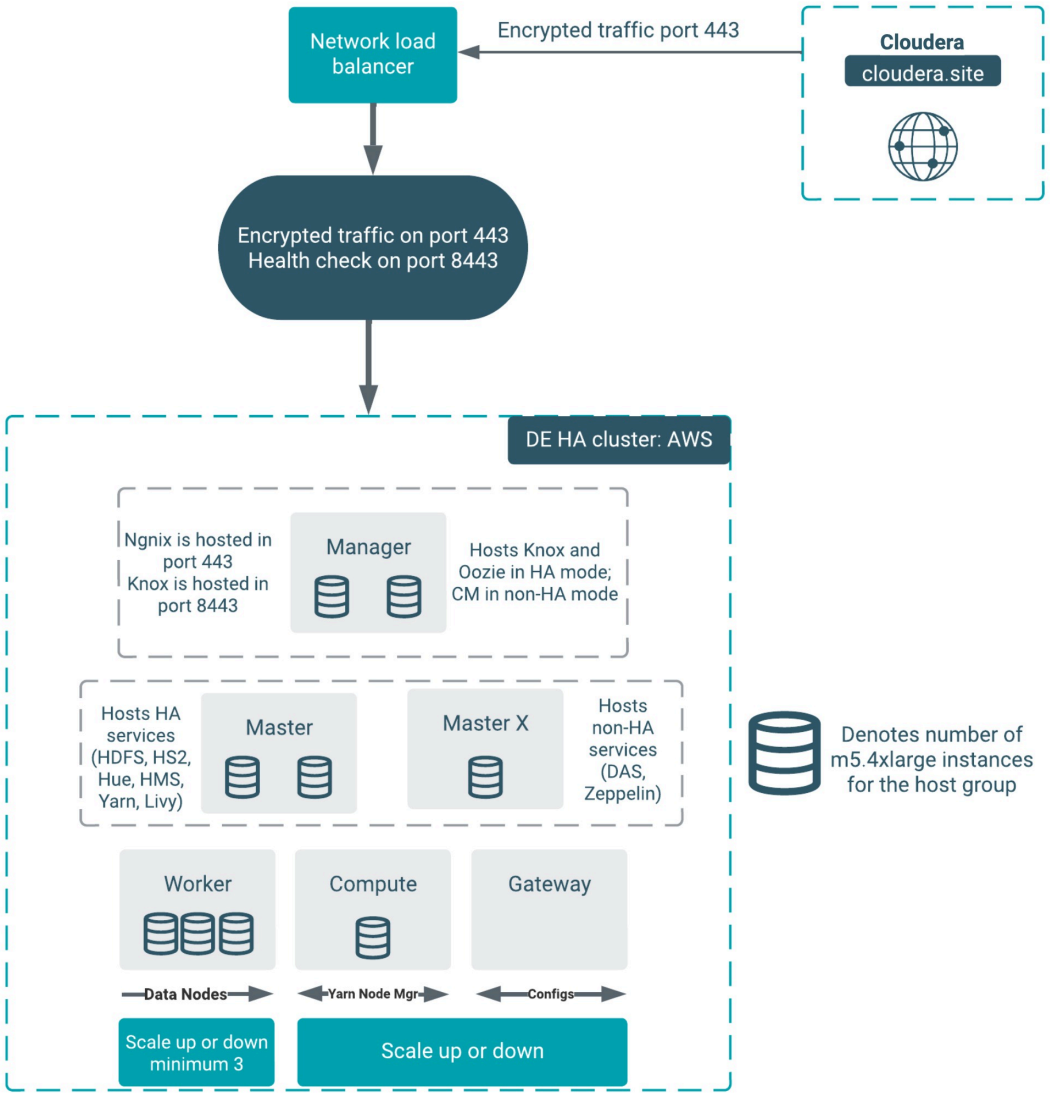
- There is a Hive Metastore Service (HMS) running in the cluster that talks to the same database instance as the Data Lake in the environment.
- If you use CLI to create the cluster, you can optionally pass an argument to create an external database for the cluster use such as Cloudera Manager, Oozie, Hue, and DAS. This database is by default embedded in the master node external volume. If you specify the external database to be of type HA or NON_HA, the database will be provisioned in the cloud provider. For all these types of databases the lifecycle is still associated with the cluster, so upon deletion of the cluster, the database will also be deleted.
- The HDFS in this cluster is for storing the intermediary processing data. For resiliency, store the data in the cloud object stores.
- For high availability requirements choose the Data Engineering High Availability cluster shape.

For AWS

Architecture of the Data Engineering HA for AWS cluster

The Data Engineering HA for AWS cluster shape provides failure resilience for several of the Data Engineering HA services, including Knox, Oozie, HDFS, HS2, Hue, Livy, YARN, and HMS.

Services that do not yet run in HA mode include Cloudera Manager, DAS, and Zeppelin.



The architecture outlined in the diagram above handles the failure of one node in all of the host groups except for the “masterx” group. See the table below for additional details about the component interactions in failure mode:

Component	Failure	User experience
Knox	One of the Knox services is down	External users will still be able to access all of the UIs, APIs, and JDBC.
Cloudera Manager	The first node in manager host group is down	The cluster operations (such as repair, scaling, and upgrade) will not work.
Cloudera Manager	The second node in the manager host group is down	No impact.
HMS	One of the HMS services is down	No impact.
Hue	One of the Hue services is down in master host group	No impact.

HS2	One of the HS2 services is down in the master host group	External users will still be able to access the Hive service via JDBC. But if Hue was accessing that particular service it will not failover to the other host. The quick fix for Hue is to restart Hue to be able to use Hive functionality.
YARN	One of the YARN services is down	No impact.
HDFS	One of the HDFS services is down	No impact.
Nginx	Nginx in one of the manager hosts is down	Fifty percent of the UI, API, and JDBC calls will be affected. If the entire manager node is down, there is no impact. This is caused by the process of forwarding and health checking that is done by the network load-balancer.
Oozie	One of the Oozie servers is down in the manager host group.	No impact for AWS as of Cloudera Runtime version 7.2.11. If you create a custom template for DE HA, follow these two rules: 1. Oozie must be in single hostgroup. 2. Oozie and Hue must not be in the same hostgroup.

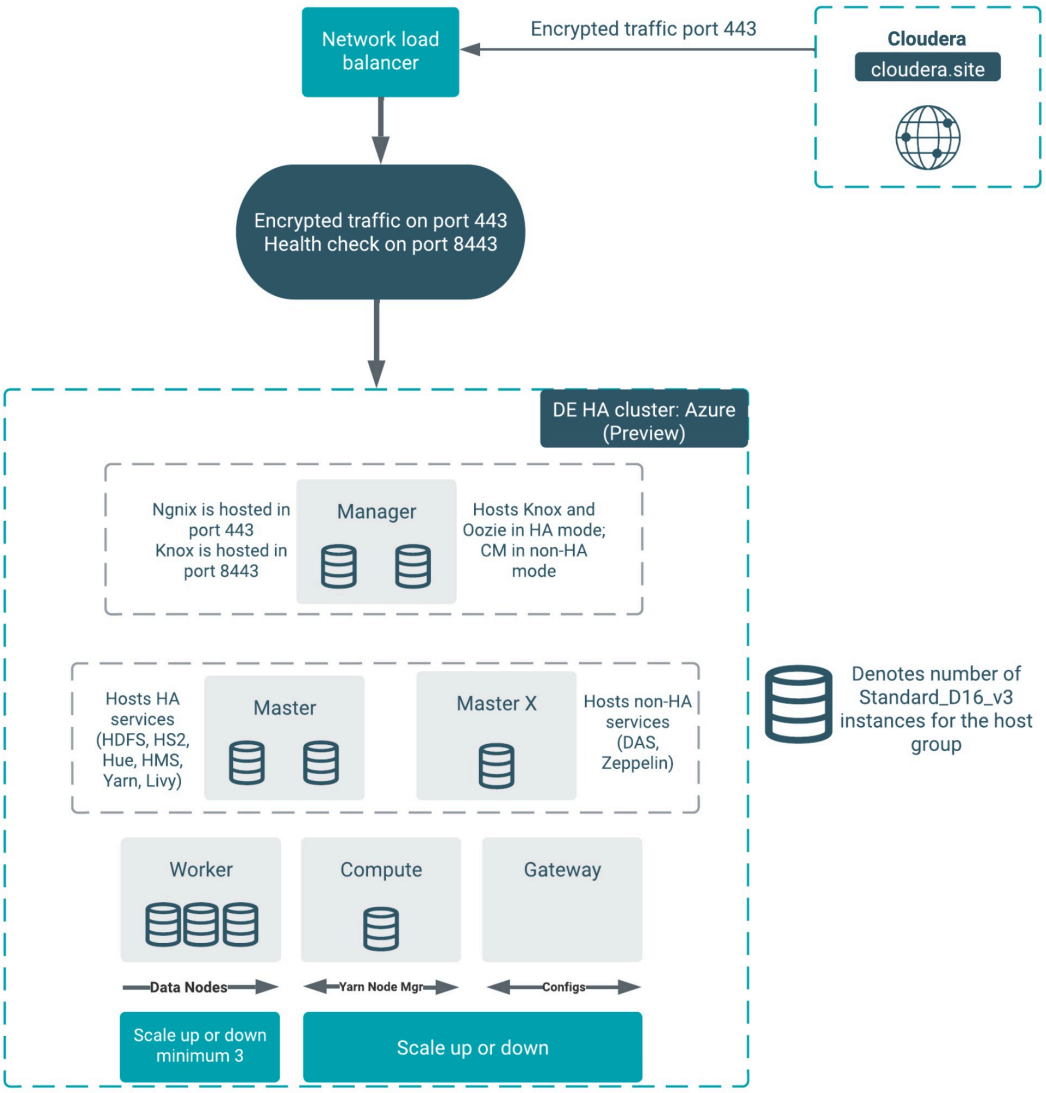
Important: If you are creating a DE HA cluster through the CDP CLI using the crea te-aws-cluster command, note that there is a CLI parameter to provision the network load-balancer in HA cluster shapes. Make sure to use the [--enable-load-balancer | --no-enable-load-balancer] parameter when provisioning a DE HA cluster via the CLI. For more information see the CDP CLI reference.

For Azure

Architecture of the Data Engineering HA for Azure cluster

The Data Engineering HA for Azure cluster shape provides failure resilience for several of the Data Engineering HA services, including Knox, Oozie, HDFS, HS2, Hue, Livy, YARN, and HMS.

Services that do not yet run in HA mode include Cloudera Manager, DAS, and Zeppelin.



Component	Failure	User experience
Knox	One of the Knox services is down	External users will still be able to access all of the UIs, APIs, and JDBC.
Cloudera Manager	The first node in manager host group is down	The cluster operations (such as repair, scaling, and upgrade) will not work.
Cloudera Manager	The second node in the manager host group is down	No impact.
HMS	One of the HMS services is down	No impact.
Hue	One of the Hue services is down in master host group	No impact.
HS2	One of the HS2 services is down in the master host group	External users will still be able to access the Hive service via JDBC. But if Hue was accessing that particular service it will not failover to the other host. The quick fix for Hue is to restart Hue to be able to use Hive functionality.

YARN	One of the YARN services is down	No impact.
HDFS	One of the HDFS services is down	No impact.
Nginx	Nginx in one of the manager hosts is down	Fifty percent of the UI, API, and JDBC calls will be affected. If the entire manager node is down, there is no impact. This is caused by the process of forwarding and health checking that is done by the network load-balancer.
Oozie	One of the Oozie servers is down in the manager host group.	No impact for Azure as of Cloudera Runtime version 7.2.11. If you create a custom template for DE HA, follow these two rules: 1. Oozie must be in single hostgroup. 2. Oozie and Hue must not be in the same hostgroup.

Important: If you are creating a DE HA cluster through the CDP CLI using the crea te-azure-cluster command, note that there is a CLI parameter to provision the network load-balancer in HA cluster shapes. Make sure to use the [--enable-load-balancer | --no-enable-load-balancer] parameter when provisioning a DE HA cluster via the CLI. For more information see the CDP CLI reference.

For GCP (preview)

Note: HA for Oozie is not yet available in the GCP template.

Custom templates

Any custom DE HA template that you create must be forked from the default templates of the corresponding version. You must create a custom cluster definition for this with the JSON parameter “enableLoadBalancers”: true , using the create-aws/azure/gcp-cluster CLI command parameter --request-template. Support for pre-existing custom cluster definitions will be added in a future release. As with the template, the custom cluster definition must be forked from the default cluster definition. You are allowed to modify the instance types and disks in the custom cluster definition. You must not change the placement of the services like Cloudera Manager, Oozie, and Hue. Currently the custom template is fully supported only via CLI.

The simplest way to change the DE HA definition is to create a custom cluster definition. In the Create Data Hub UI when you click Advanced Options, the default definition is not used fully, which will cause issues in the HA setup.

Related Information

HDFS
Hive
Hue
Livy
Oozie
Spark
YARN
Zeppelin
Zookeeper

Data Mart clusters

Learn about the default Data Mart and Real Time Data Mart clusters, including cluster definition and template names, included services, and compatible Runtime version.

Data Mart is an MPP SQL database powered by Apache Impala designed to support custom Data Mart applications at big data scale. Impala easily scales to petabytes of data, processes tables with trillions of rows, and allows users to store, browse, query, and explore their data in an interactive way.

Data Mart clusters

The Data Mart template provides a ready to use, fully capable, standalone deployment of Impala. Upon deployment, it can be used as a standalone Data Mart to which users point their BI dashboards using JDBC/ODBC end points. Users can also choose to author SQL queries in Cloudera's web-based SQL query editor, Hue, and run them with Impala providing a delightful end-user focused and interactive SQL/BI experience.

Cluster definition names

- Data Mart for AWS
- Data Mart (ARM) for AWS
- Data Mart for Azure
- Data Mart for Google Cloud

Cluster template name

Cloudera - Data Mart: Apache Impala, Hue

Included services

- HDFS
- Hue
- Impala

Compatible Cloudera Runtime versions

- 7.2.15
- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

Real Time Data Mart clusters

The Real-Time Data Mart template provides a ready-to-use, fully capable, standalone deployment of Impala and Kudu. You can use a Real Time Data Mart cluster as a standalone Data Mart which allows high throughput streaming ingest, supporting updates and deletes as well as inserts. You can immediately query data through BI dashboards using JDBC/ODBC end points. You can choose to author SQL queries in Cloudera's web-based SQL query editor, Hue. Executing queries with Impala, you will enjoy an end-user focused and interactive SQL/BI experience. This template is commonly used for Operational Reporting, Time Series, and other real time analytics use cases.

Cluster definition names

- Real-time Data Mart for AWS
- Real-time Data Mart for Azure
- Real-time Data Mart for Google Cloud

Cluster template name

Cloudera - Real-time Data Mart: Apache Impala, Hue, Apache Kudu, Apache Spark

Included services

- HDFS
- Hue
- Impala
- Kudu
- Spark 2
- Yarn

Compatible Cloudera Runtime versions

- 7.2.15
- 7.2.16
- 7.2.17

Cluster definition names

- Real-time Data Mart - Spark3 for AWS
- Real-time Data Mart - Spark3 (ARM) for AWS
- Real-time Data Mart - Spark3 for Azure
- Real-time Data Mart - Spark3 for Google Cloud

Cluster template name

Real-time Data Mart: Apache Impala, Hue, Apache Kudu, Apache Spark3

Included services

- HDFS
- Hue
- Impala
- Kudu
- Spark 3
- Yarn

Compatible Cloudera Runtime versions

- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

High availability

Cloudera recommends that you use high availability (HA), and track any services that are not capable of restarting or performing failover in some way.

Impala HA

The Impala nodes offer high availability. The following Impala services are not HA.

- Catalog service
- Statestore service

Kudu HA

Both Kudu Masters and TabletServers offer high availability.

Related Information

HDFS
Hue
Impala
Kudu

Spark
YARN

Operational Database clusters

The Operational Database (OpDB) template is removed from the Cloudera Data Hub. You can access the Cloudera Operational Database instead as a superior product.

The Cloudera Operational Database is a NoSQL database powered by Apache HBase designed to support custom OLTP applications that want to leverage the power of BigData. Apache HBase is a NoSQL, scale-out database that can easily scale to petabytes and stores tables with millions of columns and billions of rows.

Cloudera Operational Database also contains Apache Phoenix which provides a way to use HBase through an SQL interface.

Cloudera recommends you to use the Cloudera Operational Database to create Operational Database clusters.

Related Information

Getting started with Operational Database
Creating an Operational Database cluster
Cloudera Operational Database
Creating a database using Cloudera Operational Database
HDFS
HBase
Knox
Zookeeper
Phoenix

Streams Messaging clusters

Learn about the default Streams Messaging clusters, including cluster definition and template names, included services, and compatible Cloudera Runtime version.

Streams Messaging provides the following features:

- Advanced messaging and real-time processing on streaming data using Apache Kafka
- Centralized schema management using Schema Registry
- Management and monitoring capabilities powered by Streams Messaging Manager
- Cross-cluster Kafka topic replication using Streams Replication Manger
- Kafka partition rebalancing with Cruise Control

This template sets up a fault-tolerant standalone deployment of Apache Kafka and supporting Cloudera components (Schema Registry, Streams Messaging Manager, Streams Replication Manager and Cruise Control), which can be used for production Kafka workloads in the cloud or as a disaster recovery instance for on-premises. Kafka clusters.

Note:

Streams Messaging clusters have distinct planning considerations and how-to information. See the Cloudera DataFlow for Data Hub documentation for information about:

- Planning your Streams Messaging cluster deployment
- Creating your first Streams Messaging cluster
- Connecting Kafka clients to Cloudera on cloud clusters

Cluster definition names

- Streams Messaging Heavy Duty for AWS
- Streams Messaging Light Duty for AWS

- Streams Messaging HA for AWS
- Streams Messaging Heavy Duty (ARM) for AWS
- Streams Messaging Light Duty (ARM) for AWS
- Streams Messaging HA (ARM) for AWS
- Streams Messaging Heavy Duty for Azure
- Streams Messaging Light Duty for Azure
- Streams Messaging HA for Azure (Technical Preview)
- Streams Messaging Heavy Duty for GCP
- Streams Messaging Light Duty for GCP
- Streams Messaging HA for GCP (Technical Preview)

Cluster template name

- Cloudera - Streams Messaging Heavy Duty
- Cloudera - Streams Messaging Light Duty
- Cloudera - Streams Messaging High Availability

Included services

- Kafka
- Schema Registry
- Streams Messaging Manager
- Streams Replication Manager
- Cruise Control
- Kafka Connect

Compatible Cloudera Runtime version

- 7.2.15
- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

Related Information

Setting up your Streams Messaging cluster
Ingesting Data into Cloudera on cloud
Kafka
Schema Registry
Streams Messaging Manager
Streams Replication Manager

Flow Management clusters

Learn about the default Flow Management clusters, including cluster definition and template names, included services, and compatible Cloudera Runtime versions.

Flow Management delivers high-scale data ingestion, transformation, and management to enterprises from any-to-any environment. It addresses key enterprise use cases such as data movement, continuous data ingestion, log data ingestion, and acquisition of all types of streaming data including social, mobile, clickstream, and IoT data.

The Flow Management template includes a no-code data ingestion and management solution powered by Apache NiFi. With NiFi's intuitive graphical interface and 300+ processors, Flow Management enables easy data ingestion and movement between Cloudera services as well as 3rd party cloud services. NiFi Registry is automatically set up and provides a central place to manage versioned Data Flows.

Note:

Flow Management clusters have distinct planning considerations and how-to information. See the Cloudera DataFlow for Data Hub documentation for information about:

- Planning your Flow Management cluster deployment
- Creating your first Flow Management cluster
- Security considerations for Flow Management clusters
- Using Apache NiFi to ingest data into Cloudera on cloud
- Using NiFi and NiFi Registry

Cluster definition names

- Flow Management Light Duty for AWS
- Flow Management Light Duty for Azure
- Flow Management Light Duty for GCP
- Flow Management Heavy Duty for AWS
- Flow Management Heavy Duty for Azure
- Flow Management Heavy Duty for GCP

Cluster template name

- CDP - Flow Management: Light Duty
- CDP - Flow Management: Heavy Duty

Included services

- NiFi
- NiFI Registry

Compatible Cloudera Runtime versions

- 7.2.15
- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

Related Information

Setting up your Flow Management cluster
Apache NiFi documentation
Apache NiFi Registry documentation

Streaming Analytics clusters

Learn about the default Streaming Analytics clusters, including cluster definition and template names, included services, and compatible Cloudera Runtime version.

Streaming Analytics offers real-time stream processing and stream analytics with low-latency and high scaling capabilities powered by Apache Flink.

Streaming Analytics templates include Apache Flink that works out of the box in stateless or heavy state environments. Beside Flink, the template includes its supporting services namely YARN, Zookeeper and HDFS. The Heavy Duty template comes preconfigured with RocksDB as state backend, while Light Duty clusters use the default Heap state backend. You can create your streaming application by choosing between Kafka, Kudu, and HBase as datastream connectors.

You can also use SQL to query real-time data with SQL Stream Builder (SSB) in the Streaming Analytics template. By supporting the SSB service in Cloudera on cloud, you can simply and easily declare expressions that filter,

aggregate, route, and otherwise mutate streams of data. SSB is a job management interface that you can use to compose and run SQL on streams, as well as to create durable data APIs for the results.

Note:

Streaming Analytics clusters have distinct planning considerations and how-to information. See the Cloudera DataFlow for Data Hub documentation for information about:

- Planning your Streaming Analytics cluster deployment
- Creating your first Streaming Analytics cluster
- Analyzing data using Apache Flink
- Querying data using SQL Stream Builder

Cluster definition names

- Streaming Analytics Light Duty for AWS
- Streaming Analytics Light Duty for Azure
- Streaming Analytics Light Duty for GCP
- Streaming Analytics Heavy Duty for AWS
- Streaming Analytics Heavy Duty for Azure
- Streaming Analytics Heavy Duty for GCP

Cluster template name

- 7.3.1 - Streaming Analytics Light Duty
- 7.3.1 - Streaming Analytics Heavy Duty

Included services

- Flink
- SQL Stream Builder
- YARN
- Zookeeper
- HDFS
- Kafka

Important: In the Streaming Analytics cluster templates, Kafka service is included by default to serve as a background service only for the websocket ouput and sampling feature of SQL Stream Builder. The Kafka service in the Streaming Analytics cluster template cannot be used for production, you need to use the Streams Messaging cluster template when Kafka is needed for your deployment.

Compatible Cloudera Runtime version

- 7.2.15
- 7.2.16
- 7.2.17
- 7.2.18
- 7.3.1

Related Information

Setting up your Streaming Analytics cluster
Flink
YARN
Zookeeper
HDFS

Data Discovery and Exploration clusters

Learn about the default Data Discovery and Exploration clusters, including cluster definition and template names, included services, and compatible Cloudera Runtime version.

Data Discovery and Exploration

Explore and discover data sets ad-hoc. Do relevance-based analytics over unstructured data (logs, images, text, PDFs, etc). Get started with search or log analytics. Make data more accessible to everyone with Data Discovery and Exploration.

Cluster Definition Names

- Data Discovery and Exploration for AWS
- Data Discovery and Exploration for AWS
- Data Discovery and Exploration for Azure

Cluster Template Name

- Data Discovery and Exploration

Included Services

- Solr
- Spark 2
- HDFS
- Hue
- YARN
- ZooKeeper

Compatible Cloudera Runtime Versions

- 7.2.15
- 7.2.16
- 7.2.17

Cluster Definition Names

- Data Discovery and Exploration - Spark3 for AWS
- Data Discovery and Exploration (ARM) - Spark3 for AWS
- Data Discovery and Exploration - Spark3 for Azure
- Data Discovery and Exploration - Spark3 for Google Cloud

Cluster Template Name

- Data Discovery and Exploration for Spark3

Included Services

- Solr
- Spark 3
- HDFS
- Hue
- YARN
- ZooKeeper

Compatible Cloudera Runtime Version

- 7.2.18
- 7.3.1

Related Information
Solr
Spark
HDFS
Hue
YARN
Zookeeper

Building a custom cluster template

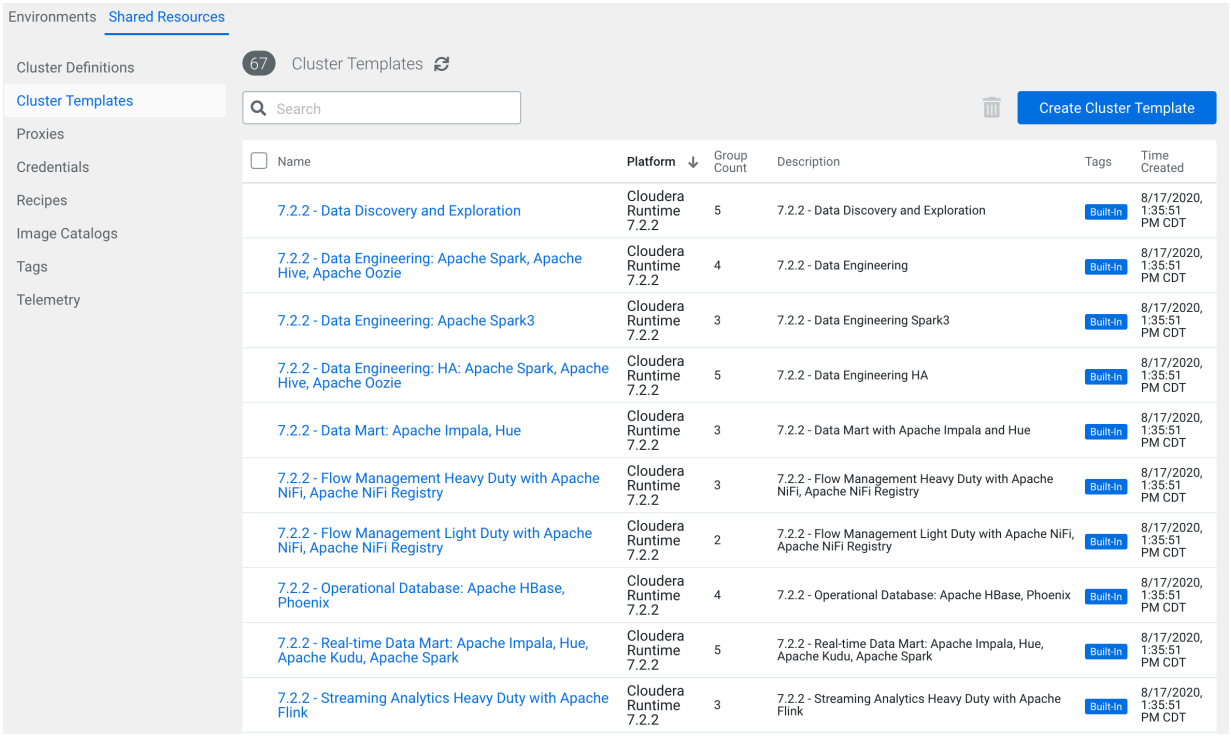
You can build a custom cluster template to modify the cluster Cloudera Runtime services, including the Cloudera Runtime configuration properties and the distribution of Runtime services across host groups. To create a custom template, modify an existing default cluster template and then upload and register the custom template.

About this task
If the default cluster templates are insufficient for the cluster that you want to create, you can build a custom cluster template. The recommended method for building a cluster template is to modify an existing default template, so that the structure of the template is mostly preserved.

Required role: EnvironmentCreator can create a shared resource and then assign users to it. SharedResourceUser or Owner of the shared resource can use the resource. The Owner of the shared resource can delete it.

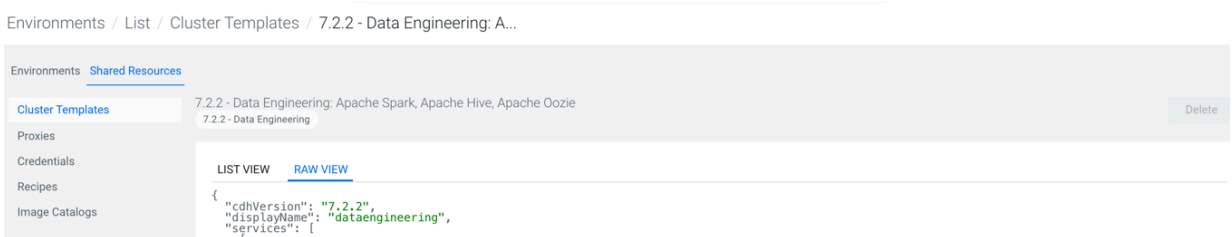
Procedure
1. Review information on the default cluster configurations to find one that includes services suitable for the type of cluster that you want to create. In general, it is best to use the templates for the current release.
2. To access the existing default cluster templates, click Shared ResourcesCluster Templates.
3. To find the newest template versions, click Platform at the top of the Platform column to sort the templates in descending order.

4. Under the Name column, click the desired default ("Built In") template.



The template opens in LIST view, which shows how the template is structured across host groups.

5. Click RAW VIEW to view the JSON structure.




```
{
  "cdhVersion": "7.2.2",
  "displayName": "dataengineering",
  "services": [
    {
      "refName": "zookeeper",
      "serviceType": "ZOOKEEPER",
      "serviceConfigs": [
        {
          "name": "service_config_suppression_server_count_validator",
          "value": "true"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "zookeeper-SERVER-BASE",
          "roleType": "SERVER",
          "base": true
        }
      ]
    },
    {
      "refName": "hdfs",
      "serviceType": "HDFS",
      "serviceConfigs": [
        {
          "name": "hdfs_verify_ec_with_topology_enabled",
          "value": false
        },
        {
          "name": "core_site_safety_valve",
          "value": "<property><name>fs.s3a.buffer.dir</name><value>${env.LOCAL_DIRS:-${hadoop.tmp.dir}}/s3a</value></property><property>
<name>fs.s3a.committer.name</name><value>directory</value></property>"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "hdfs-NAMENODE-BASE",
          "roleType": "NAMENODE",
          "base": true,
          "configs": [
            {
              "name": "role_config_suppression_fs_trash_interval_minimum_validator",
              "value": "true"
            },
            {
              "name": "role_config_suppression_namenode_java_heapsize_minimum_validator",
              "value": "true"
            },
            {
              "name": "fs_trash_interval",
              "value": "0"
            },
            {
              "name": "fs_trash_checkpoint_interval",
              "value": "0"
            },
            {
              "name": "erasure_coding_default_policy",
              "value": " "
            }
          ]
        },
        {
          "refName": "hdfs-SECONDARYNAMENODE-BASE",
          "roleType": "SECONDARYNAMENODE",
          "base": true
        },
        {
          "refName": "hdfs-DATANODE-BASE",
          "roleType": "DATANODE",
          "base": true
        },
        {
          "refName": "hdfs-BALANCER-BASE",
          "roleType": "BALANCER",
          "base": true
        },
        {
          "refName": "hdfs-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true,
          "configs": [
            {
              "name": "dfs_client_use_trash",
              "value": false
            },
            {
              "name": "role_config_suppression_hdfs_trash_disabled_validator",
              "value": "true"
            },
            {
              "name": "hdfs_client_env_safety_valve",
              "value": "HADOOP_OPTS=\"-Dorg.wildfly.openssl.path=/usr/lib64 ${HADOOP_OPTS}\""
            }
          ]
        }
      ]
    },
    {
      "refName": "yarn",
      "serviceType": "YARN",
      "serviceConfigs": [
        {
          "name": "yarn_admin_acl",
          "value": "yarn,hive,hdfs,mapred"
        },
        {
          "name": "yarn_service_mapred_safety_valve",
          "value": "<property><name>mapreduce.fileoutputcommitter.algorithm.version</name><value>1</value></property><property>
<name>mapreduce.input.fileinputformat.list-status.num-threads</name><value>100</value></property>"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "yarn-RESOURCEMANAGER-BASE",
          "roleType": "RESOURCEMANAGER",
          "base": true,
          "configs": [
            {
              "name": "resourcemanager_config_safety_valve",
              "value": "<property><name>yarn.scheduler.configuration.store.class</name><value>zk</value></property>"
            },
            {
              "name": "yarn_resourcemanager_scheduler_class",
              "value": "org.apache.hadoop.yarn.server.resourcemanager.scheduler.capacity.CapacityScheduler"
            },
            {
              "name": "yarn_scheduler_capacity_resource_calculator",
              "value": "org.apache.hadoop.yarn.util.resource.DefaultResourceCalculator"
            },
            {
              "name": "resourcemanager_capacity_scheduler_configuration",
              "value": "<configuration><property><name>yarn.scheduler.capacity.root.queues</name><value>default</value></property><property>
<name>yarn.scheduler.capacity.root.capacity</name><value>100</value></property><property><name>yarn.scheduler.capacity.root.default.capacity</name>
<value>100</value></property><property><name>yarn.scheduler.capacity.root.acl_submit_applications</name><value> </value></property><property>
<name>yarn.scheduler.capacity.root.acl_administer_queue</name><value> </value></property><property>
<name>yarn.scheduler.capacity.root.default.acl_submit_applications</name><value>*</value></property><property>
<name>yarn.scheduler.capacity.root.default.minimum-user-limit-percent</name><value>100</value></property><property>
<name>yarn.scheduler.capacity.maximum-am-resource-percent</name><value>0.33</value></property><property><name>yarn.scheduler.capacity.node-
locality-delay</name><value>0</value></property><property><name>yarn.scheduler.capacity.schedule-asynchronously.maximum-threads</name>
<value>1</value></property><property><name>yarn.scheduler.capacity.schedule-asynchronously.scheduling-interval-ms</name><value>10</value>
</property></configuration>"
            }
          ]
        },
        {
          "refName": "yarn-NODEMANAGER-WORKER",
          "roleType": "NODEMANAGER",
          "base": false
        },
        {
          "refName": "yarn-NODEMANAGER-COMPUTE",
          "roleType": "NODEMANAGER",
          "base": false
        },
        {
          "refName": "yarn-JOBHISTORY-BASE",
          "roleType": "JOBHISTORY",
          "base": true
        },
        {
          "refName": "yarn-GATEWAY-BASE",
```

6. Select all of the JSON code, copy it, and paste it to a suitable code editor. Standard text editors are not recommended.
7. If you are using a code editor such as Microsoft Visual Studio Code, you can use built-in tools to validate the JSON before you proceed. Optionally, you can save the file as-is (without having made any changes), and upload it using the instructions in the topic *Upload a cluster template*. To verify that the template JSON is functional, you can create a test cluster by selecting the template that you just registered, and see if the cluster successfully deploys.
8. When you are satisfied that you are working with a clean template, you can begin to modify the template. Return to the JSON template file in your code editor.

   Each default template consists of two main sections: the services section and the hostTemplates section. The serv ices section includes the components that make up the cluster. This is where you can add or remove services, as well as modify service configuration properties. If you want to modify a service's configuration, for example to tune Yarn or Hive, refer to the Cloudera Manager configuration properties for the desired service. You can search these properties by their API Name, which is how they appear in a Cloudera Data Hub template.

   For example, in a Data Engineering template you might want to adjust the amount of physical memory allocated for containers by configuring the yarn.nodemanager.resource.memory-mb property. If you want to configure this property to 80% of the total system RAM, for a 256 GB machine this would look like:

```
{
      "refName": "yarn",
      "serviceType": "YARN",
      "serviceConfigs": [
        {
          "name": "yarn_admin_acl",
          "value": "yarn,hive,hdfs,mapred"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "yarn-RESOURCEMANAGER-BASE",
          "roleType": "RESOURCEMANAGER",
          "base": true,
          "configs": [
            {
              "name": "yarn.nodemanager.resource.memory-mb",
              "value": 2052096
            },
```

   If you want to add a service in a template, the simplest method is to find the service in the RAW VIEW of another default template and copy it into your JSON.

   For example, if you want to add Sqoop to the services in a template, copy it from the Data Engineering or Data Engineering HA template into the services section of another template:

```
{
      "refName": "sqoop",
      "serviceType": "SQOOP_CLIENT",
      "roleConfigGroups": [
        {
          "refName": "sqoop-SQOOP_CLIENT-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "configs": [],
          "base": true
        }
      ]
```

```
    },
```

The hostTemplates section of the JSON file describes the nodes by their type and the services on the node. This section also includes a cardinality parameter, which you can set to increase or decrease the quantity of that specific node type.

For example, say that you want to create a new node type called "ZKserver" that runs a single service, Zookeeper. Assuming that Zookeeper is already a service defined in the services section of the template, you can move down to the hostTemplate section. In the master node section of this Data Engineering template, you can see that Zookeeper is already defined in the "master" node section with the string "zookeeper-SERVER-BASE":

```
"hostTemplates": [
    {
      "refName": "master",
      "cardinality": 1,
      "roleConfigGroupsRefNames": [
        "hdfs-BALANCER-BASE",
        "hdfs-NAMENODE-BASE",
        "hdfs-SECONDARYNAMENODE-BASE",
        "hdfs-GATEWAY-BASE",
        "hms-GATEWAY-BASE",
        "hms-HIVEMETASTORE-BASE",
        "hive_on_tez-HIVESERVER2-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "hue-HUE_LOAD_BALANCER-BASE",
        "hue-HUE_SERVER-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "spark_on_yarn-SPARK_YARN_HISTORY_SERVER-BASE",
        "livy-LIVY_SERVER-BASE",
        "zeppelin-ZEPPELIN_SERVER-BASE",
        "oozie-OOZIE_SERVER-BASE",
        "sqoop-SQOOP_CLIENT-GATEWAY-BASE",
        "yarn-JOBHISTORY-BASE",
        "yarn-RESOURCEMANAGER-BASE",
        "zookeeper-SERVER-BASE",
        "das-DAS_WEBAPP",
        "das-DAS_EVENT_PROCESSOR",
        "yarn-QUEUEMANAGER_WEBAPP-BASE",
        "yarn-QUEUEMANAGER_STORE-BASE",
        "yarn-GATEWAY-BASE"
      ]
```

To create our new ZKserver node, you can copy the standard node format and modify it for your purpose:

```
{
      "refName": "ZKserver",
      "cardinality": 1,
      "roleConfigGroupsRefNames": [
        "zookeeper-SERVER-BASE"
      ]
    }
```

If you also want to include dynamic parameters in your custom template, see the documentation for Dynamic cluster templates. During the cluster creation phase, dynamic parameters pick up the parameter values that you provided in the Cloudera Data Hub cluster wizard. See the *Custom Properties* documentation for a list of properties that can be dynamically replaced. You might want to use dynamic parameters when you regularly provision clusters using a specific template, but want to change a few of the property values each time you provision a new cluster.

9. When you have finished modifying the template, validate the JSON in your code editor and save the template.

What to do next

Upload the JSON file and register the template following the instructions in the *Upload a cluster template* topic. Then, when you navigate to the Data Hubs page and select Create Data Hub, be sure to select the Custom radio button underneath environment selection. Here you can provision a Cloudera Data Hub cluster using the custom template that you registered. Select your custom template from the drop-down menu before you configure any advanced options.

Related Information

Upload a cluster template

Cluster template structure

The template below is an example of a 7.2.2 Data Engineering HA cluster template in JSON view.

Many of the default templates contain properties for services that aren't displayed when you select a default template in the user interface, for example Hive on Tez or Sqoop in the default Data Engineering HA template below. If you are creating a custom template and you need to know more about the properties and services in a template, refer to the Cloudera Manager configuration properties for each Cloudera Runtime service. These properties can be helpful if you are trying to tune Yarn or Hive, for example. You can also copy a service like Sqoop from the JSON of one template to another.

```
{
  "cdhVersion": "7.2.2",
  "displayName": "dataengineering ha",
  "services": [
    {
      "refName": "zookeeper",
      "serviceType": "ZOOKEEPER",
      "roleConfigGroups": [
        {
          "refName": "zookeeper-SERVER-BASE",
          "roleType": "SERVER",
          "base": true
        }
      ]
    },
    {
      "refName": "hdfs",
      "serviceType": "HDFS",
      "serviceConfigs": [
        {
          "name": "zookeeper_service",
          "ref": "zookeeper"
        },
        {
          "name": "hdfs_verify_ec_with_topology_enabled",
          "value": false
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "hdfs-NAMENODE-BASE",
          "roleType": "NAMENODE",
          "base": true,
          "configs": [
            {
              "name": "fs_trash_interval",
              "value": "0"
            },
            {
```

```
              "name": "fs_trash_checkpoint_interval",
              "value": "0"
            },
            {
              "name": "erasure_coding_default_policy",
              "value": " "
            }
          ]
        },
        {
          "refName": "hdfs-FAILOVERCONTROLLER-BASE",
          "roleType": "FAILOVERCONTROLLER",
          "base": true
        },
        {
          "refName": "hdfs-JOURNALNODE-BASE",
          "roleType": "JOURNALNODE",
          "base": true
        },
        {
          "refName": "hdfs-DATANODE-BASE",
          "roleType": "DATANODE",
          "base": true
        },
        {
          "refName": "hdfs-SECONDARYNAMENODE-BASE",
          "roleType": "SECONDARYNAMENODE",
          "configs": [
            {
              "name": "fs_checkpoint_dir_list",
              "value": "/should_not_be_required_in_HA_setup"
            }
          ],
          "base": true
        },
        {
          "refName": "hdfs-BALANCER-BASE",
          "roleType": "BALANCER",
          "base": true
        },
        {
          "refName": "hdfs-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true,
          "configs": [
            {
              "name": "dfs_client_use_trash",
              "value": false
            },
            {
              "name": "hdfs_client_env_safety_valve",
              "value": "HADOOP_OPTS=\"-Dorg.wildfly.openssl.path=/usr/lib64
${HADOOP_OPTS}\""
            }
          ]
        }
      ]
    },
    {
      "refName": "hms",
      "serviceType": "HIVE",
      "displayName": "Hive Metastore",
      "roleConfigGroups": [
        {
```

```
          "refName": "hms-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true
        },
        {
          "refName": "hms-HIVEMETASTORE-BASE",
          "roleType": "HIVEMETASTORE",
          "base": true
        }
      ]
    },
    {
      "refName": "hive",
      "serviceType": "HIVE_ON_TEZ",
      "displayName": "Hive",
      "serviceConfigs": [
        {
          "name": "tez_container_size",
          "value": "4096"
        },
        {
          "name": "tez_auto_reducer_parallelism",
          "value": "false"
        },
        {
          "name": "hive_service_config_safety_valve",
          "value": "<property><name>fs.s3a.ssl.channel.mode</name><value>op
enssl</value></property><property><name>hive.txn.acid.dir.cache.duration</na
me><value>0</value></property>"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "hive-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true
        },
        {
          "refName": "hive-HIVESERVER2-BASE",
          "roleType": "HIVESERVER2",
          "base": true,
          "configs": [
            {
              "name": "hive_server2_transport_mode",
              "value": "http"
            },
            {
              "name": "hiveserver2_mv_files_thread",
              "value": 20
            },
            {
              "name": "hiveserver2_load_dynamic_partitions_thread_count",
              "value": 20
            }
          ]
        }
      ]
    },
    {
      "refName": "hue",
      "serviceType": "HUE",
      "serviceConfigs": [
        {
          "name": "hue_service_safety_valve",
```

```
          "value": "[desktop]\napp_blacklist=spark,zookeeper,hbase,impala
,search,sqoop,security,pig"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "hue-HUE_SERVER-BASE",
          "roleType": "HUE_SERVER",
          "base": true
        },
        {
          "refName": "hue-HUE_LOAD_BALANCER-BASE",
          "roleType": "HUE_LOAD_BALANCER",
          "base": true
        }
      ]
    },
    {
      "refName": "livy",
      "serviceType": "LIVY",
      "roleConfigGroups": [
        {
          "refName": "livy-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true
        },
        {
          "refName": "livy-LIVY_SERVER-BASE",
          "roleType": "LIVY_SERVER",
          "base": true
        }
      ]
    },
    {
      "refName": "oozie",
      "serviceType": "OOZIE",
      "roleConfigGroups": [
        {
          "refName": "oozie-OOZIE_SERVER-BASE",
          "roleType": "OOZIE_SERVER",
          "configs": [
            {
              "name": "oozie_config_safety_valve",
              "value": "<property><name>oozie.service.HadoopAccessorService
.nameNode.whitelist</name><value></value></property>"
            }
          ],
          "base": true
        }
      ]
    },
    {
      "refName": "sqoop",
      "serviceType": "SQOOP_CLIENT",
      "roleConfigGroups": [
        {
          "refName": "sqoop-SQOOP_CLIENT-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "configs": [],
          "base": true
        }
      ]
    },
    {
```

```
      "refName": "yarn",
      "serviceType": "YARN",
      "serviceConfigs": [
        {
          "name": "yarn_admin_acl",
          "value": "yarn,hive,hdfs,mapred"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "yarn-RESOURCEMANAGER-BASE",
          "roleType": "RESOURCEMANAGER",
          "base": true,
          "configs": [
            {
              "name": "resourcemanager_config_safety_valve",
              "value": "<property><name>yarn.scheduler.configuration.stor
e.class</name><value>zk</value></property>"
            },
            {
              "name": "yarn_resourcemanager_scheduler_class",
              "value": "org.apache.hadoop.yarn.server.resourcemanager.sch
eduler.capacity.CapacityScheduler"
            },
            {
              "name": "yarn_scheduler_capacity_resource_calculator",
              "value": "org.apache.hadoop.yarn.util.resource.DefaultResourc
eCalculator"
            },
            {
              "name": "resourcemanager_capacity_scheduler_configuration",
              "value": "<configuration><property><name>yarn.scheduler.capaci
ty.root.queues</name><value>default</value></property><property><name>yarn.s
cheduler.capacity.root.capacity</name><value>100</value></property><property
><name>yarn.scheduler.capacity.root.default.capacity</name><value>100</value
></property><property><name>yarn.scheduler.capacity.root.acl_submit_applicat
ions</name><value> </value></property><property><name>yarn.scheduler.capacit
y.root.acl_administer_queue</name><value> </value></property><property><name
>yarn.scheduler.capacity.root.default.acl_submit_applications</name><value>*
</value></property><property><name>yarn.scheduler.capacity.root.default.mini
mum-user-limit-percent</name><value>100</value></property><property><name>ya
rn.scheduler.capacity.maximum-am-resource-percent</name><value>0.33</value><
/property><property><name>yarn.scheduler.capacity.node-locality-delay</name>
<value>0</value></property><property><name>yarn.scheduler.capacity.schedule-
asynchronously.maximum-threads</name><value>1</value></property><property><n
ame>yarn.scheduler.capacity.schedule-asynchronously.scheduling-interval-ms</
name><value>10</value></property></configuration>"
            }
          ]
        },
        {
          "refName": "yarn-NODEMANAGER-WORKER",
          "roleType": "NODEMANAGER",
          "base": false
        },
        {
          "refName": "yarn-NODEMANAGER-COMPUTE",
          "roleType": "NODEMANAGER",
          "base": false
        },
        {
          "refName": "yarn-JOBHISTORY-BASE",
          "roleType": "JOBHISTORY",
          "base": true
```

```
        },
        {
          "refName": "yarn-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true,
          "configs": [
            {
              "name": "mapreduce_map_memory_mb",
              "value": 4096
            },
            {
              "name": "mapreduce_reduce_memory_mb",
              "value": 4096
            },
            {
              "name": "mapreduce_client_env_safety_valve",
              "value": "HADOOP_OPTS=\"-Dorg.wildfly.openssl.path=/usr/lib64
 ${HADOOP_OPTS}\""
            }
          ]
        }
      ]
    },
    {
      "refName": "spark_on_yarn",
      "serviceType": "SPARK_ON_YARN",
      "roleConfigGroups": [
        {
          "refName": "spark_on_yarn-SPARK_YARN_HISTORY_SERVER-BASE",
          "roleType": "SPARK_YARN_HISTORY_SERVER",
          "base": true
        },
        {
          "refName": "spark_on_yarn-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true,
          "configs": [
            {
              "name": "spark-conf/spark-defaults.conf_client_config_safety_v
alve",
              "value": "spark.hadoop.fs.s3a.ssl.channel.mode=openssl"
            }
          ]
        }
      ]
    },
    {
      "refName": "tez",
      "serviceType": "TEZ",
      "serviceConfigs": [
        {
          "name": "yarn_service",
          "ref": "yarn"
        },
        {
          "name": "tez.am.container.reuse.non-local-fallback.enabled",
          "value": "true"
        },
        {
          "name": "tez.am.container.reuse.locality.delay-allocation-millis",
          "value": "0"
        },
        {
          "name": "tez.am.launch.cmd-opts",
```

```
          "value": "-XX:+PrintGCDetails -verbose:gc -XX:+UseNUMA -XX:+UseG1G
C -XX:+ResizeTLAB"
        },
        {
          "name": "tez.task.launch.cmd-opts",
          "value": "-XX:+PrintGCDetails -verbose:gc -XX:+UseNUMA -XX:+UseG
1GC -XX:+ResizeTLAB"
        },
        {
          "name": "tez.grouping.split-waves",
          "value": 1.4
        },
        {
          "name": "tez.grouping.min-size",
          "value": 268435456
        },
        {
          "name": "tez.grouping.max-size",
          "value": 268435456
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "tez-GATEWAY-BASE",
          "roleType": "GATEWAY",
          "base": true
        }
      ]
    },
    {
      "refName": "das",
      "serviceType": "DAS",
      "roleConfigGroups": [
        {
          "refName": "das-DAS_WEBAPP",
          "roleType": "DAS_WEBAPP",
          "base": true,
          "configs": [
            {
              "name": "data_analytics_studio_admin_users",
              "value": "*"
            }
          ]
        },
        {
          "refName": "das-DAS_EVENT_PROCESSOR",
          "roleType": "DAS_EVENT_PROCESSOR",
          "base": true
        }
      ]
    },
    {
      "refName": "knox",
      "roleConfigGroups": [
        {
          "base": true,
          "refName": "knox-KNOX_GATEWAY-BASE",
          "roleType": "KNOX_GATEWAY"
        }
      ],
      "serviceType": "KNOX"
    },
    {
      "refName": "zeppelin",
```

```
      "serviceType": "ZEPPELIN",
      "serviceConfigs": [
        {
          "name": "yarn_service",
          "ref": "yarn"
        },
        {
          "name": "hdfs_service",
          "ref": "hdfs"
        },
        {
          "name": "spark_on_yarn_service",
          "ref": "spark_on_yarn"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "zeppelin-ZEPPELIN_SERVER-BASE",
          "roleType": "ZEPPELIN_SERVER",
          "base": true
        }
      ]
    },
    {
      "refName": "queuemanager",
      "serviceType": "QUEUEMANAGER",
      "serviceConfigs": [
        {
          "name": "kerberos.auth.enabled",
          "value": "true"
        }
      ],
      "roleConfigGroups": [
        {
          "refName": "yarn-QUEUEMANAGER_WEBAPP-BASE",
          "roleType": "QUEUEMANAGER_WEBAPP",
          "base": true
        },
        {
          "refName": "yarn-QUEUEMANAGER_STORE-BASE",
          "roleType": "QUEUEMANAGER_STORE",
          "base": true
        }
      ]
    }
  ],
  "hostTemplates": [
    {
      "refName": "gateway",
      "cardinality": 0,
      "roleConfigGroupsRefNames": [
        "hdfs-GATEWAY-BASE",
        "hive-GATEWAY-BASE",
        "hms-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "yarn-GATEWAY-BASE"
      ]
    },
    {
      "refName": "master",
      "cardinality": 2,
      "roleConfigGroupsRefNames": [
```

```
        "hdfs-FAILOVERCONTROLLER-BASE",
        "hdfs-NAMENODE-BASE",
        "hdfs-GATEWAY-BASE",
        "hive-GATEWAY-BASE",
        "hive-HIVESERVER2-BASE",
        "hms-GATEWAY-BASE",
        "hms-HIVEMETASTORE-BASE",
        "hue-HUE_LOAD_BALANCER-BASE",
        "hue-HUE_SERVER-BASE",
        "knox-KNOX_GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "yarn-RESOURCEMANAGER-BASE",
        "hdfs-JOURNALNODE-BASE",
        "zookeeper-SERVER-BASE",
        "yarn-GATEWAY-BASE"
      ]
    },
    {
      "refName": "manager",
      "cardinality": 1,
      "roleConfigGroupsRefNames": [
        "hdfs-BALANCER-BASE",
        "hdfs-GATEWAY-BASE",
        "hive-GATEWAY-BASE",
        "hms-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "livy-LIVY_SERVER-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "spark_on_yarn-SPARK_YARN_HISTORY_SERVER-BASE",
        "tez-GATEWAY-BASE",
        "yarn-JOBHISTORY-BASE",
        "oozie-OOZIE_SERVER-BASE",
        "sqoop-SQOOP_CLIENT-GATEWAY-BASE",
        "zeppelin-ZEPPELIN_SERVER-BASE",
        "das-DAS_EVENT_PROCESSOR",
        "das-DAS_WEBAPP",
        "knox-KNOX_GATEWAY-BASE",
        "hdfs-JOURNALNODE-BASE",
        "zookeeper-SERVER-BASE",
        "yarn-QUEUEMANAGER_WEBAPP-BASE",
        "yarn-QUEUEMANAGER_STORE-BASE",
        "yarn-GATEWAY-BASE"
      ]
    },
    {
      "refName": "worker",
      "cardinality": 3,
      "roleConfigGroupsRefNames": [
        "hdfs-DATANODE-BASE",
        "hdfs-GATEWAY-BASE",
        "hive-GATEWAY-BASE",
        "hms-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "yarn-NODEMANAGER-WORKER",
        "yarn-GATEWAY-BASE"
      ]
    },
    {
      "refName": "compute",
      "cardinality": 0,
```

```
      "roleConfigGroupsRefNames": [
        "hdfs-GATEWAY-BASE",
        "hive-GATEWAY-BASE",
        "hms-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "yarn-NODEMANAGER-COMPUTE",
        "yarn-GATEWAY-BASE"
      ]
    }
  ]
}
```

Dynamic cluster template parameters

You can add dynamic parameters to a cluster template. The values of the variables specified in the cluster template are dynamically replaced during the cluster creation phase, picking up the parameter values that you provided in the Cloudera Data Hub cluster wizard. Cloudera Data Hub supports the "mustache" formatting syntax for dynamic properties.

Production cluster configurations typically include certain configuration parameters, forcing you to create multiple versions of the same cluster template to handle different component configurations for these external systems. Dynamic parameters in a cluster template solve this problem by offering the ability to manage external sources outside of your cluster template. They use the cluster template as a blueprint, and Cloudera Data Hub injects the actual configurations into your cluster template. This simplifies the reuse of cluster configurations for external sources and simplifies the cluster templates themselves.

Note:

You cannot use functions in the cluster template file; only variable injection is supported.

The following provides a list of properties that can be used in a dynamic cluster template and an example:

Recipe and cluster template parameters

The following supported parameters can be specified as variables/dynamic parameters in recipes or cluster templates by using mustache formatting with "{{{ }}}" syntax.

Note: Using variable parameters is not supported for FreeIPA recipes.

The parameter keys listed below follow the following general conventions:

- { } indicates that the parameter key has multiple supported values, which are provided in this documentation. For example {fileSystemType} can be one of the following: s3, adls_gen_2, or wasb.
- [index] indicates that the parameter includes an index value for example sharedService.datalakeComponents.[index] can be "sharedService.datalakeComponents.[0]", "sharedService.datalakeComponents.[1]", and so on. There is no easy way to find out what the index will be, but you may still be able to use these parameters (for example by creating a condition to filter them).

For information on how to set these parameters dynamically in a cluster template, refer to Setting custom properties.

Custom properties

Any custom property specified in the cluster template can be used as a recipe parameter. Refer to Custom properties documentation.

General

The general parameter group includes parameters related to general cluster configuration.

Description	Example key	Example value
Name of stack	general.stackName	teststack
UUID of cluster	general.uuid	9aab7fdb-8940-454b-bc0a-62f04bce6519
Cloudera Manager user name	general.cmUserName	
Cloudera Manager password	general.cmPassword	
Cloudera Manager IP	general.clusterManagerIp	127.0.0.1
Number of nodes	general.nodeCount	5
FQDN of primary gateway instance	general.primaryGatewayInstanceDiscoveryFQDN	ip-10-0-88-28.example.com
Number indicating the Kafka replication factor (3 or 1)	general.kafkaReplicationFactor	1

Blueprint

The blueprint parameter group includes parameters related to cluster template configuration.

Parameter key	Description	Example key	Example value
blueprint.blueprintText	Blueprint text in JSON format	blueprint.blueprintText	
blueprint.version	Version of blueprint	blueprint.version	7.2.8

Cloud storage

The fileSystemConfigs parameter group includes parameters related to cloud storage configuration.

When forming the parameter keys, the {fileSystemType} should be replaced with an actual cloud storage type such as "s3", "adls_gen_2", or "wasb".

Parameter key	Description	Example key	Example value
File system common configurations			
fileSystemConfigs.{fileSystemType}.storageContainer	Name of container in Azure storage account (Cloudbreak + stackId)	fileSystemConfigs.s3.storageContainer	cloudbreak123
fileSystemConfigs.{fileSystemType}.type	Type of filesystem	fileSystemConfigs.s3.type	S3
fileSystemConfigs.{fileSystemType}.locations.[index].configFile	Configuration file used to configure the filesystem	fileSystemConfigs.s3.locations.[0].configFile	hbase-site
fileSystemConfigs.{fileSystemType}.locations.[index].property	Property key of filesystem path in defined config	fileSystemConfigs.s3.locations.[0].property	hbase.rootdir
fileSystemConfigs.{fileSystemType}.locations.[index].value	Value of filesystem path in defined config	fileSystemConfigs.s3.locations.[0].value	s3a://testranger/testrecipe2/apps/hbase/data
Amazon S3 configurations			
fileSystemConfigs.s3.storageContainer	Generated name (cloudbreak + stack id number)	fileSystemConfigs.s3.storageContainer	cloudbreak7941
fileSystemConfigs.s3.locations.[index].configFile	Hadoop component configuration file	fileSystemConfigs.s3.locations.[0].configFile	zeppelin-site
fileSystemConfigs.s3.locations.[index].property	Component property name	fileSystemConfigs.s3.locations.[0].property	zeppelin.notebook.dir

Parameter key	Description	Example key	Example value
fileSystemConfigs.s3.locations.[index].value	Component property value	fileSystemConfigs.s3.locations.[0].value	s3a://storagename/clustername/zeppelin/notebook
ADLS Gen2 configurations			
fileSystemConfigs.adls_gen_2.accountName	Name of the corresponding Azure storage account	fileSystemConfigs.adls_gen_2.accountName	testStorageaccount
fileSystemConfigs.adls_gen_2.storageContainerName	Container Name in Azure storage account	fileSystemConfigs.adls_gen_2.storageContainerName	testContainerName

External database

The rds parameter group includes parameters related to external database configuration.

When forming the parameter keys, the {rdsType} should be replaced with the actual database type such as "cloudera_manager", "beacon", "druid", "hive", "oozie", "ranger", "superset", or some other user-defined type.

Parameter key	Description	Example key	Example value
rds.{rdsType}.connectionURL	JDBC connection URL	rds.hive.connectionURL	Value is specified in the following format: jdbc:postgresql://host:port/database
rds.{rdsType}.connectionDriver	JDBC driver used for connection	rds.hive.connectionDriver	org.postgresql.Driver
rds.{rdsType}.connectionUserName	Username used for the JDBC connection	rds.hive.connectionUserName	testuser
rds.{rdsType}.connectionPassword	Password used for the JDBC connection	rds.hive.connectionPassword	TestPssword123
rds.{rdsType}.databaseName	Target database of the JDBC connection	rds.hive.databaseName	myhivedb
rds.{rdsType}.host	Host of the JDBC connection	rds.hive.host	mydbhost
rds.{rdsType}.hostWithPortWithJdbc	Host of JDBC connection with port and JDBC prefix	rds.hive.hostWithPortWithJdbc	Value is specified in the following format: jdbc:postgresql://host:port
rds.{rdsType}.subprotocol	Sub-protocol from the JDBC URL	rds.hive.subprotocol	postgresql
rds.{rdsType}.connectionString	URL of JDBC the connection. In case of Ranger, this does not contain the port	rds.hive.connectionString	Value is specified in the following format: jdbc:postgresql://host:port/database
rds.{rdsType}.databaseVendor	Database vendor	rds.hive.databaseVendor	POSTGRES
rds.{rdsType}.withoutJDBCPrefix	URL of the JDBC connection without JDBC prefix	rds.hive.withoutJDBCPrefix	Value is specified in the following format: host:port/database

Gateway

The gateway parameter group includes parameters related to Knox gateway configuration.

Parameter key	Description	Example key	Example value
gateway.ssoProvider	Path to the SSO provider	gateway.ssoProvider	/test/sso/api/v1/websso
gateway.signKey	Base64 encoded signing key	gateway.signKey	
gateway.signPub	Signing certificate (x509 format)	gateway.signPub	
gateway.signCert	Public SSH key used for signing (standard public key format)	gateway.signCert	

Shared services

The sharedService parameter group includes parameters related to Data Lake configuration.

Parameter key	Description	Example key	Example value
sharedService.rangerAdminPassword	Admin password of the Ranger component	sharedService.rangerAdminPassword	Admin1234!
sharedService.datalakeCluster	Flag indicating that the cluster is a data lake cluster	sharedService.datalakeCluster	true
sharedService.datalakeClusterManagerIp	Cloudera Manager IP of data lake cluster	sharedService.datalakeClusterManagerIp	127.0.0.1
sharedService.datalakeClusterManagerFqdn	Cloudera Manager FQDN of data lake cluster (or the IP if FQDN cannot be found)	sharedService.datalakeClusterManagerFqdn	ip-10-88-88-28.example.com

Example: A cluster template with parameters

The template below is an example of a cluster template that includes placeholder values (the {{{general.clusterName}}} property) that will be dynamically fetched and replaced during cluster creation.

```
{
  "cdhVersion": "7.0.0",
  "products": [
    {
      "version": "7.0.0-1.cdh7.0.0.p0.1309226",
      "product": "CDH"
    }
  ],
  "services": [
    {
      "refName": "hdfs",
      "serviceType": "HDFS",
      "serviceConfigs": [],
      "roleConfigGroups": [
        {
          "refName": "hdfs-NAMENODE-BASE",
          "roleType": "NAMENODE",
          "base": true,
          "displayName": null,
          "configs": [
            {
              "name": "dfs_name_dir_list",
              "value": "/hadoopfs/{{{general.clusterName}}}/namenode"
            }
          ]
        }
      ],
      "roles": null,
      "displayName": null
    }
  ],
  "hostTemplates": [
    {
      "refName": "master",
      "roleConfigGroupsRefNames": [
        "hdfs-BALANCER-BASE",
        "hdfs-NAMENODE-BASE",
        "hdfs-SECONDARYNAMENODE-BASE",
        "hms-GATEWAY-BASE",
        "hms-HIVEMETASTORE-BASE",
        "hive_on_tez-HIVESERVER2-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "hue-HUE_SERVER-BASE",
```

```
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "spark_on_yarn-SPARK_YARN_HISTORY_SERVER-BASE",
        "livy-LIVY_SERVER-BASE",
        "zeppelin-ZEPPELIN_SERVER-BASE",
        "oozie-OOZIE_SERVER-BASE",
        "yarn-JOBHISTORY-BASE",
        "yarn-RESOURCEMANAGER-BASE",
        "zookeeper-SERVER-BASE",
        "knox-KNOX_GATEWAY-BASE"
      ],
      "cardinality": 1
    },
    {
      "refName": "worker",
      "roleConfigGroupsRefNames": [
        "hdfs-DATANODE-BASE",
        "hms-GATEWAY-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "yarn-NODEMANAGER-WORKER"
      ],
      "cardinality": 1
    },
    {
      "refName": "compute",
      "roleConfigGroupsRefNames": [
        "hms-GATEWAY-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "yarn-NODEMANAGER-COMPUTE"
      ],
      "cardinality": 0
    }
  ],
  "displayName": "dataengineering",
  "cmVersion": "7.x.0",
  "repositories": [
    "http://cloudera-build-us-west-1.vpc.cloudera.com/s3/build/1309226/cdh
/7.x/parcels/"
  ],
  "clusterSpec": null
}
```

Example cluster template after {{{general.clusterName}}} is set to my-super-cluster based on the actual cluster name:

```
{
  "cdhVersion": "7.0.0",
  "products": [
    {
      "version": "7.0.0-1.cdh7.0.0.p0.1309226",
      "product": "CDH"
    }
  ],
  "services": [
    {
      "refName": "hdfs",
      "serviceType": "HDFS",
      "serviceConfigs": [],
```

```
      "roleConfigGroups": [
        {
          "refName": "hdfs-NAMENODE-BASE",
          "roleType": "NAMENODE",
          "base": true,
          "displayName": null,
          "configs": [
            {
              "name": "dfs_name_dir_list",
              "value": "/hadoopfs/my-super-cluster/namenode"
            }
          ]
        }
      ],
      "roles": null,
      "displayName": null
    }
  ],
  "hostTemplates": [
    {
      "refName": "master",
      "roleConfigGroupsRefNames": [
        "hdfs-BALANCER-BASE",
        "hdfs-NAMENODE-BASE",
        "hdfs-SECONDARYNAMENODE-BASE",
        "hms-GATEWAY-BASE",
        "hms-HIVEMETASTORE-BASE",
        "hive_on_tez-HIVESERVER2-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "hue-HUE_SERVER-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "spark_on_yarn-SPARK_YARN_HISTORY_SERVER-BASE",
        "livy-LIVY_SERVER-BASE",
        "zeppelin-ZEPPELIN_SERVER-BASE",
        "oozie-OOZIE_SERVER-BASE",
        "yarn-JOBHISTORY-BASE",
        "yarn-RESOURCEMANAGER-BASE",
        "zookeeper-SERVER-BASE",
        "knox-KNOX_GATEWAY-BASE"
      ],
      "cardinality": 1
    },
    {
      "refName": "worker",
      "roleConfigGroupsRefNames": [
        "hdfs-DATANODE-BASE",
        "hms-GATEWAY-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "livy-GATEWAY-BASE",
        "yarn-NODEMANAGER-WORKER"
      ],
      "cardinality": 1
    },
    {
      "refName": "compute",
      "roleConfigGroupsRefNames": [
        "hms-GATEWAY-BASE",
        "hive_on_tez-GATEWAY-BASE",
        "tez-GATEWAY-BASE",
        "spark_on_yarn-GATEWAY-BASE",
        "yarn-NODEMANAGER-COMPUTE"
```

```
      ],
      "cardinality": 0
    }
  ],
  "displayName": "dataengineering",
  "cmVersion": "7.x.0",
  "repositories": [
    "http://cloudera-build-us-west-1.vpc.cloudera.com/s3/build/1309226/cdh/
7.x/parcels/"
  ],
  "clusterSpec": null
}
```

For more information on setting these properties, see *Setting custom properties*.

Related Information

Setting custom properties

Uploading a cluster template

Once you have your custom template ready, upload it to the Cloudera Management Console. Once the template is uploaded, you can select it during cluster creation.

About this task

Required role: EnvironmentCreator can create a shared resource and then assign users to it. SharedResourceUser or Owner of the shared resource can use the resource.

Procedure

1. Log in to the Cloudera web interface.
2. Navigate to Shared ResourcesCluster TemplatesCreate Cluster Template.
3. Provide the following information:

Parameter	Value
Name	Enter a name for your cluster template.
Description	(Optional) Enter a description for your cluster template.
Cluster Template Source	Select one of: • Text: Paste cluster template in JSON format. • File: Upload a file that contains the cluster template. • URL: Specify the URL for your cluster template. If you use this option, the URL must be accessible from the Management Console.

4. Click Register.

What to do next

To use the uploaded cluster templates, select it when creating a cluster. The option is available on the General Configuration page of the create cluster wizard. First select the Platform Version and then select your chosen cluster template under Cluster Template.

Managing cluster templates from CLI

You can manage cluster templates from CLI using cdp datahub commands.

Required role: Required role: EnvironmentCreator can create a shared resource and then assign users to it. The Owner of the shared resource can delete it.

- Register a new cluster templates by pasting its content: cdp datahub create-cluster-template --cluster-template-name <value> --cluster-template-content <value>
- List all available cluster templates: cdp datahub list-cluster-templates
- Describe a specific cluster templates: cdp datahub describe-cluster-template --cluster-template-name <value>
- Delete one or more existing cluster templates: cdp datahub delete-cluster-templates --cluster-template-name < value>