Cloudera Data Visualization 7.2.8

Working with Data APIs

Date published: 2020-10-30
Date modified: 2024-10-30

CLOUDERA

Legal Notice

Contents

Enabling Data APIs

To enable Cloudera Data Visualization Data API, update the configuration settings and restart the Cloudera Data Visualization instance.

About this task

Note: This is only possible for Cloudera Data Visualization instances running in Cloudera Machine Learning and Cloudera Data Science Workbench.

Before you begin

Before invoking the platform-name Data API, you must first enable it in the configuration file of the Web Server and obtain an API key.

If you want to access the API outside of Cloudera Machine Learning or Cloudera Data Science Workbench jobs or sessions, you have to enable Enable Unauthenticated Access for the application. For more information and instructions, see the Public Applications section in Securing Applications.

Procedure

1. Click the Gear icon on the main navigation bar to open the Administration menu and select Site Settings.
2. Select Advanced Settings from the left navigation and scroll down to the Advanced Site Settings section.
3. Add the ENABLE_API_KEYS and DATA_API_ENABLED settings.

```
ENABLE_API_KEYS = True  # If you plan to access via API keys
DATA_API_ENABLED = True
```

4. Save your changes and restart the Cloudera Data Visualization service.

Using API keys

When you do not log into the visual layer of Cloudera Data Visualization directly, you must obtain an API key and implement it when using Data APIs.

Before you begin
Create a new API key. For instructions, see *Creating new API keys*.

Procedure

1. Use the API key that you have obtained to authenticate the user to access the dataset.
2. Invoke the data layer of Cloudera Data Visualization at the following endpoint and proceed with using the Data API interface.

```
http(s)://server:port/arc/api/data
```

For more information, see *Example of Data API usage*.

Related Information
Creating new API keys
Example of Data API usage

Accessing Data API request payload

To avoid creating Data API requests from scratch, Cloudera Data Visualization provides access to the details of the data requests.

About this task

This task is optional.

Procedure

1. Open Site Settings and scroll to Advanced Settings at the bottom of the left navigation.
2. Add the following settings.

   ```
   ENABLE_DSREQ_PERF_DISPLAY = True
   COMPRESS_DSREQ = True
   ```

3. Restart the Cloudera Data Visualization service.
4. Open an existing dashboard.
5. Use the keyboard shortcut <u>Shift</u> + <u>Control/Cmd</u> + <u>G</u> to open the Performance Profile interface.

6. Hover over a line that represents a visual ID and view the duration, type, and details of the query that runs against that visual.

The query uses the same API and includes the details of the data requests. You can use this dataset request as the payload for your Data API call.
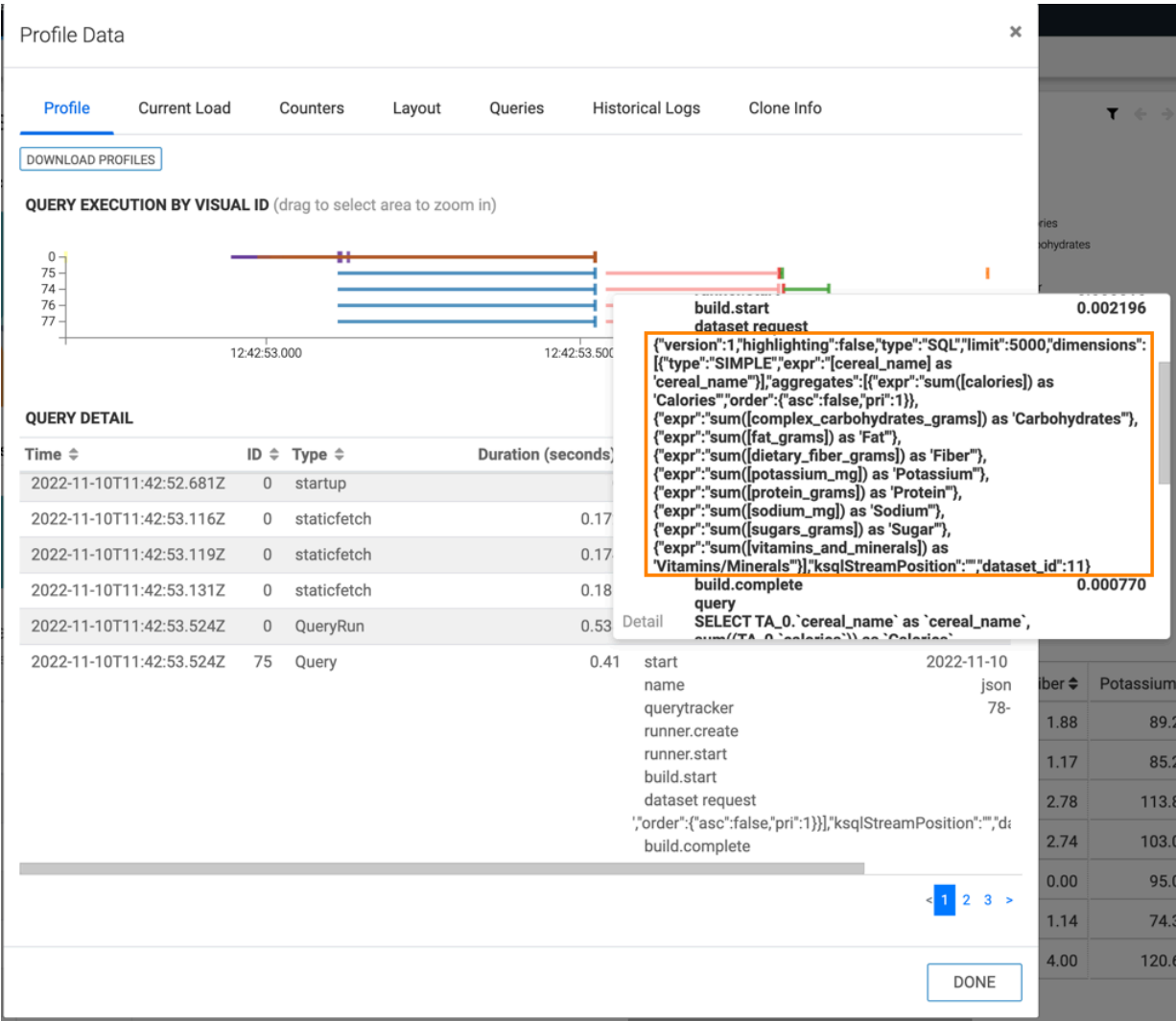
Migrating visual artifacts with REST APIs

In addition to UI-enabled export and import, Cloudera Data Visualization provides REST APIs that give more flexibility in choosing what objects to migrate. While migrating artifacts using the UI allows you to directly specify export of visuals, dashboards, and apps, REST APIs enable you to specify a wider range of migratable objects.

About this task

Note: You do not migrate events and segments separately, but as part of dataset migration. Similarly, you do not migrate thumbnails along with dashboards or visuals that they represent.

Before you begin

Enable and create the relevant data API keys, both for the source and destination machines. For instructions, see *Enabling API keys* and *Creating new API keys*. The examples used in these instructions use the secret API key that appears when you generate the keys for the first time, as in *Creating new API keys*.

Procedure

1. Export the objects by submitting a GET request on the source machine. For instructions, see *Exporting visual artifacts with REST APIs*.
2. Import the objects by submitting a POST request on the target machine. For instructions, see *Importing visual artifacts with REST APIs*.

Related Information

Enabling API keys

Creating new API keys

Exporting visual artifacts with REST APIs

Procedure

1. Authenticate the user using the data API.
2. Submit a GET request to the following address for a source machine: *source_arcviz_ip*/arc/migration/api/export/

   For instructions on import, see *Importing visual artifacts with REST APIs*.

Example

This is a sample Python code snippet for authenticating and submitting the GET request that exports dashboards with IDs 2115 and 2110, and then writing the response to the my_app.json file.

```
export requests
headers = {'AUTHORIZATION':'apikey secret_api_key'}
payload = {'dashboards': '[2115, 2110]', 'filename': 'apitestmigration',
'dry_run':'False'}
r = requests.get('source_arcviz_ip/arc/migration/api/export/', headers=hea
ders, params=payload)
with open('my_app.json', 'w') as f:
        f.write(r.text)
```

Related Information

Importing visual artifacts with REST APIs

Importing visual artifacts with REST APIs

Before you begin

Make sure that the JSON file with information about exported visual artifacts is on the destination machine.

Procedure

1. Authenticate the user using the data API.
2. Submit a POST request to the following address for a destination machine: *destination_arcviz_ip*/arc/migration/api/import/

   For instructions on export, see *Exporting visual artifacts with REST APIs*.

Example

This is a sample Python code snippet for authenticating and submitting the POST request that imports dashboards with IDs 2115 and 2110, which were earlier saved (from a metadata perspective) as my_app.json. This file is uploaded to the Cloudera Data Visualization server during the import.

```
import requests
headers = {'AUTHORIZATION':'apikey secret_api_key'}
payload = {'dry_run': False, "dataconnection_name":"data_connection"}
files = {'import_file': open('/Users/my_name/Downloads/my_app.json','r')}
r = requests.post('destination_arcviz_ip/arc/migration/api/import/',files=fi
les, data=payload, headers=headers)
print r.status_code # 200 is success
```

Related Information

Exporting visual artifacts with REST APIs