Snowflake Authenticate a Connector Instance

Last Modified on 11/02/2021 3:58 am EDT

You can authenticate with Snowflake to create your own instance of the Snowflake connector through the UI or through APIs. Once authenticated, you can use the connector instance to access the different functionalities offered by the Snowflake platform.

Authenticate Through the UI

Use the UI to authenticate with Snowflake and create a connector instance. You will need your **database host, database name, database schema name, table names, warehouse** as well as an appropriate **username and password.**

After successfully authenticating, we give you several options for next steps. Make requests using the API docs associated with the instance, map the instance to a common resource, or use it in a formula template.

Authenticate Through API

Snowflake supports two authentication methods of which one is using the JDBC driver and another via RSA Key Pair Authentication.

The following parameters are required to create a Snowflake connector Instance via JDBC drivers:

- Database Host: e.g. `123.123.1.123:3306`
- Database Schema Name
- Warehouse
- Database Username
- Database Password
- Database Tables
- Database Name

Step 1. Create an Instance

To provision your Snowflake connector, use the `/instances` API.

Below is an example of the provisioning API call.

- **HTTP Headers**: Authorization- User , Organization
- **HTTP Verb**: POST
- **Request URL**: /instances
- **Request Body**: Required – see below
- **Query Parameters**: none

Description: a connector token is returned upon successful execution of this API. This token needs to be retained by the application for all subsequent requests involving this connector instance.

A sample request illustrating the `/instances` API is shown below.

HTTP Headers:

```
Authorization: User , Organization
```

This instance.json file must be included with your instance request. Please fill your information to provision. The “key” into SAP Open Connectors Snowflake is "snowflake". This will need to be entered in the “key” field below depending on which connector you wish to instantiate.

```
{
  "element": {
    "key": "snowflake"
  },
  "configuration": {
    "db.schema": " ",
    "filter.response.nulls": "true",
    "db.host": "",
    "db.schemaname": "",
    "db.table.names": "",
    "warehouse": "xxxxxxxxx",
    "username": "xxxxxxxx",
    "password": "xxxxxxxxxxx",
    "db.name": ""
  },
  "name": ""
}
```

Here is an example cURL command to create an instance using `/instances` API.

Example Request:

```
curl -X POST
-H 'Authorization: User , Organization '
-H 'Content-Type: application/json'
-d
'{
  "name": "",
  "configuration": {
    "db.schema": " ",
    "filter.response.nulls": "true",
    "db.host": "",
    "db.schemaname": "",
    "db.table.names": "",
    "warehouse": "xxxxxxxxx",
    "username": "xxxxxxxxxxx",
    "password": "xxxxxxxxxx",
    "db.name": ""
  }
}'
```

Below is a successful JSON response:

```
{
  "id": 220937,
  "name": "",
  "createdDate": "2019-12-02T09:30:55Z",
  "token": "xxxxxxxxxxxxxxxxxxxxxx",
  "elementId": 33285,
  "tags": [],
  "provisionInteractions": [],
  "valid": true,
  "disabled": false,
  "maxCacheSize": 0,
  "cacheTimeToLive": 0,
  "configuration": {},
  "authenticationType": "custom",
  "eventsEnabled": false,
  "traceLoggingEnabled": false,
  "cachingEnabled": false,
  "organizationId": 1190,
  "accountId": 43739,
  "externalAuthentication": "none",
  "userId": 53822,
  "element": {},
  "user": {
    "id": 53822,
    "emailAddress": "",
    "firstName": "",
    "lastName": ""
  }
}
```

Instance Configuration

The content in the `configuration` section or nested object in the body posted to the `POST /instances` or `PUT /instances/{id}` APIs varies depending on which connector is being instantiated. However, some configuration properties are common to all connectors and available to be configured for all connectors. These properties are -

- `event.notification.enabled` : This property is a `boolean` property, and determines if event reception (via `webhook` or `polling`) is enabled for the connector instance. This property defaults to *false*.
- `event.vendor.type` : When `event.notification.enabled` property is set to *true*, this property determines the mechanism to use to receive or fetch changed events from the service endpoint. The supported values are `webhook` and `polling` . Most connectors support one mechanism or the other, but some like Salesforce.com support both mechanisms. This property is *optional*.
- `event.notification.type` : This property can be used to determine how an event notification should be sent to the consumer of the connector instance, in most cases your application. Currently, `webhook` is the only supported value for this property. This means that when an event is received by the connector instance, it will get forwarded to the provided `event.notification.callback.url` via a `webhook` to you. This property is *optional*.
- `event.notification.callback.url` : As mentioned above, the value of this property is an `http` or `https` URL to which we will post the event for consumption by your application. This property is *optional*.
- `filter.response.nulls` : This property defaults to *true*, i.e., it's `boolean` property, and determines if `null` values in the response `JSON` should or should not be filtered from the response returned to the consuming application. By default, all `null` values are filtered from the response before sending the response to the consuming application.

To Authenticate with Snowflake connector Instance via RSA Key Pair Authentication, refer to Using Key Pair Authentication & Key Rotation.