Cloudera Data Warehouse Public Cloud

Querying Data

Date published: 2024-01-01
Date modified: 2024-01-01

https://docs.cloudera.com/

Legal Notice

Contents

Querying data in Cloudera Data Warehouse

This topic describes how to query data in your Virtual Warehouse on Cloudera Data Warehouse (CDW).

About this task

The CDW service includes the Hue SQL editor that you can use to submit queries to Virtual Warehouses. For example, you can use Hue to submit queries to an Impala Virtual Warehouse. For detailed information about using Hue, see Using Hue.

Procedure

1. Log in to the Data Warehouse service as DWUser

   The **Overview** page is displayed.
2. Click Hue on the Virtual Warehouse tile.
3. Enter your query into the editor and submit it to the Virtual Warehouse.

Related Information

Submitting queries with Hue

Submitting queries with Hue

You can write and edit queries for Hive or Impala Virtual Warehouses in the Cloudera Data Warehouse (CDW) service by using Hue.

About this task

For detailed information about using Hue, see Using Hue.

Required role: DWUser

Before you begin

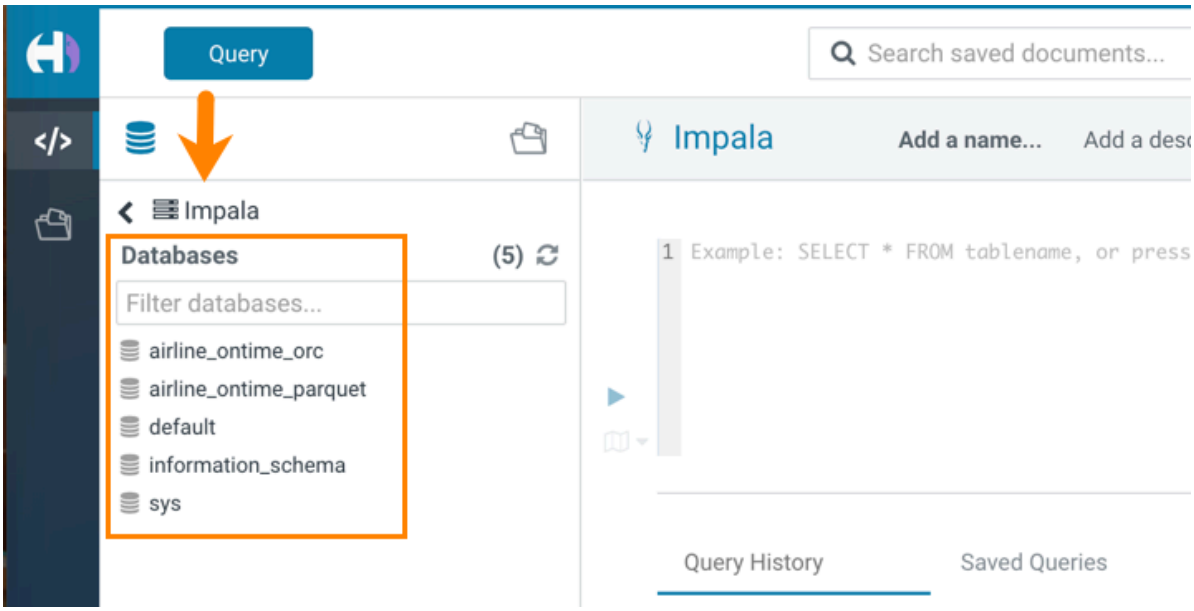
Hue uses your LDAP credentials that you have configured for the CDP Public Cloud cluster.

Procedure

1. Log into the CDP web interface and navigate to the Data Warehouse service.
2. In the Data Warehouse service, navigate to the **Overview** page.

   Note: You can also launch Hue from the **Virtual Warehouse** page using the same steps.

3. To run Impala queries:
   a) Click HUE on the Impala Virtual Warehouse tile.
      The query editor is displayed:

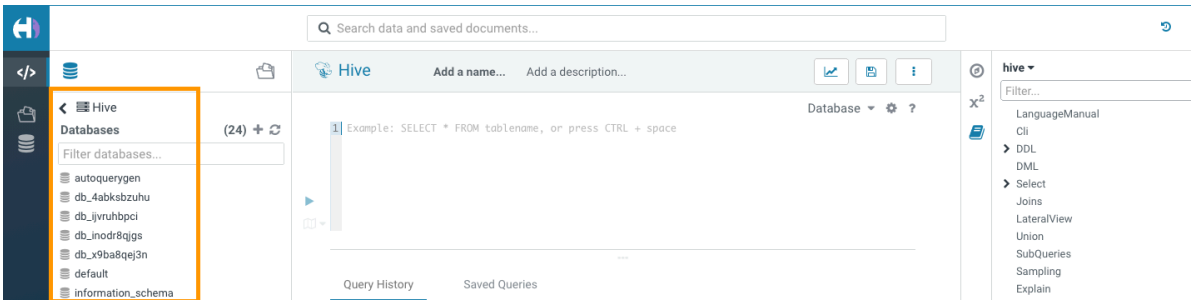


   b) Click a database to view the tables it contains.
      When you click a database, it sets it as the target of your query in the main query editor panel.
   c) Type a query in the editor panel and click the run icon to run the query.

      You can also run multiple queries by selecting them and clicking .

      Note: Use the Impala language reference to get information about syntax in addition to the SQL auto-complete feature that is built in. To view the language reference, click the book icon to the right of the query editor panel.

4. To run Hive queries:
   a) Click HUE on the Hive Virtual Warehouse tile.
      The Hive query editor is displayed:



   b) Click a database to view the tables it contains.
      When you click a database, it sets it as the target of your query in the main query editor panel.
   c) Type a query in the editor panel and click the run icon to run the query.

      You can also run multiple queries by selecting them and clicking .

      Note: Use the Hive language reference to get information about syntax in addition to the SQL auto-complete feature that is built in. To view the language reference, click the book icon to the right of the query editor panel.

Related Information

Hue overview
Using Hue

Configuring custom properties using safety valves

You can configure Hue properties that are not directly exposed through the Cloudera Data Warehouse (CDW) web UI by specifying them in the Hue Configuration called hue-safety-valve for a Virtual Warehouse. These configurations are stored in the hue.ini file.

About this task

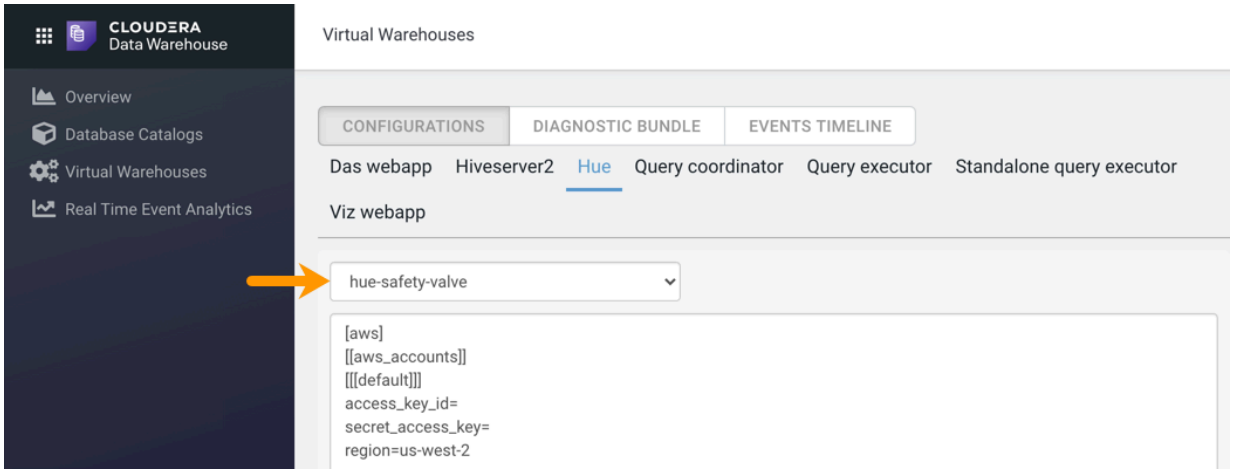
Required role: DWAdmin

Procedure

1. In the **Data Warehouse Overview** page, click the edit icon for a Virtual Warehouse associated with Hue.

2. On the **Virtual Warehouse details** page, click the Hue tab and select hue-safety-valve from the drop-down menu:

   Figure 1: Custom configurations for Hue



3. Specify the custom configuration properties in the space provided as shown in the following examples:
   For AWS:

```
[aws]
  [[aws_accounts]]
    [[[default]]]
      access_key_id=<access key id>
      secret_access_key=<secret access key>
      region=<aws region>
```

   For Azure:

```
[azure]
  [[azure_accounts]]
    [[[default]]]
      client_id=<client id>
      client_secret=<client secret>
      tenant_id=<tenant id>
```

```
[azure]
  [[abfs_clusters]]
     [[[default]]]
       fs_defaultfs=abfs://<container name>@<storage account>.dfs.core.wi
ndows.net
       webhdfs_url=https://<storage account>.dfs.core.windows.net/
```

4. Click Apply in the upper right corner of the page.

Creating tables by importing CSV files from AWS S3

You can create tables in Hue by importing CSV files stored in S3 buckets. Hue automatically detects the schema and the column types, thus helping you to create tables without using the CREATE TABLE syntax.

About this task

Only Hue Superusers can access S3 buckets and import files to create tables. To create tables by importing files from S3, you must assign and authorize use of a specific bucket on S3 bucket for your environment. The bucket then appears like a home directory on the Hue web interface.

The maximum file size supported is three gigabytes.

Before you begin

1. Create roles and synchronize users to FreeIPA.

   Make sure that the users who want to create tables by importing CSV files have EnvironmentAdmin or EnvironmentUser roles, and that you have synchronized users to FreeIPA. This is required for users accessing the data lake. Ranger will allow these users to access the additional bucket.

   To assign roles to users, see Assigning resources to users in the Management Console documentation.

   To synchronize users to FreeIPA, see Performing user sync in the Management Console documentation.
2. Add an S3 bucket to your CDW environment.

   If you add an S3 bucket and then try to access it from the Hue web interface without adding it to the CDW environment, then Impala or Hive may display the “AccessDeniedException 403” exception. Make sure that your Cloudera Data Warehouse (CDW) environment has access to the S3 buckets that you want to access from Hue.

   When you create a Virtual Warehouse in the CDW service, a cluster is created in your AWS account. Two buckets are generated:

   - One bucket is used for managed data
   - The other bucket is used for external data.

   Access to these buckets is controlled by AWS instance profiles. To add read/write access to S3 buckets under the same AWS account as the CDW service cluster, or under a different account, see the corresponding links at the bottom of the page.

   To add an S3 bucket to your CDW environment:

   a. In the Data Warehouse service, in Environments locate the environment in which you want to add the S3 bucket.
   b. Click ⋮ Edit
   c. In the **Environment Details** page, in Configurations, Add External S3 Bucket, type the name of the AWS bucket you want to configure access to..

      If the bucket belongs to another AWS account, then select the Bucket belongs to different AWS account option.
   d. Select the access mode.

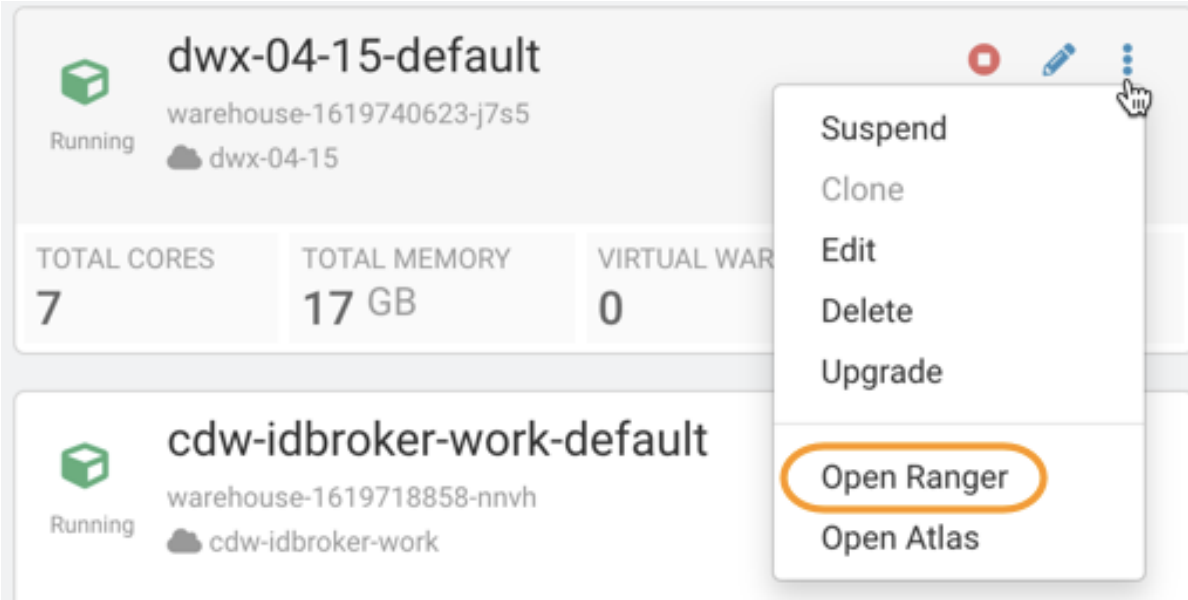
      Read-only access is sufficient to import data in Hue.
   e. Click Add Bucket to save the configuration. A success message displays at the top of the page.
   f. Click Apply to update the CDW environment.

      Note: If you configure read only access to an S3 bucket you add, there is no need to restart Virtual Warehouses. However, if you configure read/write access to an external S3 bucket, you must restart Virtual Warehouses by suspending them and starting them again.

3. Add users to the Hadoop SQL Ranger policies

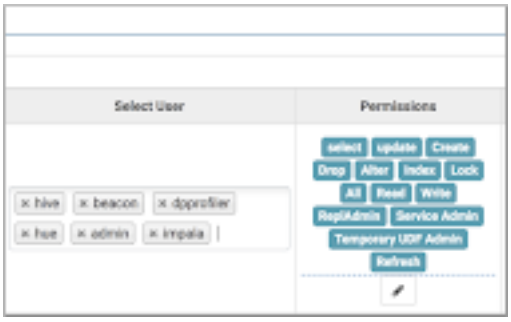
   You must grant the Hadoop SQL Ranger permissions to enable your users to access specific tables and secure your data from unauthorized access.

   a. Sign in to Cloudera Data Warehouse.
   b. Click the More… option on your Database Catalog and click Open Ranger.



   c. On the **Ranger Service Manager** page, click Hadoop SQL.
   d. Select the all - url policy.

      The **Edit Policy** page is displayed.
   e. Under the Add Conditions section, add the users under the Select User column and add permissions such as Create, Alter, Drop, Select, and so on from the Permissions column.



   f. Scroll to the bottom of the page and click Save.

4. Enable the S3 file browser in Hue.

   To enable access to S3 buckets from the Hue web UI, you must add the AWS environment details in the hue-safety-valve configuration from your Virtual Warehouse. After enabling the S3 file browser, you can browse the S3 buckets, create folders, and upload files from your computer, and import files to create tables.

   a. Sign in to Cloudera Data Warehouse.
   b. Go to the Virtual Warehouse from which you want to access the S3 buckets and click its edit icon.
   c. On the Virtual Warehouses detail page, click the Hue tab and select hue-safety-valve from the drop-down menu.
   d. Add the following configuration for Hive or Impala Virtual Warehouse in the space provided:

      For Hive Virtual Warehouse:

```
[desktop]
```

```
# Remove the file browser from the blocked list of apps.
# Tweak the app_blacklist property to suit your app configuration.
app_blacklist=oozie,search,hbase,security,pig,sqoop,spark,impala

[aws]
[[aws_accounts]]
[[[default]]]
access_key_id=[***AWS-ACCESS-KEY***]
secret_access_key=[***SECRET-ACCESS-KEY***]
region=[***AWS-REGION***]

[filebrowser]
# (Optional) To set a specific home directory path:
remote_storage_home=s3a://[***S3-BUCKET-NAME***]
```

For Impala Virtual Warehouse:

```
[desktop]

# Remove the file browser from the blocked list of apps.
# Tweak the app_blacklist property to suit your app configuration.
app_blacklist=spark,zookeeper,hive,hbase,search,oozie,jobsub,pig,sqoop,s
ecurity
[aws]
[[aws_accounts]]
[[[default]]]
access_key_id=[***AWS-ACCESS-KEY***]
secret_access_key=[***SECRET-ACCESS-KEY***]
region=[***AWS-REGION***]

[filebrowser]
# (Optional) To set a specific home directory path:
remote_storage_home=s3a://[***S3-BUCKET-NAME***]
```

e. Click Apply in the upper right corner of the page.

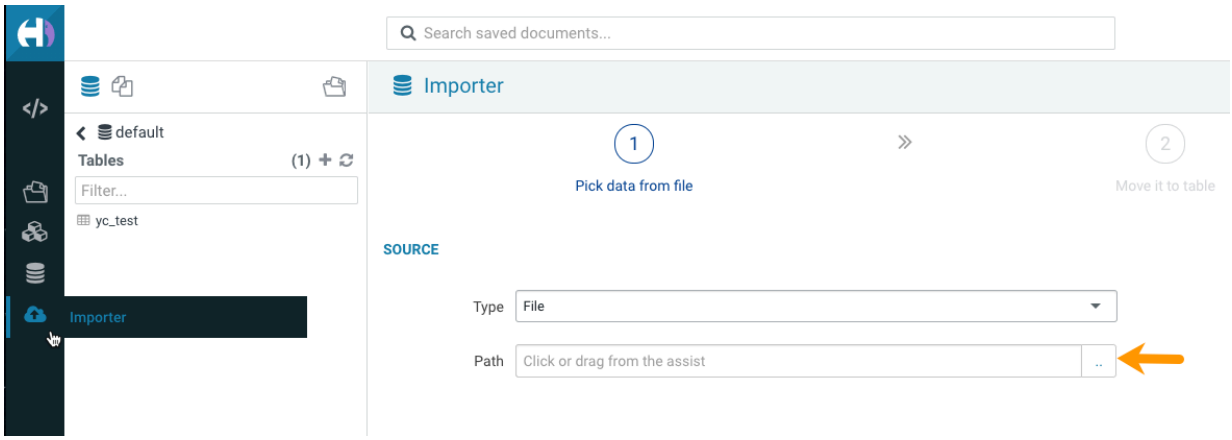
The S3 File Browser icon appears on the left Assist panel on the Hue web UI after the Virtual Warehouse restarts.

To import the CSV file and create the table:

Procedure

1. In the CDW service **Overview** page, select the Virtual Warehouse in which you want to create the table, click the options menu in the upper right corner and click Open Hue.
2. From the left assist panel, click on Importer.

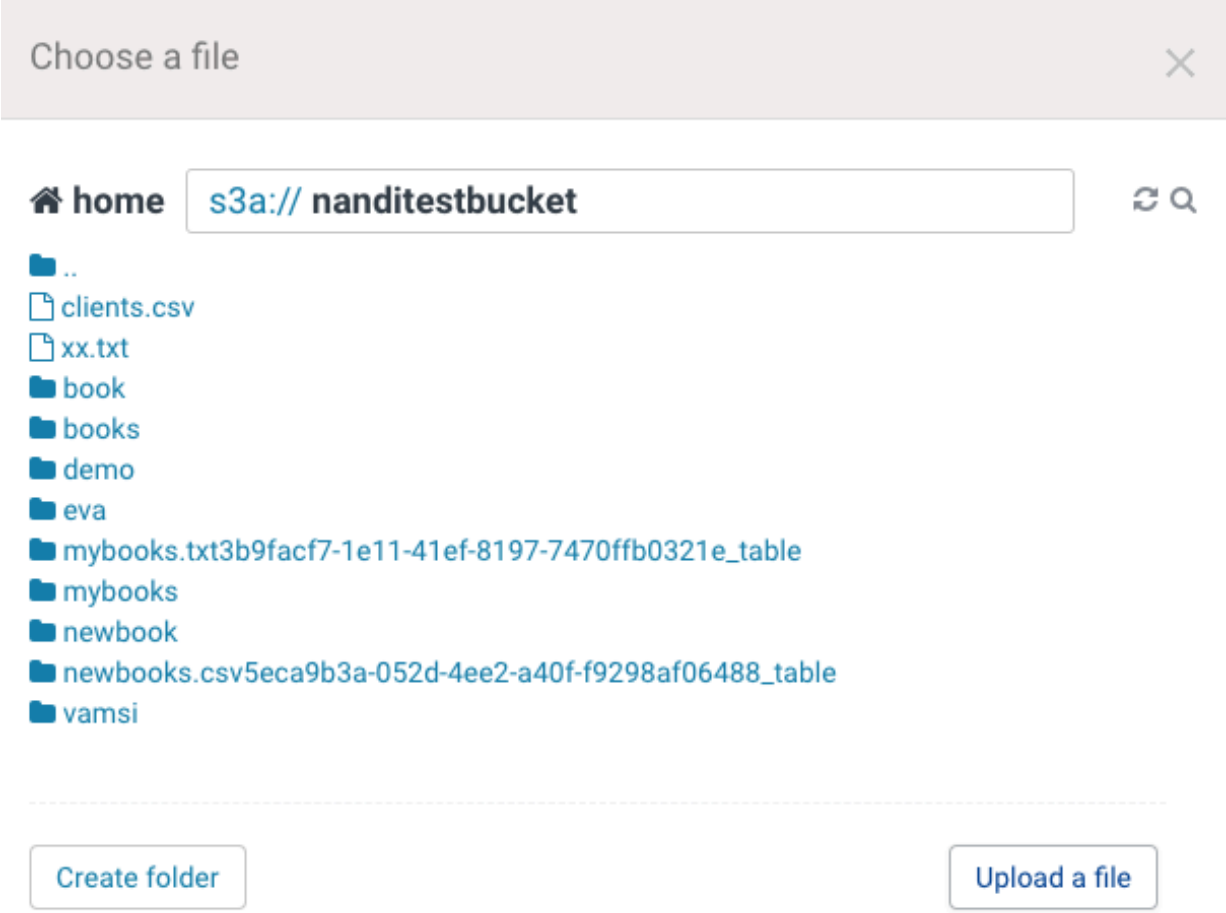
3. On the Importer screen, click .. at the end of the Path field:



Choose a file pop-up is displayed.

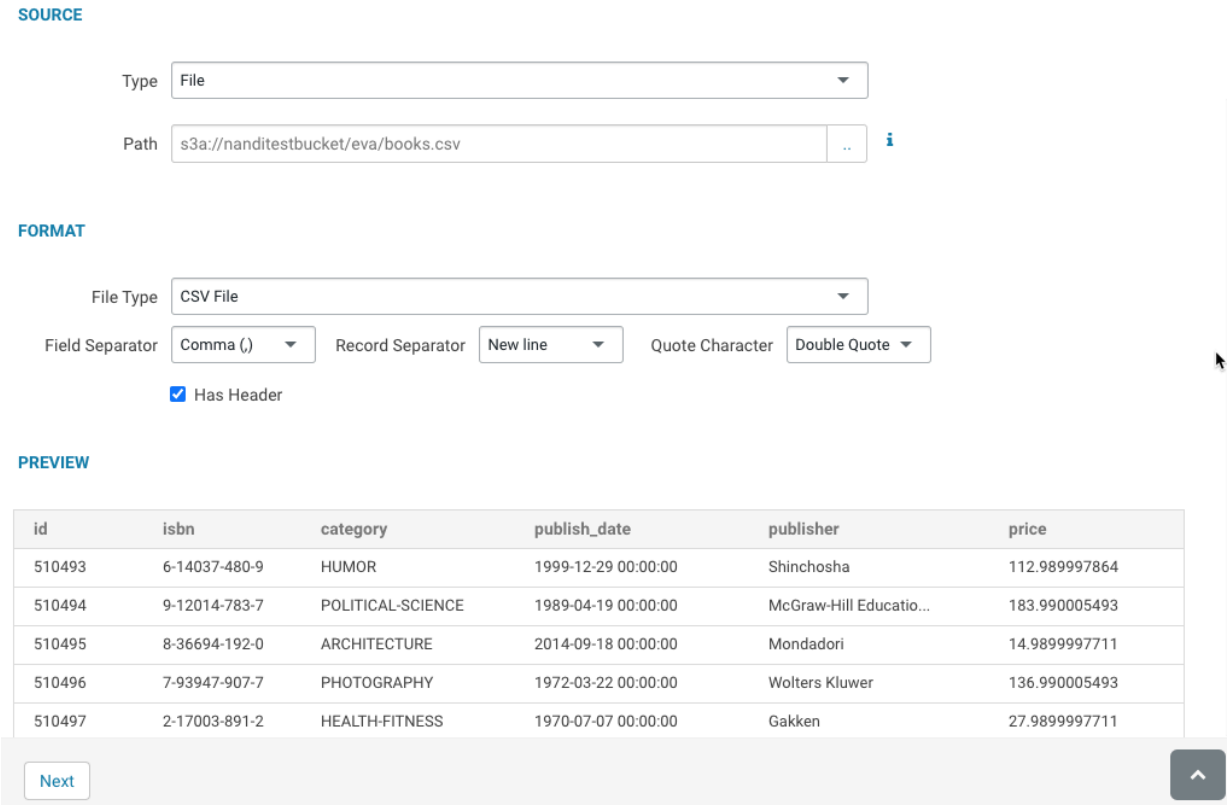
4. Type s3a:// in the address text box and press enter.

The S3 buckets associated with the CDW environment are displayed. You can narrow down the list of results using the search option.



If the file is present on your computer, then you can upload it to S3 by clicking Upload a file. To do this, you must have enabled read/write access to the S3 bucket from the CDW environment.

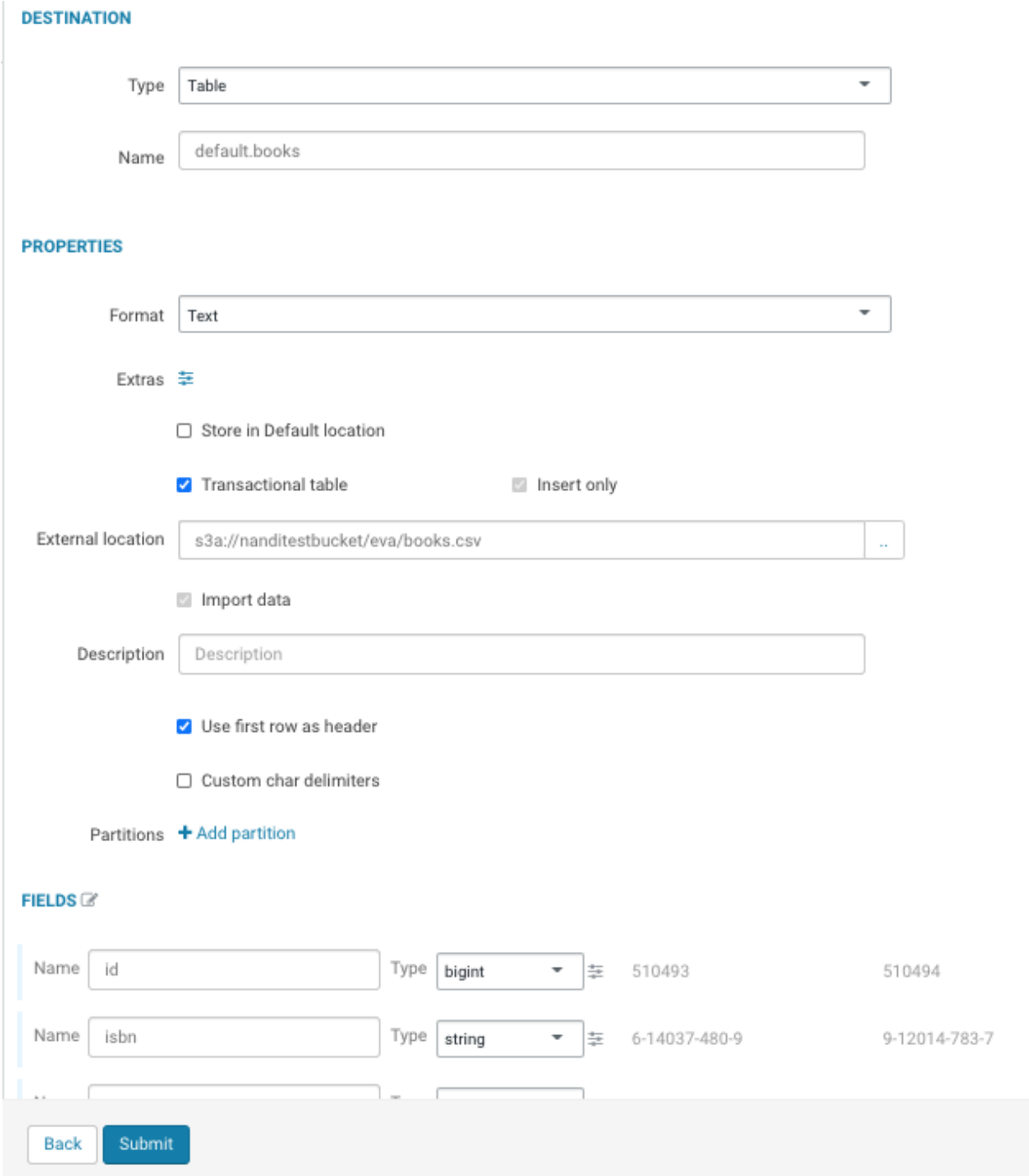
5. Select the CSV file that you want to import into Hue.
   Hue displays the preview of the table along with the format:



Hue automatically detects the field separator, record separator, and the quote character from the CSV file. If you want to override a specific setting, then you can change it by selecting a different value from the dropdown menu.

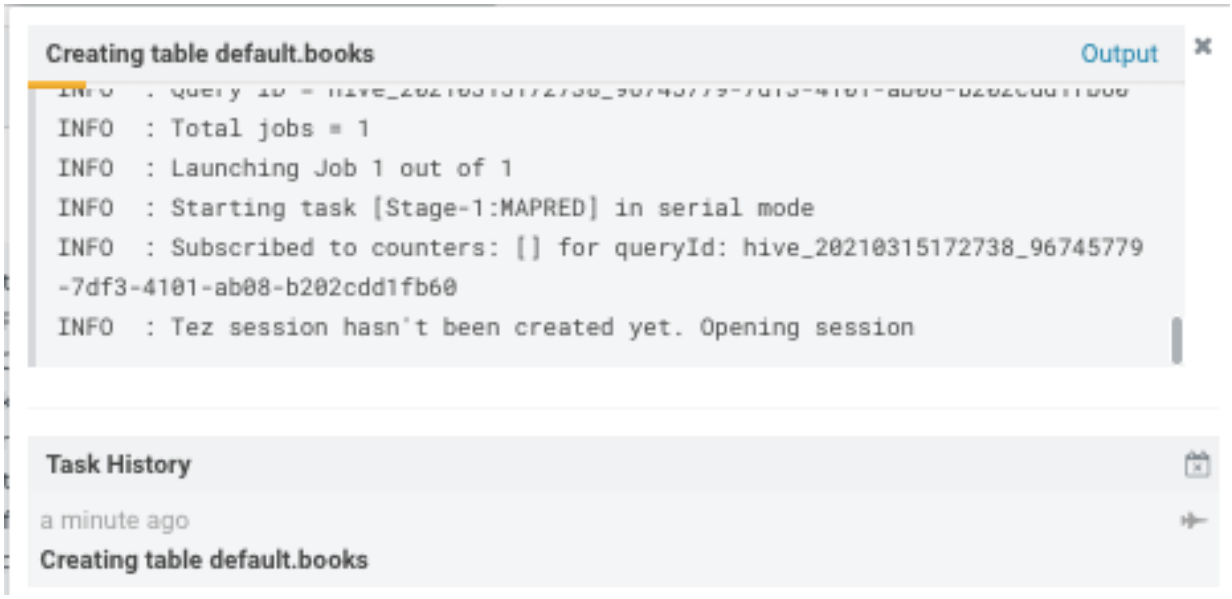
6. Click Next.

On this page, you can set the table destination, partitions, and change the column data types.



7. Verify the settings and click Submit to create the table.

   The CREATE TABLE query is triggered:



   Hue displays the logs and opens the **Table Browser** from which you can view the newly created table when the operation completes successfully.

Related Information

Assigning resources to users

Performing user sync

Adding access to external S3 buckets for CDW clusters on AWS

Adding CDW cluster access to external S3 buckets in the same AWS account

Creating tables by importing CSV files from ABFS

You can create tables in Hue by importing CSV files stored in ABFS. Hue automatically detects the schema and the column types, thus helping you to create tables without using the CREATE TABLE syntax.

About this task

Only Hue Superusers can access ABFS and import files to create tables.

The maximum file size supported is three gigabytes.

Before you begin

1. Create a storage account from the Microsoft Azure portal. On the Create storage account Advanced page of the Azure portal, enable Data Lake Storage Gen2 so that the objects and files within your account can be organized into a hierarchy of directories and nested subdirectories in the same way that the file system on your computer is organized.
2. While registering an Azure environment in CDP Management Console, set the Storage Location Base in the Data Access section as follows:

   ```
   abfs://storage-fs@[***AZURE-STORAGE-ACCOUNT-NAME***].dfs.core.windows.net
   ```

   This location is used to read and store data.

3. Enable the ABFS file browser in Hue.
   a. Sign in to Cloudera Data Warehouse.
   b. Go to the Virtual Warehouse from which you want to access the ABFS containers and click its edit icon.
   c. On the **Virtual Warehouses** detail page, go to the Hue tab and select hue-safety-valve from the drop-down menu.
   d. Add the following configuration for Hive or Impala Virtual Warehouse in the space provided:

      For Hive Virtual Warehouse:

```
[desktop]
# Remove the file browser from the blacklisted apps.
# Tweak the app_blacklist property to suit your app configuration.
app_blacklist=oozie,search,hbase,security,pig,sqoop,spark,impala
[azure]
  [[azure_accounts]]
    [[[default]]]
      client_id=[***AZURE-ACCOUNT-CLIENT-ID***]
      client_secret=[***AZURE-ACCOUNT-CLIENT-SECRET***]
      tenant_id=[***AZURE-ACCOUNT-TENANT-ID***]

    [[abfs_clusters]]
      [[[default]]]
  fs_defaultfs=abfs://[***CONTAINER-NAME***]@[***AZURE-STORAGE-ACCOUNT-N
AME***]>.dfs.core.windows.net
              webhdfs_url=https://[***AZURE-STORAGE-ACCOUNT-NAME***].d
fs.core.windows.net/
```

      For Impala Virtual Warehouse:

```
[desktop]
# Remove the file browser from the blacklisted apps.
# Tweak the app_blacklist property to suit your app configuration.
app_blacklist=spark,zookeeper,hive,hbase,search,oozie,jobsub,pig,sqoop
,security
[azure]
  [[azure_accounts]]
    [[[default]]]
      client_id=<client_id>
      client_secret=<client_secret_id>
      tenant_id=<tenant_id>
    [[abfs_clusters]]
      [[[default]]]
  fs_defaultfs=abfs://[***CONTAINER-NAME***]@[***AZURE-STORAGE-ACCOUNT-
NAME***]>.dfs.core.windows.net
              webhdfs_url=https://[***AZURE-STORAGE-ACCOUNT-NAME***].dfs
.core.windows.net/
```

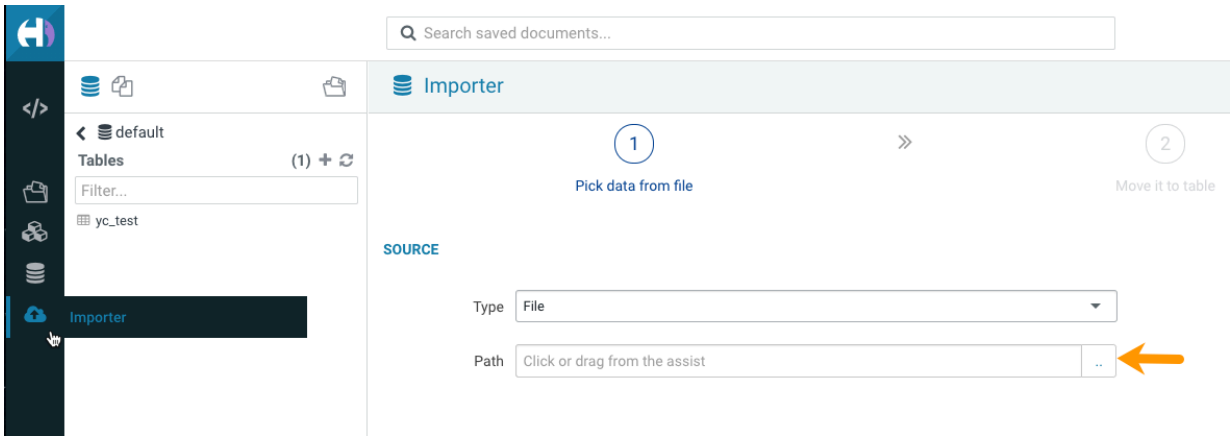
      Make sure that the container name and the Azure storage account name that you specify under the abfs_clu sters section is same as what you specified under Data Access Storage Location Base while activating the Azure environment, so that Hive or Impala has permission to access the uploaded files.
   e. Click Apply in the upper right corner of the page.

      The ABFS File Browser icon appears on the left Assist panel on the Hue web UI after the Virtual Warehouse restarts.

Procedure

1. In the CDW service **Overview** page, select the Virtual Warehouse in which you want to create the table, click the options menu in the upper right corner and click Open Hue.
2. From the left assist panel, click on Importer.

3. On the **Importer** screen, click .. at the end of the Path field:



Choose a file pop-up is displayed.

4. Type abfs://[***CONTAINER-NAME***] in the address text box and press enter.

   The ABFS containers created under the Azure storage account are displayed.

   You can narrow down the list of results using the search option.

   If the file is present on your computer, then you can upload it to ABFS by clicking Upload a file.

5. Select the CSV file that you want to import into Hue.

   Hue displays the preview of the table along with the format.

   Hue automatically detects the field separator, record separator, and the quote character from the CSV file. If you want to override a specific setting, then you can change it by selecting a different value from the drop-down menu.

6. Click Next.

   On this page, you can set the table destination, partitions, and change the column data types.

7. Verify the settings and click Submit to create the table.

   The CREATE TABLE query is triggered.

   Hue displays the logs and opens the **Table Browser** from which you can view the newly created table when the operation completes successfully.

Viewing query history in Cloudera Data Warehouse

In Cloudera Data Warehouse (CDW), you can view all queries that were run against a Database Catalog from Hue, Beeline, Hive Warehouse Connector (HWC), Tableau, Impala-shell, Impyla, and so on.

About this task

You need to set up Query Processor administrators to view the list of all queries from all users, or to restrict viewing of queries.

Procedure

1. Log in to the CDP web interface and navigate to the Data Warehouse service.
2. In the Data Warehouse service, navigate to the **Overview** page.
3. From a Virtual Warehouse, launch Hue.
4. Click on the Jobs icon on the left-assist panel.

   The **Job Browser** page is displayed.

5. Go to the **Queries** tab to view query history and query details.

Related Information

Viewing Hive query details
Viewing Impala query details
Adding Query Processor Administrator users and groups in Cloudera Data Warehouse

CDE Airflow ETL workloads in Hive Virtual Warehouses

Cloudera Data Engineering can use Apache Airflow to create jobs that run ETL workloads on Hive Virtual Warehouses in Cloudera Data Warehouse (CDW).

Hive Virtual Warehouses, which use LLAP daemons (one per executor), are optimized to run interactive queries. However, when the HiveServer query planner receives queries, it examines the scan size, and if the scan size exceeds the value specified for the hive.query.isolation.scan.size.threshold parameter, the planner runs the query in isolation mode. This means that an isolated standalone executor group is spawned to run the data-intensive ETL-type query. This applies to query workloads that originate in CDW. For more information about query isolation mode, see Hive query isolation.

When query workloads originate in Cloudera Data Engineering and are sent to Hive Virtual Warehouses in CDW using automated Airflow data pipelines, the HiveServer query planner assumes these are ETL workloads and these workloads are automatically processed in query isolation mode.

To set up automated data pipelines using Airflow in Cloudera Data Engineering, you must copy the Hive Virtual Warehouse JDBC URL in the CDW UI to a text editor. Then you must navigate to the Cloudera Data Engineering experience and configure the connection for the data pipeline using Airflow. For details about setting this up, see Automating data pipelines with CDE and CDW using Apache Airflow.

Creating a user-defined function in Cloudera Data Warehouse Public Cloud

You can call a built-in Hive function to run one of a wide-range of operations instead of performing multiple steps. You can create a user-defined function (UDF) when a built-in is not available to do what you need.

Assuming you have installed Java and a Java integrated development environment (IDE) tool, such as IntelliJ, you can create the UDF offline. For example, you can create the UDF on your laptop. You export a user-defined function (UDF) to a JAR from a Hadoop- and Hive-compatible Java project and store the JAR on the object store. Using Hive commands, you register the UDF based on the JAR, and call the UDF from a Hive query.

Setting up the development environment

You can create a Hive UDF in a development environment using IntelliJ, for example, and build the UDF. You define the Cloudera Maven Repository in your POM, which accesses necessary JARS hadoop-common-<version>.jar and hive-exec-<version>.jar.

Procedure

1. Open IntelliJ and create a new Maven-based project. Click Create New Project. Select Maven and the supported Java version as the Project SDK. Click Next.
2. Add archetype information.
   For example:
   - GroupId: com.mycompany.hiveudf
   - ArtifactId: hiveudf

3. Click Next and Finish.
   The generated pom.xml appears in sample-hiveudf.
4. To the pom.xml, add properties to facilitate versioning.
   For example:

```
<properties>
   <hadoop.version>TBD</hadoop.version>
   <hive.version>TBD</hive.version>
</properties>
```

5. In the pom.xml, define the repositories.

   Use internal repositories if you do not have internet access.

```
<repositories>
   <repository>
      <releases>
         <enabled>true</enabled>
         <updatePolicy>always</updatePolicy>
         <checksumPolicy>warn</checksumPolicy>
      </releases>
      <snapshots>
         <enabled>false</enabled>
         <updatePolicy>never</updatePolicy>
         <checksumPolicy>fail</checksumPolicy>
      </snapshots>
      <id>HDPReleases</id>
      <name>HDP Releases</name>
      <url>http://repo.hortonworks.com/content/repositories/releases/</u
rl>
      <layout>default</layout>
   </repository>
   <repository>
      <id>public.repo.hortonworks.com</id>
      <name>Public Hortonworks Maven Repo</name>
      <url>http://repo.hortonworks.com/content/groups/public/</url>
      <snapshots>
         <enabled>false</enabled>
      </snapshots>
   </repository>
   <repository>
         <id>repository.cloudera.com</id>
         <url>https://repository.cloudera.com/artifactory/cloudera-repos/<
/url>
   </repository>
 </repositories>
```

6. Define dependencies.
   For example:

```
<dependencies>
  <dependency>
       <groupId>org.apache.hive</groupId>
       <artifactId>hive-exec</artifactId>
       <version>${hive.version}</version>
    </dependency>
    <dependency>
       <groupId>org.apache.hadoop</groupId>
       <artifactId>hadoop-common</artifactId>
       <version>${hadoop.version}</version>
    </dependency>
</dependencies>
```

Create the UDF class

You define the UDF logic in a new class that returns the data type of a selected column in a table.

Procedure

1. In IntelliJ, click the vertical project tab, and expand hiveudf: hiveudf src main . Select the java directory, and on the context menu, select New Java Class , and name the class, for example, TypeOf.
2. Extend the GenericUDF class to include the logic that identifies the data type of a column.
   For example:

```
package com.mycompany.hiveudf;

import org.apache.hadoop.hive.ql.exec.UDFArgumentException;
import org.apache.hadoop.hive.ql.metadata.HiveException;
import org.apache.hadoop.hive.ql.udf.generic.GenericUDF;
import org.apache.hadoop.hive.serde2.objectinspector.ObjectInspector;
import org.apache.hadoop.hive.serde2.objectinspector.primitive.\
PrimitiveObjectInspectorFactory;
import org.apache.hadoop.io.Text;
public class TypeOf extends GenericUDF {
  private final Text output = new Text();
@Override
  public ObjectInspector initialize(ObjectInspector[] arguments) throws U
DFArgumentException {
    checkArgsSize(arguments, 1, 1);
    checkArgPrimitive(arguments, 0);
    ObjectInspector outputOI = PrimitiveObjectInspectorFactory.writableSt
ringObjectInspector;
    return outputOI;
  }

@Override
  public Object evaluate(DeferredObject[] arguments) throws HiveException
 {
    Object obj;
    if ((obj = arguments[0].get()) == null) {
      String res = "Type: NULL";
      output.set(res);
      } else {
      String res = "Type: " + obj.getClass().getName();
      output.set(res);
    }
    return output;
  }

@Override
  public String getDisplayString(String[] children) {
    return getStandardDisplayString("TYPEOF", children, ",");
  }
}
```

Building the project and uploading the JAR

First, you compile the UDF code into a JAR, and then you add the JAR to Cloudera Data Warehouse object storage.

Before you begin

You have the EnvironmentAdmin role permissions to upload the JAR to your object storage.

Procedure

1. Build the IntelliJ project.

```
...
[INFO] Building jar: /Users/max/IdeaProjects/hiveudf/target/TypeOf-1.0-S
NAPSHOT.jar
[INFO] ------------------------------------------------------------------
-------
[INFO] BUILD SUCCESS
[INFO] -------------------------------------------------------------------
--------
[INFO] Total time: 14.820 s
[INFO] Finished at: 2019-04-03T16:53:04-07:00
[INFO] Final Memory: 26M/397M
[INFO] -------------------------------------------------------------------
--------

Process finished with exit code 0
```

2. In IntelliJ, navigate to the JAR in the /target directory of the project.
3. In Cloudera Data Warehouse, click Overview Database Catalogs , select your Database Catalog, and click options ⋮ , and then click Edit.
4. Upload the JAR to the Hive warehouse on CDW object storage.
   - AWS object storage

     In AWS, upload the JAR file to a bucket on S3 that you can access, for example S3a://my-bucket/path. Add an external AWS S3 bucket if necessary.
   - Azure object storage

     In Azure, upload the JAR file to a default Azure Blob Storage (ABFS) location that you can access, for example abfs://my-storage/path.
5. In IntelliJ, click Save.
6. Click Actions Deploy Client Configuration .
7. Restart the Hive Virtual Warehouse.

Related Information

Adding access to external S3 buckets

Configuring Hive Virtual Warehouses to cache UDF JARs

After you write and compile your User Defined Function (UDF) code into a Java Archive (JAR) file, you can configure a Hive Virtual Warehouse to cache the UDF JAR in HiveServer (HS2) in Cloudera Data Warehouse (CDW).

About this task

UDFs enable you to create custom functions to process records or groups of records. Although Hive provides a comprehensive library of functions, there are gaps for which UDFs are a good solution.

In this task, you configure the Hive Virtual Warehouse to cache the JAR file for quick access by the Virtual Warehouse. After configuring the Virtual Warehouse for caching, the UDF JAR is downloaded from the object store the first time it is called, and then cached. Subsequent calls to the JAR are answered from the cache.

Configuring caching significantly improves performance for queries that use the UDF. Without caching, loading a very large UDF of several hundred MBs can take up to several minutes for each query.

Required role: DWAdmin

Before you begin

- Create a user-defined function
  - Write, compile, and export your UDF code to a JAR file.
  - Upload the UDF JAR file to a bucket or container on AWS or Azure, respectively.

Procedure

1. In the CDW UI on the Overview page, locate the Hive Virtual Warehouse that uses the bucket or container where you placed the UDF JAR file, and click Edit.
2. In Details, click CONFIGURATIONS HiveServer2.
3. Select hive-site from the drop-down list, and click +.
4. In the Add Custom Configurations, add the following configuration information, and then click ADD:

```
hive.server2.udf.cache.enabled = true
```

5. Click APPLY.
6. Verify that the configuration property and setting have been added by searching for hive.server2.udf.cache.enabled in the search box. If the property has been added, the property name is displayed in the KEY column of the table.

Registering the UDF

In Cloudera Data Warehouse (CDW), you run a command from a client interface to your Virtual Warehouse, such as Hue, to call the UDF from Hive queries. The UDF persists between HiveServer restarts.

Before you begin

You need to set up UDF access using a Ranger policy as follows:

- Log in to the Data Warehouse service and open Ranger from the Database Catalog associated with your Hive Virtual Warehouse.
- On the **Service Manager** page, under the HADOOP SQL section, select the Database Catalog associated with the Hive Virtual Warehouse in which you want to run the UDFs.
- Select the all - database, udf policy and add the users needing access to Hue. To add all users, you can specify {USER}.

About this task

Procedure

1. Open Hue from the Hive Virtual Warehouse in CDW.
2. Run the registration command by including the JAR location on your object store.

   For example, on AWS:

```
CREATE FUNCTION udftypeof AS 'com.mycompany.hiveudf.TypeOf01' USING JAR
's3a://mybucket/mypath/TypeOf01-1.0-SNAPSHOT.jar';
```

   On Azure:

```
CREATE FUNCTION udftypeof AS 'com.mycompany.hiveudf.TypeOf01' USING JAR
'abfs://mybucket/mypath/TypeOf01-1.0-SNAPSHOT.jar';
```

3. Restart the Virtual Warehouse.

4. Check that the UDF is registered.

```
SHOW FUNCTIONS;
```

   You scroll through the output and find default.typeof.

5. Run a query that calls the UDF.

```
SELECT students.name, udftypeof(students.name) AS type FROM students WHERE
 age=35;
```