Date: 24-12-2024

SNOWFLAKE cortex search

version: 1.0.0

# **OVERVIEW**

Cortex Search enables low-latency, high-quality “fuzzy” search over your Snowflake data.

## **Features:**

You can use this API to develop custom applications and integrations that:

* Perform queries using specific search services within databases and schemas.
* Retrieve metadata to manage and explore your deployment, such as available databases and schemas.

The Snowflake CORTEX SEARCH API provides operations that you can use to:

* Submit search statements for execution within a specific database and schema.
* Retrieve information about databases and schemas to enhance query execution.
* Integrate advanced query capabilities into applications with schema-specific search services for precise data access and retrieval.

**B. Percentage of re-use:**

Approximately 80% of reuse.

**2. Getting Started**

**A. Prerequisites**

## Volt Foundry

* Volt MX Iris
* [Snowflake](https://signup.snowflake.com/?utm_source=google&utm_medium=paidsearch&utm_campaign=ap-in-en-brand-trial-exact&utm_content=go-eta-evg-ss-free-trial&utm_term=c-g-snowflake%20trial-e&_bt=579374819453&_bk=snowflake%20trial&_bm=e&_bn=g&_bg=133380618168&gclsrc=aw.ds&gad_source=1&gclid=Cj0KCQiAoae5BhCNARIsADVLzZfO3xtb5qjXBOmpvJkHApciIFGNik1p4q8miBtgG3-9qs2OUtMXBvQaAjqgEALw_wcB) Account with client id and secret key.

**B. Importing the back-end service**

 To import the Snowflake backend service to Volt Foundry, do the following:

1. Download the **Snowflake back-end service** zip file from [HCL Forge](https://marketplace.hclvoltmx.com/).
2. Sign in to the [HCL Foundry](https://manage.hclvoltmx.com/) Console.
3. On the **Foundry Apps** page, click **IMPORT**.
4. On the **Import App** dialog box, drag and drop the zip file that you downloaded earlier. Alternatively, you can **browse** for the zip file on your system.  
   After the zip file is uploaded, the console displays the default **App Name** and **Version**.
5. Click **IMPORT**.

**C. Obtaining the client id and secret keys.**

1. Start a free trial by creating an account.
2. After creating the account need to provide security integration in the cortex search Worksheet of snowflake by giving those voltmx callback url and etc.

create or replace security integration DEMOHUB\_INT

type = oauth

enabled = true

oauth\_client = custom

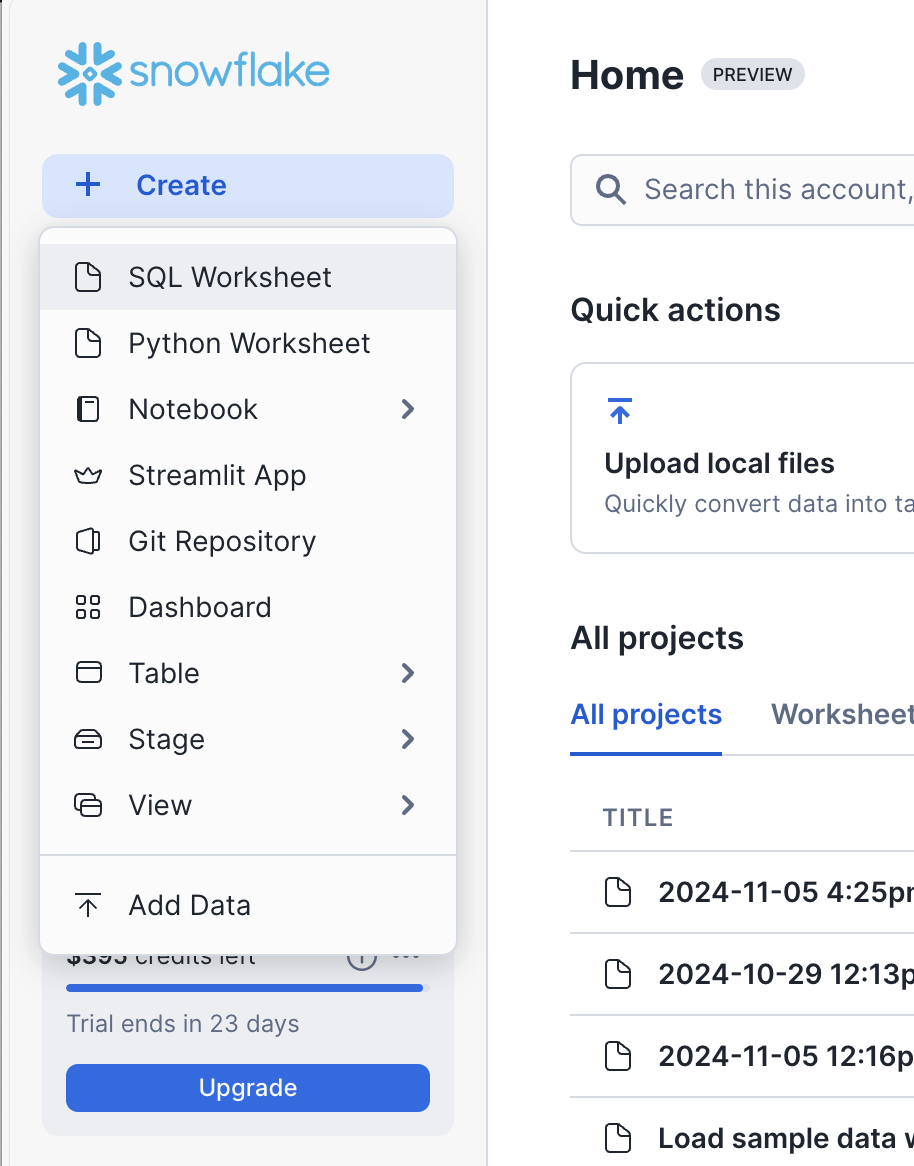
oauth\_client\_type = 'CONFIDENTIAL'

oauth\_redirect\_uri = 'https://100000017.auth.hclvoltmx.net/oauth2/callback'

oauth\_issue\_refresh\_tokens = true

oauth\_refresh\_token\_validity = 86400

;

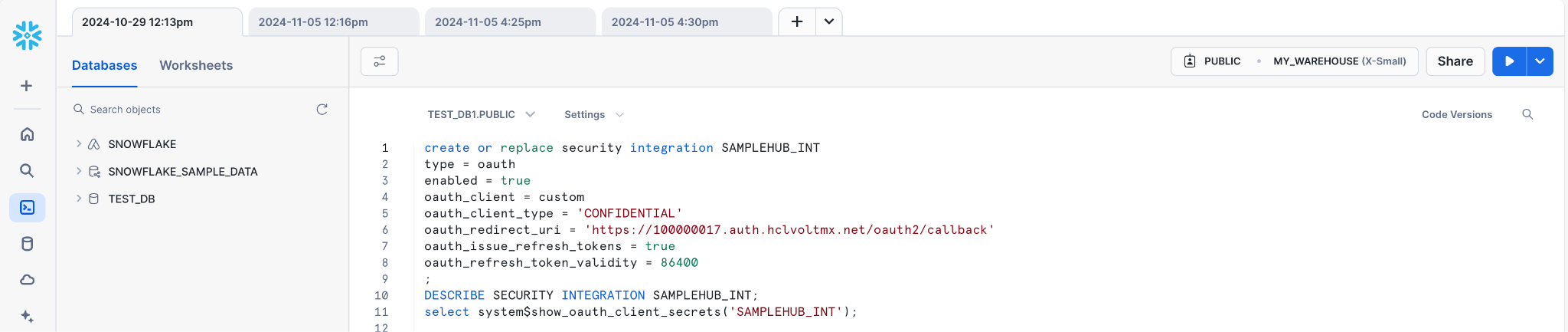


1. After we need to get the keys for that we need to give this query in cortex search worksheet. We will get client id, authorisation end point, token end point.

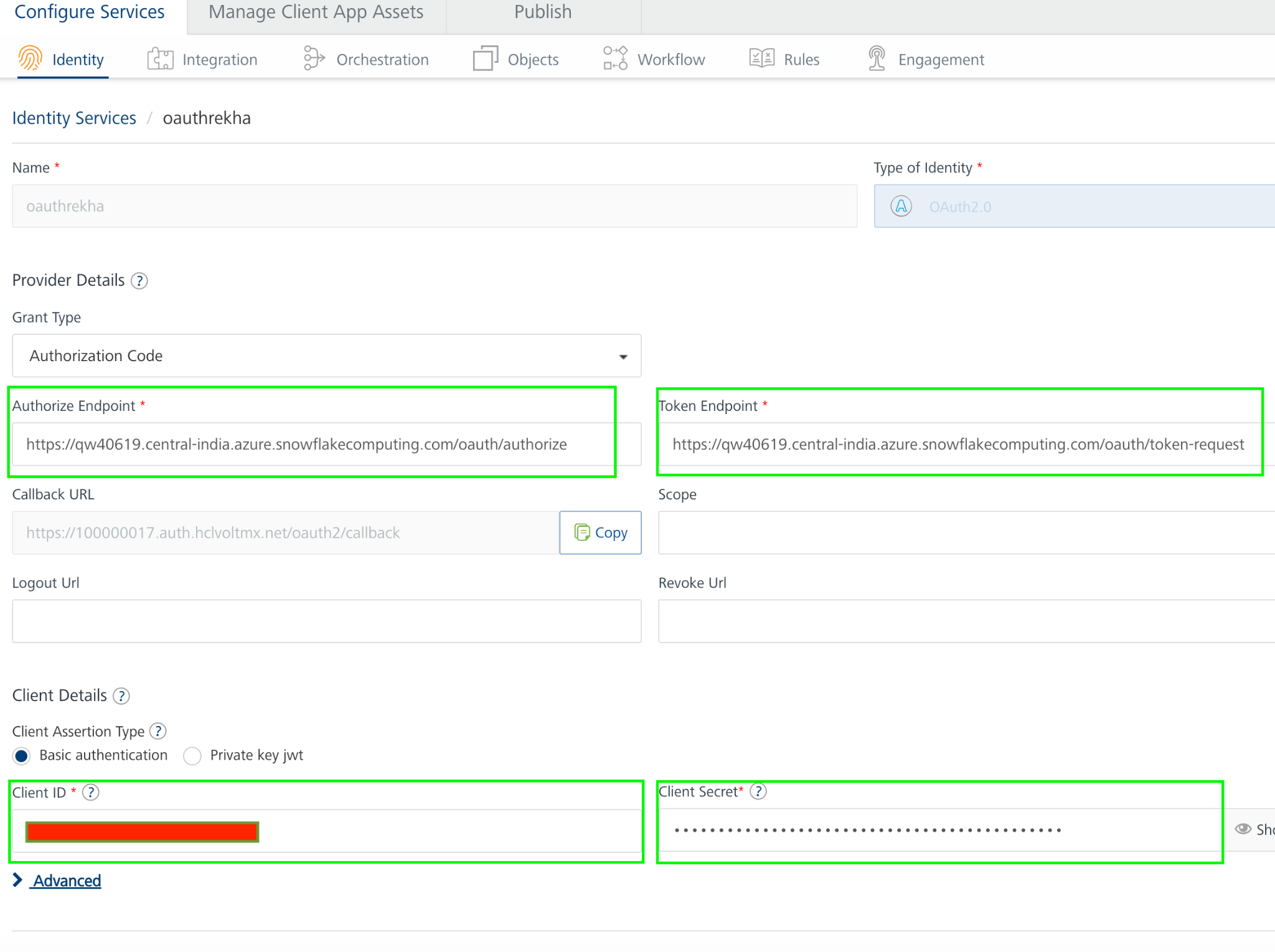
DESCRIBE SECURITY INTEGRATION DEMOHUB\_INT;

4. After that we need to get the secret key for that we need to give this query.

select system$show\_oauth\_client\_secrets('DEMOHUB\_INT');



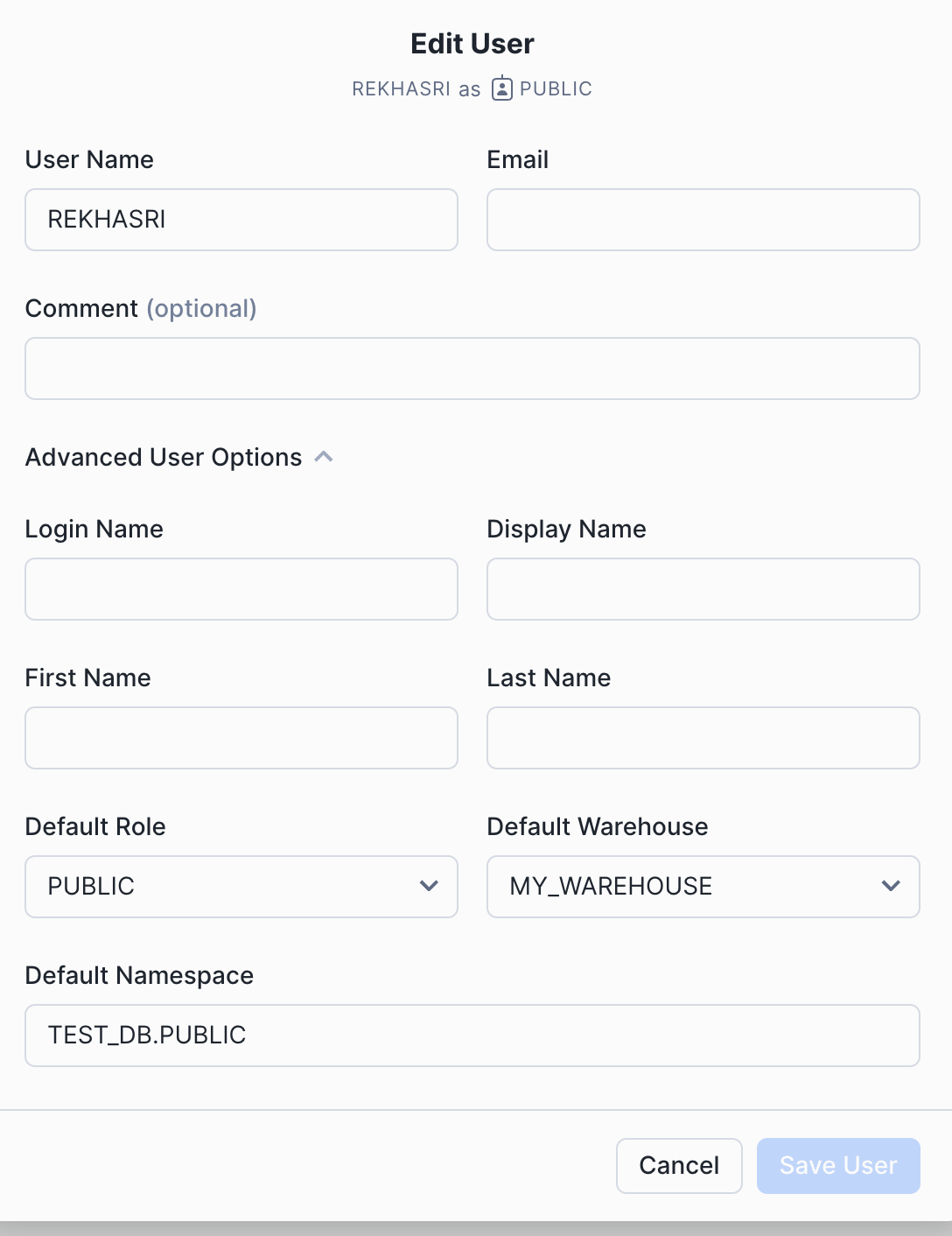
reference screenshot for using the client id, authorization end point, token end point, secret key.



5. By using these we will create Identity service in Foundry.

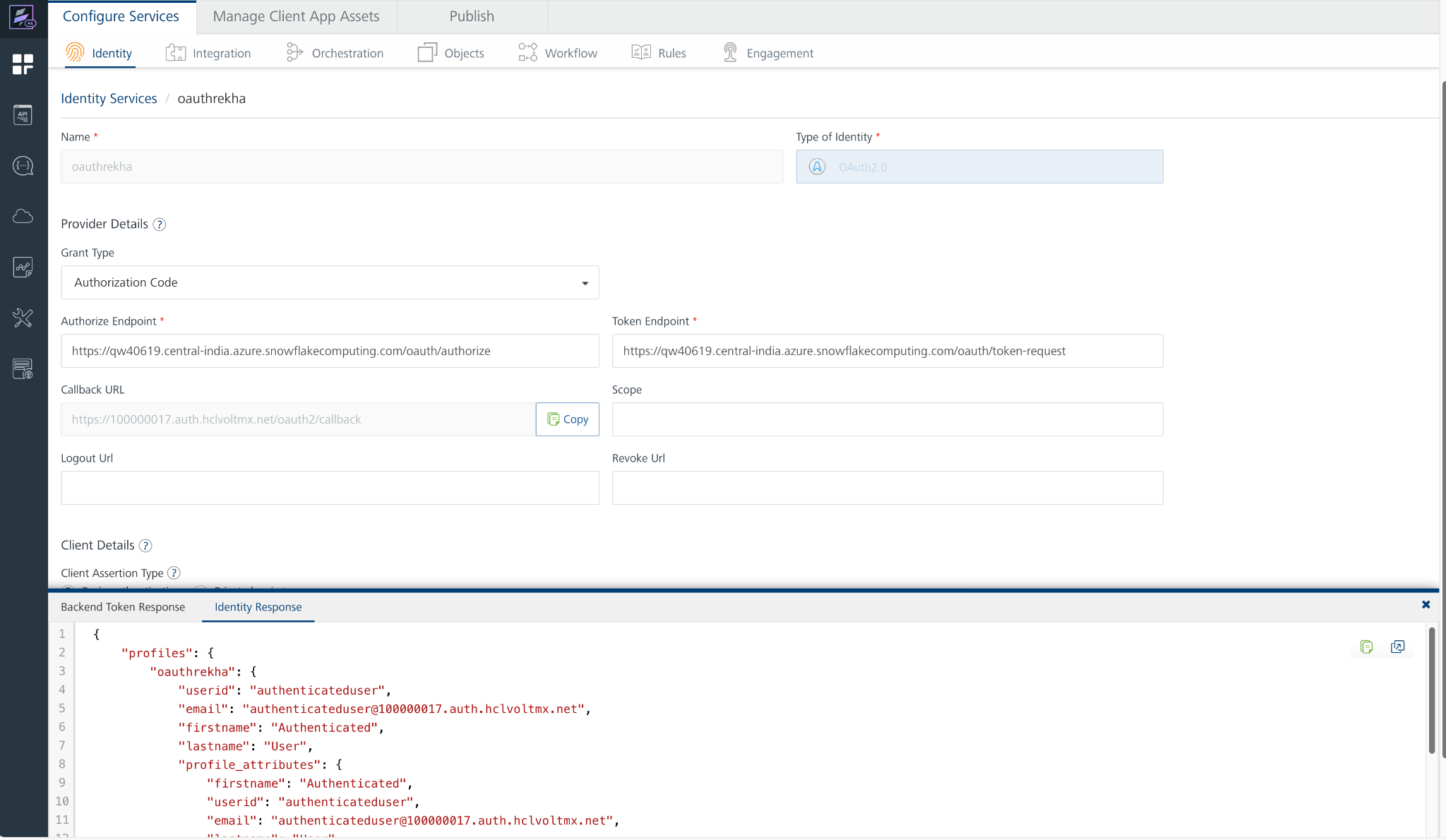
6. To get successful test login of identity service, Default role should be public in the snowflake account; the setup is as shown in below screenshot.





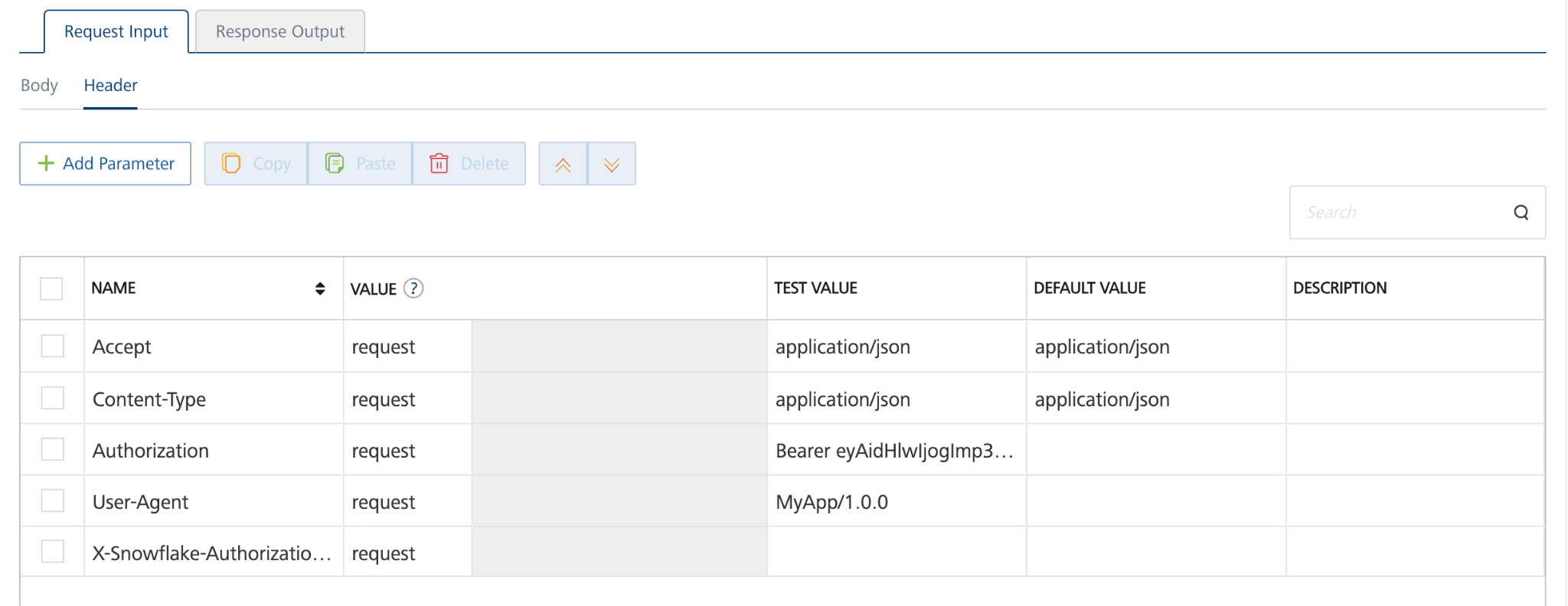
with all respective field details.

6. Reference screenshot of identity service in voltmx foundry



6. And then we will get Bearer token as well after successful test login.

7. In this response we will get claims token, that we can use as Authorization value in the header. Before value we need to add Bearer like below screenshot.



**D.Importing the adapter**

1. Sign in to the [HCL Foundry](https://manage.hclvoltmx.com/) Console.
2. From the left navigation menu, select **API Management**.

In **API Management**, select **Custom Data Adapters**.

A screenshot of a computer

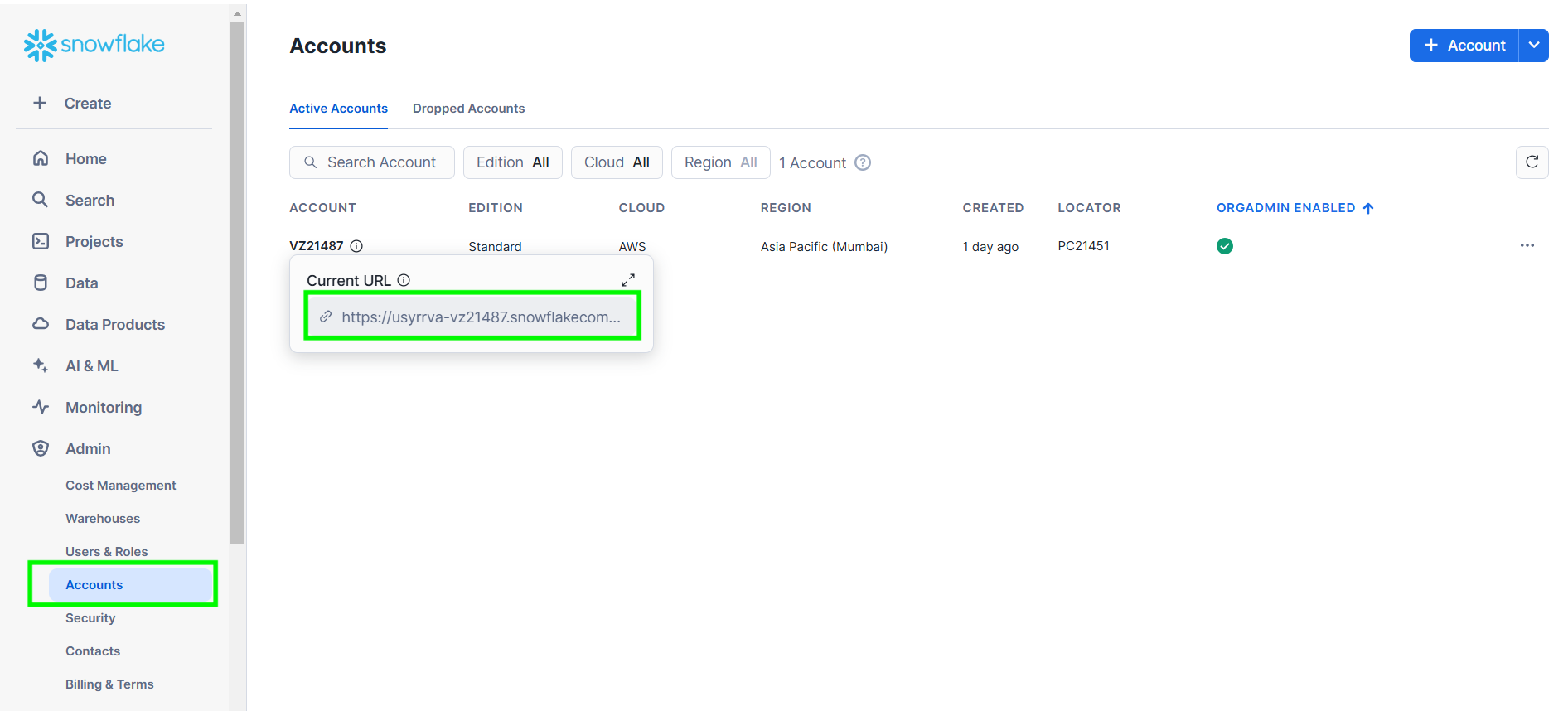
Description automatically generated

1. Click **IMPORT** to import a custom data adapter.  
   A blue and white box with text

   Description automatically generated
2. On the Import Data Adapter dialog box, click browse to import.  
   A screen shot of a computer

   Description automatically generated
3. Select data adapter zip file and click **IMPORT**.
4. After importing the Snowflake Cortex Search data adapter, Volt Foundry opens a window that shows the metadata of the data adapter.
5. After you import the data adapter, you can view it on the Custom Data Adapters page and use it to create services on VoltMX Foundry.
6. Replace the server url under connection parametres with the current url from snowflake website.  
   A screenshot of a computer

   Description automatically generated



7. From the list of operations, select an operation that you want to test.

A screenshot of a computer

Description automatically generated

8. On the **Request Input** tab, enter a **TEST VALUE** or a **DEFAULT VALUE** for the parameters.

A screenshot of a computer

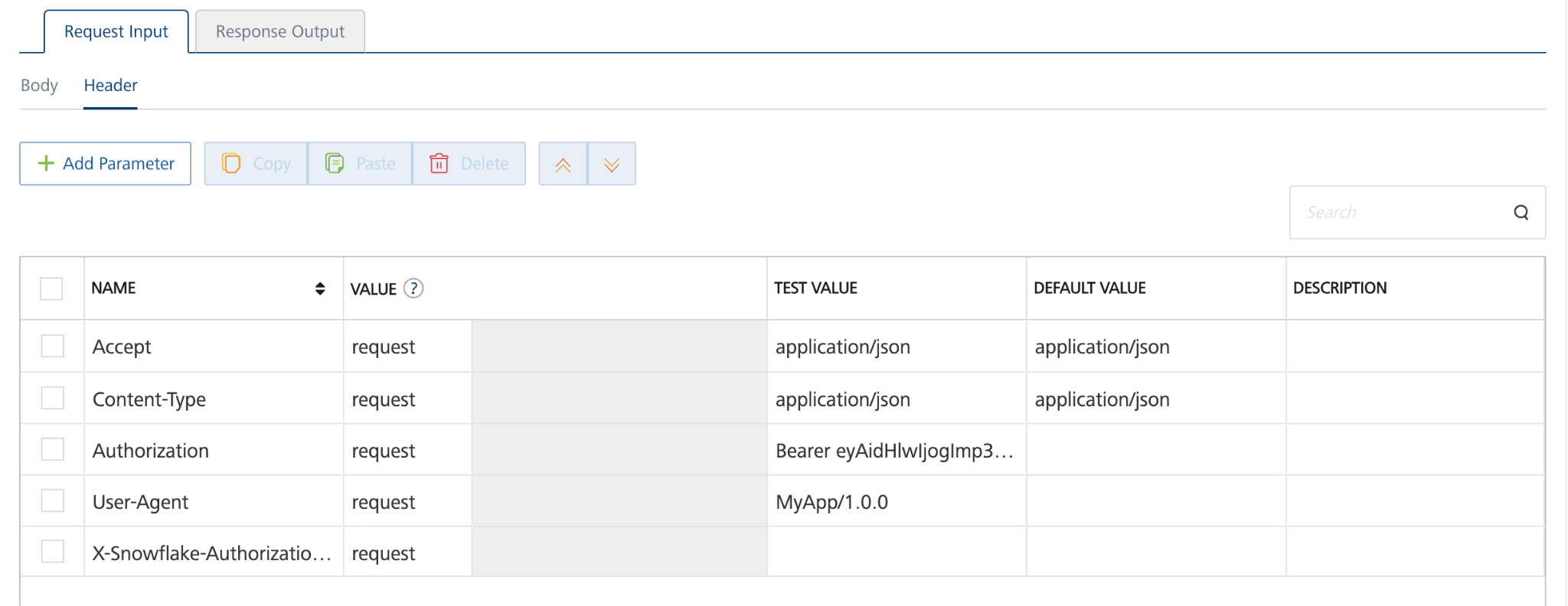
Description automatically generated

**Note:** To fetch the response onVolt MX Foundry, clear the **Enable pass-through** check box on the **Response Output** tab.

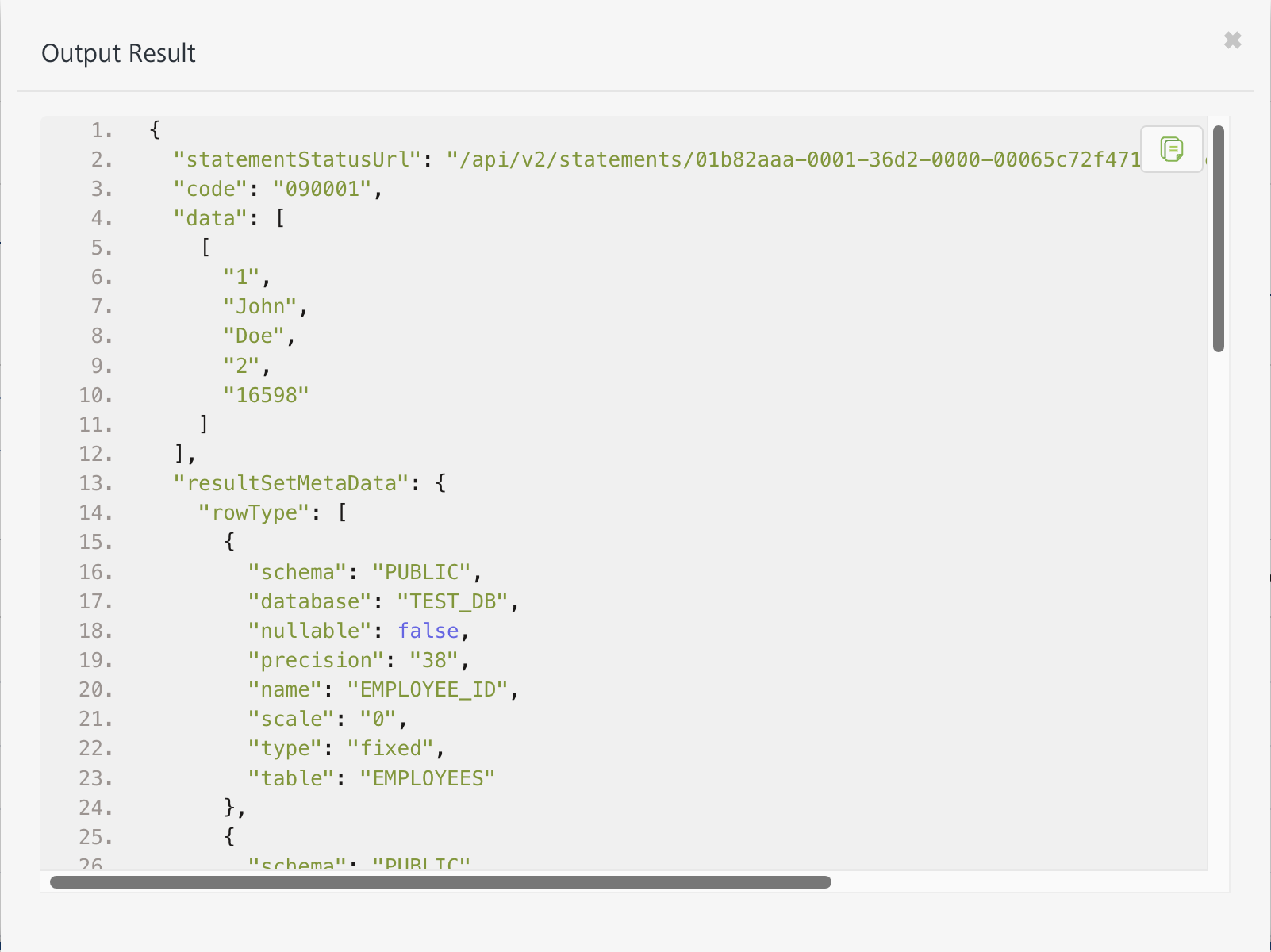
A screenshot of a computer

Description automatically generated

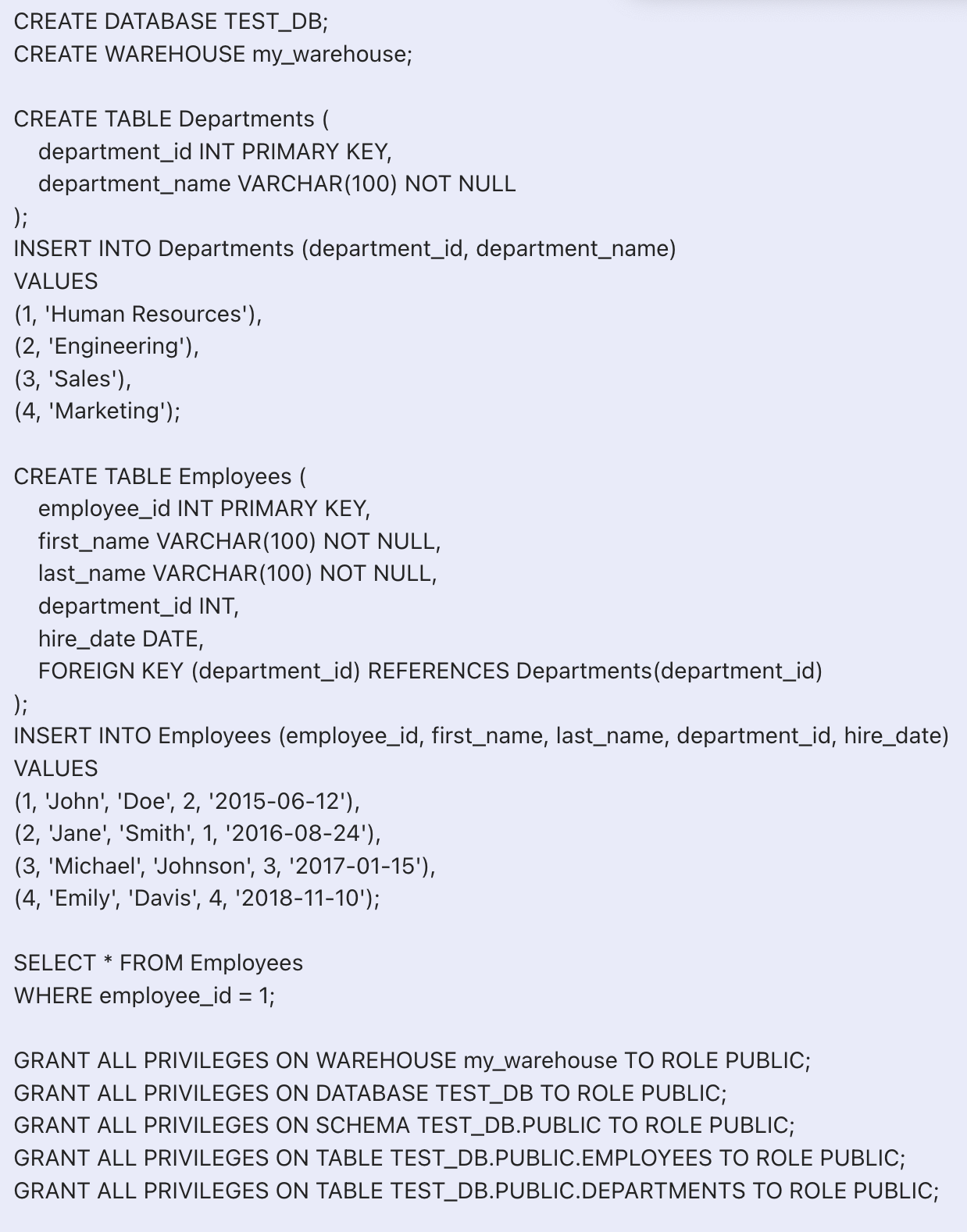
**Request Header:**

**Authorization:** Authorization:The value is the Bearer token that we got from identity service. 

9. Select a run-time environment and click **Save and Fetch Response** to get a response based on your input.

**Note:**

1. To test the operations provided in this link <https://docs.snowflake.com/en/user-guide/snowflake-cortex/cortex-search/cortex-search-overview> I have created my own sample database with sample tables and cortex search service in snowflake trial account. Adding reference screenshot for sample database, tables, respective queries to grant privileges.



**Sample code to create cortex service**

CREATE OR REPLACE CORTEX SEARCH SERVICE description\_search\_service2

ON CATEGORY

ATTRIBUTES CATEGORY

WAREHOUSE = MY\_WAREHOUSE

TARGET\_LAG = '60 second'

COMMENT = 'Search service for descriptions in the data store'

AS (

SELECT

ID,

NAME,

DESCRIPTION,

CATEGORY

FROM CORTEXVALIDATION.CORTEX\_SEARCH.DATA\_STORE

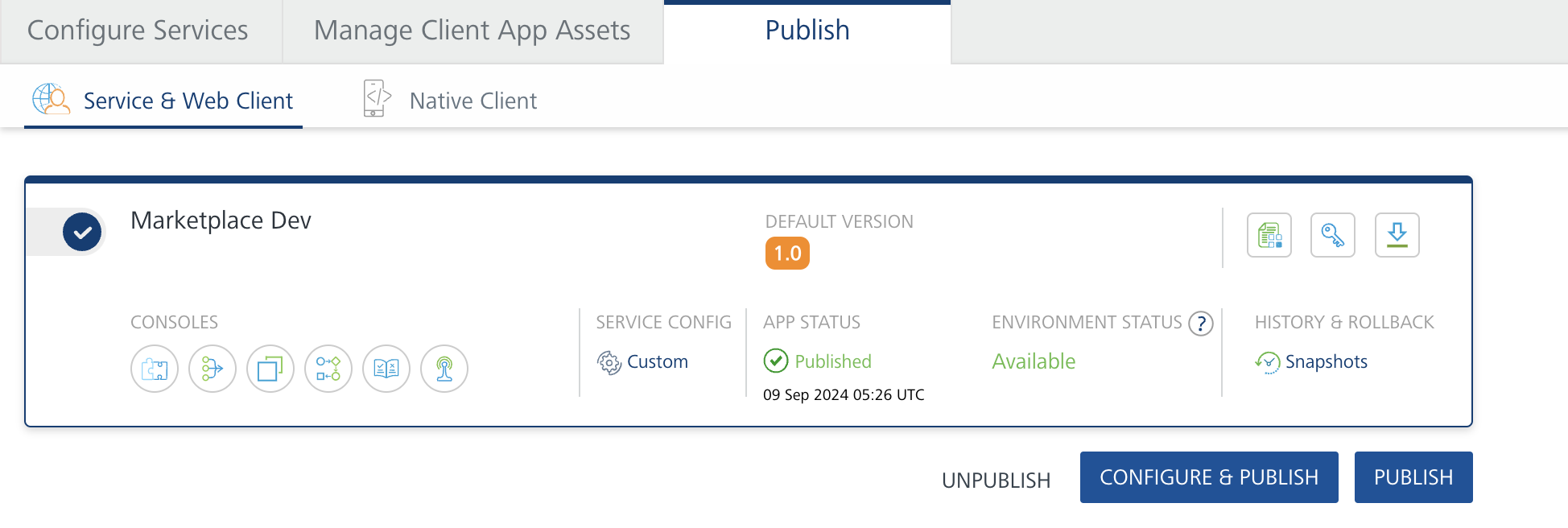
# {DATABASE}.{SCHEMA}.{TABLE}

);

1. To get successful responses in the foundry, role should be public only in the snowflake account.

## **Publishing your application**

If you want to use the services in client applications, you need to publish an app to a run-time environment. You can create the service (as described above) in an application or import the service into an application and publish the application.



## **References**

**Endpoint Documentation**

<https://docs.snowflake.com/en/user-guide/snowflake-cortex/cortex-search/query-cortex-search-service>

|  |  |
| --- | --- |
| **Endpoint** | **Description** |
| /api/v2/databases/ | Use this to retrieve a list of available databases.. |
| /api/v2/{database}/schemas | Use this to retrieve a list of schemas within a specific database. |
| /api/v2/{database}/schemas/{schema}/cortex-search-services/{service\_name}:query | Submit a search query to a specific service within a database and schema. |

**1.POST/api/v2/** **databases/{database}/schemas/{schema}/cortex-search-services/{service\_name}:query**:

including the request syntax, query parameters, request headers, request body requirements, and response codes as per your details:

|  |  |
| --- | --- |
| **Aspect** | **Details** |
| **Endpoint** | /api/v2/ databases/{database}/schemas/{schema}/cortex-search-services/{service\_name}:query |
| **Method** | POST |
| **Description** | Executes a query against a specific search service within a database and schema. |
| **Request Syntax** | POST /api/v2/ databases/{database}/schemas/{schema}/cortex-search-services/{service\_name}:query |

**Query Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Description** | **Required** |
| database | String | Database in which the service resides | Required |
| Schema | String | Schema in which the service resides. | Required |
| service\_name | String | Name of the service | Required |
| query | String | Your search query, to search over the text column in the service. | Required |
| columns | Array | A comma-separated list of columns to return for each relevant result in the response. These columns must be included in the source query for the service. | Required |
| filter | Object | filter object for filtering results based on data in the ATTRIBUTES columns. See [Filter syntax](https://docs.snowflake.com/en/user-guide/snowflake-cortex/cortex-search/query-cortex-search-service#filter-syntax). | Optional |
| limit | Number | Maximum number of results to return in response.  Maximum accepted value is 1000.  Default value is 10. | Optional |

**Request Headers**

The request must include the standard headers for Snowflake CORTEX SEARCH API operations:

|  |  |
| --- | --- |
| **Header** | **Value** |
| Authorization | Bearer <token> |
| Content-Type | application/json |
| Accept | application/json |
| User-Agent | <app\_name/version> |

**Request Body**

The request body must contain a JSON object that specifies the CORTEX SEARCH statement(s) to execute. Refer to the [Body of the POST request to /api/v2/statements/](https://docs.snowflake.com/en/developer-guide/sql-api/reference#label-sql-api-reference-post-statements) for more details

.

**Response Codes**

|  |  |
| --- | --- |
| **Code** | **Description** |
| 200 | |  | | --- | | The query was executed successfully. The response body contains the results of the search. |  |  | | --- | |  | |

**2. GET /api/v2/databases**

endpoint, including the request syntax, path parameters, query parameters, headers, and response codes.

|  |  |
| --- | --- |
| **Aspect** | **Details** |
| **Endpoint** | /api/v2/databases |
| **Method** | GET |
| **Description** | Retrieves a list of available databases. |
| **Request Syntax** | GET /api/v2/databases |

**Path Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Description** | **Required** |
| N/A | N/A | N/A | N/A |

**Query Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Description** | **Required** |
| N/A | N/A | N/A | N/A |

**Request Headers**

|  |  |  |
| --- | --- | --- |
| **Header** | **Value** | **Description** |
| Authorization | Bearer <token> | Access token for authorization. |
| Accept | application/json | Specifies that JSON is expected in response. |
| User-Agent | <app\_name/version> | Identifies the client application. |

**Response Codes**

|  |  |
| --- | --- |
| **Code** | **Description** |
| 200 | Successfully retrieved the list of databases. |

**3. POST /api/v2/{database}/schemas**

endpoint, covering request syntax, path parameters, query parameters, headers, and response codes:

|  |  |
| --- | --- |
| **Aspect** | **Details** |
| **Endpoint** | /api/v2/{database}/schemas |
| **Method** | GET |
| **Description** | Retrieves a list of schemas within a specific database. |
| **Request Syntax** | GET /api/v2/{database}/schemas |

**Path Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Description** | **Required** |
| database | String | |  | | --- | | The name of the database containing the schemas. |  |  | | --- | |  | | Required |

**Query Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Type** | **Description** | **Required** |
| |  | | --- | | N/A |  |  | | --- | |  | | |  | | --- | | N/A |  |  | | --- | |  | | |  | | --- | | N/A |  |  | | --- | |  | | |  | | --- | | N/A |  |  | | --- | |  | |

**Request Headers**

|  |  |  |
| --- | --- | --- |
| **Header** | **Value** | **Description** |
| Authorization | Bearer <token> | Access token for authorization. |
| Accept | application/json | Specifies that JSON is expected in response. |
| User-Agent | <app\_name/version> | Identifies the client application. |

**Response Codes**

|  |  |
| --- | --- |
| **Code** | **Description** |
| 200 | |  | | --- | | Successfully retrieved the list of schemas. |  |  | | --- | |  | |

**3. Revision History**

App version 1.0.0:

1. **Known Issues**

-

1. **Limitations**

-