Cloudera Runtime 7.3.2

Using Cloudera Iceberg REST Catalog for Data Sharing

Date published: 2020-07-28
Date modified: 2026-03-31

https://docs.cloudera.com/

Legal Notice

Contents

How data sharing with Cloudera Iceberg REST Catalog works

Cloudera Iceberg REST Catalog enables your clients to run their workloads from data platforms, such as Databricks or Snowflake to fetch data from Cloudera environments for analytical purposes.

Data sharing involves the creation of a Data Share with the necessary authorization and authentication mechanisms. A Data Share is an organizational unit of data, a collection of data assets. You can then share this data with your clients so that they can access the Iceberg table data created within the Cloudera environment.

The following sections describe the high-level workflow of the processes:

Data Share creation

The following tasks are part of the Data Share creation process:

- As a resource owner, use the existing Knox Token Management system to generate a token. This unique Token ID is referred to as the CLIENT_ID and the generated passcode is the CLIENT_SECRET.
- As part of the token generation process, a Ranger role and Ranger group are created. This group is a virtual group that Knox provides for the client with whom the data is shared.
- Create and maintain policies for the set of databases and tables to be shared for the Ranger role and group and thereby create a Data Share.
- Maintain the SELECT permission for the databases or tables to allow READ-only access.
- You can then share the CLIENT_ID and the CLIENT_SECRET with your clients so that they can access the shared data in the Cloudera environment.

Data Share access

After you have created a Data Share, which includes creating tokens, authoring a read-only policy within Ranger, and shared the CLIENT_ID and CLIENT_SECRET with your client, the client makes use of these credentials in their workloads to establish a handshake with Cloudera and to access the shared data. In exchange for the CLIENT_ID and the CLIENT_SECRET, Knox can return an Access Token, than can be used in requests to Iceberg REST Catalog (Hive MetaStore).

Related Information

Enabling data sharing with Cloudera Iceberg REST Catalog

Verifying external access to a Data Share

Creating a Data Share with CDP CLI

Creating a Data Share

Setting up Cloudera Iceberg REST Catalog for data sharing

Learn how to perform the preparatory configurations in Cloudera to enable data sharing. These configurations are required for the creation of a data share in Cloudera and allow your clients to access data in Cloudera environments.

Installation and Upgrade Scenarios

Note:

- This feature is available as technical preview and is under entitlement. To obtain the required entitlement, contact your Cloudera Account Representative.
- In the following processes, run all commands within the network of your Cloudera Runtime or through a VPN.

Different versions of Cloudera Runtime and Data Lakes require different steps to enable data sharing.

- Fresh 7.3.2.x Data Lake Installation on page 5
- Upgrade from Cloudera Runtime 7.2.18.x with no Cloudera Iceberg REST Catalog configurations on page 5
- Upgrade from Cloudera Runtime 7.2.18.x with Cloudera Iceberg REST Catalog / Cloudera Manager configuration on page 6
- Resizing Cloudera Data Lake from Light Duty Data Lake to Enterprise Data Lake on page 6

Fresh 7.3.2.x Data Lake Installation

Complete the following steps after a fresh installation of 7.3.2.x Data Lake to enable data sharing.

Note: For more information on installing a Data Lake, see:

- Creating and managing Cloudera deployments
- Deploy Cloudera using Terraform
- Registering an AWS environment from the Cloudera UI

1. HMS Rest Catalog configuration

   Configure the Hive Metastore (HMS) service to serve as an Iceberg REST catalog. This allows clients to use REST Catalog APIs to access the required metadata files.
2. Editing Knox topologies

   Edit your Knox topologies to define the lifetime of your Knox tokens (lifetime of your Data Shares) and the number of external users.
3. Knox configuration

   Configure properties in Knox to set up administrator privileges that allow creation of tokens.
4. IDBroker configuration

   Configure Knox IDBroker in Cloudera Manager.
5. Creating a Data Share with CDP CLI or Creating a new Data Share

   Create a Data Share in Cloudera.

Upgrade from Cloudera Runtime 7.2.18.x with no Cloudera Iceberg REST Catalog configurations

Complete the following configurations to enable Cloudera Data Sharing when upgrading the Data Lake from 7.2.18.x with no REST Catalog or Cloudera Data Sharing configured in that version.

Note: For more information on upgrading a Data Lake, see Upgrading a Data Lake.

1. Manual Installation of metering service

   Install and configure the metering server for Cloudera Data Sharing.
2. HMS Rest Catalog configuration

   Configure the Hive Metastore service to serve as an Iceberg REST catalog. This allows your clients to use REST Catalog APIs to access the required metadata files.

3. Editing Knox topologies

   Edit your Knox topologies to define the lifetime of your Knox tokens (lifetime of your Data Shares) and the number of external users.
4. Knox configuration

   Configure properties in Knox to set up administrator privileges that allow creation of tokens.
5. IDBroker configuration

   Configure Knox IDBroker in Cloudera Manager.
6. Creating a Data Share with CDP CLI or Creating a new Data Share

   Create a Data Share in Cloudera.

Upgrade from Cloudera Runtime 7.2.18.x with Cloudera Iceberg REST Catalog / Cloudera Manager configuration

Complete the following configurations to enable Data Sharing when upgrading the Data Lake from 7.2.18.x with REST Catalog / Data Sharing configured in that version.

Note: For more information on upgrading a Data Lake, see Upgrading a Data Lake.

1. Manual Installation of metering service

   Install and configure the metering server for Cloudera Data Sharing.
2. HMS Rest Catalog configuration

   Configure the Hive Metastore service to serve as an Iceberg REST catalog. This allows your clients to use REST Catalog APIs to access the required metadata files.
3. Editing Knox topologies

   Edit your Knox topologies to define the lifetime of your Knox tokens (lifetime of your Data Shares) and the number of external users.
4. Knox configuration

   Configure properties in Knox to set up administrator privileges that allow creation of tokens.
5. IDBroker configuration

   Configure Knox IDBroker in Cloudera Manager.
6. Creating a Data Share with CDP CLI or Creating a new Data Share

   Create a Data Share in Cloudera.

Resizing Cloudera Data Lake from Light Duty Data Lake to Enterprise Data Lake

Complete the following configurations to enable Cloudera Data Sharing after resizing the Data Lake from Light Duty Data Lake to Enterprise Data Lake.

Note: For more information on resizing a Data Lake, see Resizing a Data Lake.

1. HMS Rest Catalog configuration

   Configure the Hive Metastore service to serve as an Iceberg REST catalog. This allows your clients to use REST Catalog APIs to access the required metadata files.
2. Editing Knox topologies

   Edit your Knox topologies to define the lifetime of your Knox tokens (lifetime of your Data Shares) and the number of external users.
3. Knox configuration

   Configure properties in Knox to set up administrator privileges that allow creation of tokens.

4. IDBroker configuration

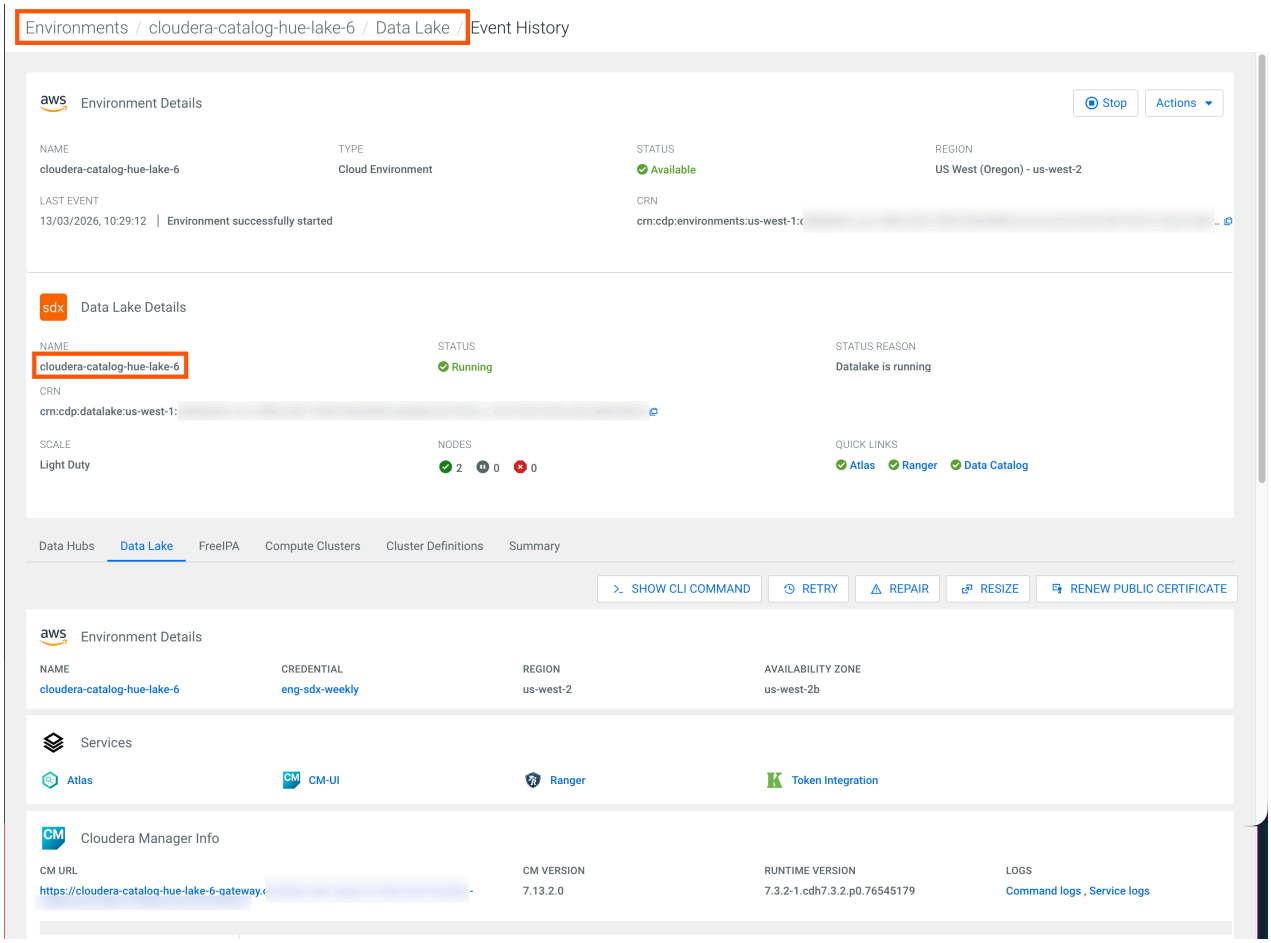
   Configure Knox IDBroker in Cloudera Manager.

Configuring Hive Metastore as a REST Catalog

To use the API endpoints provided by REST Catalog, you need to enable it in the Hive Metastore where it is deployed.

Before you begin

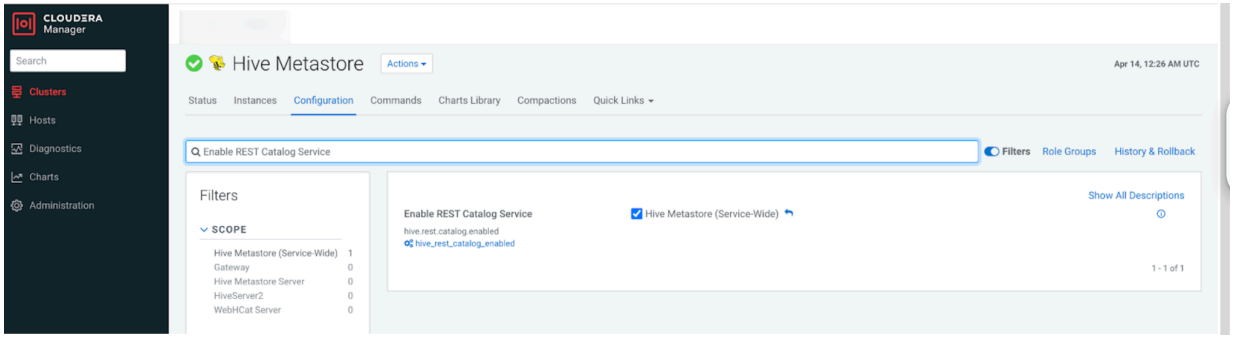
Ensure that your Cloudera Runtime is of 7.3.2 version or later in Management Console Environments ***YOUR-ENVIRONMENT*** Data Lake)



Procedure

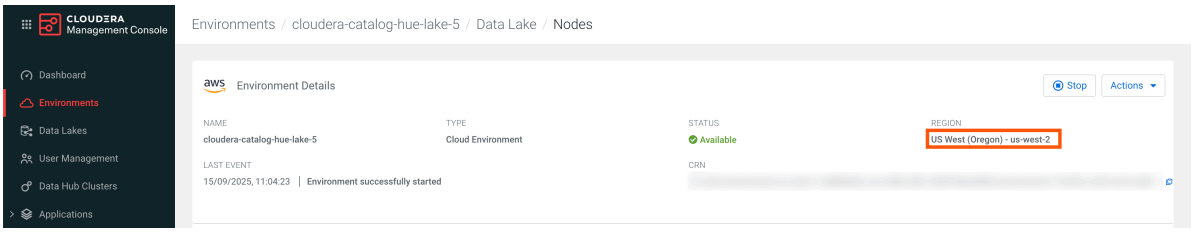
1. Log in to Cloudera Manager and click Clusters Hive Metastore Configuration .

2. Search for **Enable REST Catalog Service** and enable it to set up the REST Catalog Service in Hive Metastore.



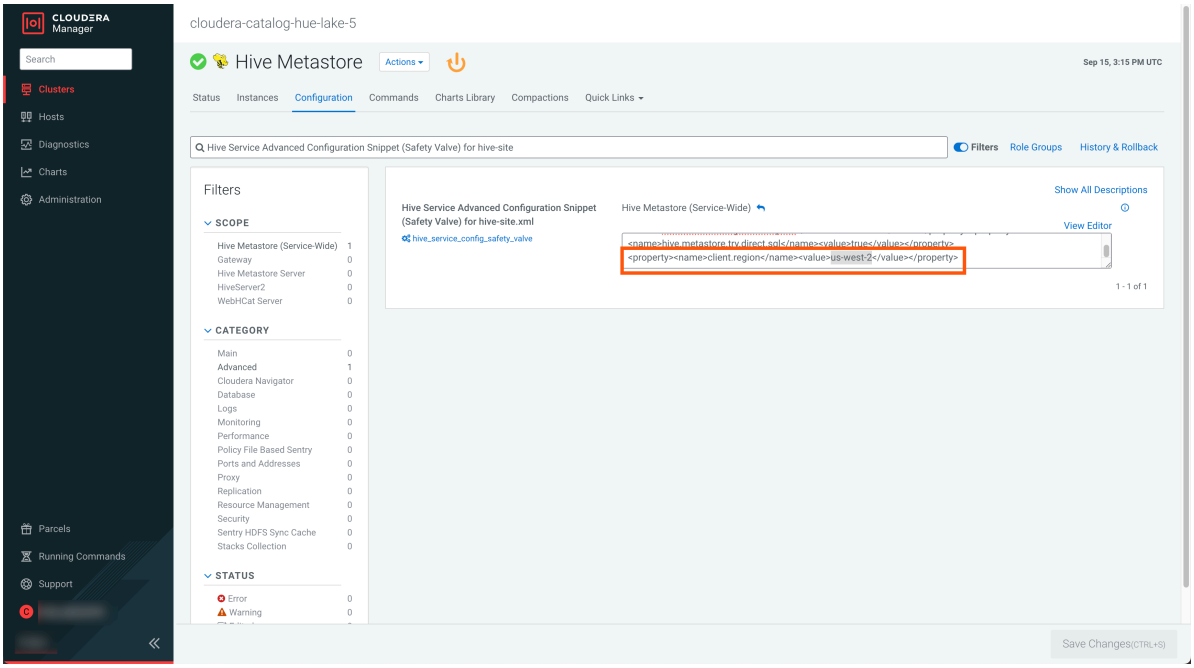
Note: Use only one IDBroker, as Hive Metastore does not support multiple IDBrokers.

3. If you use Cloudera Data Sharing with AWS Elastic MapReduce and AWS Athena notebook, add the region of AWS environment to the Hive Metastore configuration.
   a) Check your region in Cloudera Management Console Environments *[***YOUR_ENVIRONMENT***]* Environment Details .

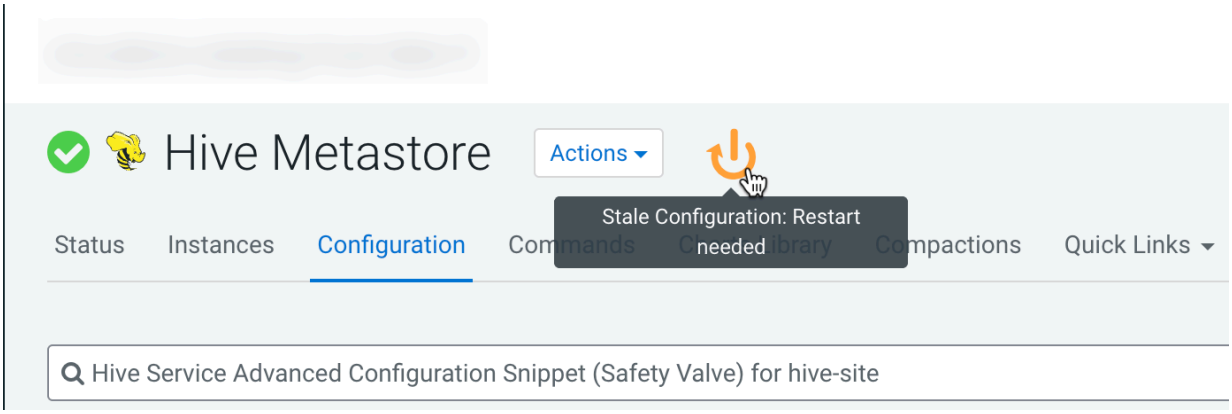


   b) If your region is different than the default us-west-2, go to Cloudera Manager Clusters Hive Metastore Configuration Hive Service Advanced Configuration Snippet (Safety Valve) for hive-site.xml .
   c) Add a property with your region.

   <property><name>client.region</name><value>*[***CLIENT_REGION_NAME***]*</value></property>



4. Save the changes and restart the Hive Metastore service.



What to do next

Continue with Editing the Knox topologies.

Editing Knox topologies

Learn how to edit Knox topologies to define for Knox how to proxy requests from external users.

Before you begin

Ensure that you have the following information before performing the steps:

- Username of the user who will generate the CLIENT_ID and CLIENT_SECRET.

About this task

The cdp-datashare-access Knox topology is automatically deployed. Editing the token lifetime and the token allowance per user (by adding the parameter KNOXTOKEN:knox.token.limit.per.user=*[***TOKENLIMIT***]*) is possible for each topology but it is optional. The following steps override the general settings in the Knox Gateway for a specific Knox topology. You can add more topologies. For more information, see Add a custom descriptor to Apache Knox.

Important: If you use multiple services using the same Knox token gateway, Cloudera strongly recommends using topology level settings by each service.

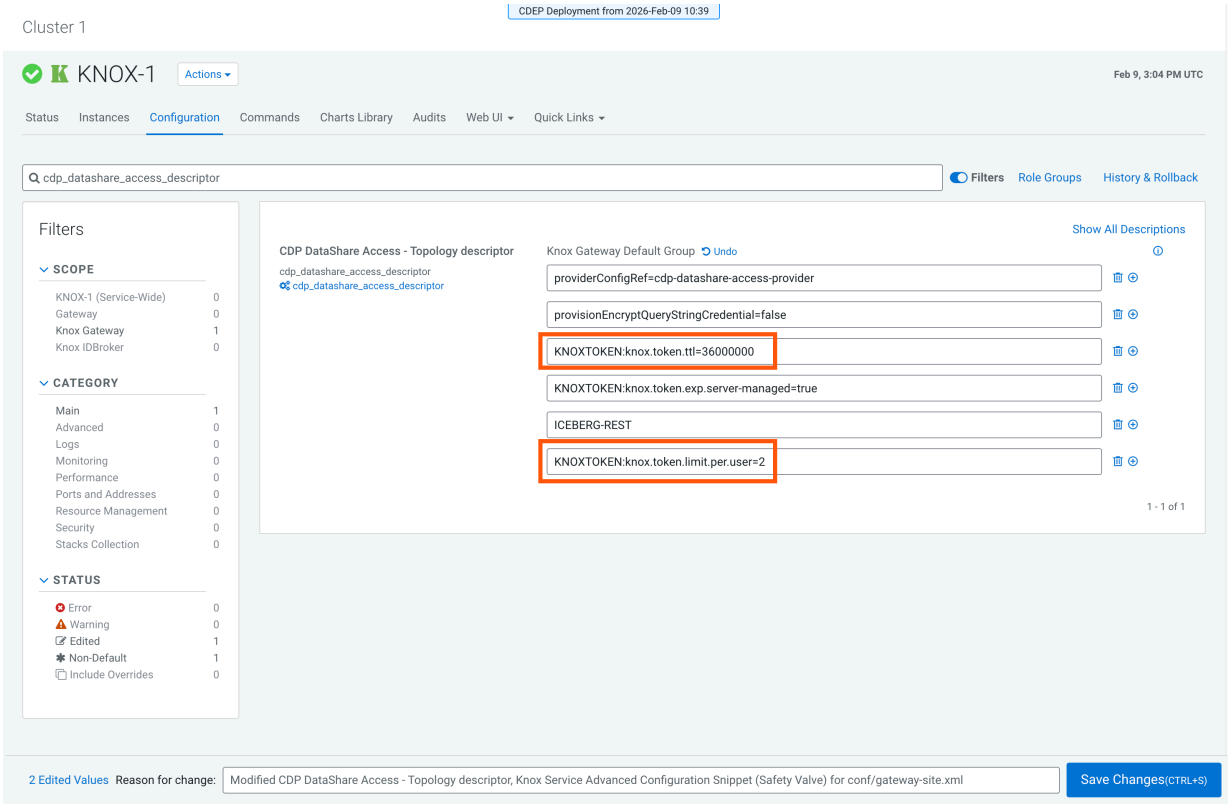
Procedure

1. Go to Cloudera Manager Knox Configuration .
2. Select the **Knox Gateway** scope.
3. Edit the default cdp-datashare-access topology by searching for cdp_datashare_access_descriptor.

4. Configure the value of KNOXTOKEN:knox.token.ttl=36000000.

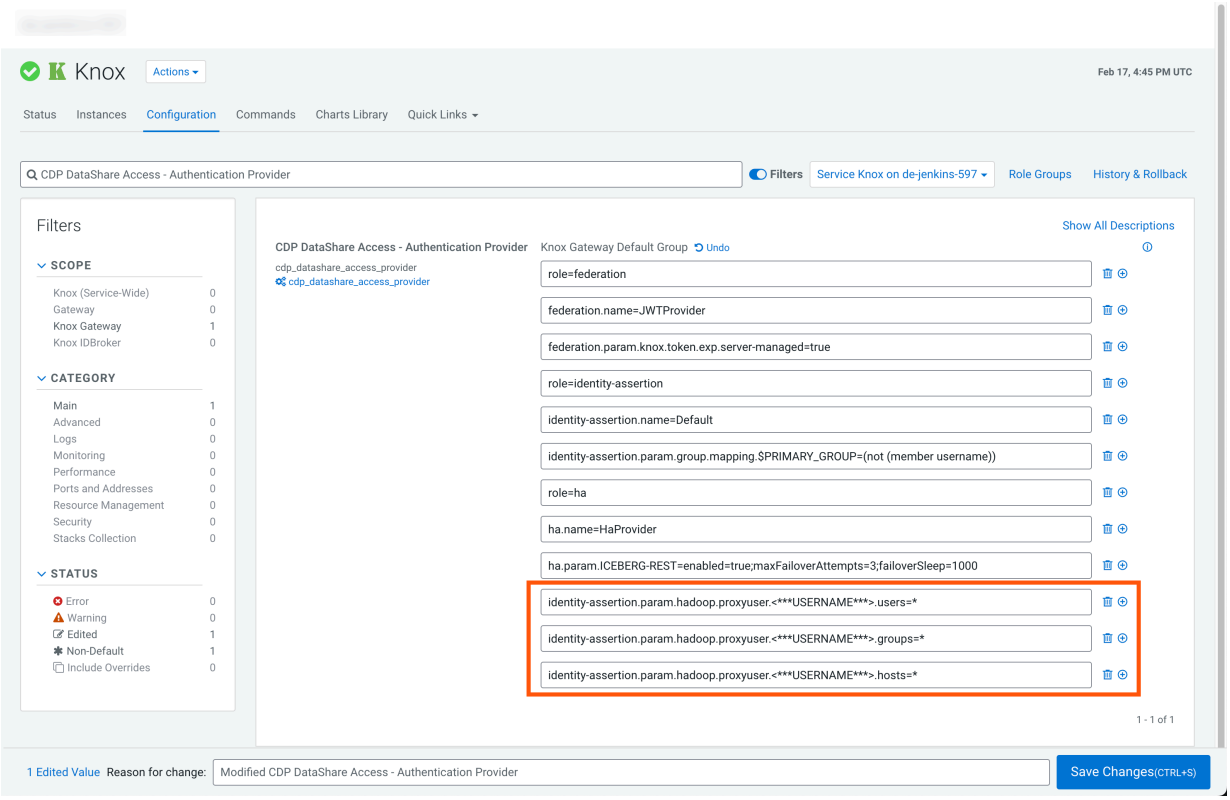
Note: The default value of 36000000 of knox.token.ttl means a 10 hour period. You can reduce this number to adjust the CLIENT_ID and CLIENT_SECRET lifetime.

Figure 1: Editing the default topology

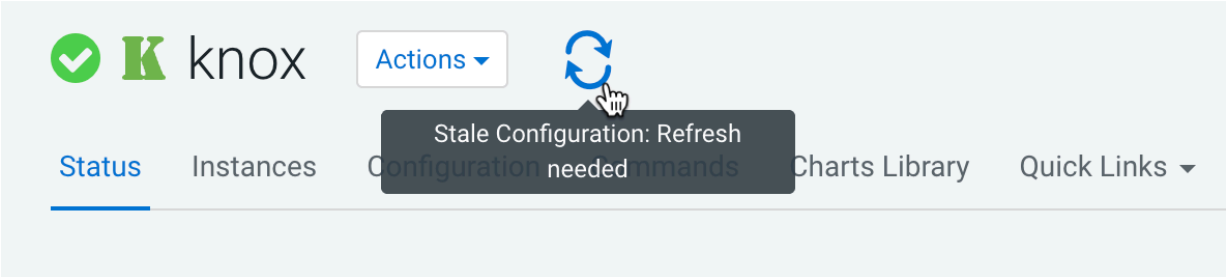


Important: knox.token.ttl controls the lifetime of your Knox access token received in exchange for the Client ID and Secret of your external users. Once it expires, external users need to request a new access token. For more information, see Editing Knox topologies.

5. If additional users are needed, the marked section needs to be duplicated to add the user in cdp_datashare_access _provider.



6. Click Save Changes and refresh the configuration as needed.



Results

The relevant Knox topologies are updated.

What to do next

Continue with configuring Knox.

Configuring the Knox gateway-site.xml

Learn how to configure Knox parameters to allow admin permissions in Knox. Admin permission is required to create the CLIENT_ID and CLIENT_SECRET.

About this task

The CLIENT_ID and CLIENT_SECRET is required for creating Data Shares to authorize your external clients.

Before you begin

- The Cloudera on cloud user must be configured as both Knox and Ranger Admin to perform the tasks required to configure Knox parameters.

  Note: For more information on setting the Ranger admin, see:

  - Cloudera account administrator
  - Administering Ranger Users, Groups, Roles, and Permissions
- Knox topologies are automatically deployed. Editing the token lifetime (KNOXTOKEN:knox.token.ttl) and the token allowance per user (KNOXTOKEN:knox.token.limit.per.user) is applicable to all topologies, but can be overridden by individual topology settings.

  Important: If you use multiple services using the same Knox token gateway, Cloudera strongly recommends using topology level settings by each service.

Procedure

1. Go to Cloudera Manager Clusters Knox Configuration Advanced Configuration Snippet (Safety Valve) for conf/gateway-site.xml
   a) Add the gateway.knox.admin.users parameter.

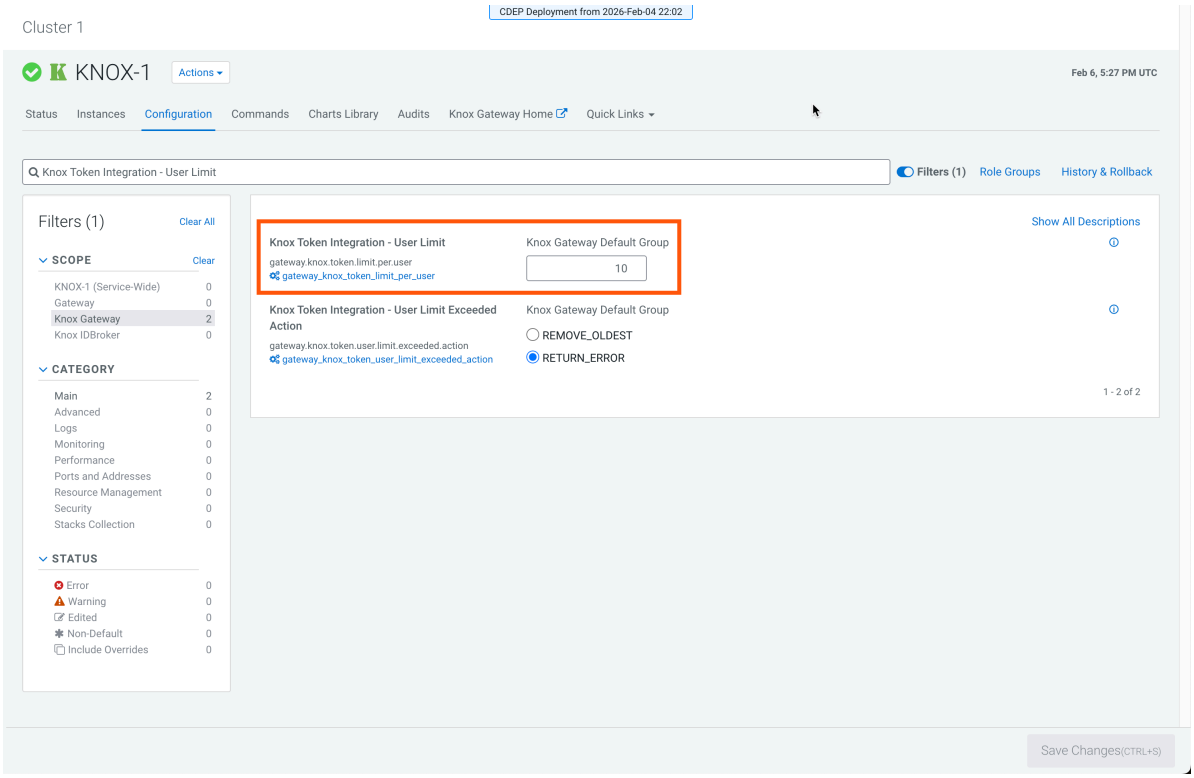
   Note:

   Users who create the Client tokens and secrets for data sharing, must be a part of the Knox admin users and groups configuration. For example, gateway.knox.admin.users = ***SHAREADMIN1***, ***SHAREADMIN2***.

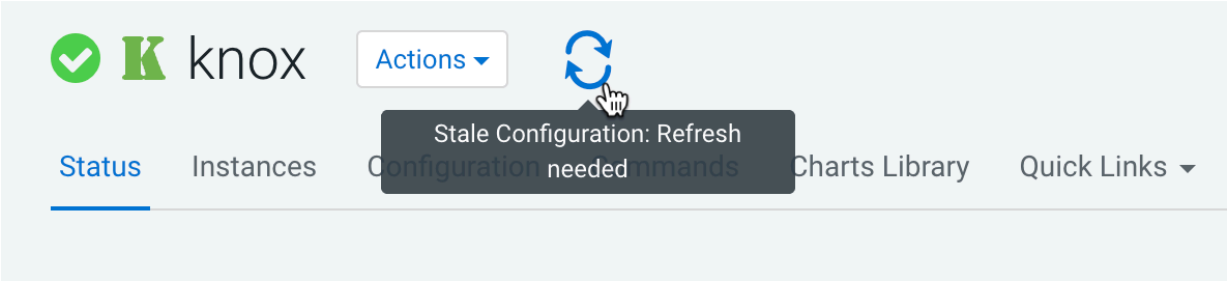
   b) Add the gateway.knox.admin.groups parameter.
2. Click Save Changes.

3. Set the Knox token limit parameter:
   a) Cloudera Manager Clusters Knox Configuration .
   b) Search for gateway.knox.token.limit.per.user, then set the value of the parameter.

Note: The value -1 means "unlimited". For example, using gateway.knox.token.limit.per.user=2 allows two external Data Share users concurrently without an expiration time limit. This setting applies to all user across all Knox topologies.



4. Click Save Changes and refresh the configuration as needed.



What to do next

Continue with configuring the Knox IDBroker.

Related Information

Editing Knox topologies

Configuring the Knox IDBroker

Learn how to configure the Knox IDBroker in Cloudera Manager.

About this task

The IDBroker must be made aware of available session policies. Configure these policies using the Cloudera Manager so that they survive restarts, upgrades, and other such events.

This policy template provides secure, read-only access to specific S3 paths for data sharing scenarios:

- Listing permissions (s3:List*): Allows listing bucket contents with prefix restrictions
- Read permissions (s3:Get*): Allows reading objects within the specified prefix path
- Dynamic scoping: The ${bucket} and ${prefix} variables are automatically substituted with the S3 bucket name and its path when REST Catalog requests credentials.
- Size optimized: The minified format ensures the policy stays under the 2048-character AWS STS plaintext limit, as well as, honors the packed policy size limit.

After REST Catalog constructs the session policy template with the data access information, the AWS Security Token Service provides the minimal required temporary credentials to read the data stored in AWS.

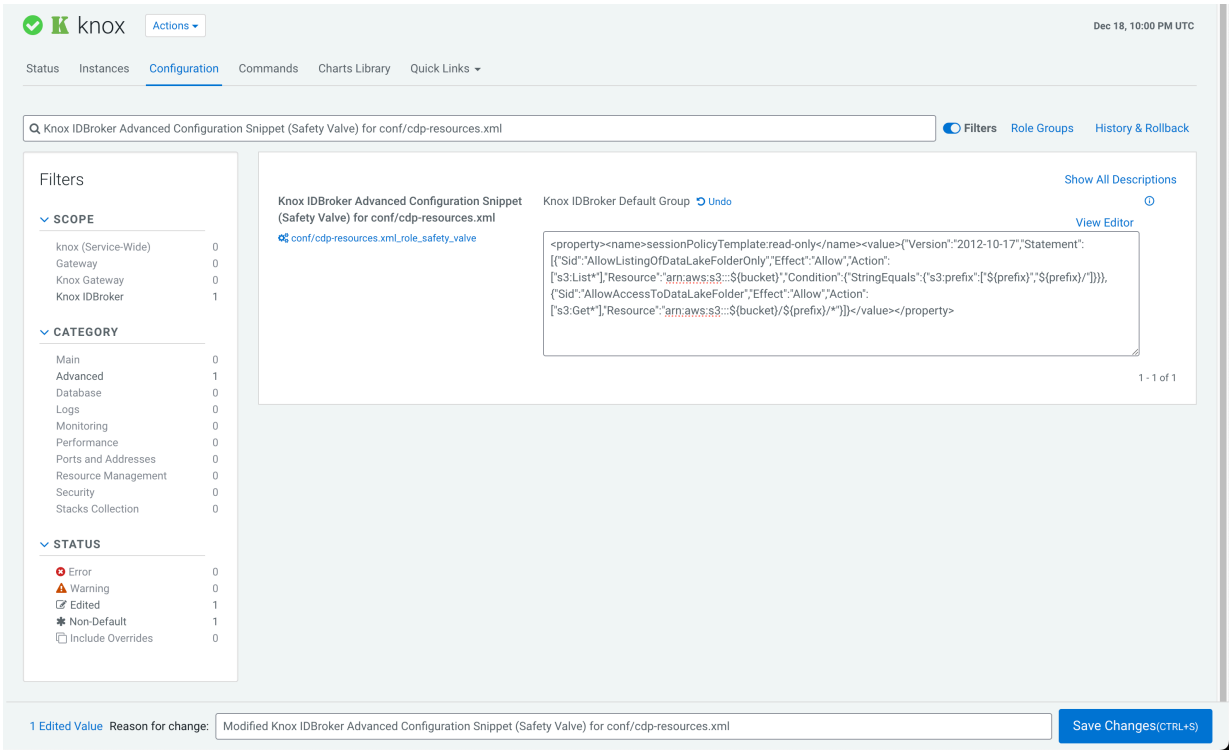
Note: The read-only policy template is automatically deployed to ${KNOX_GATEWAY_CONF_DIR}/idb-policy-templates/read-only.json during Knox IDBroker configuration. The configuration below registers this policy template so it can be applied when REST Catalog requests scoped credentials for data sharing.

Procedure

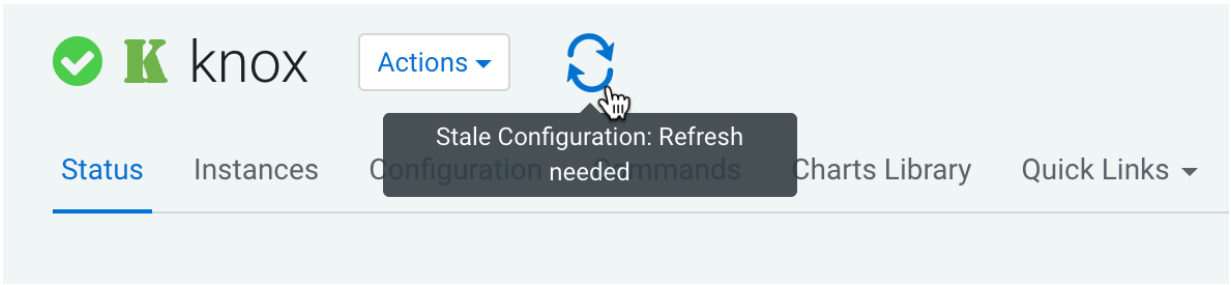
1. Go to Cloudera Manager Knox Instances Configuration Knox IDBroker Advanced Configuration Snippet (Safety Valve) for conf/cdp-resources.xml
2. After clicking View as XML, add a property named sessionPolicyTemplate:read-only with the following values:

```
<property>
<name>sessionPolicyTemplate:read-only</name>
    <value>
        {"Version":"2012-10-17",
        "Statement":[{"Sid":"AllowListingOfDataLakeFolderOnly",
        "Effect":"Allow",
        "Action":["s3:List*"],
        "Resource":"arn:aws:s3:::${bucket}",
        "Condition":{"StringEquals":{"s3:prefix":["${prefix}",
        "${prefix}/"]}}},
        {"Sid":"AllowAccessToDataLakeFolder",
        "Effect":"Allow",
        "Action":["s3:Get*"],
        "Resource":"arn:aws:s3:::${bucket}/${prefix}/*"}]}
    </value>
```

```
</property>
```



3. Save your changes and restart the Knox service.



What to do next

Continue with Creating a new Data Share.

Related Information

Creating a Data Share with CDP CLI

Creating a Data Share with CDP CLI

Learn how to register external clients in Cloudera on cloud and create Data Shares using CDP CLI commands. This process involves provisioning credentials for external users and managing data sharing through a series of CLI commands. Ensure prerequisites are met and follow the steps to securely share data assets with external users.

About this task

Resource owners or Data Share administrators who want to share their Iceberg tables in Cloudera with external clients must first register the external client in the Cloudera on cloud environment using the `cdp datacatalog create-external-users` CDP CLI command. This provisions a CLIENT_ID and CLIENT_SECRET for the external user.

After registering the external user, the resource owner creates a Data Share using the `cdp datacatalog create-data-share` CDP CLI command. The command packages specified data assets (Iceberg tables) into a shareable unit and optionally grants access to registered external users at creation time using the --external-users parameter.

The CDP CLI also provides commands to manage the full Data Share lifecycle, including listing, updating, activating, deactivating, and deleting shares, as well as managing asset membership and external user access.

Note: The following functions are all available through the Cloudera Data Catalog user interface as well.

Before you begin

- Users who run the token generation commands, must be a part of the Knox admin users and groups configuration. For more information see Knox configuration in gateway-site.xml. Having the DataShareAdmin resource role includes the knoxAdmin role. For more information, see Providing access to users.
- Run all commands within the network of your Cloudera Runtime or through a VPN.
- For Cloudera on cloud environments, you can alternatively register external users using the Cloudera Data Catalog user interface. For more information, see Creating external users.
- CDP CLI is installed and configured. For more information, see CLI client setup.

Ensure that you have the following information before performing the steps:

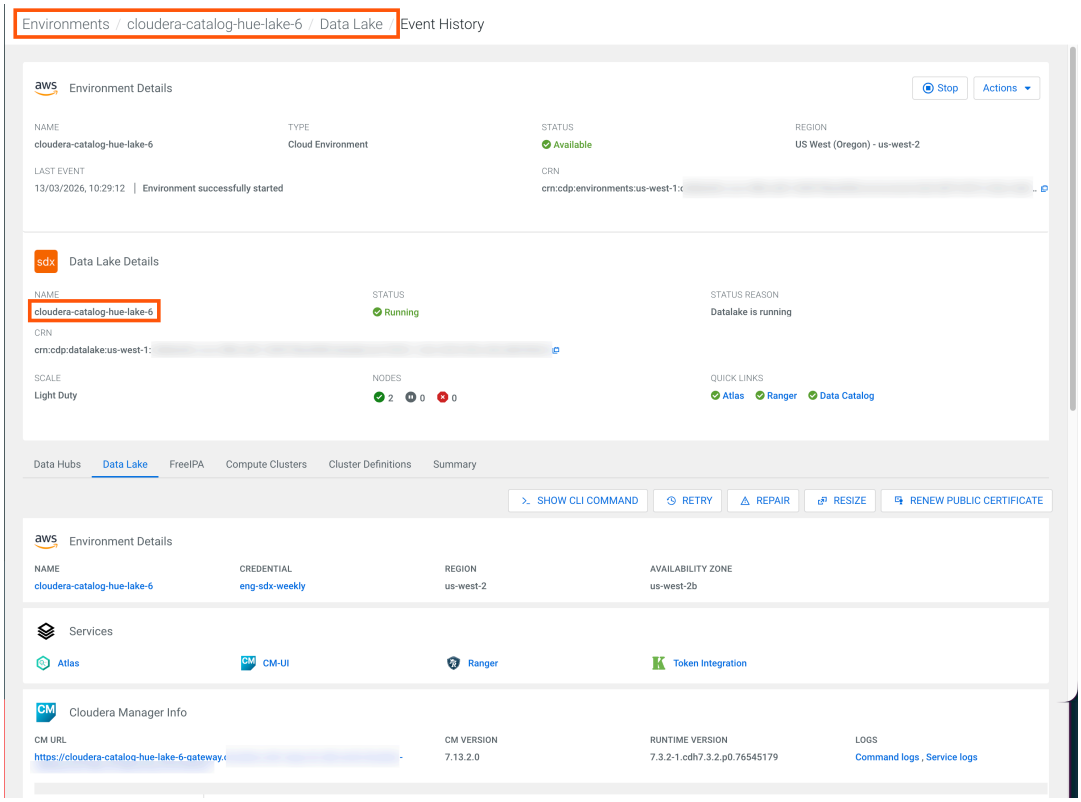
Share Admin user and password

Username and password of the Cloudera Administrator. For more information, see Cloudera account administrator.

Data Lake name

Go to Management Console Environments <***YOUR_ENVIRONMENT_NAME***> Data Lake Details and copy and make a note of the Data Lake name.

Figure 2: Data Lake Details



Data Share management commands

The following additional CDP CLI commands are available for Data Share management:

- `cdp datacatalog create-external-users` — Creates external user accounts for individuals outside Cloudera, generating a CLIENT_ID and CLIENT_SECRET for each user.
- `cdp datacatalog list-external-users` — Lists external users registered for data sharing, with optional filtering and pagination.
- `cdp datacatalog revoke-external-user-credentials` — Revokes the active credentials for an external user.
- `cdp datacatalog regenerate-external-user-credentials` — Issues a new set of credentials for an external user, invalidating the old ones.
- `cdp datacatalog delete-external-user` — Permanently deletes an external user and removes their access to all data shares.
- `cdp datacatalog create-data-share` — Creates a new data share and packages specified data assets into a shareable unit.
- `cdp datacatalog list-data-shares` — Lists all available data shares within a specified Data Lake.
- `cdp datacatalog get-data-share` — Retrieves the full details of a specific data share, including its assets and user access list.
- `cdp datacatalog update-data-share` — Updates the metadata for an existing data share, such as its name, keywords, or expiration.
- `cdp datacatalog delete-data-share` — Permanently deletes a data share.
- `cdp datacatalog share-data-share` — Activates a data share, making its assets available to the configured external users.
- `cdp datacatalog unshare-data-share` — Deactivates a data share, making its assets temporarily unavailable.
- `cdp datacatalog add-assets-to-data-share` — Adds new data assets, such as tables or views, to an existing data share.

- `cdp datacatalog remove-assets-from-data-share` — Removes one or more assets from an existing data share by resource ID.
- `cdp datacatalog grant-access-to-external-users-on-data-share` — Grants one or more external users access to a data share, with an optional expiration.
- `cdp datacatalog update-access-of-external-users-on-data-share` — Adds external users to a data share or updates their access expiration time.
- `cdp datacatalog remove-access-of-external-users-on-data-share` — Removes one or more external users' access from a specific data share.

Procedure

1. Instead using the Cloudera Data Catalog user interface, you can register external users directly using the `cdp datacatalog create-external-users` CDP CLI command:

   Run the following command to create one or more external users:

   ```
   cdp datacatalog create-external-users \
       --datalake-crn "[***DATALAKE-CRN***]" \
       --environment-crn "[***ENVIRONMENT-CRN***]" \
       --external-users email=[***EMAIL***],username=[***USERNAME***],company
   Name=[***COMPANY***]
   ```

   The command accepts the following parameters for each entry in --external-users:

   email

   Email address of the external user.

   username

   Username for the external user account.

   companyName

   Name of the external user's organization.

   On success, the command returns the created user objects including the generated credentials:

   ```
   {
       "externalUsers": [
           {
               "userId": 51,
               "username": "[***USERNAME***]",
               "email": "[***EMAIL***]",
               "companyName": "[***COMPANY***]",
               "clientId": "[***CLIENT_ID***]",
               "secret": "[***SECRET***]",
               "createdAt": "2025-07-29T14:07:05.742000+00:00",
               "error": ""
           }
       ]
   ```

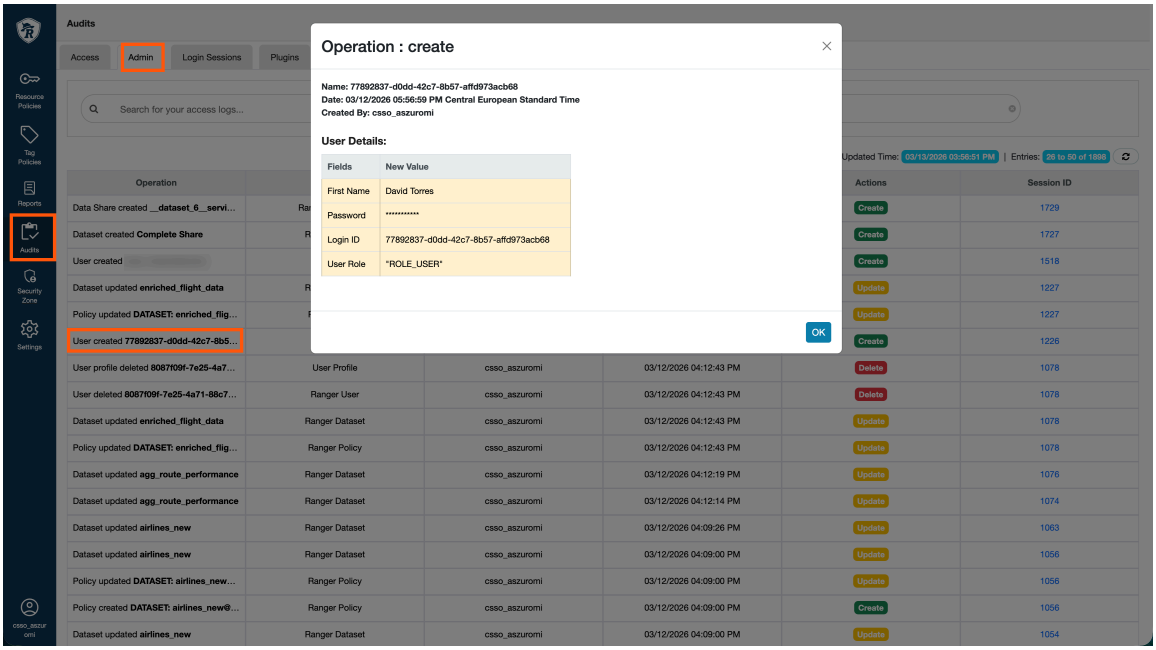
```
}
```

Important: The clientId and secret values in the response are generated credentials and are displayed only once. Securely store this information immediately and share it with the external user. If the credentials are lost, you must regenerate them.

- To verify via Ranger Audits:

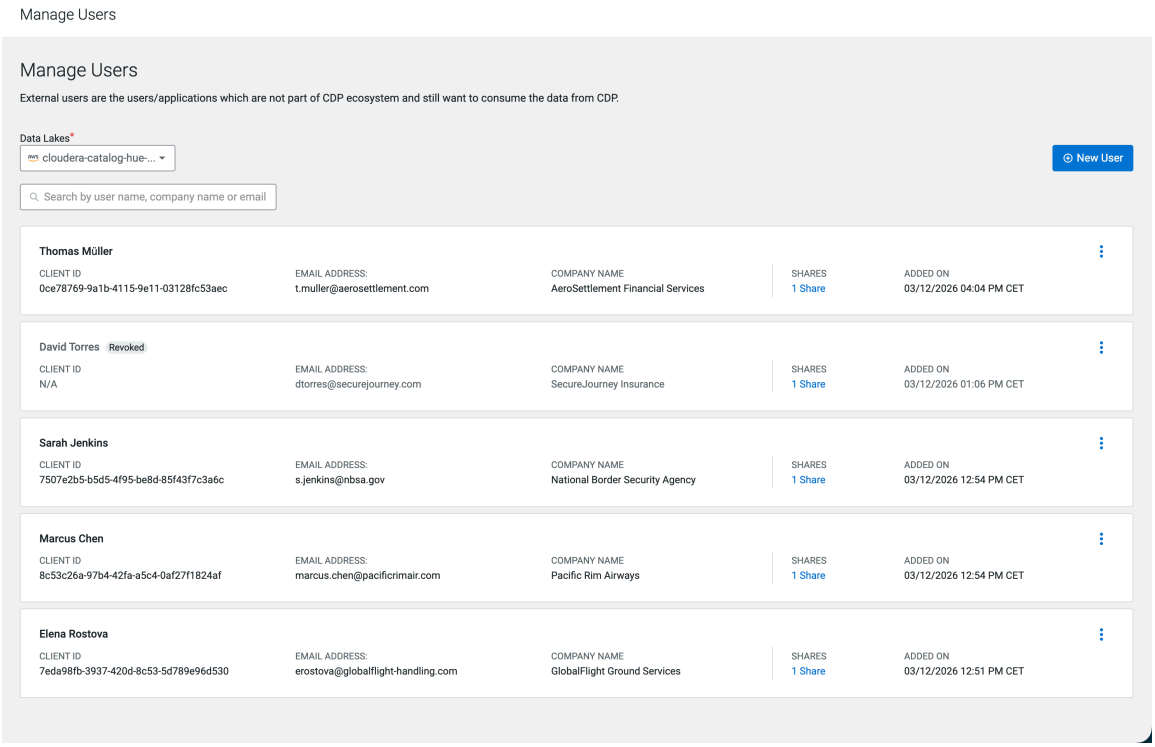
  a. To verify if the CLIENT_ID (Token ID) is generated successfully, go to Cloudera Management Console *[***ENVIRONMENT-NAME***] [***DATALAKE-NAME***]* Ranger Audits Admin . Your external user creation events are shown as User created *[***CLIENTID***]*.

Figure 3: Client ID verification in Ranger

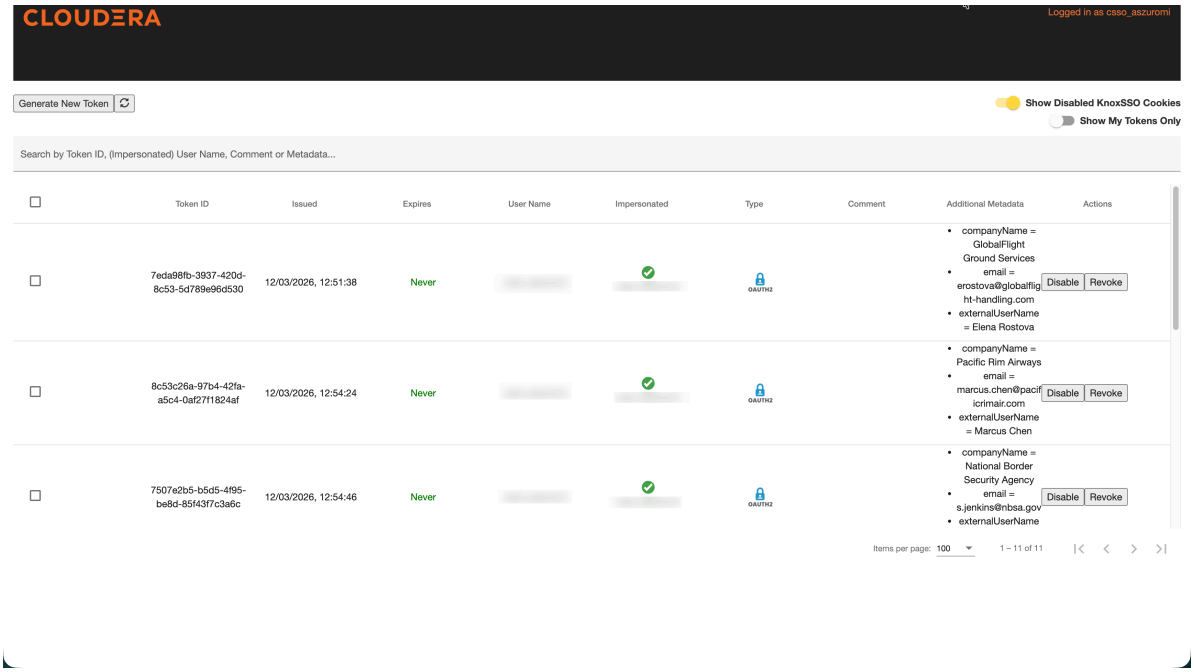


- To verify via Data Catalog user interface:
  a. In Cloudera Data Catalog, go to **Manage Users**.
  b. Select the target **Data Lake** from the dropdown at the top of the page. The user list shows all existing external users for the selected Data Lake, including their Client ID, email address, company name, associated shares, and registration date.

Figure 4: List of external users



- External user Client IDs are also visible in Knox under Cloudera Manager Environment *[***YOUR_DATALAKE***]* Token Integration Token Management



-

2. After verifying the Client IDs, create a Data Share using the CDP CLI command:

```
cdp datacatalog create-data-share \
    --datalake-crn "[***DATALAKE-CRN***]" \
    --environment-crn "[***ENVIRONMENT-CRN***]" \
    --data-share-name "[***DATA-SHARE-NAME***]" \
    --assets databaseName=[***DATABASE-NAME***],tableName=[***TABLE-
NAME***],guid=[***ASSET-GUID***]
```

The command accepts the following required parameters:

--datalake-crn

The CRN of the source Data Lake.

--environment-crn

The CRN of the associated CDP environment.

--data-share-name

A unique name for the new data share (maximum 512 characters).

--assets

The list of data tables to include in the share, specified as databaseName=*[***DATABASE-NAME***]*,tableName=*[***TABLE-NAME***]*,guid=*[***ASSET-GUID***]*. Separate multiple entries with spaces.

You can also specify the following optional parameters: --summary, --terms-of-use, --keywords, --expiry-time, and --external-users. Use --external-users to grant access to one or more external users at creation time. Each entry uses the shorthand format externalUserId=*[***USER-ID***]*,expiryTime=*[***YYYY-MM-DDTHH:MM:SS***]*. Separate multiple entries with spaces. The externalUserId is the userId returned when creating external users with `cdp datacatalog create-external-users`.

```
cdp datacatalog create-data-share \
    --datalake-crn "crn:cdp:datalake:us-west-1:..." \
    --environment-crn "crn:cdp:environments:us-west-1:..." \
    --data-share-name "q3_marketing_data" \
    --assets databaseName=marketing,tableName=campaign_results,guid=5eb2d
66b-d47d-402b-a26c-de77b403ef6a \
    --summary "Q3 marketing campaign results" \
    --expiry-time 2025-08-20T06:30:00 \
    --external-users externalUserId=344
```

On success, the command returns a JSON object with the identifiers for the new share:

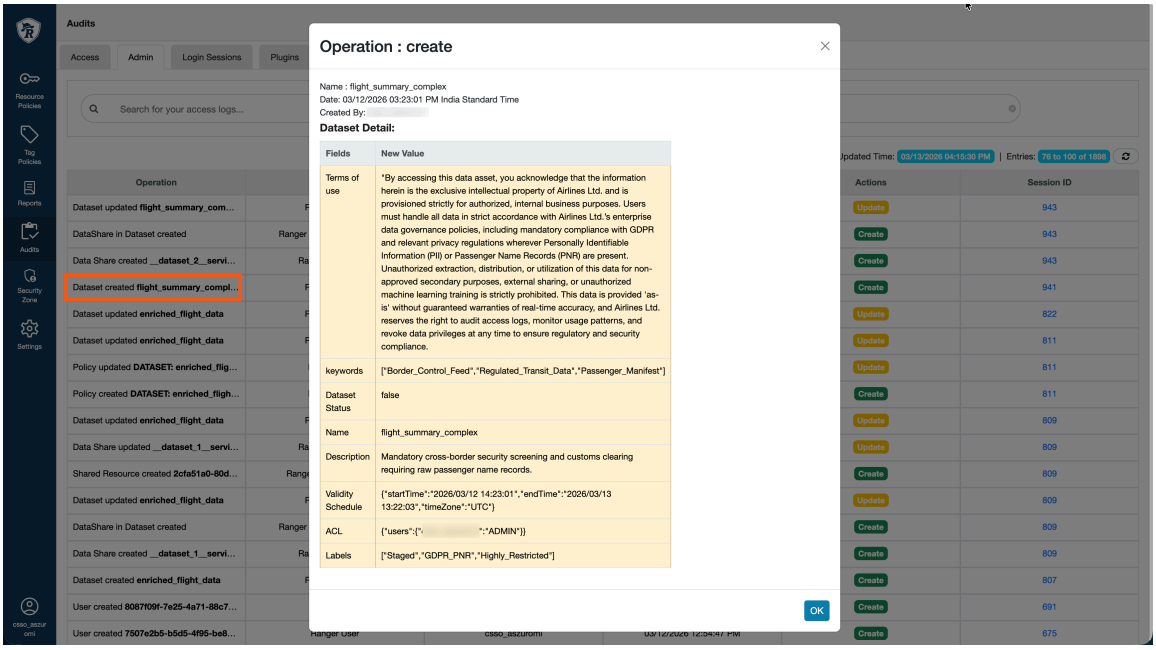
```
{
    "dataShareId": 1,
    "dataShareName": "q3_marketing_data",
    "datalakeCrn": "crn:cdp:datalake:us-west-1:..."
```

```
}
```

Note: For complex shares with many assets or users, use a JSON input file. Generate a template with cdp datacatalog create-data-share --generate-cli-skeleton > share_input.json, then run the command with --cli-input-json file://share_input.json.

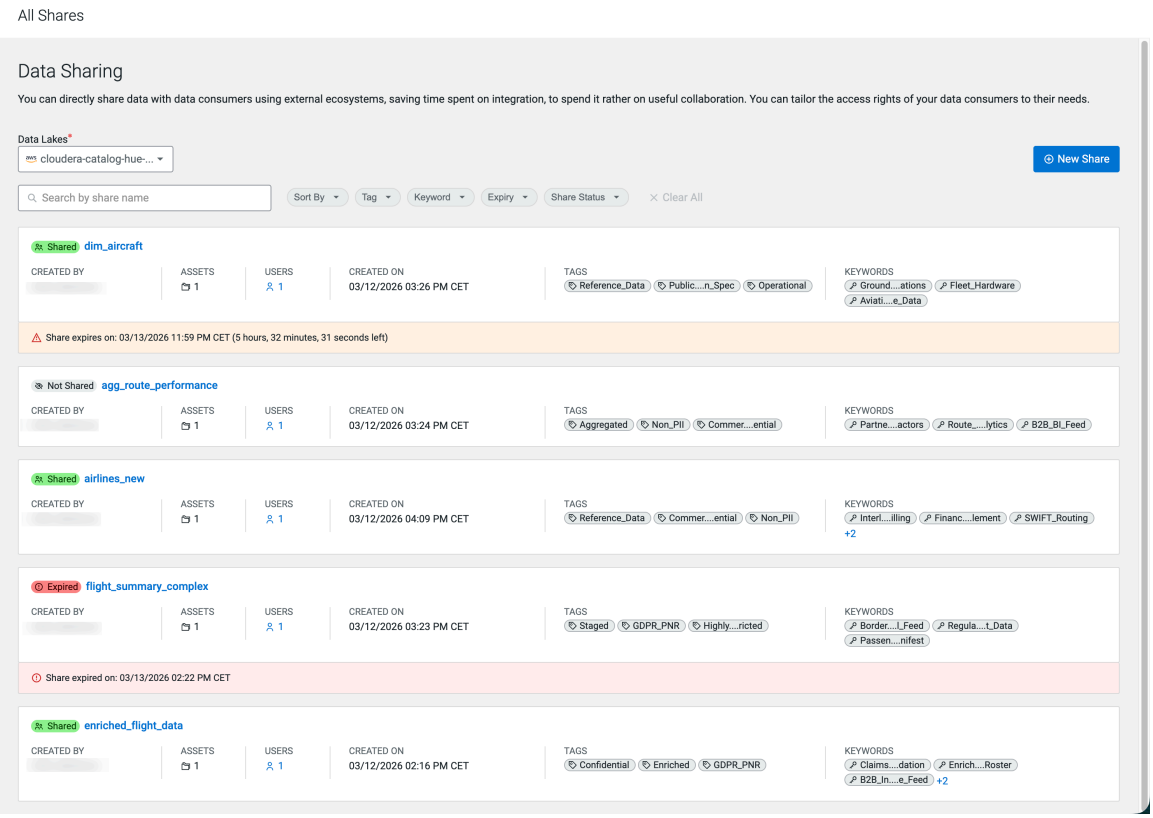
- To verify via Ranger Audits:
  a. To verify if the Data Share is generated successfully, go to Cloudera Management Console *[***ENVIRONMENT-NAME***] [***DATALAKE-NAME***]* Ranger Audits Admin . Your Data Share creation events are shown as three consecutive Ranger audit events:
    - DataShare in Dataset created
    - Data Share created ****DATASET SERVICE NAME****
    - Dataset create ****DATASHARE NAME****

Figure 5: Data Share verification in Ranger



- To verify via Data Catalog user interface:
  a. In Cloudera Data Catalog, go to **Data Sharing**. The new Data Share is listed on the **All Shares** page in Not Shared status until you publish it.

Figure 6: All Shares page



Related Information

Creating a new Data Share

Importing the Cloudera certificate in the Spark cluster

If your environment contains a custom set of trusted certificate authorities, you need to import the Cloudera certificate to your environment and apply it to the node where Spark is running.

Before you begin

Note: Run all commands within the network of your Cloudera Runtime or through a VPN.

Procedure

1. Run the following command to export a certificate from the Cloudera on cloud environment:

```
openssl s_client -showcerts -connect [***CDP-PUBLIC-CLOUD-HOSTNAME***]:443
 </dev/null 2>/dev/null | openssl x509 -outform PEM > share.pem
```

2. Convert the created share.pem file to share.cert by running the following command:

```
openssl x509 -outform der -in share.pem -out share.cert
```

3. Import the certificate into the Java Virtual Machine (JVM) of the Spark cluster.

```
keytool -importcert -alias cdpcert -keystore /usr/lib/jvm/java-11-openjdk-
amd64/lib/security/cacerts -file ${CERT_HOME}/share.cert -storepass change
it -noprompt
```

Note: The JAVA path, /usr/lib/jvm/java-11-openjdk-arm64/lib/security/cacerts is different for ARM-based machines.

Supported REST Catalog APIs for accessing the data

To use the Iceberg data sharing in Cloudera, you must employ the REST client to perform specific operations.

The REST APIs from the specification defined by Apache Iceberg are available in the REST Catalog open API specification. Cloudera currently supports the REST APIs that allow read operations on Iceberg tables:

Retrieving the access token

Retrieving the access token works the same for all supported endpoints:

```
[***MY-TOKEN-NAME***]=$(curl -k -X POST -H  "Content-Type: applic
ation/x-www-form-urlencoded" -d "client_id=[***CLIENT ID***]&cli
ent_secret=[***CLIENT SECRET***]&grant_type=client_credentials" "htt
ps://[***DATALAKE-LOADBALANCER***]/[***DATALAKE-NAME***]/cdp-share-access/hm
s-api/icecli/v1/oauth/tokens"  | jq -r '.access_token')
```

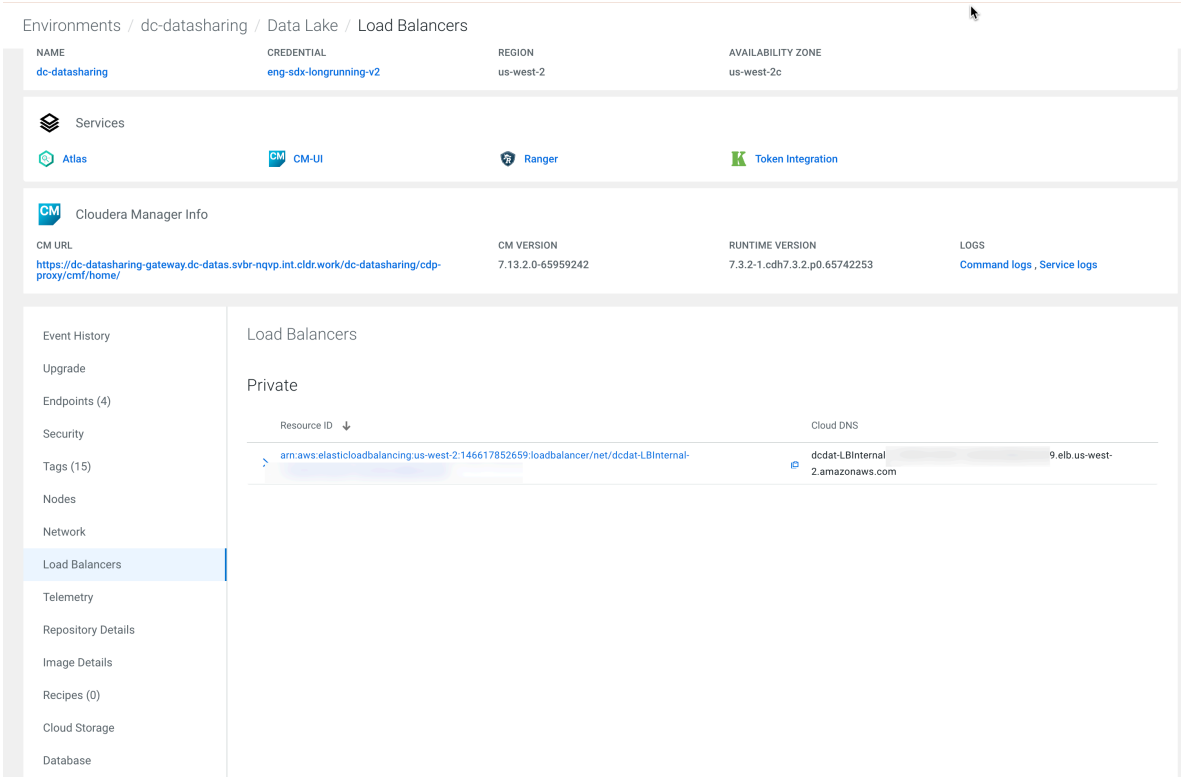
Using the endpoints must be always preceded by retrieving the access token.

List Databases: `/v1/{prefix}/namespaces`

List all namespaces at a certain level, optionally starting from a given parent namespace.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -
H "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***DATALAKE-
LOADBALANCER***]/[***DATALAKE-NAME***]/cdp-share-access/hms-api/icecli/v1/
namespaces
```

Note: The *[***DATALAKE-LOADBALANCER***]* DNS can be found via Cloudera Management Console in the **Load Balancers** section.



Example:

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-urlencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=WWpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5TmpneE1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_credentials" "https://apr24-LBInternal-1745472822408-fddbd702bfadbe45.elb.us-west-2.amazonaws.com/apr24-env2-dl/cdp-share-access/hms-api/icecli/v1/oauth/tokens"  | jq -r '.access_token')

curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Authorization: Bearer $My-token-name" https://apr24-LB-my-datalake-load-balancer.elb.us-west-2.amazonaws.com/my-datalake-name/cdp-share-access/hms-api/icecli/v1/namespaces
```

List Tables: /v1/{prefix}/namespaces/{namespace}/tables

List all table identifiers under the specified namespace.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -
H "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***DATALAKE-
LOADBALANCER***]/[***DATALAKE-NAME***]/cdp-share-access/hms-api/icecli/v1/
namespaces/<namespace>/tables
```

Example:

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-urlencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=WWpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5TmpneE1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_credentials" "https://apr24-LB-my-datalake-load-balancer.elb.us-west-2.amazona
```

```
ws.com/my-datalake-name/cdp-share-access/hms-api/icecli/v1/oauth/tokens"  |
jq -r '.access_token')

curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Au
thorization: Bearer $My-token-name" https://apr24-LB-my-datalake-load-balanc
er.elb.us-west-2.amazonaws.com/my-datalake-name/cdp-share-access/hms-api/ice
cli/v1/namespaces/hive_rest_airline_orc/tables
```

Load Tables: /v1/{prefix}/namespaces/{namespace}/tables/{table}

Load a table from the catalog.

```
curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -
H "Authorization: Bearer $[***MY-TOKEN-NAME***]" https://[***DATALAKE-
LOADBALANCER***]/[***DATALAKE-NAME***]/cdp-share-access/hms-api/icecli/v1/
namespaces/<namespace>/tables/icebergtable
```

Example:

```
My-token-name=$(curl -k -X POST -H  "Content-Type: application/x-www-form-ur
lencoded" -d "client_id=b9efc3dd-3695-4867-9e4c-523389c8a78b&client_secret=W
WpsbFptTXpaR1F0TXpZNU5TMDBPRFkzTFRsbE5HTXROVEl6TXpnNVl6aGhOemhpOjpNRGt5Tmpne
E1HUXRZak00TmkwMFpqTXhMVGczTTJZdFptTXhOekl6TmpVNFlqazI=&grant_type=client_cr
edentials" "https://apr24-LB-my-datalake-load-balancer.elb.us-west-2.amazona
ws.com/my-datalake-name/cdp-share-access/hms-api/icecli/v1/oauth/tokens"  |
jq -r '.access_token')

curl -ivk -X GET -H "Content-Type: application/x-www-form-urlencoded" -H "Au
thorization: Bearer $My-token-name" https://apr24-my-datalake-load-balancer.
elb.us-west-2.amazonaws.com/my-datalake-name/cdp-share-access/hms-api/icecli
/v1/namespaces/hive_rest_airline_orc/tables/airport_iceberg_external
```

Related Information

Sample Spark workload to access data

Data Sharing longevity test results

Iceberg REST Catalog API specification