1.19.

Configuring Data Connectors

Date published: 2023-04-05
Date modified: 2023-04-13

CLOUDƎRA

Legal Notice

Contents

Using Ozone storage with Cloudera Data Engineering Private Cloud

Apache Ozone is an object store available on the CDP Private Cloud Base cluster which enables you to optimize storage for big data workloads. You can configure Ozone as the backend storage for workloads of CDE clusters.

Data connectors enable you to access different storage using only a few configurations specific to storage. Data Connectors are bound to a CDE service. Hence, you must first create a CDE service before configuring Ozone storage through Data Connectors. All the virtual clusters managed by that CDE service can use the same data connectors.

Related Information
Adding a Cloudera Data Engineering service

Adding Ozone data connector for Cloudera Data Engineering service

You can configure Ozone as the backend storage for Cloudera Data Engineering (CDE) workloads.
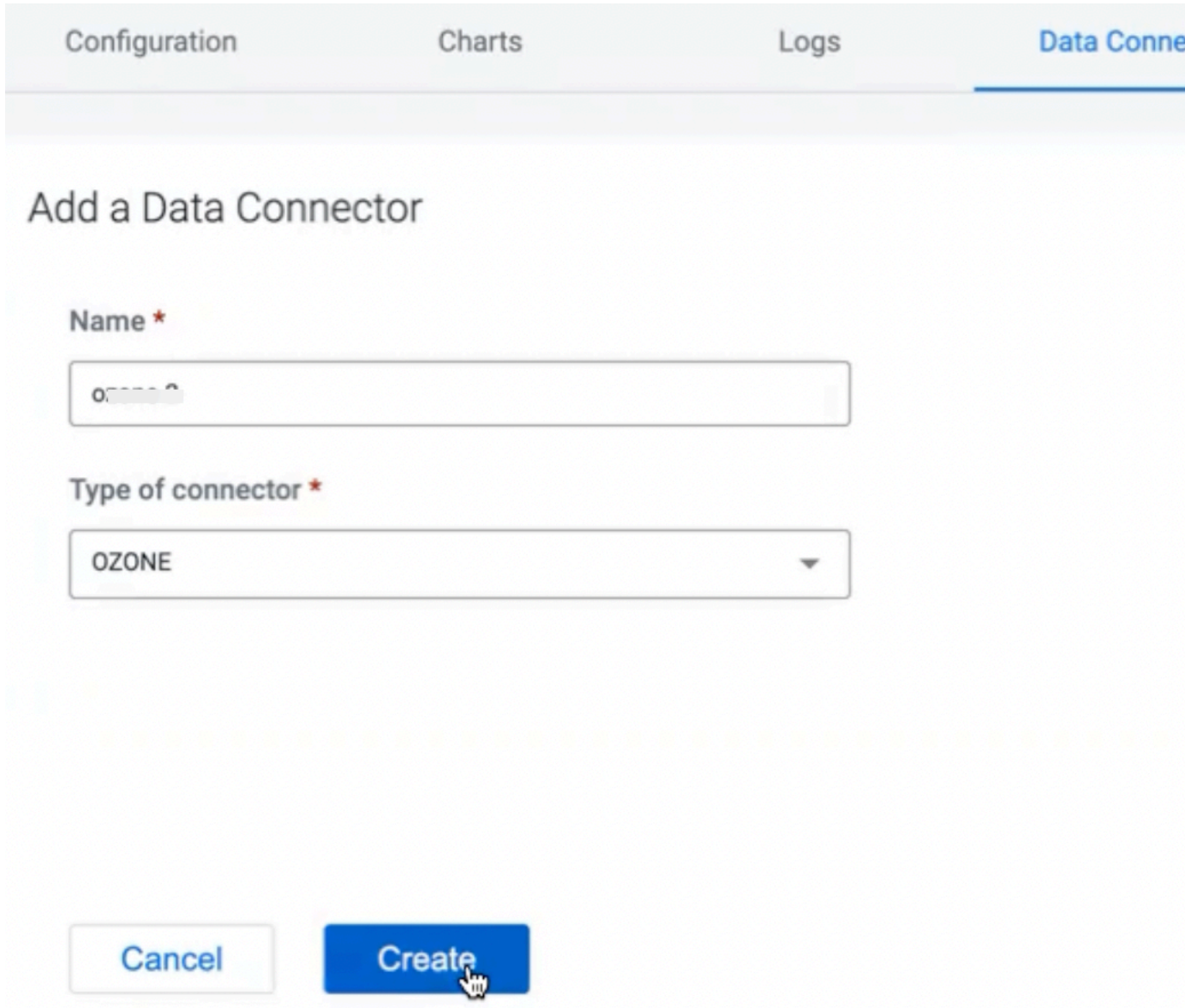
Before you begin
Data connectors are bound to a CDE service and you must first create a CDE service before configuring Ozone storage.

Procedure

1. In the Cloudera Data Platform (CDP) console, click the Data Engineering tile. The CDE Home page displays.
2. Click Administration in the left navigation menu. The Administration page displays.
3. In the Services column, click for the CDE service you want to configure for accessing the Ozone file system.
4. Click the Data Connectors tab. On the Data Connectors tab, you can view details about the existing data connectors, if any, including the service name and CDP environment.
5. Click Create. The Add a Connector pane is displayed.
6. Specify the connector's name.

7. Select Ozone from the Type of Connectordrop-down list.

8. Review the summary and click Create.

Results

After configuring the CDE service to access the Ozone filesystem, use this data connector during job creation to read and write from Apache Ozone Object Store through your workload Spark job.

Related Information

Adding a Cloudera Data Engineering service

Creating jobs in Cloudera Data Engineering

Deleting Ozone data connector for Cloudera Data Engineering service

You can delete an existing data connector.

Procedure

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

2. Click Administration in the left navigation menu. The Administration page displays.

3. In the Services column, click ✏️ icon for the CDE service you want to configure for accessing the Ozone file system.

4. Click the Data Connectors tab. On the Data Connectors tab, you can view details about the existing data connectors, if any, including the service name and CDP environment.

5. Click ⋮ in the Actions column next to the data connector, and then click Delete.

Backing up and Restoring Data Connectors

Before you upgrade your clusters or delete an existing cluster to create a new cluster, you can back up the data connectors and restore those data connectors post-upgrade on the new cluster. The data connector pre-upgrade jobs work seamlessly on the upgraded cluster after you restore those jobs.

Backing up the data connector

You must back up your data connectors before you delete or upgrade your cluster.

Before you begin

Install `jq` on the host machine where you want to run the backup and restore commands.

Procedure

1. Get the CDE Service URL.

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

   b. In the CDE Services column, click ✏️ for the CDE service.

   c. Click GRAFANA CHARTS and get the domain name.

   d. Note the URL. For example, if the URL copied is *https://service.cde-n82xsqpr.apps.apps.shared-rke-dev-01.kcloud.example.com/grafana/d/sK1XDusZz/kubernetes*, use only the *https://service.cde-n82xsqpr.apps.apps.shared-rke-dev-01.kcloud.example.com* as the CDE Service URL.

2. Get the CDE token for your CDE service URL.

```
export CDE_TOKEN=$(curl -k -L -u <user-name>:<password> https://<cde ser
vice url>/gateway/authtkn/knoxtoken/api/v1/token | jq -r .access_token)
```

   Example:

```
export CDE_TOKEN=$(curl -k -L -u dexssoadmin:Password@123 https://servic
e.cde-4x6s29mx.apps.apps.shared-os-qe-01.kcloud.example.com/gateway/auth
tkn/knoxtoken/api/v1/token | jq -r .access_token)
```

3. Note the cluster-ID.

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

   b. In the CDE Services column, click ✏️ for the CDE service.

   c. Click GRAFANA CHARTS and get the domain name.

   d. Note the Cluster ID shown on the page.

4. Provide the CDE token and Cluster ID and run the data connector backup command.

```
curl --location --request POST 'https://<cde service url>/data-connectors/
v1/backupDataConnections' \
--header "Authorization: Bearer ${CDE_TOKEN}" \
--data-raw '{
  "clusterId": "<cluster id of cluster that needs backing up>"
```

```
}' --insecure | jq .zipFile | tr -d '"'  | base64 -d > backup-dataconnect
ors.zip
```

Example:

```
curl --location --request POST 'https://service.cde-4x6s29mx.apps.apps.s
hared-os-qe-01.kcloud.cloudera.com/data-connectors/v1/backupDataConnecti
ons' \
--header "Authorization: Bearer ${CDE_TOKEN}" \
--data-raw '{
  "clusterId": "cluster-4x6s29mx"
}' --insecure | jq .zipFile | tr -d '"'  | base64 -d > backup-dataconnect
ors.zip
```

Related Information

Installing jq

Restoring the data connector

You can restore the backed-up data connectors post-upgrade or on the new CDE Service in the same cluster or across clusters.

Procedure

1. Get the CDE Service URL.

   a. In the Cloudera Data Platform (CDP) console, click the Data Engineering tile. The CDE Home page displays.
   b. In the CDE Services column, click 📝 for the CDE service.
   c. Click GRAFANA CHARTS and get the domain name.
   d. Note the URL. For example, if the URL copied is *https://service.cde-n82xsqpr.apps.apps.shared-rke-dev-01.kcloud.example.com/grafana/d/sK1XDusZz/kubernetes*, use only the *https://service.cde-n82xsqpr.apps.apps.shared-rke-dev-01.kcloud.example.com* as the CDE Service URL.

2. Get the CDE token for the CDE Service that you want to restore.

```
export CDE_TOKEN=$(curl -k -L -u <user-name>:<password> https://<cde ser
vice url>/gateway/authtkn/knoxtoken/api/v1/token | jq -r .access_token)
```

Example:

```
export CDE_TOKEN=$(curl -k -L -u dexssoadmin:Password@123 https://servic
e.cde-4x6s29mx.apps.apps.shared-os-qe-01.kcloud.example.com/gateway/auth
tkn/knoxtoken/api/v1/token | jq -r .access_token)
```

3. Encode the backed-up zip file to base64 to use it in the restore process.

```
base64 -i <backup-dataconnections.zip>
```

Example:

```
base64 -i backup-dataconnections.zip
dGhpcyBpcyBhbiBleGFtcGxlIGVuY29kZWQgc3RyaW5n
```

dGhpcyBpcyBhbiBleGFtcGxlIGVuY29kZWQgc3RyaW5n is the output.

4. Note the Cluster ID.

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

    b. In the CDE Services column, click ✎ for the CDE service.

    c. Click GRAFANA CHARTS and get the domain name.

    d. Note the Cluster ID shown on the page.

5. Run the restore API to restore the data connector.

```
curl --location --request POST '<CDE Service URL>/data-connectors/v1/res
toreDataConnections' \
--header "Authorization: Bearer ${CDE_TOKEN}" \
--data-raw '{
 "clusterId": "<cluster id to restore into>",
 "handleDuplicates":false,
 "backupZipFile": "<base64 string from last step>"
}'--insecure
```

Example:

```
curl --location --request POST 'https://service.cde-wwkd776f.apps.apps.s
hared-os-qe-01.kcloud.cloudera.com/data-connectors/v1/restoreDataConnect
ions' \
--header "Authorization: Bearer ${CDE_TOKEN}" \
--data-raw '{
 "clusterId": "cluster-wwkd776f",
 "handleDuplicates": false,
 "backupZipFile": "dGhpcyBpcyBhbbiBleGFtcGxlIGVuY29kZWQgc3RyaW5n"
}' --insecure
```

The `handleDuplicates` parameter controls the restore behavior if data connectors with the same name are found on the cluster during the restore process. If there are data connectors with the same name and `handleDuplicates` is set to `false`, the restoration fails with an error. If `handleDuplicates` is set to `true`, data connectors are restored with a different name and the format of the new name is `<old-name>-restored-<random string of length 3>`.