

Managing Your Database

Date published: 2020-08-14

Date modified: 2024-11-07

CLOUDERA

Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Creating a database using COD.....	5
Configuring a database.....	7
Configuring worker nodes.....	7
Configuring edge nodes.....	7
Upgrading a database using COD.....	8
Performing a Cloudera Runtime upgrade [Technical Preview].....	9
Performing a Cloudera operating system upgrade [Technical Preview].....	11
Rolling upgrade limitations.....	12
Changing database status using COD.....	13
Launching hbase shell.....	14
Configuring strong meta servers.....	15
Importing and restoring data into COD database.....	17
Backing up a COD table.....	18
Scaling COD instances vertically.....	21
HBase REST server scaling in COD.....	21
Configuring the HBase REST server scaling.....	22
Limitations.....	23
Fast autoscaling in COD.....	24
Enabling fast autoscaling in COD.....	26
Disabling fast autoscaling in COD.....	27
Runtime upgrade of the cluster.....	27
Ensure HBase populates pre-existing object storage data.....	28
About custom table coprocessors.....	28
Adding custom coprocessors into HBase tables.....	28

Removing custom coprocessors from HBase tables.....	29
Obtaining a list of coprocessors from a COD environment.....	29
Verifying table coprocessors in HBase tables.....	29
Managing custom images in COD.....	30
Creating a database using a custom image.....	30
Upgrading a database using a custom image.....	30
Switching image catalogs in COD.....	31
Enabling HBase region canary.....	32
Monitoring metrics in COD with Grafana.....	33
Enabling Grafana dashboard in COD.....	33
Enabling Grafana dashboard for an existing COD database.....	36
Importing a Grafana dashboard.....	39

Creating a database using COD

You can create an operational database in your registered CDP environment using the Cloudera Operational Database (COD).

About this task

Required role: You must be logged into the COD as an ODAdmin.

Welcome to Cloudera Data Platform Operational Database

There are no databases available.

Create Database

Before you begin

- Understand the CDP environment and user management. For more information, see *User Management* and *CDP Environments* topics.
- Set up an environment that gives you credential and cloud storage. For more information, see *Before you create an operational database cluster*.
- Ensure that you are authorized to create a database.

Procedure

1. In the COD web interface, click Create Database.
2. Specify the location of the database where you want to store it.
 - a) Provide a name for the database in the Database Name field.
 - b) Select the CDP environment from the list in which you want to associate the database.
 - c) Click Next.

If an environment does not exist, you can create one by clicking Create New Environment.

For more information, see *Register your first environment*.

3. Commission your database by defining a scale for your database using a predefined Data Lake template.

The template helps you to structure your database automatically thereby saving your time and cost. COD creates the predefined number of LITE or HEAVY gateway and master nodes, a set of worker nodes, and also adds additional functionalists into the new database. In case you need to modify the default number of nodes defined in the template, you can do so after the database creation.

The available templates are Micro Duty, Light Duty, and Heavy Duty. By default, Light Duty is selected.

You can create a small database using the Micro Duty template, which consists of one Gateway node and one Worker node. In a Micro database, the Gateway node carries out the processes involved in the Master or Leader nodes. You can consider using a Micro cluster for your testing and development purposes.

4. Configure your database by selecting the storage type as Cloud Storage with Caching, Cloud Storage with Caching and Data Tiering, Cloud Storage, or HDFS.
 - The storage type Cloud Storage with Caching is equivalent to using `--storage-type CLOUD_WITH_EPHEMERAL` option on CDP CLI while creating an operational database.
 - The storage type Cloud Storage, which resembles block storage, is equivalent to using `--storage-type CLOUD` option on CDP CLI while creating an operational database.
 - The storage type HDFS is equivalent to using `--storage-type HDFS` option on CDP CLI while creating an operational database.

By default, Cloud Storage with Caching is selected.

5. Check or update the settings for your database.
 - a) Check all the default settings for your database under the Default tab.
 - b) Go to the Advanced tab if you need to modify any of the default values.
 - The HDFS Volume Type option appears under the Advanced tab only if you select HDFS as the storage type in the Configuration step.
 - If you disable the Autoscaling option using the Advanced tab, the Worker Nodes and Compute Nodes options are hidden. Instead, a Node Count option appears.

The minimum and maximum number of worker nodes vary for different storage types.

- Micro duty: Minimum node count: 1. Maximum node count: 5.
- Light duty: Minimum node count: 3. Maximum node count: 100.
- Heavy duty: Minimum node count: 3. Maximum node count: 800.

6. Review the details before creating the database.

Click Show CLI Command to get the complete command details corresponding to your settings. You can use it to create the database using CDP CLI.

Alternatively, you can use the following sample command to create the database using CDP CLI.

```
cdp opdb create-database --environment-name cod-7218 --database-name test --scale-type LIGHT --storage-type CLOUD_WITH_EPHEMERAL --auto-scaling-parameters '{"minWorkersForDatabase":5, "maxWorkersForDatabase":100}' --num-edge-nodes 0 --java-version 8
```

7. Click Create Database.

Results

An information page is displayed that shows the status of the database. Your new database is ready to be used once its status becomes Available.



Note: Your database starts with a fixed size; however, it scales up and down as the workload applied to the database changes. For more information, see the *Auto Scaling* topic.

Related Information

[COD edge node overview](#)

[COD User Management](#)

[CDP Environments](#)

[Before you create an operational database cluster](#)

[Register your first environment](#)

[COD Auto Scaling](#)

[CDP CLI BETA command reference GitHub repository](#)

Configuring a database

Learn how to configure the properties of an operational database in your Cloudera Public Cloud environment.

After you create an operational database with the specified worker and edge nodes, if you want to change the node counts, you can do so for the existing database, without needing to create a new one.

Configuring worker nodes

Know how to configure an operational database's autoscaling and worker node properties in your Cloudera Public Cloud environment.

About this task

Required role: You must be logged into the Cloudera Operational Database as an OAdmin.

Procedure

1. In the Cloudera Operational Database web interface, select the Databases tab.
2. Find the database you want to configure.
- 3.

In the row of the selected database, click  (Actions) and choose Configure Database.

4. Modify the settings for Autoscaling and Worker Nodes.

Consider these points while setting the worker nodes.

- Ensure that the number of maximum worker nodes is greater than or equal to the minimum number of worker nodes; otherwise, autoscaling is disabled.

If autoscaling is enabled, the operational database automatically adjusts the maximum and minimum number of worker nodes; however, you can set these node counts to remain in the cluster.

- Non-micro clusters must have at least three worker nodes; while micro clusters can have one minimum worker node.
- The supported value range for different clusters is as follows:

Micro: 1 minimum, 5 maximum

Light: 3 minimum, 100 maximum

Heavy: 3 minimum, 800 maximum

5. Click Configure Database.

Results

The new settings are reflected in the existing database immediately after the modification.

Configuring edge nodes

Know how to configure the edge nodes of an existing operational database in your Cloudera Public Cloud environment.

About this task

Required role: You must be logged into the Cloudera Operational Database as an OAdmin.

Procedure

1. In the Cloudera Operational Database web interface, select the Databases tab.
2. Find the database you want to configure.
- 3.

In the row of the selected database, click  (Actions) and choose Configure Edge Nodes.

4. You can go to each tab Add or Delete to add or delete a specific number of edge nodes.
You can add a maximum of 20 edge nodes to the cluster.
5. Click Add Edge Nodes or Delete Edge Nodes accordingly.

Results

The new settings are reflected in the existing database immediately after the modification.

Upgrading a database using COD

Learn how to upgrade your existing Cloudera Operational Database (COD) in the CDP environment.

About this task

- You need to use the CDP Beta CLI and run the upgrade-database command to upgrade your database.
- Required role: You must be logged into the COD as an ODAAdmin.
- To use COD on a GCP environment, you must do it through CDP CLI with --use-hdfs flag.



Important: Before upgrading the Runtime version in your existing COD clusters to 7.2.16 or higher versions, you might need to perform some additional steps before upgrading. For more information, see *Data Lake upgrade validations for Python dependency* under the Management Console what's new topic and *Upgrading Data Hubs*.

Before you begin

- Before you perform the COD upgrade: In the Cloudera Manager properties, increase the `omid_max_heap_size` property for the Omid service to at least 3GB before starting the upgrade from 7.2.9/7.2.10 to 7.2.11:

Omid Max Heapsize

`omid_max_heap_size`

 [omid_max_heap_size](#)

Omid tso server Default Group [Undo](#)

- Understand CDP environment and user management. For more information, see *User Management* and *CDP Environments* topics.
- Ensure you are authorized to upgrade a database.
- You must download and install the latest CDP CLI beta version.
- Cloudera recommends that you run the `prepare-upgrade-database` command before upgrading the cluster. This command performs all validations and downloads all required parcels for the upgrade operation. For more information, see *prepare-upgrade-database*.

Procedure

1. Log in to the CDP CLI tool.

- Run the following command to prepare for the upgrade. This step is optional.

To reduce the chances of upgrade failures, COD supports a preparation phase for the runtime upgrades. During this phase, COD runs all the validations and downloads the required parcels for the machines.

```
cdp opdb prepare-upgrade-database --environment <environment-name> --database
<database-name> --runtime <runtime-version> [OR --image-id <image-id>]
```

Option	Description
--environment (string)	The name or CRN of the environment.
--database <value>	The name or CRN of the database.
--runtime <value>	The runtime version to upgrade to.
--image-id <imageId>	The image ID to upgrade to.

- Run the following command to upgrade the database.

This command upgrades an operational database in an environment to a given Runtime:

```
cdp opdb upgrade-database --environment <environment-name> --database <database-
name> --runtime <runtime-version> [--os-upgrade-only | --no-os-upgrade only]
```

Option	Description
--environment (string)	The name or CRN of the environment.
--database <value>	The name or CRN of the database.
--runtime <value>	The runtime version to upgrade to.
[--os-upgrade-only --no-os-upgrade-only]	Controls whether to perform only an Operating System upgrade.

Related Information

[Data Lake upgrade validations for Python dependency](#)

[Service pack upgrades](#)

[COD User Management](#)

[CDP Environments](#)

[prepare-upgrade-database](#)

[CDP CLI BETA command reference GitHub repository](#)

Performing a Cloudera Runtime upgrade [Technical Preview]

Learn how to perform a rolling and a non-rolling Cloudera Runtime upgrade for your existing Cloudera Operational Database (COD) in the CDP environment.

About this task

- You need to use the CDP beta CLI and run the upgrade-database command to upgrade your database.
- The Cloudera Runtime rolling upgrade is currently supported for COD clusters whose storage is selected as HDFS or cloud without ephemeral storage.
- Zero downtime upgrade or rolling upgrade is not supported on Micro COD clusters.
- It is recommended not to perform backups or run YARN jobs during COD upgrade operations.



Note: Cloudera Runtime upgrade is supported from the CDP Runtime version 7.2.16 onwards.

Before you begin

- You must have the *ODAdmin* rights to make changes to the COD database.
- In the Cloudera Manager, increase the Region Mover Threads property for the HBase service to 30 for a faster rolling upgrade.
- In the Cloudera Manager, increase the Omid Max Heapsize property for the Omid service to at least 3GB.
- You must download and install the latest CDP CLI beta version.

Procedure

1. Launch the CDP CLI tool.
2. Run the following command to check the available Cloudera Runtime upgrades.



Important: Ensure that a Cloudera Runtime upgrade is available from the specified target.

```
cdp datahub upgrade-cluster --cluster-name <cod-internal-name> --show-latest-available-image-per-runtime
```

Following is a sample output where a runtime upgrade is possible from 7.2.17 to 7.2.17.x (hotfix upgrade).

Validate the output that `upgradeCandidates` contains, which highlights a different upgrade target compared to the current one.

```
{
  "current": {
    "imageName": "ami-0b9adbe77c66a2277",
    "imageId": "0df7887b-a841-4569-aab5-269041a015eb",
    "imageCatalogName": "cdp-default",
    "created": 1694041314,
    "componentVersions": {
      "cm": "7.11.0",
      "cmGBN": "42373020",
      "cdp": "7.2.17",
      "cdpGBN": "42350016",
      "os": "centos7",
      "osPatchLevel": "2023-09-06"
    }
  },
  "upgradeCandidates": [
    {
      "imageName": "ami-0ac26feaf1ae4f9e8",
      "imageId": "42e762f0-9731-4056-83b0-b9e7bbe7c5fb",
      "imageCatalogName": "cdp-default",
      "created": 1694777926,
      "componentVersions": {
        "cm": "7.11.0",
        "cmGBN": "44461729",
        "cdp": "7.2.17",
        "cdpGBN": "44441663",
        "os": "centos7",
        "osPatchLevel": "2023-09-15"
      }
    }
  ],
  "reason": ""
}
```

3. Run the following command to perform a rolling Cloudera Runtime upgrade for the database.

The following command upgrades the Cloudera Runtime for the operational database using the rolling restart mode that ensures continuous service availability.

```
cdp opdb upgrade-database --environment <environment-name> --database <database-name> [--runtime <runtime-version> | --image <imageId>] --rolling-upgrade
```

Option	Description
--environment (string)	The name or CRN of the environment.
--database <value>	The name or CRN of the database.
--runtime <value>	The 3-digit runtime version to upgrade to. Alternatively, specify the --image option.
--image <value>	The image ID to upgrade to. Alternatively, specify the --runtime option.
--rolling-upgrade	Controls whether to perform a rolling upgrade for the COD.

Run the following command to perform a non-rolling Cloudera Runtime upgrade for the database.

```
cdp opdb upgrade-database --environment <environment-name> --database <database-name> [--runtime <runtime-version> | --image <imageId>]
```

Results

COD is upgraded to the provided runtime version in a rolling mode.

Related Information

[Upgrading Cloudera Operational Database clusters](#)

[COD User Management](#)

[CDP Environments](#)

[COD CLI command reference GitHub repository](#)

[CDP CLI BETA command reference GitHub repository](#)

Performing a Cloudera operating system upgrade [Technical Preview]

Learn how to perform a rolling and a non-rolling Operating System (OS) upgrade for your existing Cloudera Operational Database (COD) in the CDP environment.

About this task

- You need to use the CDP Beta CLI and run the `upgrade-database` command to upgrade your database.
- The operating system rolling upgrade is currently supported for COD clusters whose storage is selected as HDFS or cloud without ephemeral storage.
- Zero downtime upgrade or rolling upgrade is not supported on Micro COD clusters.
- It is recommended not to perform backups or run YARN jobs during COD upgrade operations.



Note: Cloudera operating system upgrade is supported from the CDP Runtime version 7.2.16 onwards.

Before you begin

- You must have the *ODAdmin* rights to make changes to the COD database.
- In the Cloudera Manager, increase the Region Mover Threads property for the HBase service to 30 for a faster rolling upgrade.

- In the Cloudera Manager, increase the Omid Max Heapsize property for the Omid service to at least 3GB.
- You must download and install the latest CDP CLI beta version.

Procedure

1. Launch the CDP CLI tool.
2. Run the following command to perform a rolling OS upgrade for the database.

This command upgrades the OS for the operational database using the rolling restart mode that ensures continuous service availability.

```
cdp opdb upgrade-database --environment <environment-name> --database <database-name> [--runtime <runtime-version> | --image <imageId>] --os-upgrade-only --rolling-upgrade
```

Option	Description
--environment (string)	The name or CRN of the environment.
--database <value>	The name or CRN of the database.
--os-upgrade-only	Requests OS upgrade.
--runtime <value>	The 3-digit runtime version to upgrade to. Alternatively, specify the --image option.
--image <value>	The image ID to upgrade to. Alternatively, specify the --runtime option.
--rolling-upgrade	Controls whether to perform a rolling upgrade for the COD.
--edge-upgrade-strategy <type>	Controls the upgrade strategy for the edge nodes in the cluster. Following are the supported values: <ul style="list-style-type: none"> • ALL: Upgrades all the edge nodes together. • ONE_BY_ONE: Upgrades the edge nodes, one by one.

Run the following command to perform a non-rolling OS upgrade for the database.

```
cdp opdb upgrade-database --environment <environment-name> --database <database-name> [--runtime <runtime-version> | --image <imageId>] --os-upgrade-only
```

Results

COD is upgraded to the provided operating system version in a rolling mode.

Related Information

[Upgrading Cloudera Operational Database clusters](#)

[COD User Management](#)

[CDP Environments](#)

[COD CLI command reference GitHub repository](#)

[CDP CLI BETA command reference GitHub repository](#)

Rolling upgrade limitations

Consider the following points while performing a rolling runtime and an operating system upgrade of a COD cluster.

Operating system upgrade limitations

- An unwanted rolling OS upgrade can be triggered if a runtime upgrade is not available, resulting in COD downtime. Ensure that the runtime upgrade is available for the selected runtime using the `cdp opdb describe-upgrade-database` command before using the following command.

```
cdp opdb upgrade-database --environment <environment-name> --database <database-name> --runtime <runtime-version> --rolling-upgrade
```
- COD rolling upgrade is focused on HBase and Phoenix thick clients excluding Phoenix Query Servers (PQS) because PQS connectivity depends on Knox which is not deployed in HA mode. When Knox is unavailable during the upgrade, PQS, HBase REST and other endpoints connected through Knox can be interrupted.
- The Cloudera Runtime and operating system rolling upgrades are currently supported for COD clusters whose storage is selected as HDFS or cloud without ephemeral storage.

Changing database status using COD

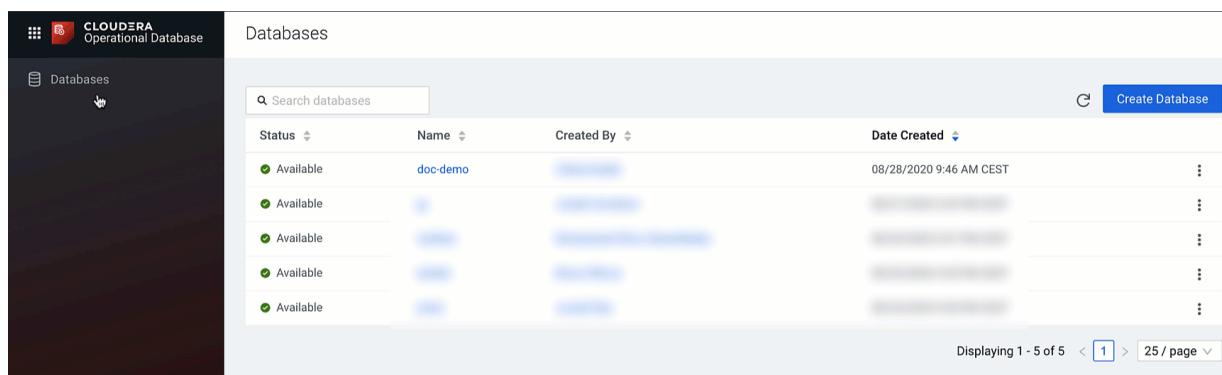
After you create an operational database, you can start, stop, or drop the database using the Cloudera Operational Database (COD) user interface.

About this task

Required role: You must be logged into the COD as an ODAdmin

Procedure

- In the COD web interface, select the Databases tab.
- Find the database you want to manage.
- In the row of the selected database, click Actions and select one of the following actions:
 - Start Database - Starting the database. When you create a new database, it is initially started by default.
 - Stop Database - Stopping the database and the underlying instances.
 - Drop Database - Dropping the database and deleting all the data that is stored in that database. However, your SDX Data Lake continues to exist and you can create another database using the same Data Lake.



Related Information

[Start database](#)

[Start database-Beta](#)

[Stop database](#)

[Start database-Beta](#)

[Drop database](#)

[Drop database-Beta](#)

Launching hbase shell

To launch the *hbase shell* and use it with Cloudera Operational Database (COD), you have to create an edge node with a configured HBase client tarball.

Before you begin

Configure an edge node. For more information, see *COD edge node overview*.

Procedure

1. In your COD web interface navigate to **Connect HBase Client Tarball**.
2. Add your Hbase Client Tarball and HBase Client Configuration.

- a) Download hbase-client-tarball.tar.gz and extract it to a location on the edge node. This is your *"HBASE_HOME"*.
 - b) Download the client configuration using the curl command.
 - c) Extract the client configuration zip file, and move the contained hbase-conf directory to *"HBASE_HOME"* with the name conf (instead of hbase-conf).
3. Launch the hbase shell.

```
$ kinit [***cdp_workload_user***]
Password: *****
$ export HBASE_HOME=hbase-2...
$ cd $HBASE_HOME
$ ./bin/hbase shell
```

4. Validate you can connect through hbase shell.
Use the list command in hbase shell to list all tables.

5. Add yarn configuration to hbase configuration.

YARN Configuration

Additional configuration required to submit YARN application.

YARN Archive

```
curl -o "yarn-config.zip" -u "csso_..." "https://cod-127zll9v4tng7-gateway0.dmx-cod.xcu2-8y8x.dev.cldr.work/clouderamanager/api/v41/clusters/cod-127zll9v4t"
```

TLS Truststore

```
curl -o "cod-truststore.jks" -u "csso_..." "https://cod-127zll9v4tng7-gateway0.dmx-cod.xcu2-8y8x.dev.cldr.work/clouderamanager/api/v41/certs/truststore?type"
```

TLS Truststore Password

```
curl -u "csso_..." "https://cod-127zll9v4tng7-gateway0.dmx-cod.xcu2-8y8x.dev.cldr.work/clouderamanager/api/v41/certs/truststorePassword"
```

- Navigate to `Connect HBase Client Tarball` and expand the `YARN Configuration` section..
- Download the `yarn-config.zip` file.
- Add the files from `yarn-conf` to `"HBASE_HOME"/conf`.
- Download the TLS truststore and remember where you place it.
- Update the `ssl-client.xml` file with the TLS TRuststore details.
- Ensure that in the `ssl-client.xml` file the `ssl.client.truststore.location` points to the `cod-truststore.jks` you have downloaded in step d.
- Ensure that the `ssl.client.truststore.password` matches the TLS Truststore Password.



Note: If you plan to use the `hbck2` command on COD client side, set the `HBASE_CLASSPATH` environment variable pointing to the HBase client tarball lib directory as follows:

```
export HBASE_CLASSPATH=/HBase\_client\_tarball\_location/hbase-client-  
tarball/lib/:
```

Results

You have an edge node with configured HBase client tarball and you can launch the hbase shell.

Related Information

COD edge node overview

Configuring strong meta servers

The strong meta feature uses an HBase RegionServerGroup (RSGroup) to provide dedicated servers to the HBase system tables. This prevents customer table regions from being hosted on the same RegionServers as the system regions and prevents the load from these custom tables from affecting system table access and initialization.

Before you begin

- You must have the `COD_STRONG_META_SERVERS` entitlement to use this feature.
- This feature is only supported on AWS environments.
- By default, meta replication is enabled and the replica number is set to 3. Therefore, ensure that each strong meta node contains at least 3 meta regions. This does not require any manual steps.

Procedure

1. Launch the CDP CLI tool.

2. Run the following command to add the desired number of strong meta servers.

```
cdp opdb update-database --environment-name envName --database-name dbName
--num-desired-strong-meta-servers 3
```



Note: The minimum number of strong meta servers to be added is 3 because the meta replication is already enabled in COD and the minimum number of supported meta regions is 3.

The above command creates 3 strong meta nodes, adds them to the cluster, and puts a region server on them.

3. To verify if the nodes are healthy, ensure that the logs on these nodes have no errors or exceptions. If data on the cluster and balancer is enabled, these nodes must serve random regions because the balancer moves regions there.
4. Ensure that the RSGroup feature is enabled on the COD cluster. To enable this, use the `list_rsgroups` command in the `hbase` shell.

If the command is not enabled, the following error is displayed.

```
org.apache.hadoop.hbase.exceptions.UnknownProtocolException: No registered
Master Coprocessor Endpoint found for RSGroupAdminService. Has it been en
abled?
```

- a) To enable the RSGroup feature, check whether `hbase.master.loadbalancer.class` property is configured. If configured, move its value to `hbase.rsgroup.grouploadbalancer.class` property.
 - b) Set the value of `hbase.master.loadbalancer.class` property to `org.apache.hadoop.hbase.rsgroup.RSGroupBasedLoadBalancer` in the `hbase-site.xml` file.
 - c) Set the value of `hbase.coprocessor.master.classes` property to `org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint` in the `hbase-site.xml` file.
 - d) Restart the master nodes for these changes to take effect.
 - e) To verify if the setup was successful, use the `list_rsgroups` command in the `hbase` shell.
5. Create a RegionServerGroup and add the strong meta nodes to it. Ensure that the system tables are also added. Execute the following steps in an `hbase` shell.

```
// verify that the feature works and every worker/strongmeta nodes are v
isible
list_rsgroups
//create the group (please note, that a different name can be also used)
add_rsgroup "strongmeta"
//verify that the group is created
list_rsgroups
//move the strongmeta nodes to the group (node names can be copied from
the list_rsgroups output)
move_servers_rsgroup "strongmeta",["cod--xxx-strongmeta0.cod-7216.xcu2-8y8
x.dev.cldr.work:16020","cod--xxx-strongmeta1.cod-7216.xcu2-8y8x.dev.cldr
.work:16020","cod--xxx-strongmeta2.cod-7216.xcu2-8y8x.dev.cldr.work:1602
0"]
//verify that the nodes are added to the group
list_rsgroups
//move the system tables to the group
move_tables_rsgroup "strongmeta",["hbase:meta"]
move_tables_rsgroup "strongmeta",["hbase:ack"]
move_tables_rsgroup "strongmeta",["hbase:namespace"]
move_tables_rsgroup "strongmeta",["hbase:rsgroup"]
//verify that the tables were moved
list_rsgroups
```

6. If the previous steps are successful, ensure that everything is functioning as expected using the HBase UI.

What to do next

Ensure that each strong meta node now only serves system table regions.

Importing and restoring data into COD database

You can import your data into your Cloudera Operational Database (COD) database by restoring your HBase table into COD.

Before you begin

- Enable HBase replication on your COD cluster. For more information, see *Cloudera Operational Database data replication*.
- Have a location in cloud storage (for example, S3 or ABFS) with an exported snapshot in it, and have the name of the snapshot.

If you do not already have an exported HBase snapshot, you can export your data to cloud storage using the following command:

```
hbase org.apache.hadoop.hbase.snapshot.ExportSnapshot -snaps
hot [***SNAPSHOT NAME***] -copy-to [***CLOUD STORAGE LOCATION***] -mappers
10
```

For example, the data-from-onprem snapshot can be exported into s3a://cod-external-bucket/hbase:

```
hbase org.apache.hadoop.hbase.snapshot.ExportSnapshot -snapshot data-from-
onprem -copy-to s3a://cod-external-bucket/hbase -mappers 10
```

- Have an edge node with a configured HBase client tarball and know how to launch the hbase shell from it. For more information, see *Launching HBase shell*.

Procedure

1. Get your CLOUD STORAGE LOCATION for your COD database using the COD web interface.
s3a://my_cod_bucket/cod-12345/hbase

Databases / doc-test

The screenshot shows the COD web interface for a database named 'doc-test'. At the top, it indicates the database is 'Available - Updated just now'. Below this, the database ID is shown as 'crm:cdp:opdb:us-west-1:9d74eee4-1cad-45d7-b645-7ccf9edbb73d:opDb:8bbef7dd-341f-44b8-acdd-95fb5e5cd953'. The 'VERSION' is '1.10.0' and the 'CREATED BY' field is blurred. At the bottom, there is a table of configuration details:

ENVIRONMENT	REGION	DATA LAKE	SQL EDITOR	CLOUD STORAGE LOCATION
cod-727-newsubnets	us-west-2	cod-727-newsubnets	Hue	s3a://cod-727-mowdev/cod-9zouq3ua3qqz/hbase

The 'CLOUD STORAGE LOCATION' field is highlighted with an orange box in the original image.

2. Add your bucket to the IAM policy used by IDBroker.

For more information, see one of the following documentation:

- [AWS Environments: Minimal setup for cloud storage](#)
- [Azure Environment: Minimal setup for cloud storage](#)

3. Launch the hbase shell from the edge node.

For more information, see *Launching HBase shell*.

4. From the edge node, run the ExportSnapshot command. Use the external bucket as the source location and the COD cloud storage as the target.

For example:

```
$ cd $HBASE_HOME
$ ./bin/hbase org.apache.hadoop.hbase.snapshot.ExportSnapshot -snapshot
"data-from-onprem" --copy-from s3a://cod-external-bucket/hbase --copy-to
s3a://my_cod_bucket/cod-12345/hbase
```

5. Use the list_snapshots command and verify that your snapshot is listed.

```
$ cd $HBASE_HOME
$ ./bin/hbase shell
$ hbase> list_snapshots
[ 'snapshot_name' ]
```

6. Use the restore_snapshot or the clone_snapshot command to reconstitute the table.



Warning: The restore_snapshot command overwrites an existing table.

- restore_snapshot: Overwrites current table state with that of the snapshot. It means that any data modification applied after the snapshot was taken would be lost. If the table does not exist in the given cluster, the command automatically creates it.
- clone_snapshot: Accepts a new table name for the table in which it restores the table schema and data.

7. Validate that all rows are present in the table using the hbase rowcounter or count command in the hbase shell.

Related Information

[Launching hbase shell](#)

[Cloudera Operational Database data replication](#)

[Using the CldrCopyTable utility to copy data](#)

Backing up a COD table

You can backup your Cloudera Operational Database (COD) table by creating a snapshot of the table and exporting it to your S3, HDFS, or ABFS bucket.

About this task

Create snapshot command takes a snapshot of the HBase table, exports the snapshot to the specified storage location, and deletes the snapshot from HBase after the export to the specified storage location is successful.

Before you begin

- Ensure that you have access to the location where you want to take the backup of the table.
- You have an S3 or ABFS bucket which you want to use to backup an HBase table in COD. For example: s3a://cod-backups/hbase
- You have an edge node configured for your COD database. For more information, see *Configure Edge Nodes*.

Procedure

1. Run the following command using `cdplici-beta`.

```
cdp opdb create-snapshot --environment-name ENVIRONMENT_NAME --database-name DATABASE_NAME --table-name TABLE_NAME --snapshot-name SNAPSHOT_NAME --snapshot-location SNAPSHOT_LOCATION
```

For example,

```
cdp opdb create-snapshot --environment-name odx-wgel6j --database-name jpfy --table-name table_6uf7xt --snapshot-name s1 --snapshot-location s3a://cod-7215/cod--yq3p370r6qro/backupdata
```

JSON output:

```
{
  "environmentName": "odx-wgel6j",
  "databaseName": "jpfy",
  "status": "IN_PROGRESS",
  "creationTime": 0,
  "commandID": 1546336350
}
```

2. Validate that the snapshot and data is present using the `list-snapshots` command.

The following command lists the snapshots created in the environment against the COD database filters like table name, command ID, and time range.

```
cdp opdb list-snapshots --environment-name ENVIRONMENT_NAME --database-name DATABASE_NAME [--table-name TABLE_NAME] [--command-id COMMAND_ID] [--from-creation-time FROM_CREATION_TIME] [--to-creation-time TO_CREATION_TIME]
```

Examples

Without filters

```
cdp opdb list-snapshots --environment-name odx-wgel6j --database-name jpfy
{
  "environmentName": "odx-wgel6j",
  "databaseName": "jpfy",
  "snapshots": [
    {
      "tableName": "table_tob8vc",
      "snapshotName": "snapshot3_table_tob8vc",
      "creationTime": 1662014265011,
      "status": "SUCCESSFUL",
      "commandID": 1546336058,
      "snapshotLocation": "abfs://qedailynat-filesystem@qedailynatstorageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase"
    },
    {
      "tableName": "table_tob8vc",
      "snapshotName": "snapshot2_table_tob8vc",
      "creationTime": 1662014023725,
      "status": "SUCCESSFUL",
      "commandID": 1546336035,
      "snapshotLocation": "abfs://qedailynat-filesystem@qedailynatstorageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase"
    },
    {
      "tableName": "table_tob8vc",
      "snapshotName": "snapshot1_table_tob8vc",
      "creationTime": 1662013782370,
```

```

        "status": "DELETED",
        "commandID": 1546336077,
        "snapshotLocation": "abfs://qedailynat-filessystem@qedailyna
tstorageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase"
      },
      {
        "tableName": "table_lpirwy",
        "snapshotName": "snapshot_table_lpirwy",
        "creationTime": 1662013568312,
        "status": "DELETED",
        "commandID": 1546336004,
        "snapshotLocation": "abfs://qedailynat-filessystem@qedailyna
tstorageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase"
      }
    ]
  }
}

```

With command ID filter

```

$ cdp opdb list-snapshots --environment-name odx-wgel6j --database-name
jpfy --command-id 1546336385
{
  "environmentName": "odx-wgel6j",
  "databaseName": "jpfy",
  "snapshots": [
    {
      "tableName": "table_6uf7xt",
      "snapshotName": "s2",
      "creationTime": 1662035967244,
      "status": "SUCCESSFUL",
      "commandID": 1546336385,
      "snapshotLocation": "abfs://qedailynat-filessystem@qedailyna
tstorageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase/"
    }
  ]
}

```

With table name filter

```

$ cdp opdb list-snapshots --environment-name odx-wgel6j --database-name
jpfy --table-name table_6uf7xt
{
  "environmentName": "odx-wgel6j",
  "databaseName": "jpfy",
  "snapshots": [
    {
      "tableName": "table_6uf7xt",
      "snapshotName": "s2",
      "creationTime": 0,
      "status": "SUCCESSFUL",
      "commandID": 1546336385,
      "snapshotLocation": "abfs://qedailynat-filessystem@qedailynats
torageaccount.dfs.core.windows.net/cod-rtzupxh843ua/hbase/"
    }
  ]
}

```

Related Information

[Launching hbase shell](#)

[Create snapshot](#)

[List snapshot](#)

Scaling COD instances vertically

You can scale up a COD cluster vertically, which means upgrading a COD cluster from a LIGHT duty to a HEAVY duty instance type. You can only upgrade COD clusters belonging to the Gateway and Master nodes.

About this task

Selecting a larger instance type adds more vCPU and/or RAM to your instances. You can scale up instances from LIGHT to HEAVY using the CDP CLI command tool.

Before you begin

- You must stop the COD cluster before you vertically scale any of the instances.
- You must grant CDP_CB_AWS_VERTICAL_SCALE entitlement for the specific tenant.
- You must stop EBS-backed instances before scaling. Vertical scaling is supported on AWS only. See *Change the instance type* in AWS documentation for more information.
- You must ensure that the COD clusters belong only to Gateway nodes or Master nodes.
- You must download and install the latest CDP CLI beta version. For more information, see *Installing Beta CDP CLI*.

Procedure

1. Log in to the CDP CLI command tool.
2. Run the following command.

```
cdp opdb update-database --environment <environment_name> --database  
<database_name> --vertical-scale <Group_type_for_the_database_nodes>
```

The following are the possible values for the group type for the database nodes:

- GATEWAY: Value of the group variable as GATEWAY.
- MASTER: Value of the group variable as MASTER.

For example,

```
cdp opdb update-database --environment-name cod-7216-micro10 --database-  
name odx2408 --vertical-scale MASTER
```

What to do next

Start the cluster after you have vertically scaled it. You can also configure the services on the cluster to use the additional or reduced resources/memory.

Related Information

[Installing Beta CDP CLI](#)

HBase REST server scaling in COD

Using Apache HBase REST API, you can scale up the HBase REST server for better connectivity to COD.



Important: To use this feature in your CDP environment, you must have the COD_RESTWORKERS entitlement enabled. Contact Cloudera Support or your Cloudera account team if you do not have this entitlement.

Multiple Knox Gateway servers are required to scale up the HBase REST servers and support the increased load, so you must have multiple gateway nodes. When you define multiple gateway nodes, the Apache Knox Gateway instances work in an HA mode, and the load is balanced among the multiple Knox instances. Each gateway node hosts an Apache Knox Gateway instance that provides access to the HBase REST server hosted on a REST worker node. The maximum number of gateway nodes supported is 10.

You can specify the required number of REST worker nodes using the `--restworker-nodes-count` option in the `create-database` command. This optional parameter can only be defined when you specify the `--gateway-nodes-count` option. Using the `update-database` command, you can scale up the number of the REST worker nodes after database creation. The default number of REST worker nodes supported is 0.

The recommended COD deployment type is Single-AZ to leverage the scaling of HBase REST servers.

Benefits of using a REST server

- When a direct line of sight between the client and all HBase servers cannot be established, you can connect the client to a COD-deployed load balancer backed by Knox and REST servers, allowing seamless connectivity with COD.
- This setup can leverage the HTTP authentication proxy, HTTP - HTTPS proxy, and OAuth proxy features of Knox. For example, Knox can act as an HTTP authentication proxy to support JWT for REST servers, with the flexibility to integrate other authentication providers as needed.
- This setup is language-independent, which means you can use any client, regardless of the programming language, to interact with COD.

Configuring the HBase REST server scaling

Know how to configure an HBase REST server scaling.

About this task

When you configure an HBase REST server:

- The heap of Knox is increased to 31 GiB on HEAVY and 2 GiB on LIGHT scale COD clusters respectively.
- The Knox parameter `gateway.httpserver.requestHeaderBuffer` is set to 1 MiB as a safety valve configuration in Cloudera Manager. For more information on this parameter, see the *Apache Knox documentation*.
- The heap size of the HBase REST server is set to 16 GiB on REST worker nodes.



Note: Cloudera recommends exercising caution while using the gateway nodes because these instance types are expensive. If you wish to save costs, use them judiciously.

Before you begin

- To utilize this functionality, you need at least two gateway nodes. The required number of gateway nodes can be specified using the `--gateway-nodes-count` option in the `create-database` command.
- To enable this functionality, you must create at least two gateway nodes.
- You must have the OAdmin rights to make changes to the COD database.
- You must download and install the latest CDP CLI beta version. For more information, see *Installing Beta CDP CLI*.
- Cloudera recommends using Java 11 to create the clusters. Java 8 has a slow TLS implementation, which significantly reduces the HTTPS throughput of both the REST server and Knox.

Procedure

1. Launch the CDP CLI tool.

2. Create the database with the desired scale type and number of gateway nodes using the `--gateway-nodes-count` option.



Important: In the case of the HEAVY scale type, the Gateway node is order of magnitude stronger than that of the LIGHT scale type (32 vCPU and 256 GiB memory vs 8 vCPU 32 GiB memory), and the allocated heap for the hosted Knox is 31 GiB. Thus the gateway node can handle a much higher load with fewer node numbers. You must consider this while planning the number of gateway nodes.

```
cdp opdb create-database --environment-name <env_name> --database-name
<database_name> --scale-type HEAVY --gateway-nodes-count integer --disable-multi-az --java-version
11
```

Optionally, you can specify the number of REST worker nodes using the `--restworker-nodes-count` option.

```
cdp opdb create-database --environment-name <env_name> --database-name
<database_name> --gateway-nodes-count <integer> --restworker-nodes-count <integer> --disable-
multi-az --java-version 11
```

What to do next

If you prefer to scale up or down an HBase REST server, use the following `update-database` command.



Note: The REST server is CPU-bound, if the CPU utilization is too high, you must increase the number of nodes.

```
cdp opdb update-database --environment-name <env_name> --database-name
<database_name> --num-desired-restworker-nodes <integer>
```

For example,

```
cdp opdb update-database --environment-name cod-env --database-name cod-db -
-num-desired-restworker-nodes 4
```

Use the following command to scale down an HBase REST server to zero instances.

```
cdp opdb update-database --environment-name <env_name> --database-name
<database_name> --remove-restworker-nodes
```

Related Information

[Installing Beta CDP CLI](#)

[Apache Knox documentation](#)

Limitations

Know the limitations and known issues while scaling an HBase REST server.

Limitations

- The number of gateway nodes cannot be modified after the database is created, so plan the number of requested gateway nodes based on the estimated load.
- You can scale the HBase REST servers in COD clusters deployed in Amazon Web Services (AWS) environments.
- The network address translation (NAT) could interfere with traffic distribution because the current setup relies on a Network Load Balancer for sticky sessions, which may cause all requests to route to a single Knox server. Future COD versions plan to use an Application Load Balancer for more effective distribution based on session cookies. Until then, NAT must be avoided if more than one Knox or REST server is in use.
- Knox nodes must be specified at the time of cluster creation, as scaling Knox nodes after deployment is currently not supported.
- A Multi-AZ setup can increase data transfer costs, as data might need to pass through multiple services, such as Knox, REST, and HBase region servers across different availability zones before reaching the client.

- HBase Client Tarball downloads might fail if Knox re-routes the request to the wrong gateway.
- The MICRO scale type does not support multiple Knox instances and scaling of HBase REST servers.
- Changing the Knox Provider Configurations and Descriptors using the Knox Admin UI only changes the provider configurations, descriptors, and the generated topologies of the Knox instance whose Admin UI is used.

Known issues

LB based client configs and tarball URLs in the describe-client-connectivity command output fails intermittently

Due to a known Cloudbreak issue, CB-26030, downloading an HBase client configuration intermittently fails when using multiple Gateway nodes.

Retry the download operation until it succeeds.

describe-database command returns incorrect output.

After creating the COD cluster with the required number of gateway and REST worker nodes, when you run the describe-database command, the output always shows 0 number of gateway and REST worker nodes.

Upgrade to the latest COD version.

Rolling operating system upgrade does not function correctly

When you try to perform a rolling upgrade on the COD cluster, it does not work with multiple gateway and REST worker nodes.

Upgrade to the latest COD version.

Fast autoscaling in COD

Cloudera Operational Database (COD) supports fast autoscaling for higher computing requirements. COD enables fast autoscaling whenever high CPU utilization or higher RPC latency is observed in the system.

The fast autoscaling is achieved using the following mechanisms.

Introduction to Compute Instance Group

A Compute node instance group in COD consists of nodes that host all the services, similar to those on Worker nodes, except the HDFS data services. The region servers on these nodes communicate with HDFS (data nodes) running on the Worker nodes.

The Compute nodes add to the higher compute capacity in the system. During scale-up operations due to higher compute requirements, the Compute nodes are scaled up using the same storage as on the existing Worker nodes. Hence, during the scale-down of these Compute nodes, the data remains in the Worker nodes and does not require data redistribution. The Worker nodes continue to host the data services and are scaled (up or down) according to the data requirements (such as Data storage and Regions per server).

Figure 1: COD Architecture



Metrics categorization

The metrics used for autoscaling are categorized into the following groups. The data metrics are used for the scaling of the Worker instance group while the compute metrics are used for the scaling of the Compute instance group.

- Worker Group (Data-related metrics)
 1. HDFS usage
 2. Region density (Regions per server)
 3. Total Store File size across region servers
- Compute Group (Compute-related metrics)
 1. CPU utilization
 2. RPC latency

Fast autoscaling mechanism

COD relies on a mechanism of suspending and resuming (start and stop) autoscaling, where the maximum allowed number of nodes in the system are instantiated but kept in a Stopped state. This mechanism is used for the scaling of the Compute instance group.

When a requirement for higher compute power arises in the cluster, COD starts the required number of Compute nodes. The scale-up time is drastically reduced because the Compute nodes are already initialized. Similarly, when the requirement for computing goes down, the Compute nodes are stopped but not dropped. They remain in the cluster in the Stopped state. The scale-up and scale-down time is reduced since the instance initialization is skipped during every scale-up or scale-down operation.

COD supports the suspending and resuming of fast autoscaling for the Compute nodes only. The Worker nodes continue to use the existing autoscaling mechanism.

Related Information

[Autoscaling clusters](#)

Enabling fast autoscaling in COD

Learn how to enable the fast autoscaling mechanism in COD.

Before you begin

- The cluster must have the COD setup in the environment.
- The node on which the steps are executed must have the CDP CLI installed.
- You must have the COD_USE_COMPUTE_ONLY_NODES entitlement.
- You must use COD 1.37.0 or a higher version.

Procedure

1. Launch the CDP CLI tool.
2. Use the CDP CLI `create-database` command to create the compute instance group.

Use the `--auto-scaling-parameters` option to specify the maximum and minimum number of compute nodes.

```
cdp opdb create-database --environment-name <env_name> --database-name <db_name> --auto-scaling-parameters '{"minComputeNodesForDatabase":<min_compute_nodes>,"maxComputeNodesForDatabase":<max_compute_nodes>}'
```

Where,

- `<max_compute_nodes>` specifies the maximum number of Compute nodes required in the system.
- `<min_compute_nodes>` specifies the minimum number of Compute nodes in the system.

When the command is successfully executed, the newly created database contains the instantiated `<max_compute_nodes>` nodes. However, in the subsequent, scaling-executor cycle, the Compute nodes are stopped until the required number of nodes or the count `<min_compute_nodes>` is reached.

Alternatively, if the database is already created on 1.37.0, we can use the update database CDP CLI command to update the autoscaling parameters mentioned above.

3. Configure the autoscaling parameter to have an appropriate `maxCpuUtilization` which represents the average CPU utilization percent across Worker nodes and Compute nodes.

```
cdp opdb update-database --environment-name <env_name> --database-name <db_name> \
--auto-scaling-parameters '{"maxCpuUtilization": 80. . . }'
```

The subsequent scaling cycles start or stop the Compute nodes as per the compute requirements.

4. Execute the `update-database` CDP CLI command to start all the Compute nodes in the system, if the cluster needs to be stopped or upgraded.

```
cdp opdb update-database --environment-name <env_name> --database-name <db_name> \
--auto-scaling-parameters '{"minComputeNodesForDatabase":<max_compute_nodes>}'
```

All the Compute nodes are started. The nodes are running after the cluster upgrade or restart.

5. Revert to the original `<min_compute_node>` after the restart or upgrade.

```
cdp opdb update-database --environment-name <env_name> --database-name <db_name> \
--auto-scaling-parameters '{"minComputeNodesForDatabase":<min_compute_nodes>}'
```

The subsequent scaling cycles stop the additional Compute nodes in the cluster.

Related Information

[CDP CLI Beta](#)

Disabling fast autoscaling in COD

You can disable the fast autoscaling in your COD cluster. However, the existing autoscaling mechanism operates as usual in the cluster.

Before you begin

- The cluster must have the COD setup in the environment.
- The node on which the steps are executed must have the CDP CLI installed.
- Fast autoscaling must be enabled in your cluster.

Procedure

1. Launch the CDP CLI tool.
2. Set the minimum and maximum Compute node counts to zero. This command deletes all the Compute nodes in the cluster.

```
cdp opdb create-database --environment-name <env_name> --database-name <db_name> \
--auto-scaling-parameters '{"minComputeNodesForDatabase":0, "maxComputeNodesForDatabase": 0}'
```

The subsequent autoscaling executions only impact the Worker instance group.

Related Information

[CDP CLI Beta](#)

Runtime upgrade of the cluster

Learn how to resume the stopped nodes in the cluster before performing a runtime upgrade.

The cluster upgrade might fail if the cluster contains nodes that are in the Stopped state. You must restart these nodes before executing the upgrade or drop the stopped nodes before the upgrade. They can be added later after the upgrade.

Ensure that you execute the following command to increase the minimum number of nodes in the cluster to the maximum number of allowed Compute nodes in the cluster. This triggers the restart of the stopped nodes. After that, the cluster can be upgraded.

```
./clients/cdpcli/cdp.sh opdb update-database --environment-name jrh16-cod-7216 \
--database-name jrh13-7216 \
--auto-scaling-parameters '{"minComputeNodesForDatabase":<MAX_COMPUTE_NODES>}'
```

Revert the `minComputeNodesForDatabase` to the original value after the upgrade.

Alternatively, drop all the stopped Compute nodes in the cluster using the following CDP CLI command.

```
cdp opdb create-database --environment-name <env_name> --database-name <db_name> --auto-scaling-parameters '{"maxComputeNodesForDatabase": <number_of_running_compute_nodes_in_the_system>}'
```

Related Information

[CDP CLI Beta](#)

Ensure HBase populates pre-existing object storage data

Cloudera Operational Database (COD) databases store data in cloud storage. The location of the data is stored in cloud storage is defined by the data access cloud storage location in the CDP environment and the COD database name .

When a COD database is dropped, any data stored in the cloud storage is not dropped. You must delete the data separately if you no longer need it. If a COD database is created against a cloud storage location which already contains COD data from a previous instance of COD, COD will automatically re-create the tables with the corresponding data in the new COD instance. Multiple COD databases must not refer to the same cloud storage location as this will cause data loss and data corruption.

If a COD database is created using the same name against a cloud data access storage location, which already contains COD data from a previous instance of COD, COD will automatically re-create the tables with the corresponding data in the new COD instance.

About custom table coprocessors

You can add table coprocessors so that HBase can run custom code on the server side against the stored data and filter local minimum or maximum value during ingestion without scanning the entire table.

You can use built-in table coprocessors from the upstream HBase releases. COD supports custom table coprocessors, which you can implement and extend from HBase coprocessors' interfaces.

You can add custom table coprocessors into the HBase tables or remove them from HBase tables, also obtain a list of coprocessors with its status on the COD environment.

You must ensure that:

- You have the CDP CLI setup.
- You have uploaded the JAR file (custom coprocessor) to a remote location that the COD instance can access.



Important: This feature is in Technical Preview and is not ready for production deployment. Cloudera encourages you to explore these technical preview features in non-production environments and provide feedback on your experiences.

Related Information

[Coprocessor Introduction](#)

Adding custom coprocessors into HBase tables

Learn how to add custom table coprocessors into the HBase table.

Procedure

1. Launch the CDP CLI tool.

2. Upload your custom coprocessor to a remote location. For example, S3.
3. Run the following `add-coprocessor` command to add a specific coprocessor using its canonical class name.

```
# add table-level coprocessor
cdp opdb add-coprocessor \
--environment ENVIRONMENT \
--database DATABASE \
--table-name TABLE_NAME \
--coprocessor-canonical-name COPROCESSOR_CANONICAL_NAME \
[--coprocessor-location-url COPROCESSOR_LOCATION_URL] \
[--coprocessor-args COPROCESSOR_ARGS]
```

Removing custom coprocessors from HBase tables

Learn how to remove the custom coprocessors from the HBase tables.

Procedure

1. Launch the CDP CLI tool.
2. Run the following `remove-coprocessor` command to remove a specific coprocessor using its canonical class name.

```
# remove table-level coprocessor
cdp opdb remove-coprocessor \
--environment ENVIRONMENT \
--database DATABASE \
--table-name TABLE_NAME \
--coprocessor-canonical-name COPROCESSOR_CANONICAL_NAME \
[--force] \
[--no-force]
```

Obtaining a list of coprocessors from a COD environment

Learn how to obtain a list of coprocessors from a COD environment.

Procedure

1. Launch the CDP CLI tool.
2. Run the following `list-coprocessors` command to obtain a list of all the added coprocessors.

```
# Get a list of coprocessors with its status on the COD environment
cdp opdb list-coprocessors \
--environment ENVIRONMENT \
--database DATABASE \
[--table-name TABLE_NAME] \
[--command-id COMMAND_ID]
```

Verifying table coprocessors in HBase tables

After you add or remove the table coprocessors through CDP CLI, you can verify whether they are added successfully.

Procedure

1. Open the HBase UI from the Cloudera Manager.

2. Click User Tables under Tables.
3. Check the coprocessors details in the Description column for the respective HBase tables.

Managing custom images in COD

You can customize a default image for compliance or security reasons. You can then use the CDP CLI to register a custom image catalog and set the custom image within the custom image catalog. Later, you can use this custom image to create an operational database.

To know more about the custom images, see *Custom images and image catalogs*.

Related Information

[Custom images and image catalogs](#)

Creating a database using a custom image

Cloudera Operational Database (COD) allows you to create a database using a custom image for compliance or security purposes. You can inherit pre-installed packages or software libraries from the custom image while creating an operational database.

Before you begin

- You must download and install the latest CDP CLI beta version.
- You must have a working CDP environment.
- You must ensure that the custom image resides in your CDP environment.

Procedure

1. Run the following command to create a database using a custom image.

```
cdp opdb create-database --environment <env_name> --database <database-name> --image <"id":<image_id>,"catalog":<catalog_name>>
```

Parameter value	Description
<env_name>	Name of the environment that holds the new database.
<database-name>	Name of the new database for which you want to use the custom image.
<image_id>	Image ID of the database image.
<catalog_name>	Name of the database image catalog.

For example,

```
cdp opdb create-database --environment-name cod-7215 --database-name testdb --image '{"id":"78aeac7f-4a37-4395-a270-833d655c20fb","catalog":"cdp-default"}'
```

2. Verify that the database creation is successful for the same image.

Navigate to the Data Hub Clusters service or to the Management Console Data Hub Clusters , and click the Image Details tab of the newly created Data Hub cluster.

Upgrading a database using a custom image

You can upgrade an existing database using a custom image. The database inherits the packages and software libraries of the custom image, and is upgraded to the provided database image.

Before you begin

- You must download and install the latest CDP CLI beta version.
- You must have a working CDP environment.
- You must ensure that the custom image resides in your CDP environment.

Procedure

1. Run the following commands sequentially to upgrade a database using a custom image.

- a. `cdp opdb upgrade-database --environment <env_name> --database <database_name> --image-id=string`
- b. `cdp opdb upgrade-database --environment <env_name> --database <database_name> --image-id=string --os-upgrade-only`

Parameter value	Description
<env_name>	Name of the environment that holds the existing database.
<database_name>	Name of the existing database for which you want to use the custom image.
--image-id	Image ID of the database image to which the existing database is upgraded.
--os-upgrade-only	Only perform an Operating System upgrade.

For example,

```
cdp opdb upgrade-database --environment cod-7216-micro3 --database tempd
b12 --image-id a188a3d7-067f-42f1-9b89-16d40faab24c
```

```
cdp opdb upgrade-database --environment cod-7216-micro3 --database tempd
b12 --image-id a188a3d7-067f-42f1-9b89-16d40faab24c --os-upgrade-only
```

2. Verify that the database upgrade is successful for the same image.

Navigate to the Data Hub Clusters service or to the Management Console Data Hub Clusters , and click the Image Details tab of the upgraded Data Hub cluster.

Switching image catalogs in COD

You can switch the image catalog of an existing operational database. You may want to switch the image catalog for a database to restrict which Runtime version can be upgraded to, or to move to custom images for an existing database.

Before you begin

- You must download and install the latest CDP CLI beta version.
- You must have a working CDP environment.
- You must ensure that the custom image resides in your CDP environment.

Procedure

1. Run the following command to switch an image catalog of an existing database.

```
cdp opdb update-database --environment <env_name> --database <database_name> --
catalog <catalog_name>
```

Parameter value	Description
<env_name>	Name of the environment that holds the existing database.

Parameter value	Description
<database_name>	Name of the existing database for which you want to switch the image catalog.
<catalog_name>	Name of the image catalog that you want to update for the database.

For example,

```
cdp opdb update-database --environment-name cod-7216-micro2 --database-name testdb --catalog v3dev-1
```

2. Verify that the image catalog switching is successful for the database.

Navigate to the Data Hub Clusters service or to the Management Console Data Hub Clusters , and click the Image Details tab of the updated Data Hub cluster.

Enabling HBase region canary

You can enable the HBase canary, which is an optional service, to monitor the health of the HBase RegionServer.

About this task

COD provides the CLI option `--enable-region-canary` to add the HBase canaries to monitor the RegionServer health. You can use this option while creating an operational database using COD CLI.

The HBase canaries are added under the HBase service in the Cloudera Manager. You can view them under Cloudera Manager HBase Service Configuration tab .

Before you begin

Ensure that you have created a CDP environment and is ready to use it.

Procedure

1. Log in to the terminal that has the CDP CLI client installed.
2. Use the following command to enable the HBase region canaries.

```
cdp opdb create-database --environment-name ENVIRONMENT_NAME --database-name DATABASE_NAME --enable-region-canary
```

For example,

```
cdp opdb create-database --environment-name cdp_7215 --database-name cod_1 --enable-region-canary
```

Results

The following HBase canaries are added on the Cloudera Manager UI.

- hbase_region_health_canary_enabled (HBase Region Health Canary)
- hbase_region_health_canary_slow_run_alert_enabled (HBase Region Health Canary Slow Run Alert Enabled)
- hbase_canary_alert_unhealthy_region_percent_threshold (HBase Canary Unhealthy Region Percentage Alert Threshold)

Related Information

[Create database](#)

Monitoring metrics in COD with Grafana

Cloudera Operational Database (COD) provides a pre-defined solution to visualize the COD metrics comprehensively. COD uses Grafana to store and visualize the metrics. You can seamlessly access all the COD metrics using this Grafana solution.

Grafana deployment for COD aims to provide the following benefits.

- *Enables access restriction to Cloudera Manager:* You can obtain necessary information and perform the required operations through the COD API, CLI, or UI while maintaining appropriate access restrictions to Cloudera Manager for other operations.
- *Utilizes ready-made dashboards with advanced widgets:* COD utilizes the pre-built dashboards that Grafana offers, to visualize and monitor the performance of various components.
- *Integrates external metrics sources:* COD allows you to incorporate metrics from external sources such as S3 storage. This integration provides a comprehensive view of COD, which helps to understand COD performance problems more effectively.

Consider the following aspects while enabling Grafana for COD.

- *Grafana server upgrade:* COD focuses on utilizing the existing version to meet the requirements and the upgrade of the Grafana server is not within the scope of this solution.
- *Manual dashboard updates:* You must explicitly download the updated dashboards from the Cloudera Public repository. This allows flexibility and ensures that you have the latest dashboard versions whenever necessary.
- *Operating System support:* The existing solution works on Red Hat, CentOS, RHEL, and Fedora OS because of an RPM-based installation. That is why COD does not allow you to install Grafana with custom images having different OS.
- *Data Lake metrics support:* COD does not include the Data Lake metrics in this solution. Only individual COD (Data Hub) metrics appear in the Grafana dashboard.
- *HA Knox support:* Currently, COD only supports a single Gateway host with a single Knox Gateway.
- *Foursquare plugin support:* The Cloudera Manager foursquare datasource plugin does not support sending alerts. That is why alerts cannot be created on HBase, HDFS, and ZooKeeper dashboard panels.

Enabling Grafana dashboard in COD

Learn how to enable the Grafana URL to visualize the Cloudera Operational Database (COD) metrics.

Before you begin

- You must whitelist the *Cloudera archive* URL so that the necessary RPM packages for Grafana can be installed in the instances.
- You must whitelist the *Cloudera repository* so that the dashboards are created automatically.
- You must whitelist the *Grafana RPM packages* URL so that Grafana can be installed in the instances.
- You must whitelist the *Google API storage* URL so that the Cloudera Manager foursquare plugin can be installed.
- You must attach the following policy for CloudWatch plugin under cdp-infra2-logs-role (or the role with which the ec2 instances are created) to enable the Amazon S3 metrics in Grafana dashboard. To attach a policy under the cdp-infra2-logs-role, see *Create a cross-account IAM role*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowReadingMetricsFromCloudWatch",
      "Effect": "Allow",
```

```

    "Action": [
      "cloudwatch:DescribeAlarmsForMetric",
      "cloudwatch:DescribeAlarmHistory",
      "cloudwatch:DescribeAlarms",
      "cloudwatch:ListMetrics",
      "cloudwatch:GetMetricStatistics",
      "cloudwatch:GetMetricData",
      "cloudwatch:GetInsightRuleReport"
    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingLogsFromCloudWatch",
    "Effect": "Allow",
    "Action": [
      "logs:DescribeLogGroups",
      "logs:GetLogGroupFields",
      "logs:StartQuery",
      "logs:StopQuery",
      "logs:GetQueryResults",
      "logs:GetLogEvents"
    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingTagsInstancesRegionsFromEC2",
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeTags",
      "ec2:DescribeInstances",
      "ec2:DescribeRegions"
    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingResourcesForTags",
    "Effect": "Allow",
    "Action": "tag:GetResources",
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingAcrossAccounts",
    "Effect": "Allow",
    "Action": [
      "oam:ListSinks",
      "oam:ListAttachedLinks"
    ],
    "Resource": "*"
  }
]
}

```

- You must also create a CloudWatch metrics configuration to enable the Amazon S3 metrics in Grafana dashboard. See the steps mentioned under Using the S3 console in *Creating a CloudWatch metrics configuration for all the objects in your bucket*.

Procedure

1. Log in to the CDP CLI command tool.

2. Run the following command.

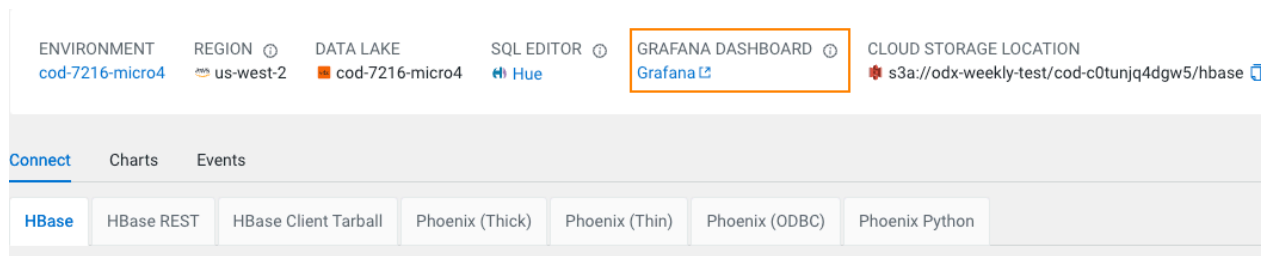
```
cdp opdb create-database --environment <environment_name> --database
<database_name> --enable-grafana
```

For example,

```
cdp opdb create-database --environment-name cod-7216-micro10 --database-
name odx2408 --enable-grafana
```

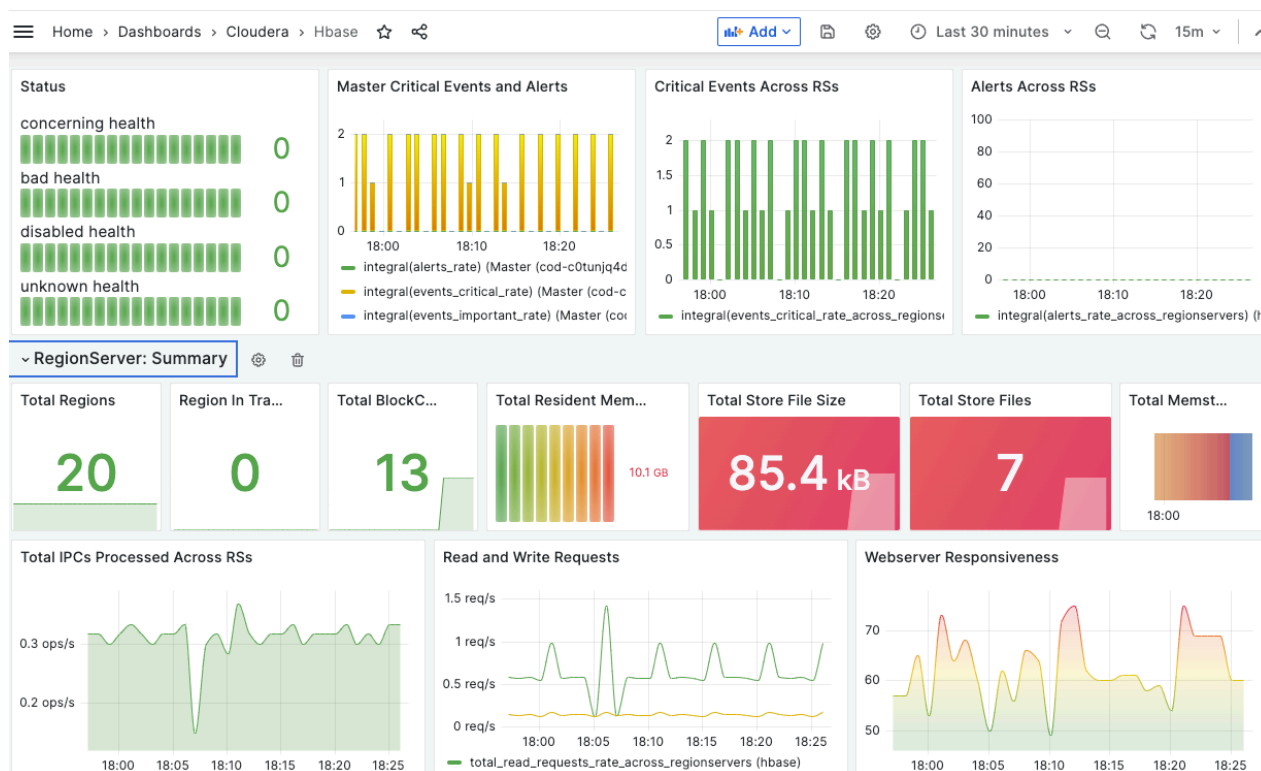
Results

On successfully executing the command, Grafana URL is added under the GRAFANA DASHBOARD option inside the COD database as shown in the following figure.



When you click on the Grafana URL, it takes you to the Grafana dashboard.

Here is an example of the HBase dashboard using Grafana.



Related Information

[CDP CLI Beta](#)

[Cloudera archive](#)

[Cloudera repository](#)

[Grafana RPM packages](#)

[Google API storage](#)

[Create a cross-account IAM role](#)

[Creating a CloudWatch metrics configuration for all the objects in your bucket](#)

Enabling Grafana dashboard for an existing COD database

Learn how to enable the Grafana URL to visualize the Cloudera Operational Database (COD) metrics for an existing COD database.

Before you begin

- You must whitelist the *Cloudera archive* URL so that the necessary RPM packages for Grafana can be installed in the instances.
- You must whitelist the *Cloudera repository* so that the dashboards are created automatically.
- You must whitelist the *Grafana RPM packages* URL so that Grafana can be installed in the instances.
- You must whitelist the *Google API storage* URL so that the Cloudera Manager fousquare plugin can be installed.
- You must attach the following policy for CloudWatch plugin under cdp-infra2-logs-role (or the role with which the ec2 instances are created) to enable the Amazon S3 metrics in Grafana dashboard. To attach a policy under the cdp-infra2-logs-rollee, see *Create a cross-account IAM role*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowReadingMetricsFromCloudWatch",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData",
        "cloudwatch:GetInsightRuleReport"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowReadingLogsFromCloudWatch",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups",
        "logs:GetLogGroupFields",
        "logs:StartQuery",
        "logs:StopQuery",
        "logs:GetQueryResults",
        "logs:GetLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowReadingTagsInstancesRegionsFromEC2",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeTags",
        "ec2:DescribeInstances",
        "ec2:DescribeRegions"
      ]
    }
  ]
}
```

```

    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingResourcesForTags",
    "Effect": "Allow",
    "Action": "tag:GetResources",
    "Resource": "*"
  },
  {
    "Sid": "AllowReadingAcrossAccounts",
    "Effect": "Allow",
    "Action": [
      "oam:ListSinks",
      "oam:ListAttachedLinks"
    ],
    "Resource": "*"
  }
]
}

```

- You must also create a CloudWatch metrics configuration to enable the Amazon S3 metrics in Grafana dashboard. See the steps mentioned under Using the S3 console in *Creating a CloudWatch metrics configuration for all the objects in your bucket*.

Procedure

- Go to the *Cloudera repository* and copy these files (grafana-install-configure-v2.sh and configure-knox-for-grafana.sh) to the gateway node of the COD cluster.

If you are using CDH version 7.2.15, download configure-knox-for-grafana7215.sh file.

Following is an example command.

```

scp -i ~/.ssh/odx-developers.pem ./grafana-install-configure-v2.sh cloudbreak@10.1.1.1:
scp -i ~/.ssh/odx-developers.pem ./configure-knox-for-grafana.sh cloudbreak@10.1.1.1:

```

- Connect to the gateway node.

Following is an example command.

```
ssh -i ~/.ssh/odx-developers.pem cloudbreak@10.8.2.5
```

- Get the root permission for the folder where you copied the script files.

Following is an example command.

```
sudo -i cd /home/cloudbreak/
```

- Set the owner permission for the script files so that it can run successfully.

```
chown root:root *
```

- Run the Grafana installation script.

Following is an example command.

```
./grafana-install-configure-v2.sh
```

```

extracted string: cod--186yjxqvwchow
Failed to set locale, defaulting to C

```

```

Loaded plugins: fastestmirror, versionlock
Determining fastest mirrors
epel/x86_64/metalink

                | 27 kB  00:00:00
* base: download.cf.centos.org
* centos-sclo-rh: download.cf.centos.org
* centos-sclo-sclo: download.cf.centos.org
* epel: ftp.osl.osuosl.org
* extras: download.cf.centos.org
* updates: download.cf.centos.org
  base

                | 3.6 kB  00:00:00
  cdp-infra-tools

                | 2.9 kB  00:00:00
  centos-sclo-rh

                | 3.0 kB  00:00:00
...
Created symlink from /etc/systemd/system/multi-user.target.wants/grafana-
server.service to /usr/lib/systemd/system/grafana-server.service.

```

If the command is successful, you must see the symlink created message.

6. Check the Grafana service status using the `systemctl` command.

```
systemctl status grafana-server.service
```

```

grafana-server.service - Grafana instance
Loaded: loaded (/usr/lib/systemd/system/grafana-server.service; enabled; v
endor preset: disabled)
Active: active (running) since Tue 2023-08-22 10:51:08 UTC; 16s ago
Docs: http://docs.grafana.org
Main PID: 25273 (grafana)
CGroup: /system.slice/grafana-server.service
#25273 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/var/run/grafana/grafana-server.pid --packaging=rpm c
fg=default.paths.logs=/var/log/gr...

```

7. Run the Knox configuration script.

Following is an example command.

```
./configure-knox-for-grafana.sh
```

If you are using CDH version 7.2.15, use the following command `./configure-knox-for-grafana7215.sh`.

```

Installing knox service configs for grafana
Adding GRAFANA service in the cdp-proxy topology
Adding GRAFANA service in the cdp-proxy topology in the active directory

```

8. Restart the Knox service using Cloudera Manager.

Results

The following is a sample Grafana dashboard URL.

<https://<GATEWAY-FQDN>/grafanacod/dashboards>

You can obtain the value of `<GATEWAY-FQDN>` from COD DATAHUB Nodes Gateway . In the listed table, you can find the FQDN column. For example, <https://cod--186yjqvwowh-gateway0.cod-7216.xcu2-8y8x.dev.cldr.work/grafanacod/dashboards>.

Related Information

[CDP CLI Beta](#)

[Cloudera archive](#)

[Cloudera repository](#)

[Grafana RPM packages](#)

[Google API storage](#)

[Create a cross-account IAM role](#)

[Creating a CloudWatch metrics configuration for all the objects in your bucket](#)

Importing a Grafana dashboard

Know how to import an existing Grafana dashboard into your COD environment.

Before you begin

- You must whitelist the *Cloudera archive* URL so that the necessary RPM packages for Grafana can be installed in the instances.
- You must whitelist the *Cloudera repository* so that the dashboards are created automatically.

Procedure

1. Download the dashboard JSON files (for example, S3.json) in your local computer from the *Cloudera repository*.
2. Open your Grafana portal and go to the Dashboard page.
3. Choose your folder where you want to install the dashboards (for example, *Cloudera*).
4. Click **New Import** from the drop-down menu.
5. Upload the dashboard JSON file which you had downloaded earlier.
After uploading the file you can see that the dashboard is created in your chosen folder.
6. Select the dashboard to see the graphs.

Related Information

[Cloudera archive](#)

[Cloudera repository](#)