Machine Learning

ML Discovery & Exploration

Date published: 2020-07-16
Date modified: 2024-04-01

Legal Notice

Contents

Exploratory Data Science and Visualization

Exploratory Data Science and Visualization makes it simple for data scientists to get started on a data science project.

When you start a data science project, you are presented with a blank notebook, and no indication of what data sources are available on the CDP platform, or how to access them. The Exploratory Data Science and Visualization feature automatically discovers the data sources available to you, from within the standard CML user interface.

The Exploration Data Science and Visualization experience enables all features for Exploratory Data Analysis. From the Data tab you can:

- Connect to data sources that are available in your project
- Explore the data with SQL to understand its basic shape and characteristics
- Create named datasets that you can reference later
- Quickly create visualizations of the data to understand its properties
- Create dashboards that you can share with your team

For more information on what you can do in the Data tab, see the documentation for *Cloudera Data Visualization*.

Related Information

Cloudera Data Visualization

Prerequisites

Machine Learning Discovery and Exploration has a few prerequisites.

- For both Hive and Impala connections, the HADOOP_USERNAME and WORKLOAD_PASSWORD must be set. The HADOOP_USERNAME is set automatically by the environment. As a data scientist, you must set the WORKLOAD_PASSWORD. For more information, see *Setting the workload password*.
- To manually create Impala or Hive data connections, you must have the related JDBC URL. You can obtain this from the option menu for the virtual warehouse to which you want to connect. These warehouses must be already created in the environment. For more information on how to create a data warehouse, see *Adding a New Virtual Warehouse*.
- To manually create a Spark connection, you need the Data Lake S3 bucket. This can be retrieved from the Data Lake page. For Spark data connections, you must use ML Runtimes, and specifically the Spark Runtime Add-on must be enabled before starting a workload (job or session).
- For Spark data connections, you must have permissions set correctly for external access to the S3 bucket.
- Hive and Impala data connections also require ML Runtimes. Legacy engines may work, but are not supported.
- There must be auto-discovered data connections, or your Administrator must create data connections to virtual warehouses or other data sources in your CDP environment.

Known Issue

- Only Hive virtual warehouses with SSO disabled are supported.

Related Information

Create a Dashboard

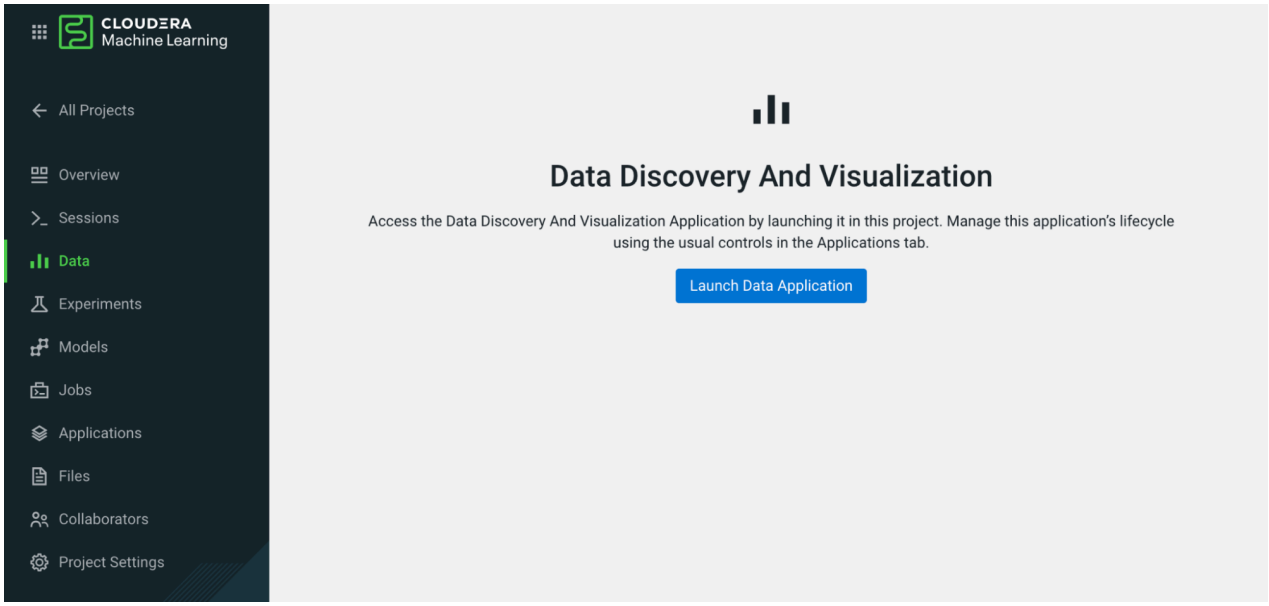
Adding a New Virtual Warehouse

Setting the workload password

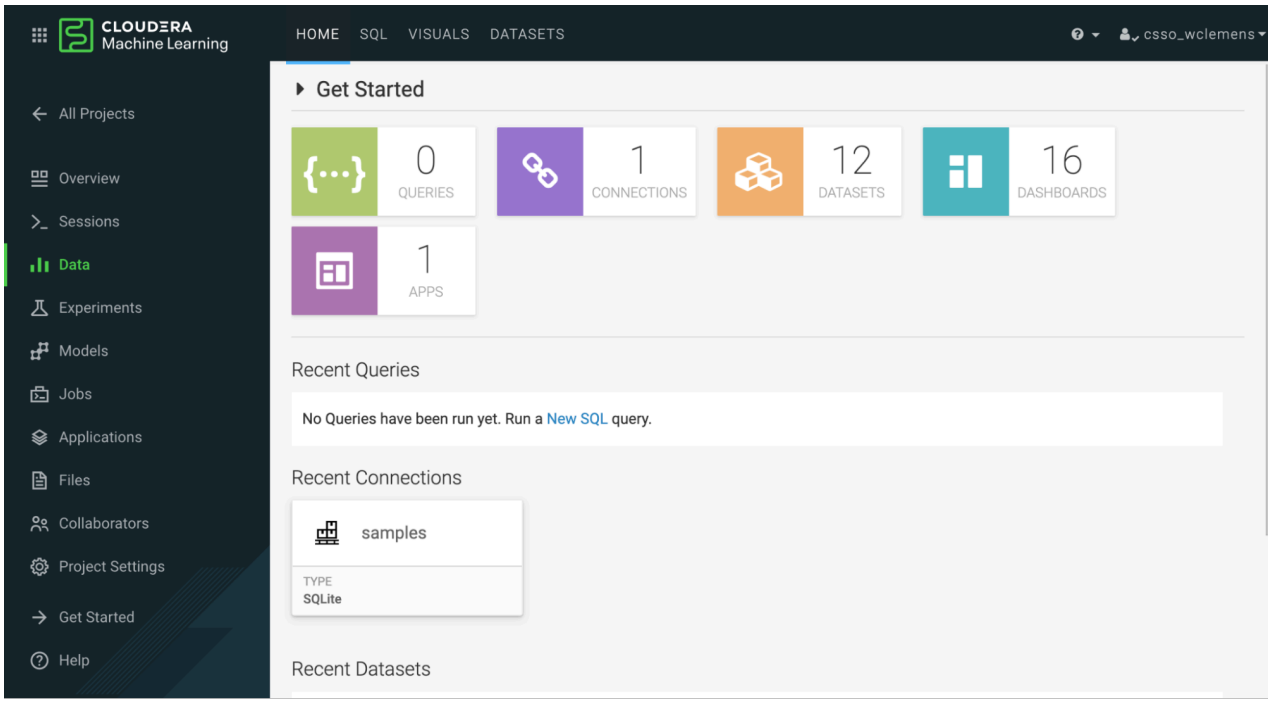
Starting Data Discovery and Visualization

You can start the Exploratory Data Science and Visualization application within Cloudera Machine Learning (CML).

1. Create or open a project.
2. In the project, click Data in the left navigation pane.
3. If the application was previously started, it appears in the UI.
4. If not, click Launch Data Application to start the application. It takes a few minutes to start the first time.



When the application starts, you can see four tabs at the top of the UI. From here, you can follow the steps in Get Started, or try some of the following tasks.



Related Information

Creating datasets
Create a Dashboard
Troubleshooting: 401 Unauthorized

Working with Data Discovery and Visualization

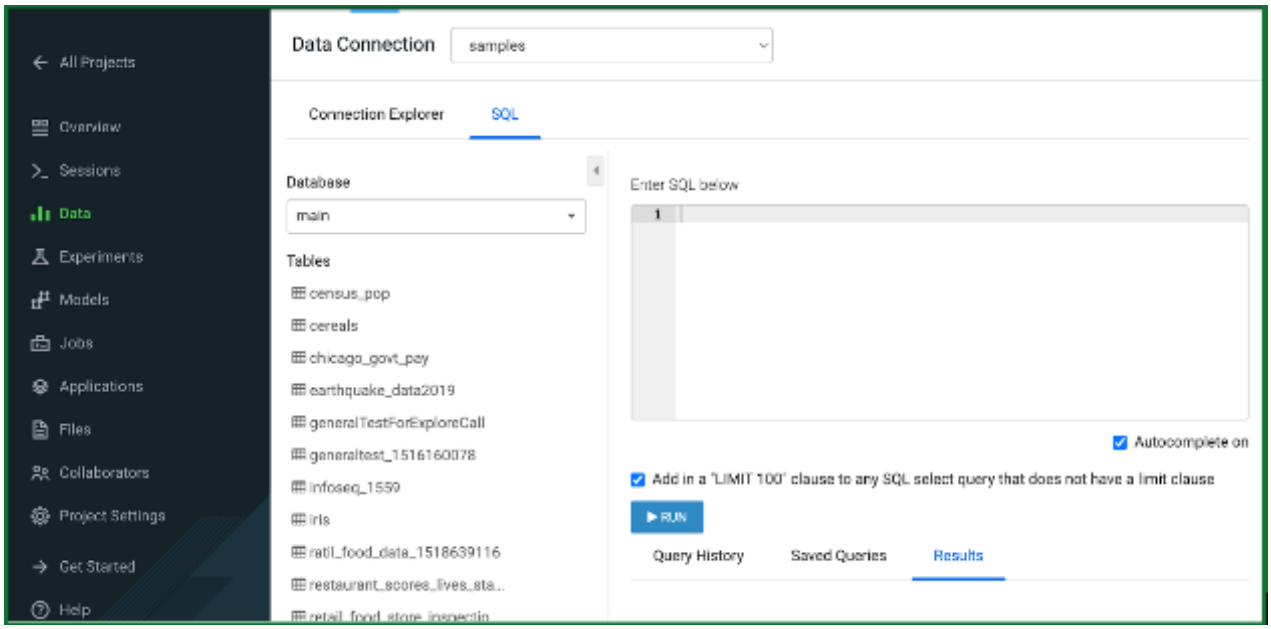
This section provides examples of the tasks you can perform using Data Discovery and Visualization.

Run an SQL query

On the SQL tab, select a Data Connection from the dropdown box.

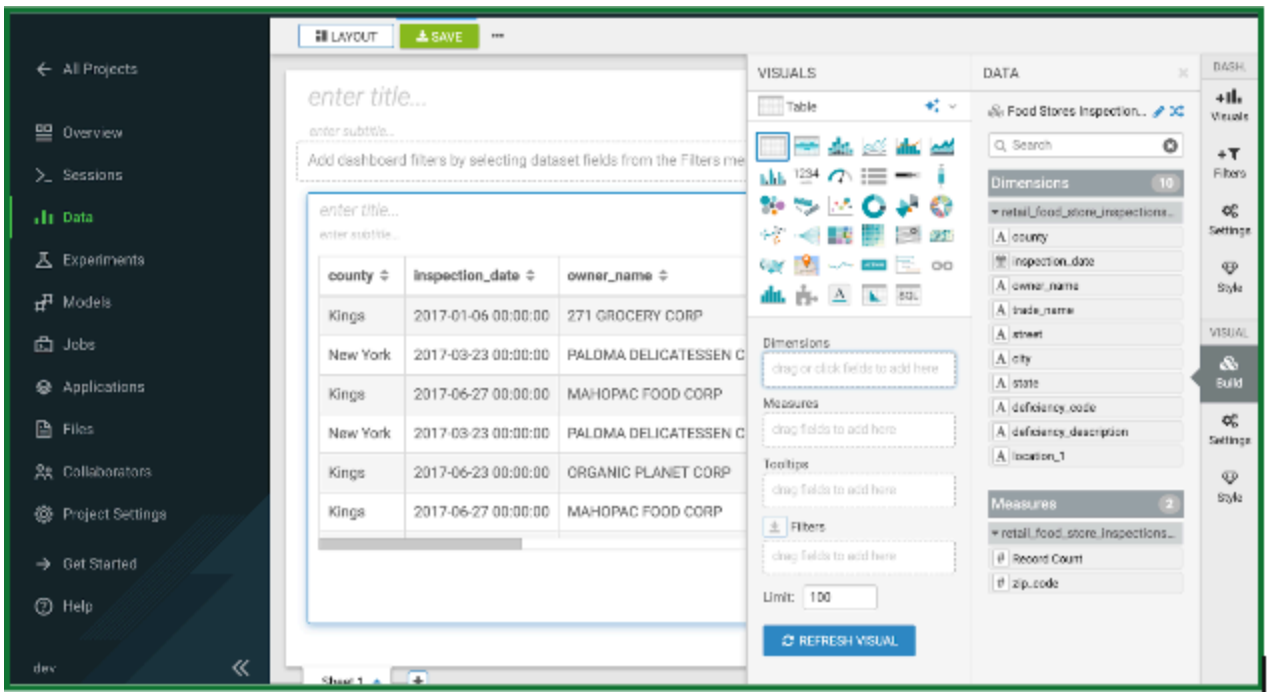
Note: If you do not see the data connections that you have configured in the Project, refer to the Troubleshooting section for more information.

Once you select a Data Connection, you can explore the available databases and tables. You can start writing an SQL query and click the table names to insert them into the query string. For more information, see *Creating a dataset*.



Build a dashboard

Click New Dashboard to access the Dashboard Designer. You can select from a wide variety of graphs, click to add them to your dashboard, and select the columns from your dataset to use in the visual. For more information, see *Create a Dashboard*.



Related Information

Creating a dataset

Create a dashboard

Troubleshooting: 401 Unauthorized

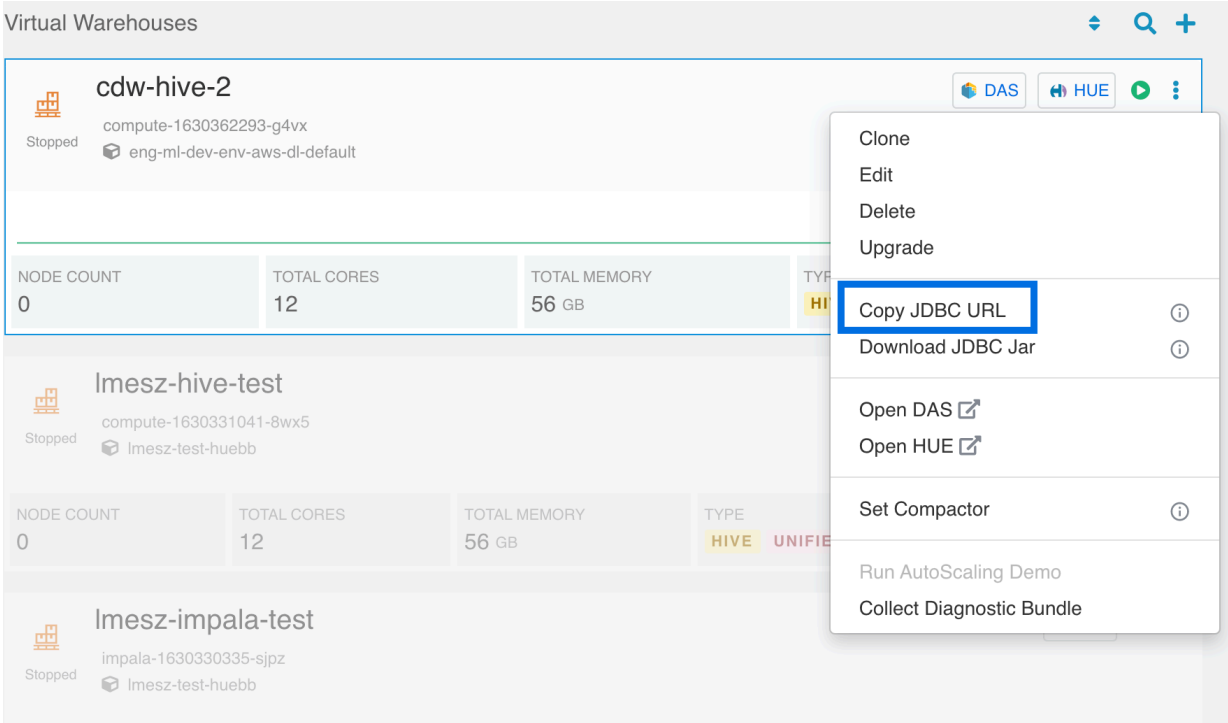
Set up a Hive or Impala data connection manually

Data connections to Hive or Impala virtual warehouses within the same environment as the CML workspace are automatically discovered and configured. You can also set up a data connection manually, which works across CDP environments. Follow this procedure to set up a Hive or Impala data connection.

Procedure

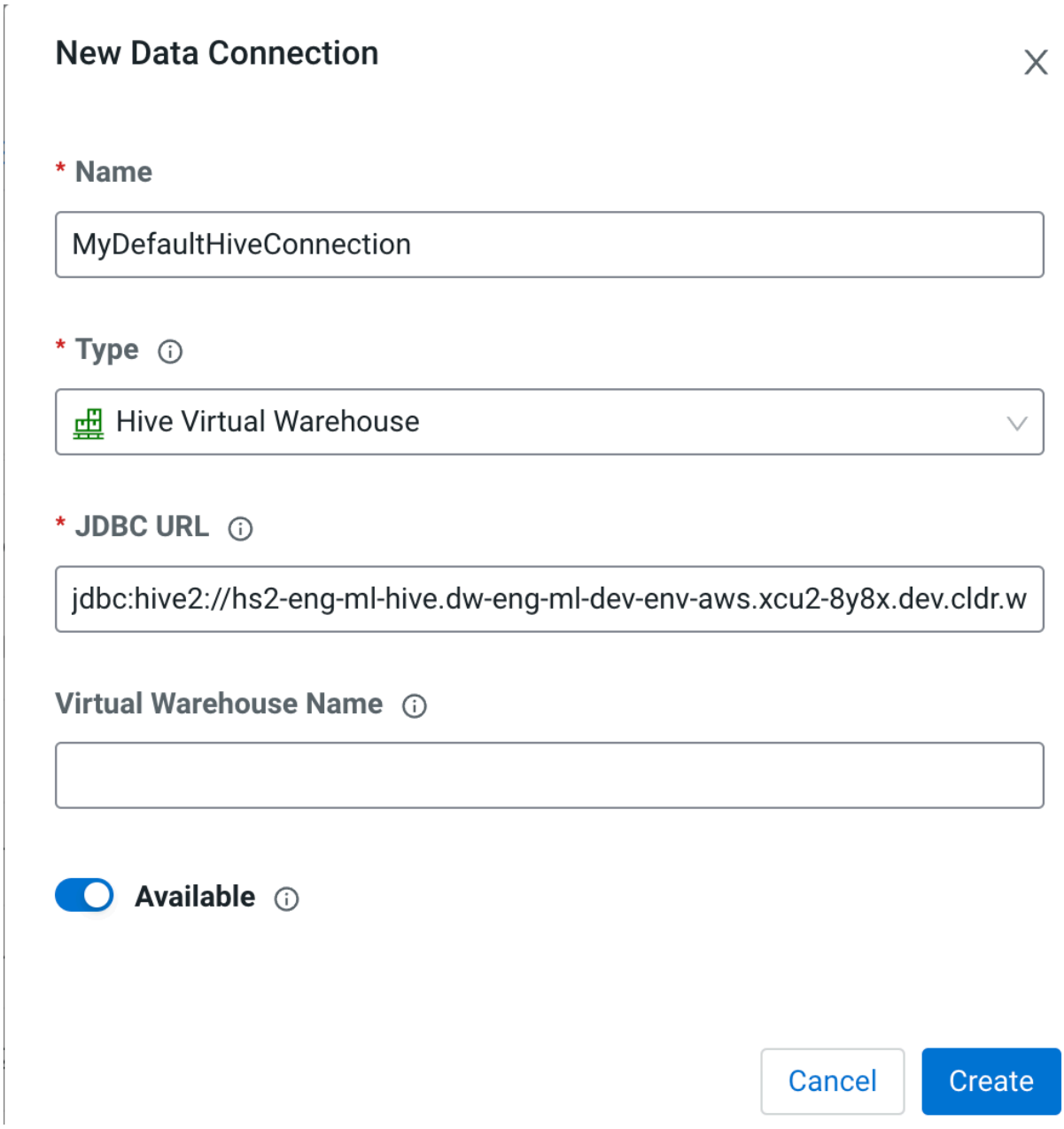
1. Log into the CDP web interface and navigate to the Data Warehouse service.
2. In the Data Warehouse service, select Virtual Warehouses in the left navigation panel.

3. Select the options menu for the warehouse you want to access, and select Copy JDBC URL.



4. Return to the Machine Learning service. In Site Administration Data Connections , select New Connection.
5. Enter the connection name. You cannot have duplicate names for data connections within a workspace or within a given project.
6. Select the connection type:
   a. Hive Virtual Warehouse
   b. Impala Virtual Warehouse
7. Paste the JDBC URL for the data connection.

8. (Optional) Enter the Virtual Warehouse Name. This is the name of the warehouse in Cloudera Data Warehouse.



Results

The data connection is available to users by default. To change availability, click the Available switch. This switch determines if the data connection is displayed in Projects created within the workspace.

Set up a Spark data connection

Spark data connections within the same environment as CML are automatically discovered, but you can also set up a connection manually. Follow this procedure to set up a Spark data connection.

Procedure

1. In the Workspaces UI, select the link environment for the workspace you are using. This takes you to the Environments UI.

2. In Environments, select Data Lake Cloud Storage tabs.
3. Select the directory path shown for Hive Metastore External Warehouse, and copy it.
4. In Project Settings > Data Connections, click New Connection.
5. Enter a name for the connection.
6. Select the type: Spark Data Lake
7. Paste the value you copied in step 3 into Datalake Hive Metastore External Warehouse Directory.
8. Click Create.

Results

The data connection is available to users by default. To change availability, click the Available toggle.

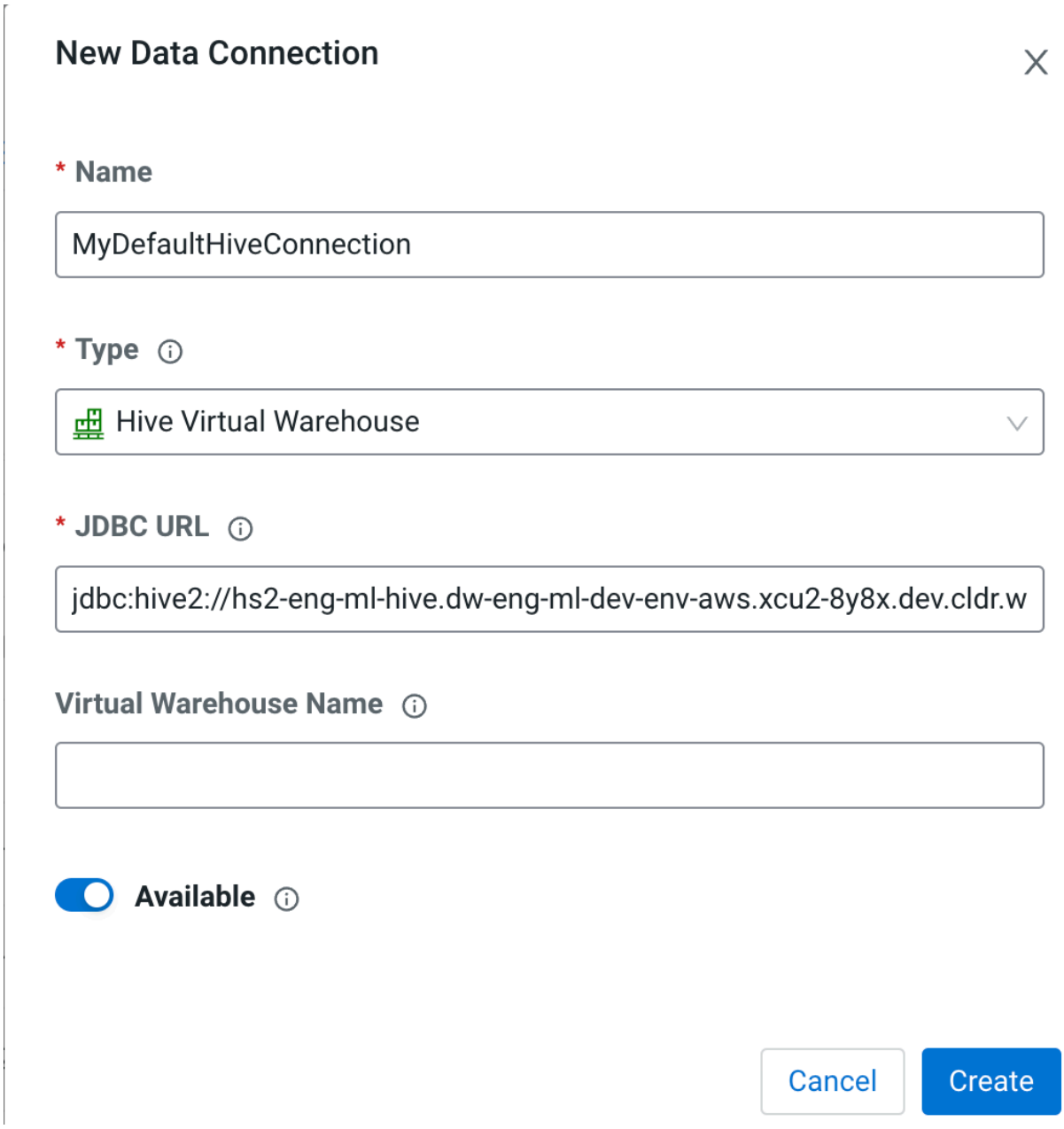
Set up a data connection to CDP DataHub

You can set up a data connection to the DataHub cluster. You can set up a connection using the New Connection dialog, or by using raw code inside your project. Both approaches are shown below.

Procedure

1. Log into the console web UI.
2. Depending on which connection you want to use, click on either Connect to Hive or Connect to Impala tile.
3. Ensure your Data Hub cluster name is correct in the popup.
4. Copy the JDBC URL string.
5. Now click on the Build a Data Science Project tile to log into the the ML workspace.
6. In Site Administration Data Connections , select New Connection.
7. Return to the Machine Learning service. In Site Administration Data Connections , select New Connection.
8. Enter the connection name. You cannot have duplicate names for data connections within a workspace or within a given project.
9. Select the connection type:
   a. Hive Virtual Warehouse
   b. Impala Virtual Warehouse
10. Paste the JDBC URL for the data connection.

11. (Optional) Enter the Virtual Warehouse Name. This is the name of the warehouse in Cloudera Data Warehouse.



Results

The data connection is available to users by default. To change availability, click the Available switch. This switch determines if the data connection is displayed in Projects created within the workspace.

Set up a DataHub data connection using raw code

It is recommended to use the New Connection dialog to create a new data connection. If needed, you can also set up a data connection in your project code by using and adapting the following code snippet.

Example

```
from impala.dbapi import connect

#Example connection string:
```

```
# jdbc:hive2://my-test-master0.eng-ml-i.svbr-nqvp.int.cldr.work/;ssl=true;
transportMode=http;httpPath=my-test/cdp-proxy-api/hive

conn = connect(
    host = "my-test-master0.eng-ml-i.svbr-nqvp.int.cldr.work",
    port = 443,
    auth_mechanism = "LDAP",
    use_ssl = True,
    use_http_transport = True,
    http_path = "my-test/cdp-proxy-api/hive",
    user = "csso_me",
    password = "Test@123")
cursor = conn.cursor()
cursor.execute("select * from 3yearpop")

for row in cursor:
    print(row)
cursor.close()
conn.close()
```

Data connection management

There are a few things to keep in mind about data connections.

- Manage data connections in a workspace

  At the workspace level, you can check the data connections that are available in a workspace. In Project Settings Data Connections , check that your desired data source is present. You can also set the availability for any discovered connections, if necessary.
- Manage data connections in a project

  All the data connections that were available in the workspace when the project was created will be automatically created in the project as well. The available connections can then further be marked as unavailable if so desired. You can update any changes to the connections that were made at the workspace level by clicking Sync with Workspace. Any changes made here only apply to your project.
- Data connection availability

  Keep in mind these two scenarios for setting data connection availability.

  1. If a workspace data connection is marked Unavailable, and you then create a project, the data connection will not appear in the project. If the connection is then changed to Available, and then the Sync with Workspace button is clicked, the connection will appear in the project.
  2. If a workspace data connection is marked Available, and you then create a project, the connection shows up. If the workspace data connection is then toggled to Unavailable, and you click Sync with Workspace in the project, the data connection will remain available in the project.

Setting the workload password

Set the workload password to connect your code to the data.

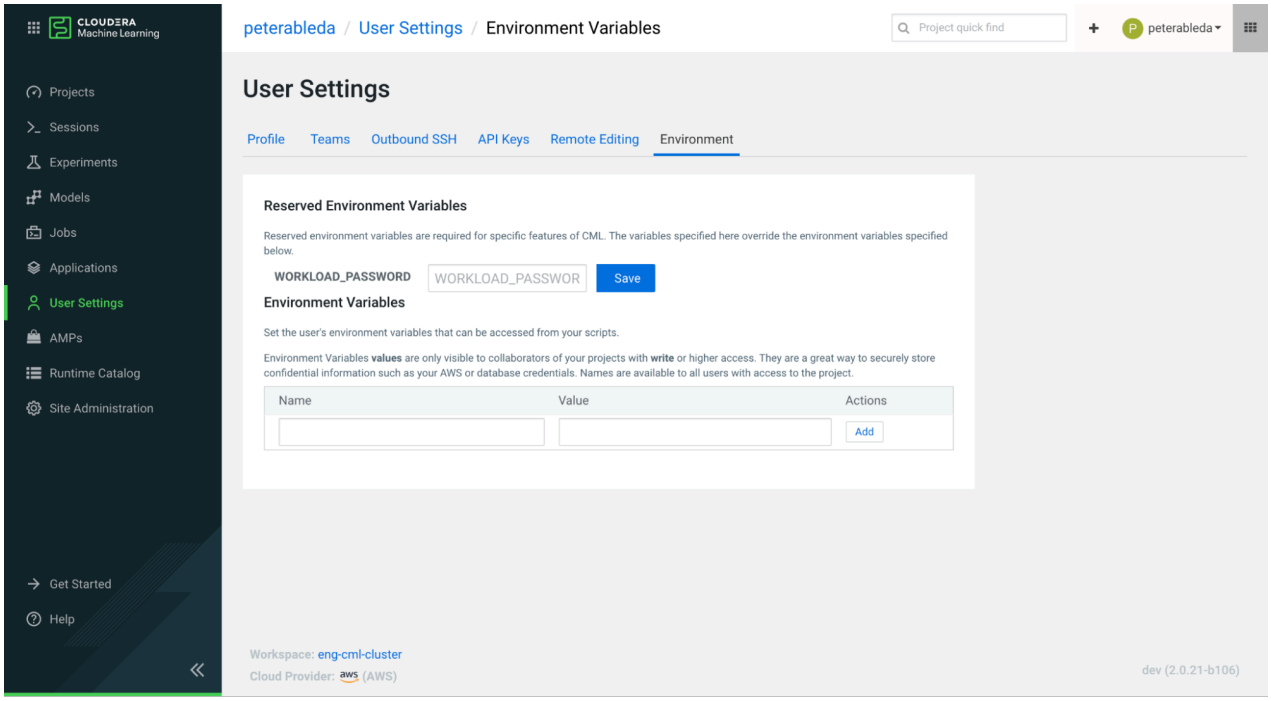
You need to set an environmental variable for the password (WORKLOAD_PASSWORD). This environmental variable is needed to access a Hive or Impala warehouse. The password you set in Machine Learning must match the password you set in Management Console. For more information on setting the workload password in Management Console, see *Setting the workload password*.

To set the password:

1. In Machine Learning, go to User Settings Environment .

2. Under the Reserved Environment Variables set your WORKLOAD_PASSWORD.
3. Click Save.
4. In Management Console, select your user name in the left navigation pane, and select Profile.
5. Click Set Workload Password.
6. Enter the same password as in step 2, and click Set Workload Password.

Note: You need to restart any sessions or applications that are running in your workspace.



Related Information

Setting the workload password

Set up a Custom Data Connection

Data connections that point to data sources outside of CDP or require custom configurations can be created and made available to end users with Custom Data Connections. These Python implementations of the CML Data library are stored in the Data Connections Registry. Workspace users can track and connect to any data source and connection implementation a CML Administrator makes available.

About this task

Keep in mind the following:

- Custom connections can only be created in projects created by the Administrator.
- The project source selection list in the Data Connection creation dialogue only displays projects created by the user.
- Team projects or projects with multiple collaborators will also not be displayed, only those directly created by the user.
- Custom connections at workspace level can only be edited by the creator, not other Administrator users. Attempts at editing workspace level custom connections will result in an error.

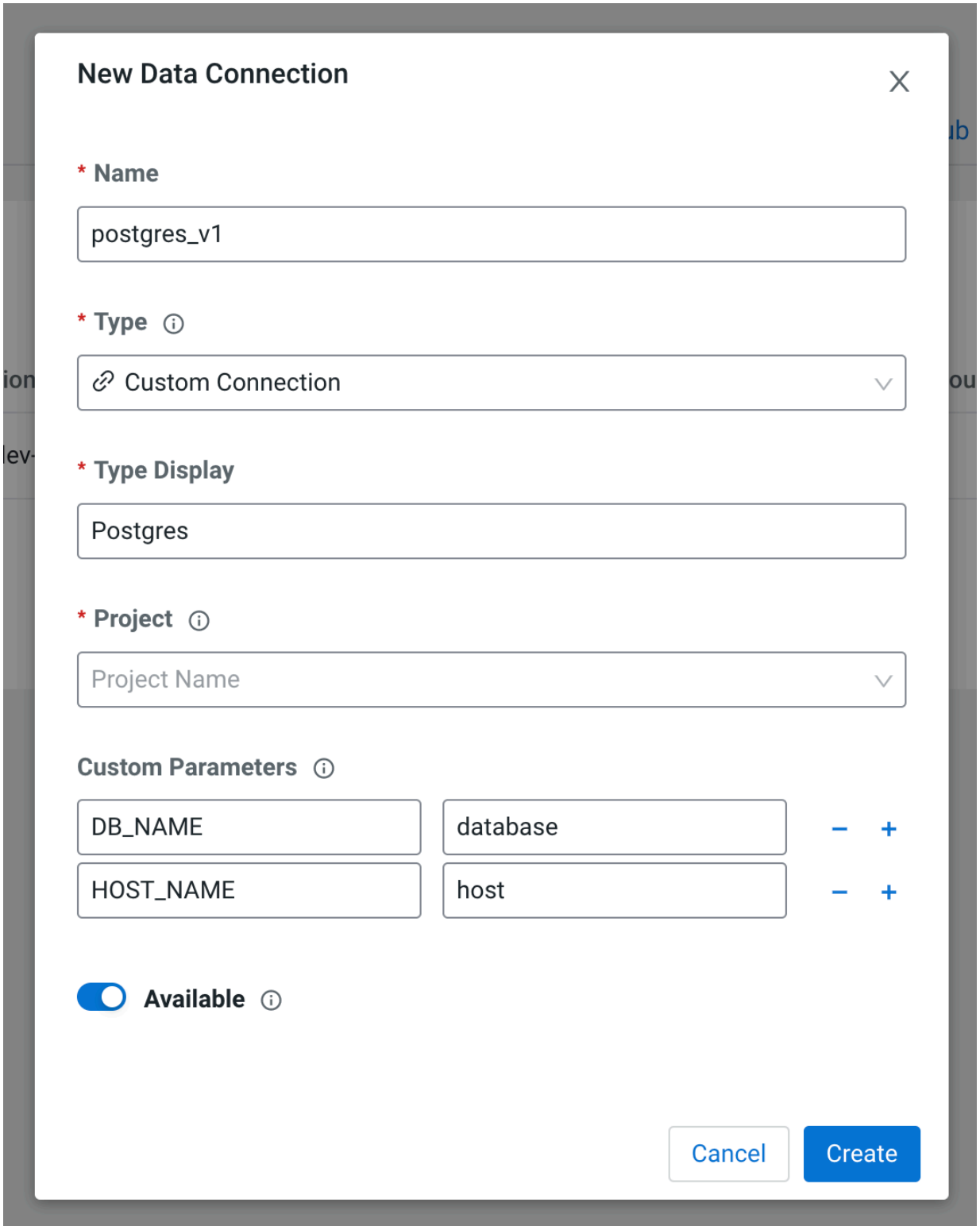
Before you begin

Before setting up a custom connection, you might want to create a dedicated CML Team to collaborate on external connections. A good practice is to separate the connection code projects and and configure collaborators on the Team level to build and maintain the connection code.

Procedure

1. Develop your own custom data connection (see *Developing a Custom Data Connection*) in a CML project, or clone an existing custom data connection files directory into a CML project.
2. In Site Administration Data Connections , select New Connection.
3. Enter the connection name. You cannot have duplicate names for data connections within a workspace or within a given project.
4. Select the connection type: Custom Connection
5. Enter the Type Display name. This should be a descriptive label to help CML project owners identify what this custom connection could be used for.
6. Select the CML Project and Project directory which contains your custom connection implementation
   a. Connection files must be in a directory and not in the root of your project.
   b. A snapshot of all implementation files in the directory will be uploaded to the CML Custom Data Connection registry located in the workspace.
   c. These uploaded files are safe from any changes to the originating project. To make changes to the files, create a new custom data connection.

7. (Optional) Enter any custom parameters. These are available during a session and can be validated or overridden depending on the interface implementation for the custom data connection. Refer to the implementation of your custom data connection for specific details on required keys and values.



8. Click Create.

Results

The data connection is now available to all users. To change availability, click the Available switch. This switch determines if the data connection is displayed in Projects created within the workspace. Refer to *Data connection management* for availability of your newly created custom connections in new and existing CML Projects (https://docs.cloudera.com/machine-learning/cloud/mlde/topics/ml-mlde-data-conn-mgmt.html)

Related Information

Data connection management

Custom Data Connection Development

Custom data connections can be developed from within CML Workbench and Python Sessions using the CML Python Data Library and implementing the CML Custom Connection Interface.

You can view CustomConnection interface help descriptions within in a session:

```
import cml.data_v1 as cmldata
help(cmldata.customconnection)
```

Alternatively, you can inspect the source content as follows:

```
import cml.data_v1 as cmldata
import inspect
print(inspect.getsource(cmldata.customconnection))
```

Your custom connection code must implement the CustomConnection interface for the cml.data_v1 library to load your module dynamically (see *Loading custom connections*)

Two functions are already implemented so that the CML Python Data Library can dynamically load your Python module implementation and make custom parameters available in self.parameters. In most cases, you will not need to reimplement these:

1. __init__(self, properties)
2. update_properties(self, properties)

The rest of the interface functions are included as common functions that you may want to implement.

1. get_base_connection(self)
2. get_pandas_dataframe(self, query)
3. get_cursor(self)
4. print_usage(self)
5. override_parameters(self)

See *Developing and testing your first custom connection* for a simple example of how to implement these.

Related Information

Loading custom connections

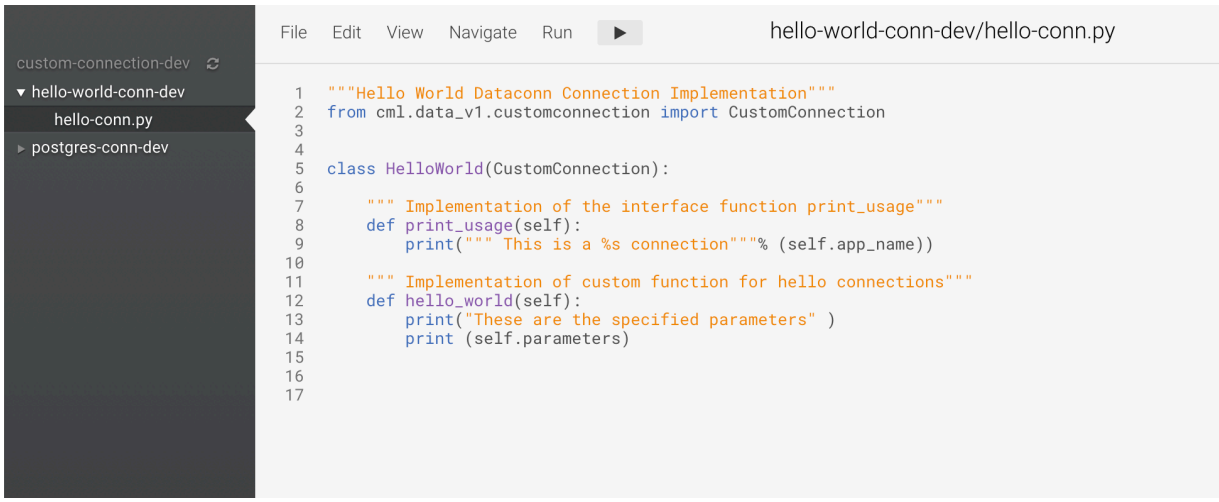
Developing and testing your first custom connection

Developing and testing your first custom connection

You can develop a custom connection in your own project, as shown in this example.

Procedure

1. Create a new directory hello-world-conn-dev and file hello-conn.py to contain your custom data connection files.



2. Implement your custom data connection in hello-conn.py. (There must only be one class which implements `CustomConnection` in this directory.)

```
"""Hello World Custom Connection Implementation"""
from cml.data_v1.customconnection import CustomConnection

class HelloWorld(CustomConnection):

    """ Implementation of the interface function print_usage"""
    def print_usage(self):
        print(""" This is a %s connection"""% (self.app_name))

    """ Implementation of custom function for hello connections"""
    def hello_world(self):
        print("These are the specified parameters" )
        print (self.parameters)
```

3. Test your custom data connection locally (for more information see *Loading custom connections*)

```
test_params = {"PARAM_1": "foo", "PARAM_2": "bar"}
import cml.data_v1 as cmldata

conn = cmldata.get_custom_connection_from_local(
  "hello-world-conn-dev",
  "my-hello-connection",
  test_params)
conn.print_usage()
```

```
conn.hello_world()
```

Project | Terminal Access | Data | Clear | Interrupt | Stop | Sessions

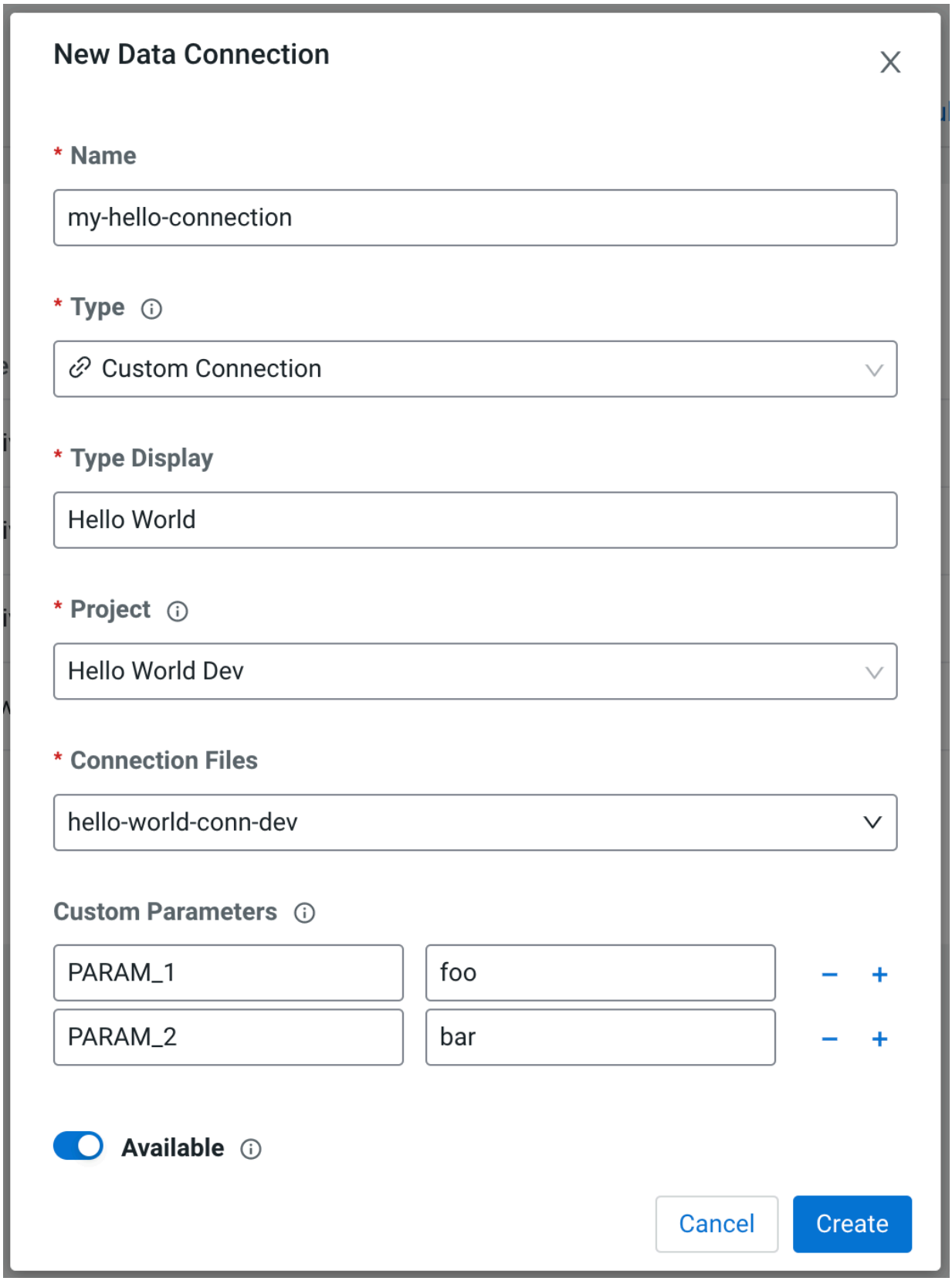
Untitled Session Running

By CDEP CREATED ACCOUNT — Session — 2 vCPU / 4 GiB Memory — a few seconds ago

Session | Logs | Collapse | Share | Export PDF

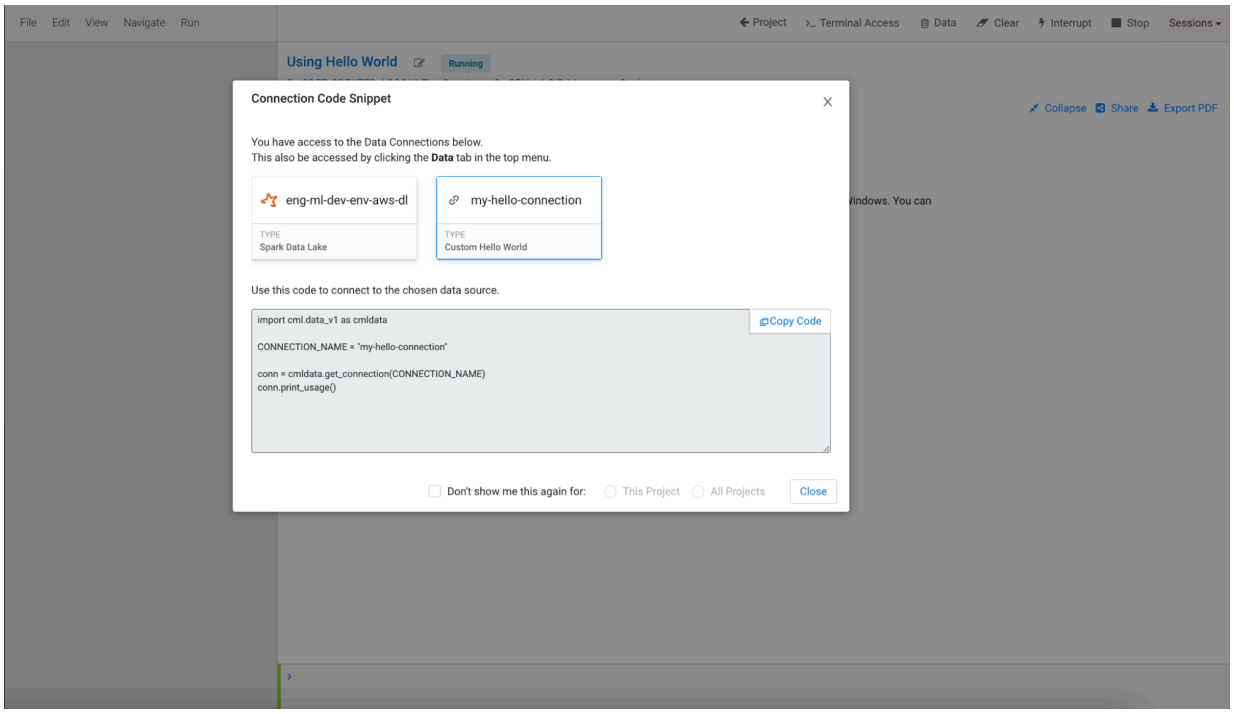
```
> test_params = {"PARAM_1": "foo", "PARAM_2": "bar"}
> conn = cmldata.get_custom_connection_from_local(
    "hello-world-conn-dev",
    "my-hello-connection",
    test_params)

Loaded custom connection my-hello-connection

> conn.print_usage()

 This is a my-hello-connection connection

> conn.hello_world()

These are the specified parameters
{'PARAM_1': 'foo', 'PARAM_2': 'bar'}
```

4. In Site Administration Data Connections , select New Connection, and fill in the fields as shown below.

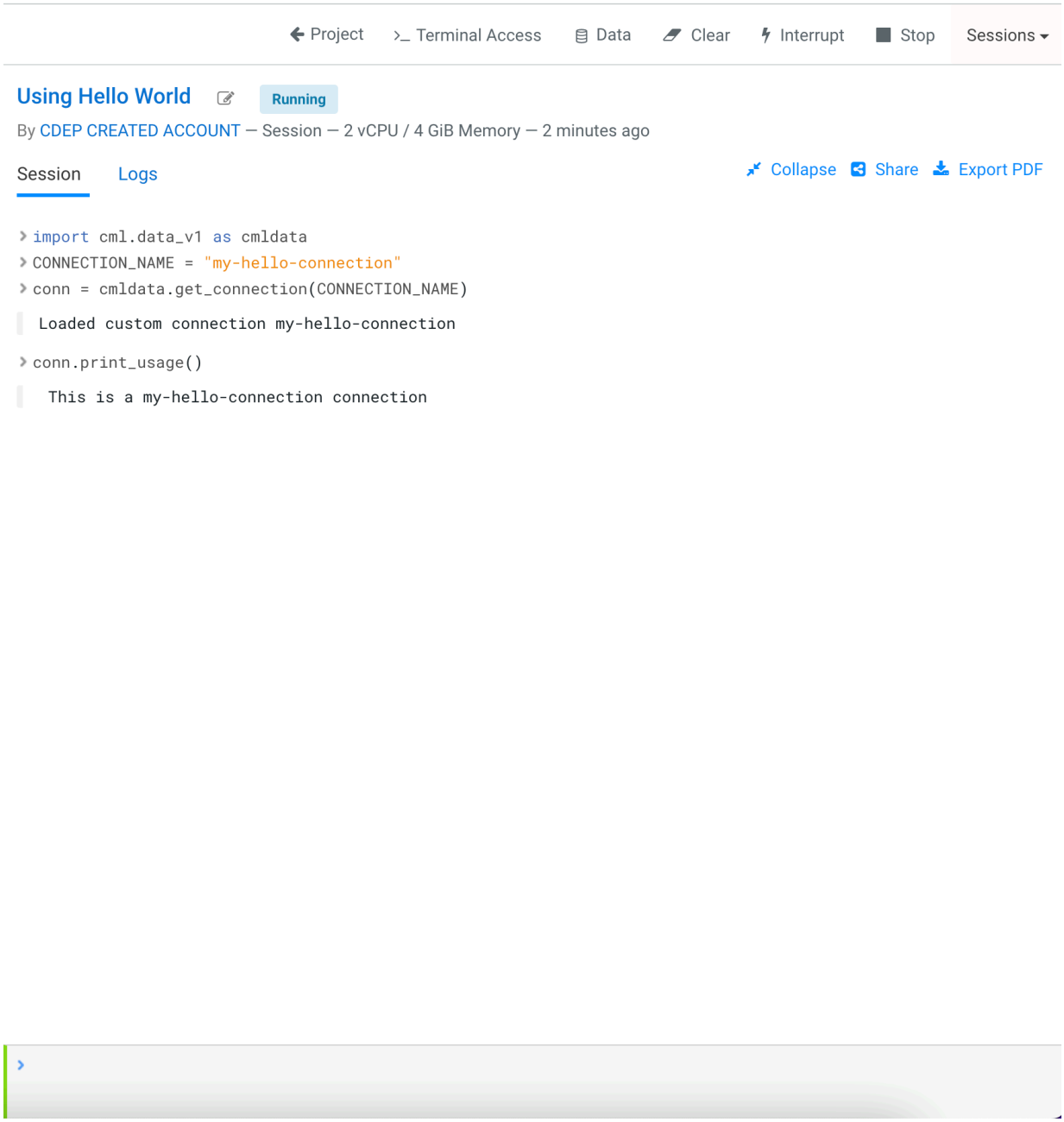


5. See *Data connection management* to set the availability of your newly created custom connection in new and existing CML projects.

6. Create a project, and start a session to use your completed Hello World connection!



Note: A default custom connection code snippet is automatically displayed to the user.



Note: When exporting custom connection files, git is recommended. Alternatively, download the custom connection directory only, instead of all project files.

Related Information

Loading custom connections

Data connection management

Using data connection snippets

Loading custom connections

You can instantiate a local connection for testing, using the name of your custom connection directory, a sample connection name, and an optional dictionary of parameters. This local connection object can then be used to test and implement functions in your custom connection.

The following code sample loads custom connection package directories in the same way that the CML Data Library imports a registered custom connection when called with get_connection:

```
get_custom_connection_from_local(package_name, connection_name, parameters={
})
```

Returns a CML Custom Data Connection object. For testing in-development Data Connection code.

Parameters:

- package_name (str): The accessible package name containing custom connection code to load.
- connection_name (str): The connection name to be used in Custom Connection loading.
- parameters (dict of str: str): Mapping of custom parameter keys and values that will be loaded by the custom connection code.

Return:

A custom connection object that implements cml.data_v1.customconnection.CustomConnection

Usage: conn = load_custom_connection_source("myconndir", {"HOSTNAME": "my.instance.host.com"})

Note: When you make changes to your custom connection file, get_custom_connection_from_local will dynamically re-import the contents, so the latest code on disk is instantiated for the local connection.

To load any created data connection that is available in the CML project, use the get_connection instruction.

```
get_connection(dataconnection_name, parameters=None)
```

Usage: conn = get_connection(connection_name)

If the specified connection is of type “Custom” a snapshot of the Custom Data Connection files specified at the time of Connection creation is imported.

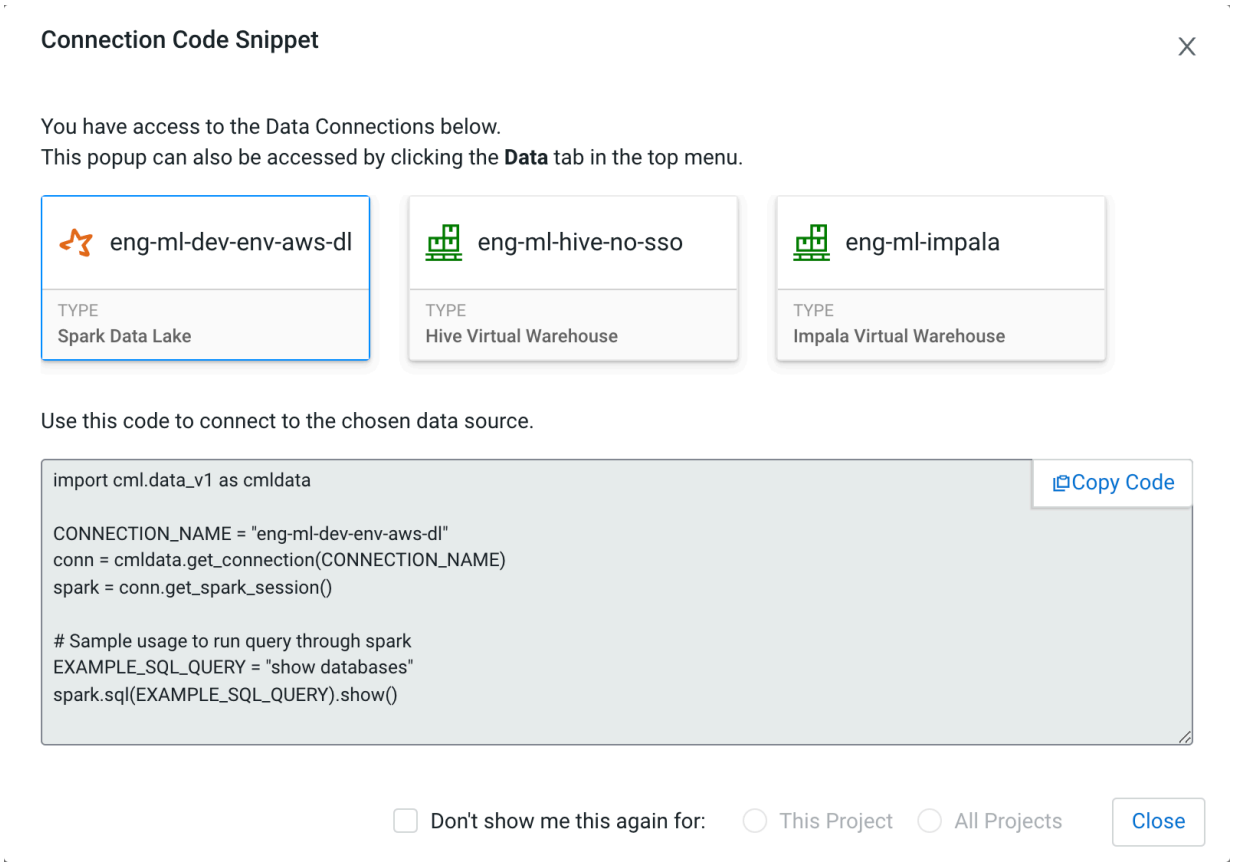
Using data connection snippets

As a data scientist, you can connect your project to data with a data connection snippet.

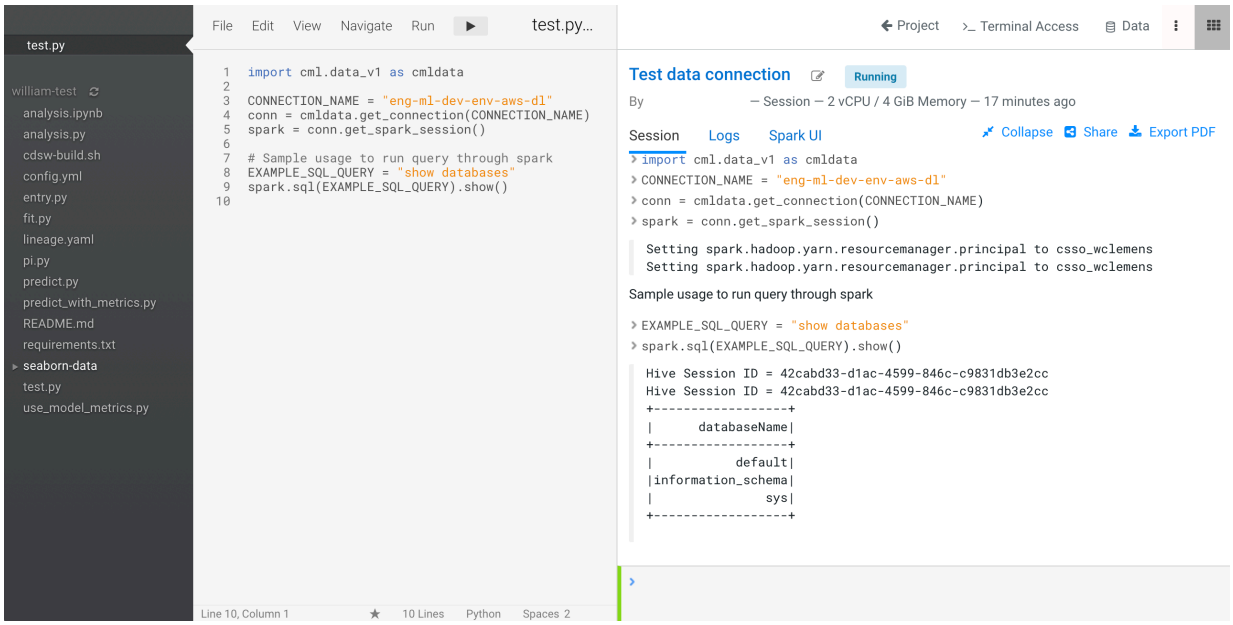
Procedure

1. Select New Project.
2. Enter the project name.
3. Start a new session. To use a Spark connection, make sure to select an ML Runtime engine.

4. In the Connection Code Snippet pane, select the data connection to use.



5. Select Copy Code.
6. Paste the snippet into your code.



7. Edit the username and password, if necessary. (This is not necessary for the Spark Data Lake connection, or if the WORKLOAD_PASSWORD environmental variable is set up for the user.)
8. Enter the data connection name.
9. Uncomment and edit the SQL query.

10. Select Run.

Results

The results of the SQL query display in the command window.

What to do next

When you have finished exploring the available data sources, you can select Don't show me this again for this project on the Connection Code Snippet pane, and it will no longer display when you start a session in the project.

Managing default and backup data connections

As a data scientist, you may want to set default and backup data connections (virtual warehouses). This makes it easy to manage the case where the default data source (virtual warehouse or data lake) becomes unavailable, for example.

Before you begin

You need Admin privileges to perform this task. Here, the data connection names of default and backup are used as examples.

Procedure

1. In Site Administration Data connections , click Edit to change the name of the default data connection to default.
2. Change another connection to backup.
3. In Project Settings, click Sync with Workspace to update the names of the data connections. If two different data connections have the same name, an error occurs during synchronization.
4. If the default data connection becomes unavailable, the Workspace Administrator can go to the Data Connections tab, and rename the connections.

   For example, after changing default to unavailable, change backup to default. Projects that use library code for their connection to the default data connection will continue to operate, because it is now using the new default connection. Note: you need to click Sync with Workspace at the project level to get the updated connections.

API Permissions For Projects

The tasks available to collaborators depends on their level of access.

A collaborator with read-only access is able to do the following:

- List data connections
- Get data connection details

Note: The list of data connections is unavailable in the UI to viewer-only users because the Setting tab in a project is hidden for viewer collaborators. However, they can use curl to submit api requests to list the data connections.

A collaborator with write access can:

- Create data connections
- Edit data connections
- Delete data connections
- List data connections
- Get data connection details

Troubleshooting: 401 Unauthorized

Problem: Session returns a 401: Unauthorized error.

When you are in a session and try to run the code for a Hive or Impala connection, the session returns a 401: Unauthorized HTTP error.

```
Modify next 2 lines to update your credentials

> USERNAME = os.getenv('HADOOP_USER_NAME')
> PASSWORD = os.getenv('WORKLOAD_PASSWORD')
> conn = cmldata.getConnection({
            'CONNECTION_NAME': CONNECTION_NAME,
            'USERNAME': USERNAME,
            'PASSWORD': PASSWORD
        })
HttpError: HTTP code 401: Unauthorized
HttpError                                 Traceback (most recent call last)
<ipython-input-1-a69c5e2af83b> in <module>
      2          'CONNECTION_NAME': CONNECTION_NAME,
      3          'USERNAME': USERNAME,
----> 4          'PASSWORD': PASSWORD
      5        })

~/.local/lib/python3.7/site-packages/cmldata.py in getConnection(properties)
    105     fmt_codesnippet = _getConnectionSnippet(properties)
    106     scope = {}
--> 107     exec(fmt_codesnippet, scope)
    108     return scope["conn"]

<string> in <module>

<string> in getCursor(self)

/usr/local/lib/python3.7/site-packages/impala/hiveserver2.py in cursor(self, user, configuration, convert_type
    127         log.debug('cursor(): getting new session handle')
>
```

Solution: You need to set your workload password. See *Setting the workload password* for details. After setting the workload password, start a new session. If this error occurs in the Data tab, then restart the Data Discovery and Exploration app.

Related Information

Setting the workload password

Troubleshooting: 401 Unauthorized when accessing Hive

Problem: Session returns a 401: Unauthorized error when you are accessing a Hive database.

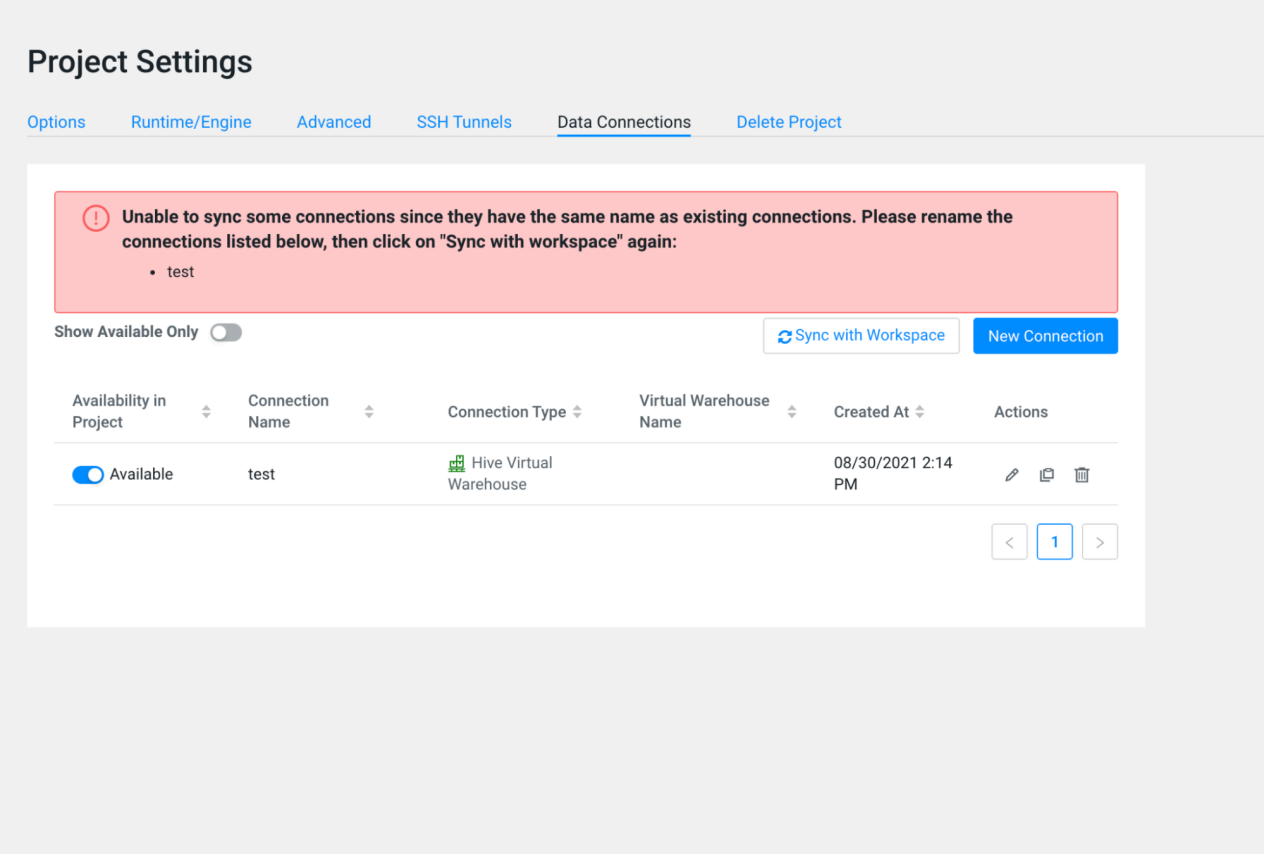
Check that the Hive data warehouse is working.

Solution: Follow these steps to check that Hive data warehouse is running, and restart if needed.

1. In the control plane, go to Data Warehouses.
2. Click Virtual Warehouses, and select the data warehouse for the connection.
3. Check the status of that data warehouse, and make sure that it is running or in a good state.

Troubleshooting: Existing connection name

Problem: When the user attempts to sync data connections, an error message displays, stating the crn or name is a duplicate.



Solution: This indicates a project connection (one that is not copied from the workspace) has the same name or crn as a workspace connection. To resolve this, you need to change the name or crn of the data connection at the project level.

Troubleshooting: Empty data page

You launch the application and see the spinner, but then the empty data page returns.

On the applications tab, click Data Discovery and Visualization application, and check the logs. Check if another user stopped or restarted the application.

On the applications tab, click on Settings Resource Profile , and check if the available resources are sufficient. You can increase the resources if necessary, then restart the application.

Troubleshooting: Some connections not shown

The data application doesn't show all data connections.

See the error message to fix the workload password. To set the workload password, follow the procedure in *Setting the workload password*. You will need to restart the data application after setting the workload password.

Alternatively, you may need to click Sync under the Data Connections to get all of the data connections available at the project level.

Related Information

Setting the workload password