Cloudera Streaming Analytics 1.6.0

Connectors

Date published: 2019-12-17
Date modified: 2021-12-15



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

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.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

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

Connector support in SSB 4
Kafka connectors 4
CDC connectors 5
JDBC connector 7
Filesystem connector 7
Datagen connector 8
Faker connector 8
Blackhole connector 9

Managing connectors and data formats 9
Adding new connectors 10
SSB fails with missing JAR file error 12
Adding data formats 12

Connector support in SSB

SQL Stream Builder (SSB) supports different connector types and data formats for Flink SQL tables to ease development and access to all kinds of data sources.

The following table summarizes the supported connectors and how they can be used in SSB:

Connector	Type	Description
Kafka	source/sink	Supported as exactly-once-sink
Hive	source/sink	Can be used as catalog
Kudu	source/sink	Can be used as catalog
Schema Registry	source/sink	Can be used as catalog
JDBC	source/sink	Can be used with Flink SQL. PostgreSQL, MySQL and Hive are supported.
Filesystems	source/sink	Filesystems such as HDFS, S3 and so on. Can be used with Flink SQL
Debezium CDC	source	Can be used with Flink SQL. PostgreSQL, MySQL, Oracle DB are supported.
Webhook	sink	Can be used as HTTP POST/PUT with templates and headers
PostgreSQL	sink	Materialized View connection for reading views. Can be used with anything that reads PostgreSQL wire protocol
REST	sink	Materialized View connection for reading views. Can be used with anything that reads REST (such as notebooks, applications, and so on)
BlackHole	sink	Can be used with Flink SQL.

Kafka connectors

When using the Kafka connector, you can choose between using an internal or external Kafka service. Based on the connector type you choose, there are mandatory fields where you must provide the correct information.

You can choose from the following Kafka connectors when creating a table in Streaming SQL Console:

Template: local-kafka

Using the Kafka service that is installed on your cluster.

Type: source/sink

The following fields are mandatory to use the connector:

- scan.startup.mode: Startup mode for the Kafka consumer. group-offsets is the default value. You can choose from earliest-offset, latest-offset, timestamp and specific-offsets as startup mode.
- topic: The topic from which data is read as a soruce, or the topic to which data is written to. No default value is specified. You can also add a topic list in case of sources. In this case, you need to separate the topics by semicolon. You can only specify the topic-pattern or topic for the sources.
- format: The format used to deserialize and serialize the value part of Kafka messages. No default value is specified. You can use either the format or the value.format option.

Template: kafka

Using an external Kafka service as a connector. To connect to the external Kafka service, you need to speficy the Kafka brokers that are used in your deployment.

Type: source/sink

The following fields are mandatory to use the connector:

- properties.bootstrap.servers: Specifying a list of Kafka brokers that are separated by comma. No default value is specified.
- topic: The topic from which data is read as a soruce, or the topic to which data is written to. No default value is specified. You can also add a topic list in case of sources. In this case, you need to separate the topics by semicolon. You can only specify the topic-pattern or topic for the sources.
- format: The format used to deserialize and serialize the value part of Kafka messages. No default value is specified. You can use either the format or the value.format option.

Template: upsert-kafka

Using the upsert Kafka service as a connector. For more information about the upsert Kafka connector, see the Apache Flink documentation.

Type: source/sink

- properties.bootstrap.servers: Specifying a list of Kafka brokers that are separated by comma. No default value is specified.
- topic: The topic from which data is read as a soruce, or the topic to which data is written to. No default value is specified. You can also add a topic list in case of sources. In this case, you need to separate the topics by semicolon. You can only specify the topic-pattern or topic for the sources.
- key.format: The format used to deserialize and serialize the key part of Kafka messages. No default value is specified. Compared to the regular Kafka connector, the key fields are specified by the PRIMARY KEY syntax.
- value.format: The format used to deserialize and serialize the value part of Kafka messages. No default value is specified. You can use either the format or the value.format option.

Using the Kafka connectors

You can access and import the templates of the Kafka connectors from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select one of the Kafka templates you want to use.

   The template is imported to the SQL window.
5. Provide information to the mandatory fields of the template.
6. Click Execute.

CDC connectors

You can use the Debezium Change Data Capture (CDC) connector to stream changes in real-time from MySQL, PostgreSQL, Oracle, Db2 and feed data to Kafka, JDBC, the Webhook sink or Materialized Views using SQL Stream Builder (SSB).

Concept of Change Data Capture

Change Data Capture (CDC) is a process to capture changes in a source system, and update the data within a downstream system or application with the changes.

The Debezium implementation offers CDC with database connectors from which real-time events are updated using Kafka and Kafka Connect. Debezium captures every row-level change in each database table of an event stream. Applications read these streams to see the change events in the same order as they occurred. The change events are routed to a Kafka topic from which Kafka Connect feeds the records to other systems and databases.

For more information about Debezium, see the official Debezium site.

CDC in Cloudera Streaming Analytics (CSA) does not require Kafka or Kafka Connect as Debezium is implemented as a library within the Flink runtime. This means that the captured changes are propagated downstream to any connector that Flink supports. CSA allows queries to be issued at change data capture time, which means filtering, grouping, joining, and so on, can be performed on the change stream as it comes from the source database.

For more information about the Flink implementation of Debezium, see the official Apache Flink documentation.

From the supported set of Debezium connectors, MySQL, PostgreSQL, Oracle and Db2 are supported in Cloudera Streaming Analytics.



Note: The Db2 connector is only supported for Linux operating system.



Note: You need to configure the databases, users and permissions for the supported connectors before you are able to use them in SSB. For more information about setting up the databases, see the MySQL, PostgreSQL, Oracle, Db2 documentation.

Using the CDC connectors

You can access and import the templates of the CDC connectors from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select one of the CDC templates you want to use.

   The template is imported to the SQL window.
5. Provide information to the mandatory fields of the template.
6. Click Execute.

Using Db2 CDC connector

When you want to use Db2 from the CDC connectors, you can use the following reference example to create a Db2 CDC table:

```
CREATE TABLE db2_cdc_source (
 'COLUMN_NAME' INT,
 'COLUMN_NAME' STRING
) WITH (
    'connector' = 'db2-cdc',
 'hostname' = '...',
 'port' = '...',
 'username' = '...',
 'password' = '...',
```

```
  'database-name' = '...',
        'schema-name' = '...',
  'table-name' = '...'
)
```

JDBC connector

When using the JDBC connector, you can choose between using a PostgreSQL, MySQL or Hive databases. Based on the connector type you choose, there are mandatory fields where you must provide the correct information.

Template: jdbc

Using either PostgreSQL, MySQL or Hive as databases. When you use the JDBC connector, you must specify the JDBC database which are going to be used for the connection.

Type: source/sink

The following fields are mandatory to use the connector:

- url: The URL of the JDBC database. No default value is specified.
- table-name: The name of the JDBC table in the database that you need to connecto to. No default value is specified.

Using the JDBC connector

You can access and import the templates of the JDBC connector from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select the JDBC template.

   The template is imported to the SQL window.
5. Provide information to the mandatory fields of the template.
6. Click Execute.

Filesystem connector

When using the Filesystem connector, you can choose between HDFS, S3 and so on storage systems. Based on the connector type you choose, there are mandatory fields where you must provide the correct information.

Template: filesystem

Using either HDFS, S3 or any type of storage system. When you use the Filesystem connector, you must specify the path to the file system which are going to be used for the connection.

Type: source/sink

The following fields are mandatory to use the connector:

- path: Path to the root directory of the table data. No default value is specified.
- format: The format used to for the file system. No default value is specified.

Using the Filesystem connector

You can access and import the templates of the Filesystem connector from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select the Filesystem template.

   The template is imported to the SQL window.
5. Provide information to the mandatory fields of the template.
6. Click Execute.

Datagen connector

The Datagen connector can be used a source to generate random data. The Datagen connector works out of the box, no mandatory field is required to use the connector. The Datagen connector is useful when you want to experiment and try out SQL Stream Builder or to test your SQL queries using random data.

Template: datagen

You do not need to provide any type of information when using the Datagen connector and template.

Type: source

Using the Datagen connector

You can access and import the templates of the Datagen connector from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select the Datagen template.

   The template is imported to the SQL window.
5. Click Execute.

Faker connector

The Faker connector can be used a source to generate random data. To use the Faker connector, you need to specify the usage The Datagen connector is useful when you want to experiment and try out SQL Stream Builder or to test your SQL queries using random data.

Template: faker

You do not need to provide any type of information when using the Datagen connector and template.

Type: source

The following fields are mandatory to use the connector:

- fields.#.expression: The Java Faker expression to generate the values for a specific field. For more information about the list and use of the Faker expressions, see the flink-faker documentation.

Using the Faker connector

You can access and import the templates of the Datagen connector from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select the Faker template.
5. Provide information to the mandatory fields of the template.
6. Click Execute.

Blackhole connector

The Blackhole connector can be used a sink where you can write any type of data into. The Blackhole connector works out of the box, no mandatory field is required to use the connector. The Blackhole connector is useful when you want to experiment and try out SQL Stream Builder or to test your SQL queries using random data.

Template: blackhole

You do not need to provide any type of information when using the Blackhole connector and template.

Type: sink

Using the Blackhole connector

You can access and import the templates of the Blackhole connector from Streaming SQL Console:

1. Navigate to the Streaming SQL Console.
   a. Go to your cluster in Cloudera Manager.
   b. Click on SQL Stream Builder from the list of Services.
   c. Click on the SQLStreamBuilder Console.

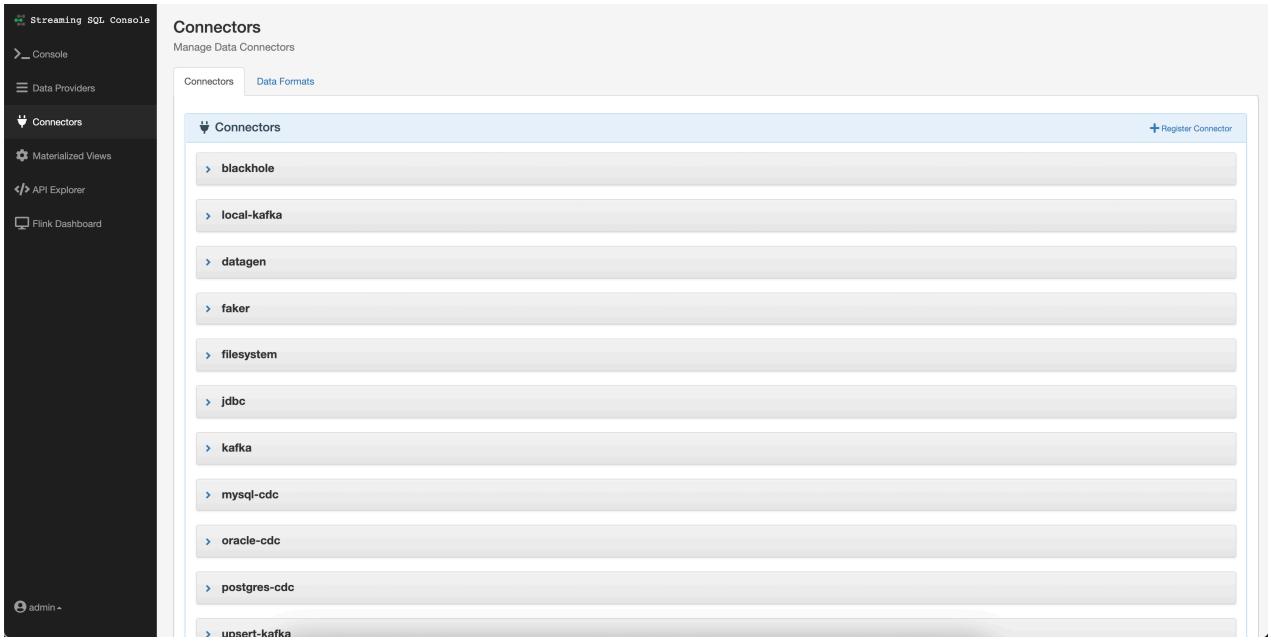
   The Streaming SQL Console opens in a new window.
2. Select Console from the main menu.
3. Click Templates under the SQL window.
4. Select the Blackhole template.

   The template is imported to the SQL window.
5. Click Execute.

Managing connectors and data formats

You can add new connectors and data formats in SQL Stream Builder (SSB) using the Streaming SQL Console. The newly added connectors and data formats will appear in Templates under the SQL window on the Compose page.

The newly added connectors and data formats are automatically added to the list of Templates in SSB. The templates are built based on the properties and data formats defined for the connectors.



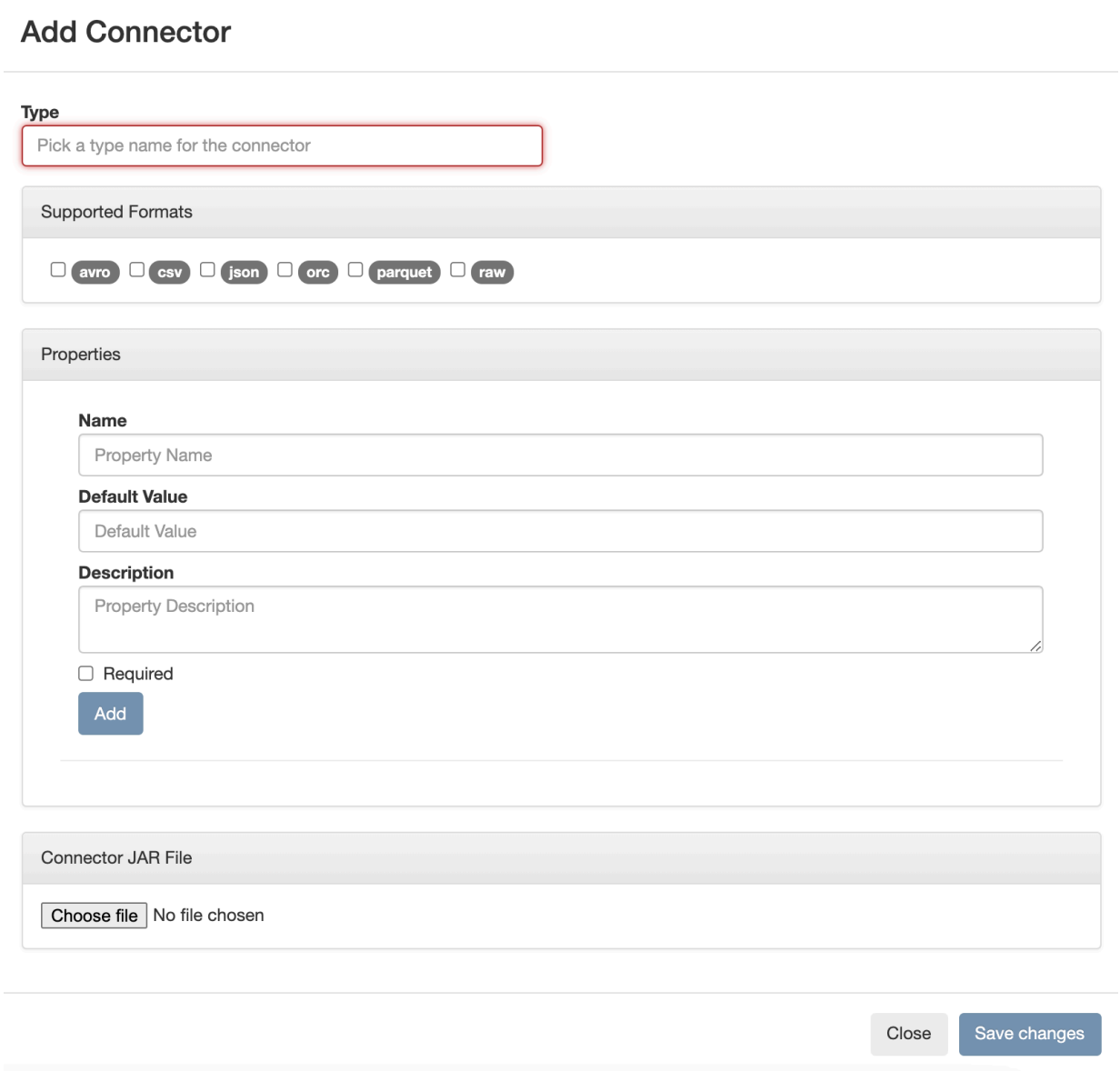
Adding new connectors

When adding new connectors to SQL Stream Builder, you need to specify the property list, data types and must upload the connector JAR file to the Console.

Procedure

1. Select Connectors on the main menu.

2. Click Register Connector.



3. Provide a name for the connector as Type.
4. Select a data format to be supported for the connector from the Supported Formats.
5. Provide properties that are mandatory or optional for the connector.
   a) Add a name to the property.
   b) Add a default value to the property.
   c) Add a description to the property.
   d) Check the Required checkbox to make a property mandatory.
   e) Click Add.

   You can specify as many properties as needed for the connector.

6. Upload the connector JAR file.

   Note: When uploading a JAR file larger than 1 MB, you need to set the server.tomcat.max-swallow-size in Cloudera Manager using the following steps:

   a. Open your cluster in Cloudera Manager.
   b. Select SQL Stream Builder from the list of services.
   c. Select Configuration.
   d. Search for Streaming SQL Engine Advanced Configuration Snippet (Safety Valve) for ssb-conf/application.properties in the search bar.
   e. Add server.tomcat.max-swallow-size=2000MB to the **Safety Valve**.
   f. Click Save.
   g. Restart the SQL Stream Builder service.

7. Click Save changes.

   The newly added connector is listed on the Connectors page.

SSB fails with missing JAR file error

Condition

When creating a custom connector by uploading a JAR file and adding it to Streaming SQL Console, after a period of time the job submission fails when using the newly created connector with a JAR file missing error. This can be solved by changing the location of the storage directory in Cloudera Manager.

Cause

The uploaded JAR files are stored under /tmp/ssb_artifacts in SQL Stream Builder (SSB). This temporary folder can be cleaned up periodically by the Operating System (OS), which deletes the JAR files.

Remedy

Procedure

1. Open your cluster in Cloudera Manager.
2. Select SQL Stream Builder from the list of services.
3. Select Configuration.
4. Search for Streaming SQL Engine Advanced Configuration Snippet (Safety Valve) for ssb-conf/application.properties in the search bar.
5. Add ssb.artifact.storage.dir=*[***NEW DIRECTORY PATH***]* to the **Safety Valve**.
6. Click Save.
7. Restart the SQL Stream Builder service.

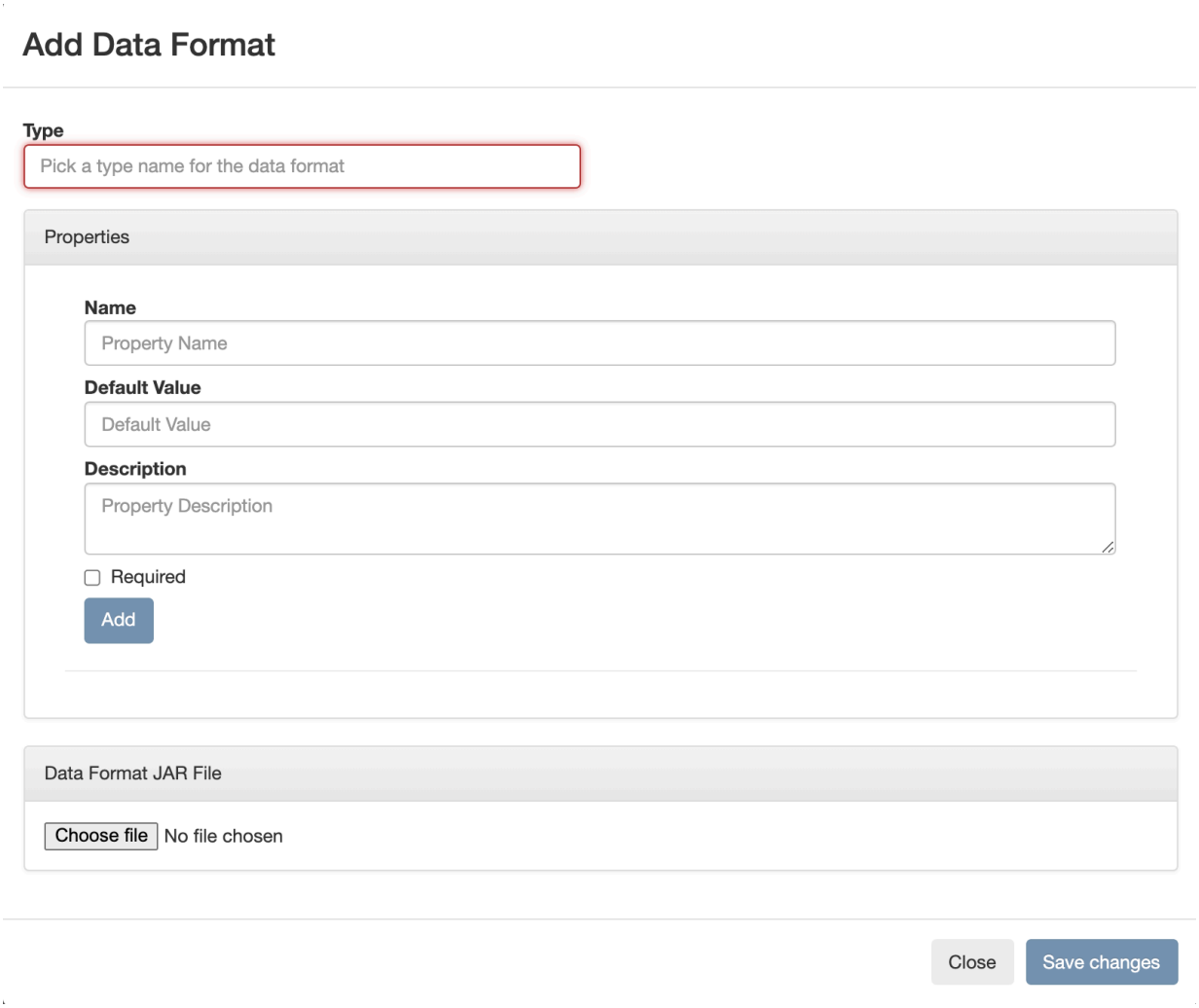
Adding data formats

When adding new data formats to SQL Stream Builder, you need to specify the property list and must upload the data format JAR file to the Console.

Procedure

1. Select Connectors on the main menu.
2. Click Data Formats.

3. Click Register Data Format.



4. Provide a name for the data format as Type.
5. Provide properties that are mandatory or optional for the data format.
   a) Add a name to the property.
   b) Add a default value to the property.
   c) Add a description to the property.
   d) Check the Required checkbox to make a property mandatory.
   e) Click Add.

   You can specify as many properties as needed for the data format.
6. Upload the data format JAR file.
7. Click Save changes.

   The newly added data format is listed on the Data Formats page.