Cloudera Stream Processing 7.2.10

Using Schema Registry

Date published: 2019-11-08
Date modified: 2021-06-08

https://docs.cloudera.com/

Legal Notice

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Adding a new schema

To add a new schema to Schema Registry, provide information about the schema entities, select a compatibility policy and upload the schema text from a file. Schema entities are the collection of information that help you organize and sort your schemas. They include group, version, and informational metadata.

Before you begin

Ensure that you understand compatibility policies. Once selected, you cannot change the compatibility policy for a schema.

Procedure

1. From the Schema Registry UI, click the + icon.
2. Add the Schema metadata as follows:
   - Name – A unique name for each schema. Used as a key to look up schemas.
   - Description – A short description of the schema.

     Note:

     Avro is currently the only supported type.
   - Schema Type – The schema format.
   - Schema Group – Allows you to group schemas in any logical order.
   - Compatibility – Sets the compatibility policy for the schema. Once set, this cannot be changed.
3. To allow schema to evolve over time by creating multiple versions, select the Evolve checkbox.

   Note:

   Deselecting Evolve means that you can only have one version of a schema.
4. Click Choose File to upload a new schema.

Querying a schema

You can use the Search box at the top of the Schema Registry UI to search for schemas by name, description, or schema text. To search, you can enter a schema name, key words, or any text string to return schemas with that value.

To return to your full list of schemas, clear the search bar and press Enter on your keyboard.

Evolving a schema

You evolve a schema when you create a new version. Schema Registry tracks the changes made to your schema, and stores each set of changes in a separate version of the schema. When multiple versions exist, you can select which version you want to use. Ensure that you understand compatibility policies, as they determine how you can evolve your schemas.

Before you begin

- You have selected the Evolve checkbox when initially adding the schema.
- You have the schema you want to evolve saved to a file.

Procedure

1. From the Schema Registry UI, identify the schema that you want to version.
2. Click the pencil icon to open the Edit Version dialog.
3. Add a description of what has changed in this new version of the schema. You can view the description in the Schema Registry UI to easily understand what has changed in each version of the schema so we recommend that you add as much detail as you can.
4. Click Choose File to upload the schema you want to evolve.
5. Click Ok.

Deleting a schema

You can use the Schema Registry REST API to delete schemas.

About this task

You can access the Schema Registry API Swagger documentation directly from the UI. To do this, append your URL with: /swagger/

For example: http://localhost:9090/swagger/

Procedure

1. Find the DELETE /api/v1/schemaregistry/schemas/{schema_name} API under "Schema".
2. Replace {schema_name} with the schema you wish to delete.
3. Invoke the API.

Importing Confluent Schema Registry schemas into Cloudera Schema Registry

If you are migrating away from Confluent's Schema Registry, you can use the schema-import tool to import your schemas to the Cloudera Schema Registry service.

About this task

Confluent Schema Registry stores all schemas in a Kafka topic usually named _schemas. To import these schemas you must copy the content to a text file and upload to Cloudera's Schema Registry service.

Before you begin

You must have the export-import access policy with the Create permission assigned to your user account.

For more information on Schema Registry access policies, see *Schema Registry Authorization through Ranger Access Policies*

Procedure

1. Copy the contents of the Kafka topic (usually named _schemas), to a text file. Use the following command:

   ```
   kafka-console-consumer --from-beginning --topic _schemas --formatter kafka.tools.DefaultMessageFormatter --property print.key=true --property print.value=true --bootstrap-server `hostname`:9092
   ```

   Here is an example of a text file with the contents of the Kafka topic:

   ```
   {"keytype":"SCHEMA","subject":"Kafka-key","version":1,"magic":1}
   {"subject":"Kafka-key","version":1,"id":1,"schema":"\"string\"","deleted":false}
   {"keytype":"SCHEMA","subject":"Car","version":1,"magic":1}      {"subject":"Car","version":1,"id":2,"schema":"{\"type\":\"record\",\"name\":\"Car\",\"namespace\":\"com.cloudera\",\"fields\":[{\"name\":\"model\",\"type\":\"string\"},{\"name\":\"color\",\"type\":\"string\",\"default\":\"blue\"},{\"name\":\"price\",\"type\":\"string\",\"default\":\"0\"},{\"name\":\"year\",\"type\":[\"null\",\"string\"],\"default\":null}]}","deleted":false}
   {"keytype":"SCHEMA","subject":"Car","version":2,"magic":1}      {"subject":"Car","version":2,"id":3,"schema":"{\"type\":\"record\",\"name\":\"Car\",\"namespace\":\"com.cloudera\",\"fields\":[{\"name\":\"model\",\"type\":\"string\"},{\"name\":\"color\",\"type\":\"string\",\"default\":\"blue\"},{\"name\":\"price\",\"type\":\"string\",\"default\":\"0\"}]}","deleted":false}
   ```

2. Upload the text file to Cloudera's Schema Registry service using the following endpoint: /api/v1/schemaregistry/import

   Use the following command:

   ```
   curl -u : --negotiate --form file='<filename>' http://`hostname`:7788/api/v1/schemaregistry/import?format=1&failOnError=true
   ```

   Where:

   - format: Can be 0 or 1. However, for schema content from Confluent, the file must be 1.
   - failOnError=true: This setting is useful if the process fails to import all the schemas when one or more schemas already exist in the target Schema Registry.
   - file='<filename>: Enter the name of the file.

   Here is an example of the command:

   ```
   curl -u : --negotiate --form file='@schemadump.json' http://`hostname`:7788/api/v1/schemaregistry/import?format=1&failOnError=true
   ```

   Here is an example of the response:

   ```
   {"successCount":45,"failedCount":0,"failedIds":[]}
   ```