

Snowflake OAUTH Builtin and External OKTA and AzureAD

- Built-In Snowflake OAuth
- Snowflake Side Configuration
 - Step 1: Snowflake New Setup
 - Create Security Integration
 - Collect newly created Security Integration complete details
 - Collect newly created Security Integration Client ID and Secret
 - Delete the above Config (Optional)
 - Step 2: Alation Configuration for Builtin OAUTH
 - Step 3: After Authentication Test in Builtin Oauth
- OKTA External:
 - Reference <https://docs.snowflake.com/en/user-guide/oauth-okta>
 - Stage1.1 OKTA Auth Client:
 - Create application OIDC Sign in method
 - Choose Web application Type
 - Set Auth code and refresh token Grant type
 - Set redirect URI to Alation Callback
 - Set For internal testing -allow everyone
 - Change issuer from Dynamic to Static in Signon tab - openid
 - Click save
 - Full config sample
 - Stage1.2 OKTA Auth Server:
 - Step1: Create Auth Server in OKTA
 - In Input for Audience give our Snowflake Server Address and Save
 - Step2: Change Issuer from Dynamic to Static and save
 - Step3: On scopes tab> Add scope > SESSION:ROLE-ANY
 - Step4: Add policy, can create a policy to allow all users' use of this scope.
 - Map the Client Application created for snowflake under this policy
 - Now Add rule and create rule
 - Sample :
 - Collect Info
 - Issuer Okta URL <https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKFV85d7>
 - Click on Metadata URI
 - link <https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKFV85d7/.well-known/oauth-authorization-server>
 - Used in Snowflake
 - Used in Alation
 - Stage2: In Snowflake: set up External OAuth with Okta;
 - Stage3: In Alation: enable and configure the OAuth for your Snowflake data source.
 - TESTING OKTA:
 - Logs at OKTA Server Side for Success-full Login
 - Azure AD_Snowflake_Alation:
 - Prerequisite Step: Determine the OAuth Flow in Azure AD
 - Stage1.1 Create App registration for AUTH SERVER
 - Set Application ID URI (We will use <https://alation.com/> as Application ID URI)
 - Now click Add scope
 - Provide:
 - Stage1.2 Create App registration for AUTH Client
 - CREATE AUTH CLIENT
 - Generate Client Secret and Copy
 - Add API Permissions to Auth Server created
 - Add permission
 - Switch to My API's on popup
 - Double click Auth Server Created
 - Press Delegated Permissions ,select and add Permission
 - Collect Azure AD Information for Snowflake
 - Token endpoint v2<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/token>
 - jwks_uri Locate Locate the "jwks_uri" here
 - Next Capture Entity ID:
 - Stage2: Snowflake Create a Security Integration Azure type
 - Just for understand what to put in Alation for User Field Name (Like EMail or Username or UPN)
 - Sample Users Mapping
 - In Azure AD:
 - In Snowflake: We use Login Name
 - Stage3 Configure Alation
 - Test Azure Ad Flow
 - Error 1: No reply Address
 - Solution: After Adding Client Redirect URI on Auth Client this got fixed
 - Error2 Callback/ vs callback
 - Tracing Success Routing Azure:
 - Step1 On Opening connection -on compose Redirect OAUTH : As soon we open Compose with Respective DataSource
 - Step2: On Add SSO User
 - Step 2.1 SSO page1
 - Here we can see the username

- Step2.2 SSO page2
- Step2.3 Alation Gets Code for OAUTH call
- In summary we can see UPN from OKTA to Azure
- Step 3:Compose uses code to get token
- Step5: Re Connect if broken
- retry
- Flow Chart :
- Code and token
- Snowflake call
- Now lets run the Query again
- RAN NEW QUERY:query_run
- Results:execution_result_data
- Try Switching Roles with Builtin Oauth
 - Switch using JDBC URL
 - Default Role here ALATION
 - Adding new role
 - ADD role=role_name in JDBC url
 - Validate User is mapped with the role
 - Test
 - Allow OAUTH
 - Validate current role
 - Try Switching Roles with External Oauth
 - Before Connection
 - Tried creating a JDBC URL with role parameter
 - After successful authentication

Built-In Snowflake OAuth

References:

- [CREATE SECURITY INTEGRATION \(Snowflake OAuth\) | Snowflake Documentation](#)
- [Configure Snowflake OAuth for Custom Clients | Snowflake Documentation](#)
- [Alation Help Center: Snowflake SSO](#)

Overview of the steps to perform:

The configuration workflow has 2 Stages:

Stage 1: Configure Snowflake with Alation Instance details

- Stage 1 Input: Run the **security integration register** query with the Alation endpoint details
- Stage 1 Output: Client ID , Client Secret , Authorization endpoint , Token Endpoint

Stage 2: Configure Alation Instance with Snowflake details

- Stage 2 Input: Use Stage 1 output in Alation's data source Compose connection setting

Snowflake Side Configuration

Step 1: Snowflake New Setup

Create Security Integration

You can run a query similar to the one shown below. Following adjustments are required:

- Change name from `SURAJ_SNOWFLAKE_BUILTIN_OAUTH` to yours. Avoid special characters.
- Change `OAUTH_REDIRECT_URI` to your own `https://ALATION_INSTANCE_FQDN/api/datasource_auth/oauth/callback`

```

1 CREATE SECURITY INTEGRATION
2   "SURAJ_SNOWFLAKE_BUILTIN_OAUTH"
3   TYPE = OAUTH
4   ENABLED = TRUE
5   OAUTH_CLIENT = CUSTOM
6   OAUTH_CLIENT_TYPE = 'CONFIDENTIAL'
7   OAUTH_REDIRECT_URI = 'https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback'
8   OAUTH_REFRESH_TOKEN_VALIDITY = 776000
9   OAUTH_ISSUE_REFRESH_TOKENS = TRUE ;

```

```

3   CREATE SECURITY INTEGRATION
4     "SURAJ_SNOWFLAKE_BUILTIN_OAUTH"
5       TYPE = OAUTH
6       ENABLED = TRUE
7       OAUTH_CLIENT = CUSTOM
8       OAUTH_CLIENT_TYPE = 'CONFIDENTIAL'
9       OAUTH_REDIRECT_URI = 'https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback'
10      OAUTH_REFRESH_TOKEN_VALIDITY = 7776000
11      OAUTH_ISSUE_REFRESH_TOKENS = TRUE ;
12
13
14
15 CREATE SECURITY INTEGRATION

```

Objects Editor Results Chart

	status
1	Integration SURAJ_SNOWFLAKE_BUILTIN_OAUTH successfully created.

Collect newly created Security Integration complete details

Execute the following query to get the details about security integration we created earlier on:

```
1 DESC SECURITY INTEGRATION "SURAJ_SNOWFLAKE_BUILTIN_OAUTH";
```

73
74 | DESC SECURITY INTEGRATION "SURAJ_SNOWFLAKE_BUILTIN_OAUTH";
75
76
77

Objects Editor Results Chart

property	property_type	property_value	...	property_default
1 ENABLED	Boolean	true		false
2 OAUTH_REDIRECT_URI	String	https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback		
3 OAUTH_CLIENT_TYPE	String	CONFIDENTIAL		CONFIDENTIAL
4 OAUTH_ISSUE_REFRESH_TOKENS	Boolean	true		true
5 OAUTH_REFRESH_TOKEN_VALIDITY	Integer	7776000		7776000
6 OAUTH_ENFORCE_PCKE	Boolean	false		false
7 OAUTH_USE_SECONDARY_ROLES	String	NONE		NONE
8 OAUTH_CLIENT_ID	String	EfVgJaZDQzdnZU7HbmGe7PRqjFw=		
9 OAUTH_AUTHORIZATION_ENDPOINT	String	https://esa45191.us-east-1.snowflakecomputing.com/oauth/authorize		
10 OAUTH_TOKEN_ENDPOINT	String	https://esa45191.us-east-1.snowflakecomputing.com/oauth/token-request		
11 OAUTH_ALLOWED_AUTHORIZATION_ENDPOINTS	List	https://esa45191.us-east-1.snowflakecomputing.com/oauth/authorize,https://alati...	...	
12 OAUTH_ALLOWED_TOKEN_ENDPOINTS	List	https://esa45191.us-east-1.snowflakecomputing.com/oauth/token-request,https://...	...	
13 PREAUTHORIZED_ROLES_LIST	List		...	
14 BLOCKED_ROLES_LIST	List	ACCOUNTADMIN,SECURITYADMIN	...	
15 OAUTH_ALLOW_NON_TLS_REDIRECT_URI	Boolean	false		false
16 OAUTH_CLIENT_RSA_PUBLIC_KEY_Fingerprint	String			

In the result set you will find entries for:

- **Authorization Endpoint** as `OAUTH_AUTHORIZATION_ENDPOINT`
- **Token Endpoint** as `OAUTH_TOKEN_ENDPOINT`

Collect newly created Security Integration Client ID and Secret

Retrieve the Client ID and Secret by running this query (replace the name of the security integration):

```
1 SELECT SYSTEM$SHOW_OAUTH_CLIENT_SECRETS('<SECURITY_INTEGRATION_NAME>');
2 -- in example:
3 SELECT SYSTEM$SHOW_OAUTH_CLIENT_SECRETS('SURAJ_SNOWFLAKE_BUILTIN_OAUTH');
```

Here `OAUTH_CLIENT_SECRET_2` can be ignored:

Security Integration Values (example output):

```
1 {"OAUTH_CLIENT_SECRET_2":"riWm/G1qCo6pUQdH8cpJ4IwJTbp+QCA81wtT0kPxogQ=",
2 "OAUTH_CLIENT_SECRET":"ObBSF783PI9j3D6WK7xE+h3wPX12XcfChphLxRgf3w=",
3 "OAUTH_CLIENT_ID":"EfVgJaZDQzdnZU7HbmGe7PRqjFw="}
```

75
76
77

Objects Editor Results Chart

SYSTEM\$SHOW_OAUTH_CLIENT_SECRETS('SURAJ_SNOWFLAKE_BUILTIN_OAUTH')

1 {"OAUTH_CLIENT_SECRET_2":"riWm/G1qCo6pUQdH8cpJ4IwJTbp+QCA81wtT0kPxogQ=","OAUTH_CLIENT_SECRET":"ObBSF783PI9j3D6WK7xE+h3wPX12XcfChphLxRgf3w=","OAUTH_CLIENT_ID":"EfVgJaZDQzdnZU7HbmGe7PRqjFw="}

Delete the above Config (Optional)

⚠ Only do this if you were testing the setup and want to clear down everything!

In case you want to drop the configuration (because you are just testing this setup), you can use this:

```
1 DROP SECURITY INTEGRATION IF EXISTS "<SECURITY_INTEGRATION_NAME>" CASCADE;
```

Step 2: Alation Configuration for Builtin OAUTH

Under **Data Source > Settings > Compose**. Toggle **Enable in Compose** (if it is not already switched on) scroll down and edit it as follows:

Compose Connections

Title	URI
Default Connection	snowflake://<hostname>:<port>/?warehouse=<warehouse_name>&db=<db_name>

Compose Connection Sharing

Shared connections across tabs
More performant, but may prevent users from running multiple queries in parallel, even from separate Compose tabs

Separate connection per tab
Enables users to run multiple queries simultaneously, at the possible expense of system performance. Also: temporary tables created in one tab may not be "visible" to queries in another.

OAuth Connection

Enable OAuth 2.0 in Compose

Client ID Client Secret

Request Refresh Tokens Enable PKCE

Authorization Endpoint

Token Endpoint

Default Scope

Username Field/Claim JWT

Access token parameter name

OAuth enablers

Enable OAuth 2.0 for all Compose Connections

- Query Forms
- Data Upload
- Dynamic Sampling/Profiling

Save

1. Add `authenticator=oauth` to the connection string.
2. Tick **Enable OAuth 2.0 in Compose** checkbox
3. Provide **Client ID** and **Client Secret**.
4. Tick **Request Refresh Tokens** checkbox.
5. Provide **Authorization Endpoint**.
6. Provide **Token endpoint**.
7. **Default Scope**: Leave Blank
8. **Refresh Scope**: Insert `refresh_token`
9. **Username Field**: Insert `username` . (here we specify the username since the same is configured at Snowflake side for user identification)
10. **Access token parameter**: Insert `token`
11. **OAuth Enablers**: Insert `authenticator=oauth`

Compose Connections

Title	URI
Default Connection	snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth

Compose Connection Sharing

Shared connections across tabs
More performant, but may prevent users from running multiple queries in parallel, even from separate Compose tabs

Separate connection per tab
Enables users to run multiple queries simultaneously, at the possible expense of system performance. Also: temporary tables created in one tab may not be "visible" to queries in another.

OAuth Connection

Enable OAuth 2.0 in Compose

Client ID: EFVgJaZDQzdnZU7hb Client Secret

Request Refresh Tokens Enable PKCE

https://esa45191.us-east-1.snowflakecomputing.com/oauth/authorize https://esa45191.us-east-1.snowflakecomputing.com/oauth/token-request

Default Scope

refresh_token

username JWT

token

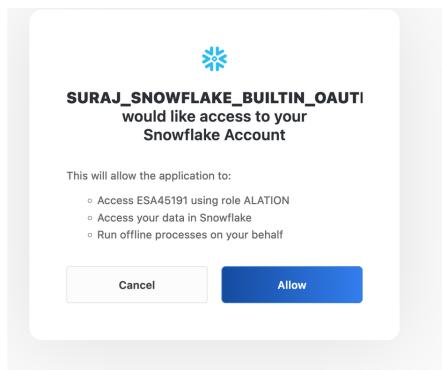
authenticator=oauth

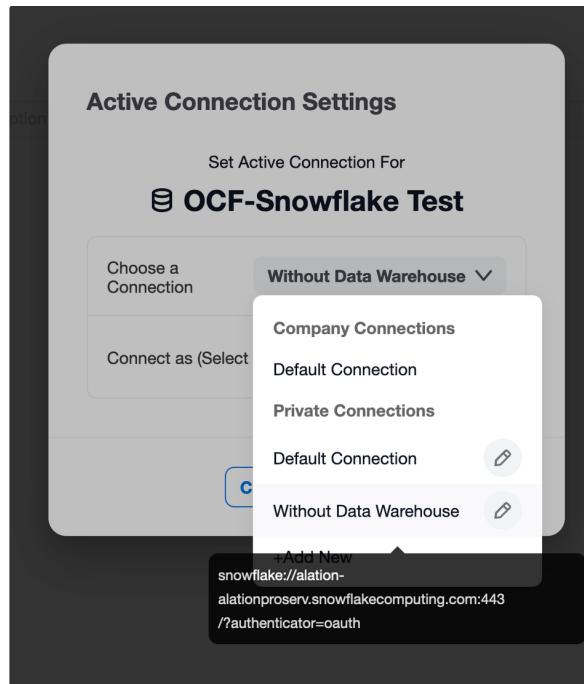
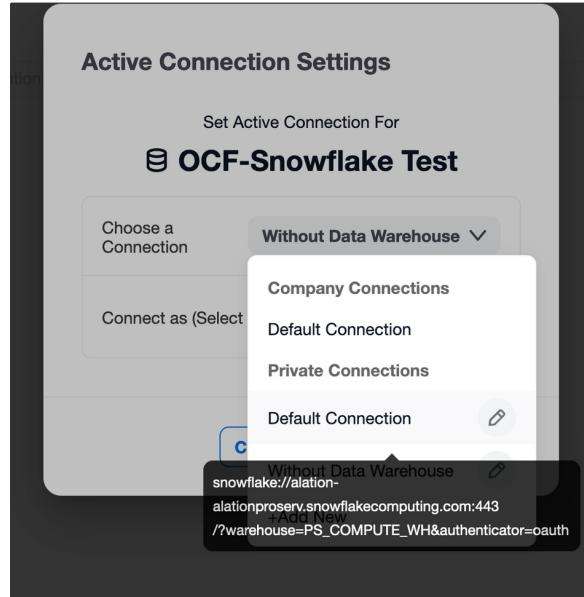
Enable OAuth 2.0 for all Compose Connections and also:

- Query Forms
- Data Upload
- Dynamic Sampling/Profiling

Save

Step 3: After Authentication Test in Builtin Oauth





OKTA External:

Reference <https://docs.snowflake.com/en/user-guide/oauth-okta>

Configuration Work Flow in 3 Stages:

Stage1.1 Configure OKTA Client with Alation Instance callback details

Stage1.1 Input- Alation callback url

Stage1.1 Output- client id and secret

Stage1.2 Configure OKTA Auth Server with Snowflake required details

Stage1.2 Input- Give Snowflake URL as Audience and Snowflake role in scope

Stage1.2 Output-Endpoints

Stage2: Configure Snowflake with OKTA Auth Server details

Stage2 Input- run integration register query with OKTA endpoint details

Stage3: Configure Alation Instance with OKTA Client ID and OKTA Auth server endpoints Details details

Stage3Input: Use Stage1 input in Alation's data source compose connection setting

- In Snowflake: if this has not been done yet, set up External **OAuth** with Okta;
- In Alation: enable and configure the **OAuth** for your Snowflake data source.

Stage1.1 OKTA Auth Client:

1. When creating the client application in Okta, use Web as the Application type.

2. In the client application settings:

Under General > General Settings > Application > Allowed grant types, ensure that the Authorization Code and Refresh Token checkboxes are selected;

(Optional) Under General > General Settings > User Consent > User consent, select Require consent;

Under General > General Settings > LOGIN > Login redirect URIs, add https://<alation.company.com>/api/datasource_auth/oauth/callback/

Ensure the URI ends with “/”

HTTP can be specified without having to change any TLS settings

Create a new app integration

Sign-in method

Learn More [Open](#)

OIDC - OpenID Connect
Token-based OAuth 2.0 authentication for Single Sign-On (SSO) through API endpoints. Recommended if you intend to build a custom app integration with the Okta Sign-In Widget.

SAML 2.0
XML-based open standard for SSO. Use if the Identity Provider for your application only supports SAML.

Create application OIDC Sign in method

Application type

Web Application
Server-side applications where authentication and tokens are handled on the server (for example, Go, Java, ASP.NET, Node.js, PHP)

Single-Page Application
Single-page web applications that run in the browser where the client receives tokens (for example, Javascript, Angular, React, Vue)

Native Application
Desktop or mobile applications that run natively on a device and redirect users to a non-HTTP callback (for example, iOS, Android, React Native)

Cancel **Next**

Choose Web application Type

New Web App Integration

General Settings

App integration name

Logo (Optional)

Grant type

Learn More [Open](#)

Client acting on behalf of itself
 Client Credentials
 Client acting on behalf of a user
 Authorization Code
 Refresh Token
 Implicit (hybrid)

Sign-in redirect URIs

Allow wildcard * in sign-in URI redirect.
 [X](#)

Set Auth code and refresh token Grant type

Set redirect URI to Alation Callback

https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback/

Sign-in redirect URIs

Okta sends the authentication response and ID token for the user's sign-in request to these URIs

Allow wildcard * in sign-in URI redirect.

inbox-r53.alationproserv.com/api/datasource_auth/oauth/callback/ X

[Learn More](#) + Add URI

Set For internal testing -allow everyone

Assignments

Controlled access

Select whether to assign the app integration to everyone in your org, only selected group(s), or skip assignment until after app creation.

Allow everyone in your organization to access

Limit access to selected groups

Skip group assignment for now

Enable immediate access (Recommended)

Recommended if you want to grant access to everyone without pre-assigning your app to users and use Okta only for authentication.

Enable immediate access with **Federation Broker Mode**

i To ensure optimal app performance at scale, Okta End User Dashboard and provisioning features are disabled. Learn more about [Federation Broker Mode](#).

Save Cancel

Change issuer from Dynamic to Static in Signon tab - openid

Click save

Full config sample

The screenshot shows the Okta Application Settings page for an application named "snowflake_okta_ation_compose". The application is marked as "Active". The "General" tab is selected, showing the "Collect Client Info" section. Under "Client Credentials", the Client ID is "Ooa8jq9eoczePGg1g5d7" and the Client authentication method is set to "Client secret". There is also a checkbox for "Require PKCE as additional verification". Under "CLIENT SECRETS", a new secret was generated on Mar 1, 2023, with the secret value "Ee9T-LftzgpbCCvyED52h8P6cgCsWiRi8ga_wsZ".

- Client ID Ooa8jq9eoczePGg1g5d7
- Client Secret Ee9T-LftzgpbCCvyED52h8P6cgCsWiRi8ga_wsZ

Stage1.2 OKTA Auth Server:

In the Okta Authorization Server for Snowflake settings:

1. Ensure that the audience URL does not have a trailing slash,

working example: https://alationi_test.us-east-1.snowflakecomputing.com

2. Consider adding SESSION:ROLE-ANY as a Scope to avoid adding a Scope entry for each role. You can create a policy to allow all users' use of this scope. This will allow users to connect to Snowflake as any role to which they have access in Snowflake and will also enable role switching in Compose with the USE ROLE statement. Otherwise, create Scopes linked to specific Snowflake roles and add policies limiting use of these roles to specific users.

input:Our Snowflake Server:<https://esa45191.us-east-1.snowflakecomputing.com>

The screenshot shows the Snowflake Server General Tab. At the top, there's a navigation bar with links for Databases, Shares, Marketplace, Warehouses, Worksheets (which is the active tab), History, and Account. Below the navigation is a message: "Our updated Worksheets experience now includes autocomplete, an improved object browser, and a new i". There are "Go to Worksheet" and "Remind me later" buttons. The main area shows a "New Worksheet" button and a search bar. A sidebar on the left lists "Find database objects" and "Starting with..." with "AAV2" selected. The bottom of the screen displays the "Snowflake Server General Tab" title.

The screenshot shows the Okta Admin Console. The left sidebar has sections for Dashboard, Directory, Customizations, Applications, Security (which is highlighted with a green oval), HealthInsight, Authentication, Multifactor, Identity Providers, Delegated Authentication, Networks, and LOGI. Under LOGI, there are Sign-in and API sections, with the API section highlighted with a green oval. The main content area is titled "Step1: Create Auth Server in OKTA" and "APPLICATION". It shows an "Add Authorization Server" button. The "Name" field is set to "default", "Audience" to "api://default", and "Issuer URI" to "https://dev-66731772-admin.okta.com/oauth2/as". The "Issuer" dropdown is set to "Dynamic (based on request domain)". The URL in the browser is <https://dev-66731772-admin.okta.com/admin/oauth2/as>. The title of the window is "okta".

In Input for Audience give our Snowflake Server Address and Save

Our Snowflake Server:<https://esa45191.us-east-1.snowflakecomputing.com>

The screenshot shows the "Add Authorization Server" dialog box. It has fields for "Name" (External Snowflake Auth Server), "Audience" (<https://esa45191.us-east-1.snowflakecomputing.com>), and "Description" (External Snowflake Auth Server). There are "Save" and "Cancel" buttons at the bottom. The URL in the browser is <https://dev-66731772-admin.okta.com/admin/oauth2/as>.

Step2:Change Issuer from Dynamic to Static and save

From Dynamic issuer

External Snowflake Auth Server

The screenshot shows the "External Snowflake Auth Server" settings page. The "Settings" tab is selected. The "Name" is "External Snowflake Auth Server", "Audience" is "<https://esa45191.us-east-1.snowflakecomputing.com>", "Description" is "External Snowflake Auth Server", "Issuer" is "Dynamic (based on request domain)", "Metadata URI" is "<https://dev-66731772.okta.com/oauth2/aus8@jrwUVSKV85d77well-known/oauth-authorization-server>", and "Signing Key Rotation" is "Automatic". There are "Edit" and "Save" buttons at the top right.

To Static OKTA Issuer URL

Issuer	Okta URL (https://dev-66731772.okta.com/oauth2/ ...)
Metadata URI	Dynamic (based on request domain)
Signing Key Rotation	Automatic

External Snowflake Auth Server

Active ▾

Settings Scopes Claims Access Policies Token Preview

Settings

Edit

Name	External Snowflake Auth Server
Audience	https://esa45191.us-east-1.snowflakecomputing.com
Description	External Snowflake Auth Server
Issuer	Okta URL (https://dev-66731772.okta.com/oauth2/aus8jqjrwUVSKfV85d7)
Metadata URI	https://dev-66731772.okta.com/oauth2/aus8jqjrwUVSKfV85d7/well-known/auth-authorization-server
Signing Key Rotation	Automatic
Last Rotation	2 Mar 2023

Step3: On scopes tab> Add scope > SESSION:ROLE-ANY

External Snowflake Auth Server

Active ▾

Settings Scopes Claims Access Policies Token Preview

Add Scope

Add Scope

Name	SESSION:ROLE-ANY
Display phrase	Use any Snowflake role
Description	Accept any Snowflake role
User consent	<input checked="" type="checkbox"/> Require user consent for this scope <input checked="" type="checkbox"/> Block services from requesting this scope
Default scope	<input type="checkbox"/> Set as a default scope
Metadata	<input type="checkbox"/> Include in public metadata

Create Cancel

External Snowflake Auth Server

Active ▾

Settings Scopes Claims Access Policies Token Preview

Name	Display Name	Description	User Consent	Block Services	Default Scope	Metadata Publish
SESSION:ROLE-ANY	Use any Snowflake role	Accept any Snowflake role	Yes	Yes	No	No

Step4: Add policy, can create a policy to allow all users' use of this scope.

Add policy

External Snowflake Auth Server

Active ▾

Settings Scopes Claims Access Policies Token Preview

No access policies added

Policies and rules are required for clients to access t

Add Policy

Add Policy

Name	Allow all users to use this scope
Description	
Assign to	<input checked="" type="radio"/> All clients <input type="radio"/> The following clients:
Create Policy Cancel	

Map the Client Application created for snowflake under this policy

Edit Policy

Name	Allow all users to use this scope
Description	Allow all users to use this scope
Assign to	<input type="radio"/> All clients <input checked="" type="radio"/> The following clients: <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">snowflake_okta_alation_compose</div>
Update Policy Cancel	

External Snowflake Auth Server

Active		Access Policies			Token Preview	
Add New Access Policy						
1	Allow all users to use this scope	Active	Edit	Delete		
	Description	Allow all users to use this scope				
	Assigned to clients	<div style="border: 1px solid #ccc; padding: 2px;">snowflake_okta_alation_compose</div>				

Now Add rule and create rule

Add Rule

Rule Name	AuthZ server policy rule		
IF Grant type is	<input checked="" type="checkbox"/> Client acting on behalf of itself <input checked="" type="checkbox"/> Client Credentials <input checked="" type="checkbox"/> Client acting on behalf of a user <input checked="" type="checkbox"/> Authorization Code <input checked="" type="checkbox"/> Implicit (hybrid) <input checked="" type="checkbox"/> Resource Owner Password <input checked="" type="checkbox"/> SAML 2.0 Assertion <input checked="" type="checkbox"/> Device Authorization <input checked="" type="checkbox"/> Token Exchange		
AND User is	<input checked="" type="radio"/> Any user assigned the app <input type="radio"/> Assigned the app and a member of one of the following:		
AND Scopes requested	<input checked="" type="radio"/> Any scopes <input type="radio"/> The following scopes:		
THEN Use this inline hook	<div style="border: 1px solid #ccc; padding: 2px;">None (disabled)</div>		
AND Access token lifetime is	1	Hours	
AND Refresh token lifetime is		Unlimited	
but will expire if not used every	7	Days	
Create rule Cancel			

Sample :

Edit Rule

Rule Name
Generic Rule

[IF] Grant type is
 Client Credentials
 Client acting on behalf of itself

[AND] User is
 Any user assigned the app
 Assigned the app and a member of one of the following:

[AND] Scopes requested
 Any scopes
 The following scopes:

[THEN] Access token lifetime is
1 Hours

[AND] Refresh token lifetime is
100 Days

but will expire if 7 Days not used every

[Update rule](#) [Cancel](#)

Collect Info

Issuer Okta URL <https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7>

External Snowflake Auth Server

Active

Settings [Edit](#)

Name External Snowflake Auth Server

Audience <https://esa45191.us-east-1.snowflakecomputing.com>

Description External Snowflake Auth Server

Issuer [Okta URL \(https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7\)](https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7)

Metadata URI <https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/.well-known/oauth-authorization-server>

Signing Key Rotation [Edit](#) Automatic

Last Rotation 2 Mar 2023

Click on Metadata URI

link <https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/.well-known/oauth-authorization-server>

In the **Metadata** document:

1. Copy the **Metadata URI** value, open a browser tab, and paste the URL in the address bar.
2. You should see JSON text in the browser. You can work with this text in a text editor or in the browser itself.
3. Locate the "`jwks_uri`" parameter and copy its value. Its format should resemble `https://dev-11111.oktapreview.com/oauth2/ausLh9j9v9fej7NfT0h7/v1/keys`. This endpoint will be known as the <OKTA_JWS_KEY_ENDPOINT>.
4. Locate the "`token_endpoint`" parameter and copy its value. Its format should resemble `https://dev-11111.oktapreview.com/oauth2/ausLh9j9v9fej7NfT0h7/v1/token`. This endpoint will be known as the <OKTA_OAUTH_TOKEN_ENDPOINT> in the following steps.

Used in Snowflake

"`issuer`": "<https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7>",
"`jwks_uri`": "<https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/v1/keys>",

Audience from Auth Server <https://esa45191.us-east-1.snowflakecomputing.com>

Used in Alation

Stage2: In Snowflake: set up External OAuth with Okta;

In the security integration created for the Okta Authorization Server in Snowflake, run the following query:

Run below query with inputs from OKTA

```
create security integration NAME_OF_CLIENT_REGISTERING
type = external_oauth
enabled = true
external_oauth_type = okta
external_oauth_issuer = 'ISSUER FROM OKTA AUTH SERVER'
```

```

external_oauth_jws_keys_url = 'KEYS FROM OKTA AUTH SERVER'
external_oauth_audience_list = ('AUDIENCE LIST FROM OKTA AUTH SERVER')
external_oauth_token_user_mapping_claim = 'sub'
external_oauth_snowflake_user_mapping_attribute = 'login_name'
external_oauth_any_role_mode = 'ENABLE';

```

Note: to enable role switching, set the EXTERNAL_OAUTH_ANY_ROLE_MODE field to ENABLE.

```

1  create security integration SURAJ_SNOWFLAKE_OKTA_EXTERNAL_OAUTH
2    type = external_oauth
3    enabled = true
4    external_oauth_type = okta
5    external_oauth_issuer = 'https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7'
6    external_oauth_jws_keys_url = 'https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/v1/keys'
7    external_oauth_audience_list = ('https://esa45191.us-east-1.snowflakecomputing.com')
8    external_oauth_token_user_mapping_claim = 'sub'
9    external_oauth_snowflake_user_mapping_attribute = 'login_name'
10   external_oauth_any_role_mode = 'ENABLE';

42
43  create security integration SURAJ_SNOWFLAKE_OKTA_EXTERNAL_OAUTH
44    type = external_oauth
45    enabled = true
46    external_oauth_type = okta
47    external_oauth_issuer = 'https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7'
48    external_oauth_jws_keys_url = 'https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/v1/keys'
49    external_oauth_audience_list = ('https://esa45191.us-east-1.snowflakecomputing.com')
50    external_oauth_token_user_mapping_claim = 'sub'
51    external_oauth_snowflake_user_mapping_attribute = 'login_name'
52    external_oauth_any_role_mode = 'ENABLE';
53

```

status
1 Integration SURAJ_SNOWFLAKE_OKTA_EXTERNAL_OAUTH successfully created.

Stage3: In Alation: enable and configure the OAuth for your Snowflake data source.

Empty Config At Alation Side

Compose Connection Sharing

- Shared connections across tabs
 More performant, but may prevent users from running multiple concurrent queries in separate Compose tabs.
- Separate connection per tab
 Enables users to run multiple queries simultaneously at the possible expense of system performance. Also: temporary tables created in one tab may not be visible to queries in another.

OAuth Connection

Enable OAuth 2.0 in Compose

Client ID Client Secret

Request Refresh Tokens Enable PKCE

Authorization Endpoint Token Endpoint

Default Scope Refresh Scope

Username Field/Claim JWT

Access token parameter name authenticator=oauth

Enable OAuth 2.0 for all Compose Connections and also:

- Drop Function
- Data Upload
- Dynamic Sampling/Profiling

Data at Snowflake Side:

```

1  DESC SECURITY_INTEGRATION "SURAJ_SNOWFLAKE_OKTA_EXTERNAL_OAUTH";

```

property	property_type	property_value
ENABLED	Boolean	true
EXTERNAL_OAUTH_ISSUER	String	https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7
EXTERNAL_OAUTH_JWS_KEYS_URL	Object	https://dev-66731772.okta.com/oauth2/aus8jqjjrwUVSKfV85d7/v1/keys
EXTERNAL_OAUTH_ANY_ROLE_MODE	String	ENABLE
EXTERNAL_OAUTH_RSA_PUBLIC_KEY	String	
EXTERNAL_OAUTH_RSA_PUBLIC_KEY_2	String	
EXTERNAL_OAUTH_BLOCKED_ROLES_LIST	List	ACCOUNTADMIN,SECURITYADMIN
EXTERNAL_OAUTH_ALLOWED_ROLES_LIST	List	
EXTERNAL_OAUTH_AUDIENCE_LIST	List	https://esa45191.us-east-1.snowflakecomputing.com
EXTERNAL_OAUTH_TOKEN_USER_MAPPING_CLAIM	Object	{'sub'}
EXTERNAL_OAUTH_SNOWFLAKE_USER_MAPPING_ATTRIBUTE	String	login_name
COMMENT	String	

Data from OKTA Client Side

- Client ID, 0oa8jq9eoczePGg1g5d7
- Client Secret, Ee9T-LftzgpbcCvyeD52h8P6cgCsWiRi8ga_wsZ

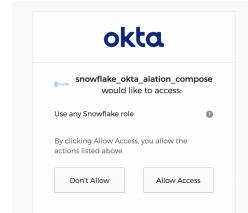
Data from OKTA Server Side

- Authorization endpoint, <https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKfV85d7/v1/authorize>
- Token endpoint, <https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKfV85d7/v1/token>

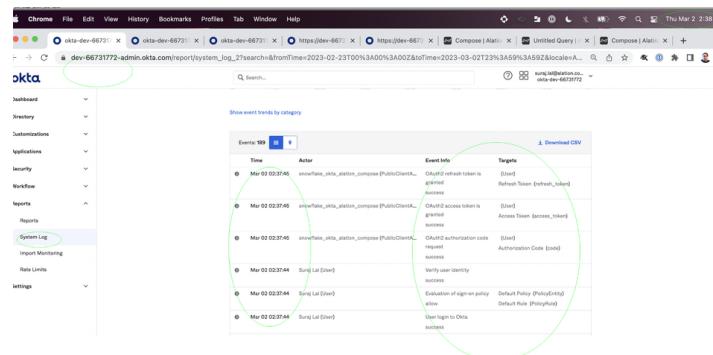
The screenshot shows the 'Compose Connections' page in Alation. A connection named 'Default Connection' is selected, pointing to 'snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth'. Under 'Compose Connection Sharing', 'Separate connection tab' is selected. In the 'OAuth Connection' section, 'Enable OAuth 2.0 in Compose' is checked. The client ID is set to '00a8jg9ecze9gjg5d7'. Below it, 'Request Refresh Tokens' is checked, while 'Enable PKCE' is unchecked. Two URLs are listed under 'Client URLs': 'https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKfV85d7/v1/authorize' and 'https://dev-66731772.okta.com/oauth2/aus8qjjrwUVSKfV85d7/v1/token'. The 'Save' button is at the bottom. A blue callout box highlights the 'Enable OAuth 2.0 for all Compose Connections and also:' section, which includes 'Query Forms', 'Data Upload', and 'Dynamic Sampling/Profiling'.

TESTING OKTA :

- After OKTA Redirect from Alation
- User Authenticates and then
- Consent from client application



Logs at OKTA Server Side for Success-full Login



severity	event_type	display_message	uuid	version	timestamp
INFO	user.session.start	User login to Okta	bd138fdb-b8c4-11ed-b66d-41d1699663fc	0	2023-03-02T06:37:44
INFO	user.authentication.verify	Verify user identity	bd1abbc8-b8c4-11ed-b66d-41d1699663fc	0	2023-03-02T06:37:44
INFO	app.oauth2.as.token.grant.refresh_token	OAuth2 refresh token is granted	be00afba-b8c4-11ed-8905-81624f8eed9	0	2023-03-02T06:37:45
INFO	policy.evaluate_sign_on	Evaluation of sign-on policy	bd1871dc-b8c4-11ed-b66d-41d1699663fc	0	2023-03-02T06:37:44
INFO	app.oauth2.as.authorize.code	OAuth2 authorization code request	bd95808b-b8c4-11ed-a2b6-f5f5729c420f	0	2023-03-02T06:37:45
INFO	app.oauth2.as.token.grant.access_token	OAuth2 access token is granted	be0088a9-b8c4-11ed-8905-81624f8eed9	0	2023-03-02T06:37:45

in	timestamp	outcome.result	outcome.reason	actor.id
0	2023-03-02T06:37:44.363Z	SUCCESS		00u7qelljmzf70Y7F5d7
0	2023-03-02T06:37:44.410Z	SUCCESS		00u7qelljmzf70Y7F5d7
0	2023-03-02T06:37:45.917Z	SUCCESS		0oa8jq9eoczePGg1g5d7
0	2023-03-02T06:37:44.395Z	ALLOW	Sign-on policy evaluation resulted in ALLOW	00u7qelljmzf70Y7F5d7
0	2023-03-02T06:37:45.216Z	SUCCESS		0oa8jq9eoczePGg1g5d7
0	2023-03-02T06:37:45.916Z	SUCCESS		0oa8jq9eoczePGg1g5d7

actor.id	actor.type	actor.display_name	actor.alternate_id	authentication_context.authentication_step	at
00u7qelljmzf70Y7F5d7	User	Suraj Lal	sural.lal@alation.com		0
00u7qelljmzf70Y7F5d7	User	Suraj Lal	sural.lal@alation.com		0
0oa8jq9eoczePGg1g5d7	PublicClientApp	snowflake_okta_alation_compose	0oa8jq9eoczePGg1g5d7		0
00u7qelljmzf70Y7F5d7	User	Suraj Lal	sural.lal@alation.com		0
0oa8jq9eoczePGg1g5d7	PublicClientApp	snowflake_okta_alation_compose	0oa8jq9eoczePGg1g5d7		0
0oa8jq9eoczePGg1g5d7	PublicClientApp	snowflake_okta_alation_compose	0oa8jq9eoczePGg1g5d7		0

Azure AD_Snowflake_Alation:

<https://docs.snowflake.com/en/user-guide/oauth-ext-overview>

<https://docs.snowflake.com/en/user-guide/oauth-azure>

Prerequisite Step: Determine the OAuth Flow in Azure AD

Azure AD supports two different OAuth flows in which an OAuth Client can get an access token.

1. The authorization server can grant the OAuth client an access token on behalf of the user.(We will User based)
2. The authorization server can grant the OAuth client an access token for the OAuth client itself.

Configuration Work Flow in 3 Stages:

Stage1.1 Create App registration for AUTH SERVER

Stage1.1 Input- Give Snowflake URL as Audience , Snowflake role in scope, and Application UID(can be anything that is there in trusted domain of AzureAD)
 Stage1.2 Output-Endpoints

Stage1.2 Create App registration for AUTH Client

Stage1.1 Input- Alation callback url, Generate Client ID
 Stage1.1 Output- client id and secret

Stage2: Configure Snowflake with Azure Auth Server details

Stage2 Input- run integration register query with Azure Auth Server endpoint details

Stage3: Configure Alation Instance with Azure Client ID and Azure Auth server endpoints Details details

Stage3Input: Use Stage1 input in Alation's data source compose connection setting

- Complete either sub-step 10 or 11 in [Step 1: Configure the OAuth Resource in Azure AD](#).
- Complete either sub-step 13 or 14 in [Step 2: Create an OAuth Client in Azure AD](#).

Stage1.1 Create App registration for AUTH SERVER

The screenshot shows the Azure portal interface. At the top, there is a search bar with the text "portal.azure.com/#home". Below the search bar is a blue header bar with the word "Azure" and a "Search resources" button. Underneath the header, the text "Azure services" is displayed. A search bar is present, and below it, there is a list of service icons. One icon, labeled "App registrations", is highlighted with a white background and a thin gray border, indicating it is the selected item.

Home > App registrations >
Register an application ...

* Name

The user-facing display name for this application (this can be changed later).

SURAJ_AD_SNOWFLAKE_AUTH_SERVER ✓

Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Alation, Inc. only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- Personal Microsoft accounts only

[Help me choose...](#)

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web iandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback/✓

Provide details:

- Name SURAJ_AD_SNOWFLAKE_AUTH_SERVER
- Platform type web callbackurl (Note No forward)
- https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback

SURAJ_AD_SNOWFLAKE_AUTH_SERVER | Authentication

Platform configurations

Depending on the platform or device this application is targeting, additional configuration may be required such as redirect URIs, specific authentication settings, or fields specific to the platform.

Web

Redirect URIs

The URLs we will accept as destinations when returning authentication responses (tokens) after successfully authenticating or signing out users. The redirect URI you send in the request to the login server should match one listed here. Also referred to as reply URLs. [Learn more about Redirect URIs and their restrictions](#)

https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback ✓

Set Application ID URI (We will use <https://alation.com/> as Application ID URI)

- Note Since only allowed domains URL placeholders in our case are
- https://portal.azure.com/#view/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/-/Domains

Alation, Inc. | Custom domain names

Looking to move an on-premises application to the cloud and use Azure Active Directory Domain Services?

Name	Status	Federated	Primary
alation.com	Verified	✓	
alation.microsoft.com	Available	✓	
alationsupport.onmicrosoft.com	Unverified		

Microsoft Azure

Home > App registrations > SURAJ_AD_SNOWFLAKE_AUTH_SERVER

SURAJ_AD_SNOWFLAKE_AUTH_SERVER | Expose an API

Search Got feedback?

Overview Quickstart Integration assistant

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API**

Application ID URI Set

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking for application-only scopes, type. Go to App roles.

+ Add a scope

Scopes	Who can consent
No scopes have been defined	

Application ID URI <https://alation.com/suraj>

Home > App registrations > SURAJ_AD_SNOWFLAKE_AUTH_SERVER

SURAJ_AD_SNOWFLAKE_AUTH_SERVER | Expose an API

Search Got feedback?

Overview Quickstart Integration assistant

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API**
- App roles

Application ID URI <https://alation.com/suraj>

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking for application-only scopes, type. Go to App roles.

+ Add a scope

Scopes	Who can consent
No scopes have been defined	

Now click Add scope

Home > App registrations > SURAJ_AD_SNOWFLAKE_AUTH_SERVER

SURAJ_AD_SNOWFLAKE.AUTH_SERVER | Expose an API

Search Got feedback?

Overview Quickstart Integration assistant

Manage

- Branding & properties
- Authentication
- Certificates & secrets
- Token configuration
- API permissions
- Expose an API**
- App roles
- Owners
- Roles and administrators
- Metrics

Application ID URI <https://alation.com/suraj>

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the API. An application that requires API access must define at least one scope for the API to request to one or more of these scopes.

Adding a scope here creates only delegated permissions. If you are looking to create application-only scopes, use type. Go to App roles.

+ Add a scope

Scope name *

Who can consent

Admins and users

Admins consent display name *

Admins consent description *

User consent display name *

User consent description *

Status Enabled

Provide:

- Scope Name: SESSION:ROLE-ANY
- Who can access: Admins and users
- Consent :any

Add a scope

Scope name * [Get feedback?](#)

Application URL [Edit](#) [Delete](#)

Who can consent? **Admins and users** Admins only

Admin consent display name *

Admin consent description *

User consent display name

User consent description

Date Enabled Disabled

[Add scope](#) [Cancel](#)

[Save](#) [Discard](#) [Delete](#)

Scope name * [Get feedback?](#)

https://alation.com/suraj/SESSION:ROLE:ANY

Who can consent? **Admins and users** Admins only

Admin consent display name *

Admin consent description *

User consent display name

User consent description

State Enabled Disabled

Scope

Expose an API	+ Add a scope	Scopes	Who can consent	Admin consent display ...	User consent display na...	State
App roles		https://alation.com/suraj/SESSION:ROLE:ANY	Admins and users	any	any	Enabled

Stage1.2 Create App registration for AUTH Client

CREATE AUTH CLIENT

Step 2: Create an OAuth Client in Azure AD

Requires the correct Azure AD tenant and authentication.

- Navigate to the Azure Active Directory.
- Click on App Registration.
- Click on New registration.
- Enter a name for the client such as **Suraj Multi Client**.
- Select the application type as Single sign-on.
- Click Register.
- In the Overview section, copy the **Client ID** for the Application (client ID) field. This will be stored in Step 3.
- Click on Certificates & secrets and then the **New client secret**.
- Enter a name for the secret and set its expiration date.
- Set secret expires.
- For testing purposes, select secrets that never expire.
- Click on Save changes.
- For production clients that are requesting an Access Token in a general or user context, configure permissions as follows:
 - Click on Add Permissions.
 - Click on Microsoft Graph.
 - Click on the **ReadWriteMulti** Resource that you created in Step 1 Configure the OAuth Resource in Microsoft Graph.
 - Click on the Delegated Permissions tab.
 - Check the **Delegated Permissions** related to the scopes defined in the Application (client ID) of the Application (client ID).
 - Click Add Permissions.
 - Click on the **Suraj Admin Consent** button to grant the permission to the client. Note that for testing purposes, permissions are configured this way. However, in a production environment, granting permissions to this resource is not recommended.
 - Click Yes.
- For production clients that are requesting an Access Token for themselves, configure API permissions for Applications as follows:
 - Click on API permissions.
 - Click on Add Permissions.
 - Click on My API.
 - Click on the **ReadWriteMulti** Resource that you created in Step 1 Configure the OAuth Resource in Microsoft Graph.
 - Click on the Application Permissions.
 - Check the **Application Permissions** related to the scopes normally defined in the Application (client ID) of the Application (client ID).
 - Click Add Permissions.
 - Click on the **Suraj Admin Consent** button to grant the permission to the client. Note that for testing purposes, permissions are configured this way. However, in a production environment, granting permissions to this resource is not recommended.
 - Click Yes.

- SURAJ_AD_SNOWFLAKE.AUTH_CLIENT
- Add Web URL https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback

Register an application

* Name

The user-facing display name for this application (this can be changed later).

Supported account types

Who can use this application or access this API?

- Accounts in this organizational directory only (Alation, Inc. only - Single tenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant)
- Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts
- Personal Microsoft accounts only

[Help me choose...](#)

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is changed later, but a value is required for most authentication scenarios.

Select a platform

e.g. https://example.com/auth

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by [a](#)

By proceeding, you agree to the Microsoft Platform Policies [↗](#)

[Register](#)

SURAJ_AD_SNOWFLAKE_AUTH_CLIENT | Authentication [↗](#) [...](#)

[Search](#) << [Got feedback?](#)

[Overview](#)

[Quickstart](#)

[Integration assistant](#)

[Manage](#)

[Branding & properties](#)

[Authentication](#)

[Certificates & secrets](#)

[Token configuration](#)

[API permissions](#)

[Expose an API](#)

[App roles](#)

[...](#)

Platform configurations

Depending on the platform or device this application is targeting, additional configuration may be required : redirect URIs, specific authentication settings, or fields specific to the platform.

[Add a platform](#)

Web

Redirect URIs

The URLs we will accept as destinations when returning authentication responses (tokens) after successfully authenticating the user. The URL sent in the request to the login server should match one listed here. Also referred to as reply URLs. [Learn more](#)

[Add URI](#)

Generate Client Secret and Copy

1. Click on **Certificates & secrets** and then **New client secret**.
2. Add a description of the secret.
3. Select **never expire**. For testing purposes, select secrets that never expire.
4. Click **Add**. Copy the secret. This will be known as the <OAUTH_CLIENT_SECRET> in the following steps.

SURAJ_AD_SNOWFLAKE_AUTH_CLIENT | Certificates & secrets

Credentials enable confidential applications to identify themselves to the authentication service. For a higher level of assurance, we recommend using a certificate (instead of a client secret).

Description	Expires	Value	Secret ID
Secret Client expires in 2025	3/2/2025	WyC8Q~7JXkY4uJipFw5a2fVWr6O315qAA4CEGc0D be07805b-16ec-4660-8f07-b182de4db407	

Add a client secret

Description	Secret Client expires in 2025
Expires	Custom
Start	03/02/2023
End	03/02/2025

Description	Expires	Value	Secret ID
Secret Client expires in 2025	3/2/2025	WyC8Q~7JXkY4uJipFw5a2fVWr6O315qAA4CEGc0D be07805b-16ec-4660-8f07-b182de4db407	

- CLIENT SECRET FOR CLIENT AUTH WyC8Q~7JXkY4uJipFw5a2fVWr6O315qAA4CEGc0D
- FOR CLIENT AUTH CLIENT ID 98d037d4-60df-4cfe-900a-73f0dd16c8e4

Essentials

Display name	: SURAJ AD SNOWFLAKE AUTH CLIENT	Copied	Client
Application (client) ID	: 98d037d4-60df-4cfe-900a-73f0dd16c8e4	Copy	Redirect URI
Object ID	: 1c8e2cf8-8c63-4bf4-b21c-eadb3e010fea		Applic
Directory (tenant) ID	: f8e32097-5c65-485f-9eb7-f40149462461		Manage
Supported account types	: My organization only		

Add API Permissions to Auth Server created

For programmatic clients that will request an Access Token on behalf of a user, configure Delegated permissions for Applications as follows.

- Click on **API Permissions**.
- Click on **Add Permission**.
- Click on **My APIs**.
- Click on the **Snowflake OAuth Resource** that you created in [Step 1: Configure the OAuth Resource in Azure AD](#).
- Click on the **Delegated Permissions** box.
- Check on the Permission related to the Scopes defined in the Application that you wish to grant to this client.
- Click **Add Permissions**.
- Click on the **Grant Admin Consent** button to grant the permissions to the client. Note that for testing purposes, permissions are configured this way. However, in a production environment, granting permissions in this manner is not advisable.
- Click **Yes**.

Home > App registrations > SURAJ_AD_SNOWFLAKE_AUTH_CLIENT

SURAJ_AD_SNOWFLAKE_AUTH_CLIENT | API permissions

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process or in organizations where this app will be used. [Learn more about permissions and consent](#)

<input type="button" value="Add a permission"/> API / Permissions name	Type	Description	Admin consent required
<input type="button" value="Microsoft Graph (1)"/> User.Read	Delegated	Sign in and read user profile	No

To view and manage consented permissions for individual apps, as well as your tenant's consent settings, try [Enterprise app consent](#).

Add permission

.SNOWFLAKE_AUTH_CLIENT | AKE_AUTH_CLIENT | API permission

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process or in organizations where this app will be used. [Learn more about permissions and consent](#)

<input type="button" value="Add a permission"/> API / Permissions name	Type	Description	Admin consent required
<input type="button" value="Microsoft Graph (1)"/> User.Read	Delegated	Sign in and read user profile	No

Switch to My APIs on popup

Request API permissions

Select an API

[Microsoft APIs](#) [APIs my organization uses](#) [My APIs](#)

Applications that expose permissions are shown below

Name	Application (client) ID
PowerBI_SE	61da3f97-bcc9-4990-bc6d-d85933
AMD_poc	ad48b7fd-fdd5-45ff-9e3f-ce623a6
SURAJ_AD_SNOWFLAKE_AUTH_SERVER	fa827156-44af-4864-b718-0caf071
TMobile-POC	1669e132-5e18-4307-817c-75dfdf

Double click Auth Server Created

AVID_POC	ad48b7fd-fdd5-45ff-9e3f-ce623a6
SURAJ_AD_SNOWFLAKE_AUTH_SERVER	fa827156-44af-4864-b718-0caf071
TMobile-POC	1669e132-5e18-4307-817c-75dfdf

Request API permissions

[◀ All APIs](#)

SU SURAJ_AD_SNOWFLAKE_AUTH_SERVER
<https://alation.com/suraj>

What type of permissions does your application require?

Delegated permissions

Your application needs to access the API as the signed-in user.

Your application needs user.

Request API permissions

[◀ All APIs](#)

 SURAJ_AD_SNOWFLAKE_AUTH_SERVER
https://alation.com/suraj

What type of permissions does your application require?

Delegated permissions

Your application needs to access the API as the signed-in user.

Application permissions

Your application runs as a signed-in user.

Select permissions

Start typing a permission to filter these results

 The "Admin consent required" column shows the default value for an organization. However, user permission, user, or app. This column may not reflect the value in your organization, or in organization [more](#)

Permission

∨ Permissions (1)

SESSION:ROLE-ANY ⓘ
any

[Add permissions](#)

[Discard](#)

SURAJ_AD_SNOWFLAKE_AUTH_CLIENT | API permissions

Configured permissions

API / Permissions name	Type	Description	Admin consent required
User Read	Delegated	Sign in and read user profile	No
SESSION_ROLE_ANY	Delegated	any	No

Collect Azure AD Information for Snowflake

1. Navigate to the [Microsoft Azure Portal](#) and authenticate.
2. Navigate to Azure Active Directory.
3. Click on **App Registrations**.
4. Click on the **Snowflake OAuth Resource** that you created in [Step 1: Configure the OAuth Resource in Azure AD](#).
5. Click on **Endpoints** in the **Overview** interface.
6. On the right-hand side, copy the **OAuth 2.0 token endpoint (v2)** and note the URLs for **OpenID Connect metadata** and **Federation Connect metadata**.
 - The **OAuth 2.0 token endpoint (v2)** will be known as the <AZURE_AD_OAUTH_TOKEN_ENDPOINT> in the following configuration steps. The endpoint should be similar to <https://login.microsoftonline.com/90288a9b-97df-4c6d-b025-95713f21cef9/oauth2/v2.0/token>.
 - For the **OpenID Connect metadata**, open in a new browser window.
 - Locate the "jwks_uri" parameter and copy its value.
 - This parameter value will be known as the <AZURE_AD_JWS_KEY_ENDPOINT> in the following configuration steps. The endpoint should be similar to <https://login.microsoftonline.com/90288a9b-97df-4c6d-b025-95713f21cef9/discovery/v2.0/keys>.
 - For the **Federation metadata document**, open the URL in a new browser window.
 - Locate the "entityID" parameter in the XML Root Element and copy its value.
 - This parameter value will be known as the <AZURE_AD_ISSUER> in the following configuration steps. The entityID value should be similar to <https://sts.windows.net/90288a9b-97df-4c6d-b025-95713f21cef9/>.

SURAJ_AD_SNOWFLAKE_AUTH_CLIENT

Endpoints

SURAJ_AD_SNOWFLAKE_AUTH_SERVER

Endpoints

Display name	:	SURAJ AD SNOWFLAKE AUTH SERVER
Application (client) ID	:	fa827156-44af-4864-b718-0caf071fa142
Object ID	:	c78abd78-a261-4e81-95a6-96dae8dc08bf
Directory (tenant) ID	:	f8e32097-5c65-485f-9eb7-f40149462461
Supported account types	:	My organization only

OAuth 2.0 token endpoint (v2)

<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/token>

Token endpoint v2 <https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/token>

<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/v2.0/.well-known/openid-configuration>

OpenID Connect metadata document

<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/v2.0/.well-known/openid-configuration>

Microsoft Graph API endpoint

jwks_uri Locate Locate the "jwks_uri" here

```
← → C 🔍 login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/v2.0/.well-known/openid-configuration

{"token_endpoint": "https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/.auth/v2.0/token", "token_endpoint_auth_methods_supported": ["client_secret_post", "private_key_jwt", "client_secret_basic"], "jwks_uri": "https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/discovery/v2.0/keys", "response_modes_supported": ["query", "fragment", "form_post"], "subject_types_supported": ["pairwise"], "id_token_signing_alg_params": ["RS256"], "response_types_supported": ["code", "id_token", "code id_token", "id_token token"], "scopes_supported": ["openid", "profile", "email", "offline_access"], "issuer": "https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461"}, {"x-ms-keyvault-kv-uri": "https://keyvaultname.vault.azure.net/"}
```

"jwks_uri":"<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/discovery/v2.0/keys>"

Next Capture Entity ID:

<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/federationmetadata/2007-06/federationmetadata.xml>

Federation metadata document

<https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/federationmetadata/2007-06/federationmetadata.xml>

Locate entityID here

```
← → C 🔍 login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/federationmetadata/2007-06/federationmetadata.xml

This XML file does not appear to have any style information associated with it. The document tree is shown below.

<EntityDescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" ID="29413a42-e03f-4eb5-b129-d6503ccb4cd5" entityID="https://sts.windows.net/f8e32097-5c65-485f-9eb7-f40149462461/>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
</Signature>
```

entityID="<https://sts.windows.net/f8e32097-5c65-485f-9eb7-f40149462461/>"

Stage2:Snowflake Create a Security Integration Azure type

```
1 create security integration SURAJ_SNOWFLAKE_AZUREAD_EXTERNAL_OAUTH
2   type = external_oauth
3   enabled = true
4   external_oauth_type = azure
5   external_oauth_issuer = 'https://sts.windows.net/f8e32097-5c65-485f-9eb7-f40149462461/'
6   external_oauth_jws_keys_url = 'https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/discovery/v2.0/keys'
7   external_oauth_audience_list = ('https://alation.com/suraj')
8   external_oauth_any_role_mode = 'ENABLE'
9   external_oauth_token_user_mapping_claim = 'upn'
10  external_oauth_snowflake_user_mapping_attribute = 'login_name';

11 create security integration SURAJ_SNOWFLAKE_AZUREAD_EXTERNAL_OAUTH
12   type = external_oauth
13   enabled = true
14   external_oauth_type = azure
15   external_oauth_issuer = 'https://sts.windows.net/f8e32097-5c65-485f-9eb7-f40149462461/'
16   external_oauth_jws_keys_url = 'https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/discovery/v2.0/keys'
17   external_oauth_audience_list = ('https://alation.com/suraj')
18   external_oauth_any_role_mode = 'ENABLE'
19   external_oauth_token_user_mapping_claim = 'upn'
20   external_oauth_snowflake_user_mapping_attribute = 'login_name';

21
```

Objects Editor Results Chart

status
Integration SURAJ_SNOWFLAKE_AZUREAD_EXTERNAL_OAUTH successfully created.

property	property_type	property_value	...
1 ENABLED	Boolean	true	
2 EXTERNAL_OAUTH_ISSUER	String	https://sts.windows.net/f8e32097-5c65-485f-9eb7-f40149462461/	
3 EXTERNAL_OAUTH_JWS_KEYS_URL	Object	https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/discovery/v2.0/keys	
4 EXTERNAL_OAUTH_ANY_ROLE_MODE	String	ENABLE	
5 EXTERNAL_OAUTH_RSA_PUBLIC_KEY	String		
6 EXTERNAL_OAUTH_RSA_PUBLIC_KEY_2	String		
7 EXTERNAL_OAUTH_BLOCKED_ROLES_LIST	List	ACCOUNTADMIN,SECURITYADMIN	
8 EXTERNAL_OAUTH_ALLOWED_ROLES_LIST	List		
9 EXTERNAL_OAUTH_AUDIENCE_LIST	List	https://alation.com/suraj	
10 EXTERNAL_OAUTH_TOKEN_USER_MAPPING_CLAIM	Object	['upn']	
11 EXTERNAL_OAUTH_SNOWFLAKE_USER_MAPPING_ATTRIBUTE	String	login_name	
12 COMMENT	String		

Just for understand what to put in Alation for User Field Name (Like Email or Username or UPN)

Sample Users Mapping

In Azure AD:

we used UPN (USER Principal name) external_oauth_token_user_mapping_claim = 'upn'

The screenshot shows the Microsoft Azure AD 'Users' page. A search bar at the top contains 'suraj'. Below it, a table lists one user found: 'Suraj Lal' with the User principal name 'suraj.lal@alation.com'. Other columns include 'User type' (Member), 'On-premises sync status' (No), 'Identities' (alation.onmicrosoft.com), 'Company name' (alation.onmicrosoft.com), and 'Creation type'.

In Snowflake: We use Login Name

The screenshot shows the Alation Data Catalog interface. On the left, a sidebar menu for 'Suraj Lal' (ACCOUNTADMIN) includes 'Worksheets', 'Dashboards', 'Data', 'Marketplace', 'Activity', 'Admin' (with 'Usage', 'Warehouses', and 'Resource Monitors' sub-options), and 'Users & Roles' (which is selected). On the right, a detailed user profile for 'SURAJLAL' is displayed under the heading 'About'. It shows the 'Login Name' as 'SURAJ.LAL@ALATION.COM', 'Default Warehouse' as '—', and 'Last Login' as '2 minutes ago'. Below this is a section titled 'Privileges'.

Stage3 Configure Alation

Configure Alation Instance with Azure Client ID and Azure Auth server endpoints Details details

The screenshot shows the 'Compose Connection Sharing' section of the Alation connection configuration. It offers two options: 'Shared connections across tabs' (disabled) and 'Separate connection per tab' (selected). Below this is the 'OAuth Connection' section. Under 'Enable OAuth 2.0 in Compose', there is a note about a saved client ID and a link to change it. The 'Request Refresh Tokens' checkbox is checked. A note indicates that tokens are stored in the browser. The 'Save' button is at the bottom.

Title Default Connection **URI** snowflake://alation-alationprosver.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth

Compose Connection Sharing

Shared connections across tabs
More performant, but may prevent users from running multiple queries in parallel, even from separate Compose tabs

Separate connection per tab
Enables users to run multiple queries simultaneously, at the possible expense of system performance. Also temporary tables created in one tab may not be visible to queries in another.

OAuth Connection

Enable OAuth 2.0 in Compose
Currently saved client ID is 98e037d4-60df-4cfe-908a-73f0dd16c8e4. Click here to change it and the client secret.

Request Refresh Tokens Enable PKCE

Enable OAuth 2.0 for all Compose Connections and also:
• Query Forms
• Data Upload
• Dynamic Sampling/Profiling

https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oas

https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oas

https://alation.com/suraj/SESS

offline_access

upn WFO

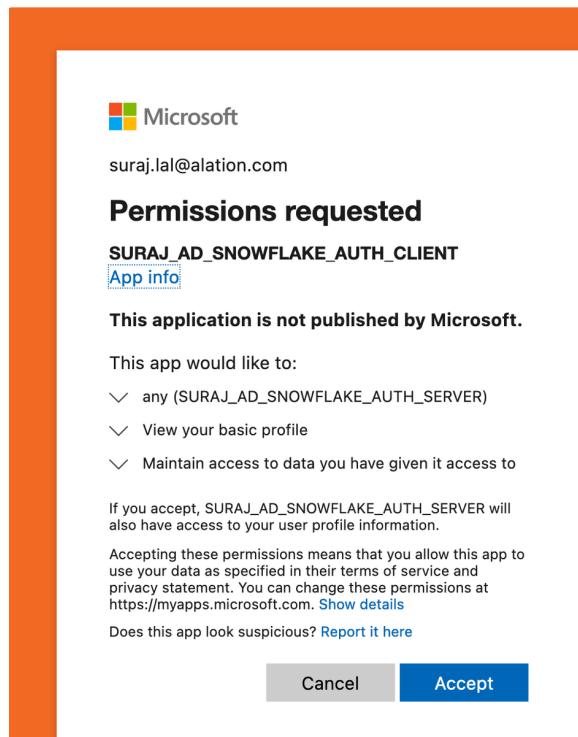
token

authenticator.oauth

Save

Test Azure Ad Flow

Error 1: No reply Address



This screenshot shows the Azure portal's 'Authentication' blade for the 'SURAJ_AD_SNOWFLAKE_AUTH_CLIENT' application. The 'Web' section is expanded, showing the 'Redirect URIs' configuration. The 'Add URI' button is visible. The URL 'https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback' is listed as a redirect URI. Other sections like 'General' and 'Certificates & secrets' are partially visible on the left.

Error2 Callback/ vs callback

Here i tried with callback/ but it failed as the request was from callback only not callback/ with slash

Success Image



Tracing Success Routing Azure:

Step1 On Opening connection -on compose Redirect OAuth : As soon we open Compose with Respective DataSource

1st call: GET is redirected and added with parameter ds_id=here(26)

```
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?ds_id=26&uri=snowflake%3A%2F%2Fålation-alationproserv.snowflakecomputing.com%3A443%2F%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth
HTTP/1.1 Host: suraj-ha-sandbox-r53.alationproserv.com User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Accept-Language: en-CA,en-US;q=0.7,en;q=0.3 Accept-Encoding: gzip, deflate, br DNT: 1 Connection: keep-alive Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28/ Cookie: sessionid=qk3sj6yxbn1eqb5ozc63y57nweyjhh2; csrfToken=af2ZGBOpadQygCTKd4fZHkSrULr4FKWsxEF7j0bNiLnlt1zGOUeAYqzUEpGthrn Upgrade-Insecure-Requests: 1 Sec-Fetch-Dest: document Sec-Fetch-Mode: navigate Sec-Fetch-Site: same-origin

HTTP/2.0 302 Found date: Fri, 03 Mar 2023 21:43:47 GMT content-type: text/html; charset=utf-8 content-length: 0 location:
https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A/alation-alationproserv.snowflakecomputing.com%3A443%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth&ds_id=26 server: nginx allow: GET, HEAD, OPTIONS etag: "d41d8cd98f00b204e9800998ecf8427e" vary: Cookie x-content-type-options: nosniff nosniff x-xss-protection: 1; mode=block x-request-id: 4e4b2019-55a4-4633-a285-8ede47d80d73 content-security-policy: default-src * 'unsafe-eval' 'unsafe-inline' blob: data: X-Firefox-Spdy: h2
```

```
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?ds_id=26&uri=snowflake%3A%2F%2Fålation-alationproserv.snowflakecomputing.com%3A443%2F%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth
HTTP/1.1 Host: suraj-ha-sandbox-r53.alationproserv.com User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Accept-Language: en-CA,en-US;q=0.7,en;q=0.3 Accept-Encoding: gzip, deflate, br DNT: 1 Connection: keep-alive Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28/ Cookie: sessionid=qk3sj6yxbn1eqb5ozc63y57nweyjhh2; csrfToken=af2ZGBOpadQygCTKd4fZHkSrULr4FKWsxEF7j0bNiLnlt1zGOUeAYqzUEpGthrn Upgrade-Insecure-Requests: 1 Sec-Fetch-Dest: document Sec-Fetch-Mode: navigate Sec-Fetch-Site: same-origin

HTTP/2.0 302 Found date: Fri, 03 Mar 2023 21:43:47 GMT content-type: text/html; charset=utf-8 content-length: 0 location:
https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A/alation-alationproserv.snowflakecomputing.com%3A443%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth&ds_id=26 server: nginx allow: GET, HEAD, OPTIONS etag: "d41d8cd98f00b204e9800998ecf8427e" vary: Cookie x-content-type-options: nosniff nosniff x-xss-protection: 1; mode=block x-request-id: 4e4b2019-55a4-4633-a285-8ede47d80d73 content-security-policy: default-src * 'unsafe-eval' 'unsafe-inline' blob: data: X-Firefox-Spdy: h2
```

```
HTTP Parameters
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?ds_id=26&uri=snowflake%3A%2F%2Fålation-alationproserv.snowflakecomputing.com%3A443%2F%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth HTTP/1.1
Host: suraj-ha-sandbox-r53.alationproserv.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: keep-alive
Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28/
Cookie: sessionid=qk3sj6yxbn1eqb5ozc63y57nweyjhh2; csrfToken=af2ZGBOpadQygCTKd4fZHkSrULr4FKWsxEF7j0bNiLnlt1zGOUeAYqzUEpGthrn
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin

HTTP/2.0 302 Found
date: Fri, 03 Mar 2023 21:43:47 GMT
content-type: text/html; charset=utf-8
content-length: 0
location:
https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A/alation-alationproserv.snowflakecomputing.com%3A443%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth&ds_id=26
server: nginx
allow: GET, HEAD, OPTIONS
etag: "d41d8cd98f00b204e9800998ecf8427e"
vary: Cookie
x-content-type-options: nosniff
x-xss-protection: 1; mode=block
x-request-id: 4e4b2019-55a4-4633-a285-8ede47d80d73
content-security-policy: default-src * 'unsafe-eval' 'unsafe-inline' blob: data:
X-Firefox-Spdy: h2
```

Step2:On Add SSO User

The movement we want to add SSO user , it is redirected to Authorize URL configured in Compose with client id, response_type=code and redirect URI = alation callback url

With state value to return back and final scope variable is passed for matching Audience names

```
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A/alation-alationproserv.snowflakecomputing.com%3A443%3Fwarehouse%3DPS_COMPUTE_WH%26authenticator%3Doauth&ds_id=26
HTTP/1.1 Host: suraj-ha-sandbox-r53.alationproserv.com User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 Accept-Language: en-CA,en-US;q=0.7,en;q=0.3 Accept-Encoding: gzip, deflate, br Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28/ DNT: 1 Connection: keep-alive Cookie: sessionid=qk3sj6yxbn1eqb5ozc63y57nweyjhh2; csrfToken=af2ZGBOpadQygCTKd4fZHkSrULr4FKWsxEF7j0bNiLnlt1zGOUeAYqzUEpGthrn Upgrade-Insecure-Requests: 1 Sec-Fetch-Dest: document Sec-Fetch-Mode: navigate Sec-Fetch-Site: same-origin

HTTP/2.0 302 Found date: Fri, 03 Mar 2023 21:43:47 GMT content-type: text/html; charset=utf-8 content-length: 0 location:
https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cfe-900a-73f0dd16c8e4&response_type=code&redirect_uri=https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback&state=eyJjb25uZWNoaW9uX3VyaS16ICJzbm93Zmxha2U6Ly9hbGF0aW9uLWFsYXRpb25wcm9zZXJ2LnNub3dmGFrZWNvbXB1dGluZy5jb206NDQzLz93YXJlaG91c2U9UFNfQ09NUFVURV9XSCZhdXRozW50aNWhdG9yPW9hdXRoiwgImRzX2lkjogMjYsJC0aW1lc3RhXAlOiaIMjAyMy0wMy0wMyAyMTo0Mzo0Ny4yNzQ3OTkfQ%3D%D&scope=https://alation.com/suraj/SESSION:ROLE-ANY server: nginx etag: "d41d8cd98f00b204e9800998ecf8427e" vary: Cookie x-content-type-options: nosniff x-xss-protection: 1; mode=block x-request-id: 480d9301-d594-4ef9-a5cf-c90433b29a7f content-security-policy: default-src * 'unsafe-eval' 'unsafe-inline' blob: data: X-Firefox-Spdy: h2
```

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?ds_id=2&uri=snowflake%3A%2F%2Falation-alationproserv.snowflake

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?uri=snowflake%3A%2F%2Falation-alationproserv.snowflakecomputing.com%

GET https://login.microsoftonline.com/fbe32097-5c65-485f-9eb7-40149462461/oauth2/v2.0/authorize?client_id=980374-60df-4fc6-900a-730dd16c6e4&resp

GET https://login.live.com/Ms.htm?v=3

GET https://adodcon.msauthimages.net/dbd5a2dd-x1gyth6ysafan8qbknn9khm4ece8tmisav0u2vh8e/logintenantbranding/0/bannerlogo/?ts=637649945424069

HTTP Parameters

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A%2F%2Falation-alationproserv.snowflakecomputing.com%2F%2Fwarehouse%3DPS COMPUTE_WH&authenticator=%2Doauth&ds_id=26 HTTP/1.1

Host: suraj-ha-sandbox-r53.alationproserv.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5312.102 Safari/537.36

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Accept-Language: en-CA,en-US;q=0.7,en;q=0.3

Accept-Encoding: gzip, deflate

Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28

DNT: 1

Connection: keep-alive

Cookie: sessionid=qk3sjyjxybkleqb5oc63y57nveygjhb2; csrfToken=af22d809padQycGTCkd4ffHVKsRULz4FKKexEF7j0hNIlNlt1zGOUeaYqzUEp0thr

Upgrade-Insecure-Requests: 1

Sec-Fetch-Dest: document

Sec-Fetch-Mode: navigate

Sec-Fetch-Site: same-origin

HTTP/2.0 302 Found

date: Fri, 03 Mar 2023 21:43:47 GMT

content-type: text/html; charset=utf-8

content-length: 114

location: https://login.microsoftonline.com/fbe32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/authorize?client_id=980374-60df-4fc6-900a-730dd16c6e4&response_type=code&redirect_uri=https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback&state=d1d18cd98f00b204e800998ecf8427e&scope=https://alation.com/SESSION;ROLE=ANY

server: Microsoft-IIS/10.0

set-cookie: ebsessionid=d1d18cd98f00b204e800998ecf8427e; etag: "d1d18cd98f00b204e800998ecf8427e"

vary: Cookie

x-content-type-options: nosniff

x-xss-protection: 1; mode=block

x-request-id: 480d9101-d594-4ef9-a5cf-c90433b29a7f

content-security-policy: default-src * 'unsafe-eval' 'unsafe-inline' blob: data:

X-Firefox-Spdy: h2

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/auth/redirect/?uri=snowflake%3A%2F%2Falation-alationproserv.snowflakecomputing.com%

GET https://login.microsoftonline.com/fbe32097-5c65-485f-9eb7-40149462461/oauth2/v2.0/authorize?client_id=980374-60df-4fc6-900a-730dd16c6e4&resp

GET https://login.live.com/Ms.htm?v=3

GET https://adodcon.msauthimages.net/dbd5a2dd-x1gyth6ysafan8qbknn9khm4ece8tmisav0u2vh8e/logintenantbranding/0/bannerlogo/?ts=637649945424069

HTTP Parameters

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect/?uri=snowflake%3A%2F%2Falation-alationproserv.snowflakecomputing.com%2F%2Fwarehouse%3DPS COMPUTE_WH&authenticator=%2Doauth&ds_id=26 HTTP/1.1

Ur: https://suraj-ha-sandbox-t53.alationproserv.com/api/datasource_auth/raun/requires/os_id=ze&uri=snowmake%3A%2F%2F-alation-alationproserv.snowflakecomputing.com%2Fdatasource%2Fos_id%2Fze%2F%2097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cde-800a-73f0531c8a4&resp_type=token

GET https://suraj-ha-sandbox-t53.alationproserv.com/api/datasource_auth/oauth/redirec/?uri=snowflake%3A//alation-alationproserv.snowflakecomputing.com%2Fdatasource%2Fos_id%2Fze%2F%2097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cde-800a-73f0531c8a4&resp_type=token

GET https://login.live.com/Met.htm?v=3

GET https://aadcdn.msauiimages.net/0db5a2dd-x1gyth6jsafem8qbkm9kilm4eoce8tniawav0u2vh8e/loginenantbranding/0/bannerlogo?ts=637649945424069&v=1

POST https://suraj-ha-sandbox-t53.alationproserv.com/metrics/collect/

GET https://suraj-ha-sandbox-t53.alationproserv.com/version/

GET https://suraj-ha-sandbox-t53.alationproserv.com/version/

GET https://suraj-ha-sandbox-t53.alationproserv.com/version/

POST https://suraj-ha-sandbox-t53.alationproserv.com/metrics/collect/

POST https://login.microsoftonline.com/common/GetCredentiaType?lmkt=en-CA

GET https://alation.okta.com/app/office365xkr8uqj5PxJa46357/ssowefed/pasive?client-request-id=e8e1b664-5ed3-44d4-9d14-32299f61348c&WS-FED

GET https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkbr8u8n15PxJa46357%2Fssso%2Fwsfed%2Fpassive%3Fclient-request-id%3D

GET https://login.okta.com/discovery/frame.html

HTTP Parameters

GET https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cde-800a-73f0531c8a4&resp_type=token

```
POST /_api/_search?size=500&version=true&versionType=internal&filterPath=_score HTTP/1.1
Host: search-01.alaition.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://uraaj-ha-sandbox-r53.alaitionproserv.com/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site

HTTP/1.1 200 OK
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Type: text/html; charset=utf-8
Content-Encoding: gzip
Expires: -1
Vary: Accept-Encoding
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Link: <https://aadddn.msauth.net>; rel=dns-prefetch, <https://aadddn.msauth.net>; rel=dns-prefetch,
<https://aadddn.msauth.net>; rel=dns-prefetch
X-DNS-Prefetch-Control: on
P3P: CP="IDC FIN"
x-ms-request-id: 5d2030e3-4899-4264-b987-375e191c0a01
x-ms-eets-server: 2.1.14649.20 - WUS2 Prodslices
Referrer-Policy: strict-origin-when-cross-origin
X-Content-Type-Options: nosniff
Set-Cookie: build=0.ARcAlydj-GVcX0iet_QBSUYkydq30JjtfP5MkApz8N0WyoQQAAA.AQABAAEAAA--DLA3JVO7qddJg7Wevr2TfI5crZAMz1NhdDC7vCwHlWvtxuK1j5m5mCvq81oCK3vJVN04fsF5zCaJzOs0EdDN9Urz1z4udXXJF_a_e37BhvXF26sUwawq; path=/; expires=Mon, 02-Apr-2023 21:43:47 GMT; secure; HttpOnly; SameSite=None; fnc=ahB2zKw_r15conex-WlRufotkiCkvHAGAhA3Nh13MeGhAAb; enoiresRun=; 02-Apr-2023 21:43:47 GMT; path=/; secure; HttpOnly;
122 requests received (45 hidden)
```

GET	https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/redirect?url=snowflake%3A/alation-alationproserv.snowflakecomputing.com%
GET	https://login.live.com/me.htm?r=3
GET	https://aadcdn.msauthimages.net/dbd5a2dd-x1gyfhvysafam8qbk9kilm4e0ce8tmiwv0u2vh8e/logintenantbranding/0/bannerlogo?ts=637649945424069
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA
GET	https://alation.okta.com/app/office365/exkbr8u8n15PxJa46357/sso/wsfed/passive?client-request-id=8fe1b664-5ed3-44d4-9d14-3229bf61348c&WS-FED
GET	https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkbr8u8n15PxJa46357%2Fssos%2Fwsfed%2Fpassive%3Fclient-request-id%3
GET	https://login.okta.com/discovery/iframe.html

Now User is prompted with default page at SSO Server (Azure AD)

On passing correct user details like username@domain.com .

AD will redirect to respective Authentication page

In our case Azure AD was configured with OKTA

Ignore Metrics and Version calls from Alation

GET	https://suraj-ha-sandbox-r53.alationproserv.com/images/logo/Alation_Logo_White.png
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET	https://suraj-ha-sandbox-r53.alationproserv.com/version/
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA

Continuation : we see that SSO server (Azure AD calls) Internal method with reference data from Alation compose call to get proper Authentcion for this user

<https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA>

POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
POST	https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA
GET	https://alation.okta.com/app/office365/exkbr8u8n15PxJa46357/sso/wsfed/passive?client-request-id=8fe1
GET	https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkbr8u8n15PxJa46357%2F
GET	https://login.okta.com/discovery/iframe.html
GET	https://alation.okta.com/auth/services/devicefingerprint
POST	https://alation.okta.com/api/v1/internal/device/nonce

HTTP

```
POST https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA HTTP/1.1
Host: login.microsoftonline.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: application/json
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth/v2.0/authorize?client_id=98d037d4-60df-4cfe-900a-73f0dd16c8e4&response_type=code&redirect_uri=https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback&state=eyJjb25uZWN0aW9u3VyaSI6ICzbn932mxha2U6y9hbGF0aW9uLWFsYXRp25wcm9zZXJ2LnNub3dmbGFrZWNvblxBlDluZy5jb206NDQzLz93YXJlaG91c2U9UFNFQ09NUFVURV9XSC2hdXRoZw50aNhwdG9yPW9hdXRoIwgImRzX2lkjogMjysICJ0aW1c3rbhXai0iaMjAyMy0wMy0wMyMto0Mzo0Ny4yNz03OTk1f0==&scope=https://alation.com/suraj/SESSION:ROLE-ANY
```

Step 2.1 SSO page1

Here for this specific user company , SSO Server (Azure AD) is configured with OKTA SSO server with WS federation method

So Azure passes all auth request to OKTA

POST	https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA
GET	https://alation.okta.com/app/office365/exkbr8u8n15PxJa46357/sso/wsfed/passive?client-request-id=8fe1b664-5ed3-44d4-9d14-3229bf61348c&WS-FED
GET	https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkbr8u8n15PxJa46357%2Fssos%2Fwsfed%2Fpassive%3Fclient-request-id%3
GET	https://login.okta.com/discovery/iframe.html
GET	https://alation.okta.com/auth/services/devicefingerprint
POST	https://alation.okta.com/api/v1/internal/device/nonce

HTTP Parameters

```
GET https://alation.okta.com/app/office365/exkbr8u8n15PxJa46357/sso/wsfed/passive?client-request-id=8fe1b664-5ed3-44d4-9d14-3229bf61348c&WS-FED
GET https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkbr8u8n15PxJa46357%2Fssos%2Fwsfed%2Fpassive%3Fclient-request-id%3
GET https://login.okta.com/discovery/iframe.html
GET https://alation.okta.com/auth/services/devicefingerprint
POST https://alation.okta.com/api/v1/internal/device/nonce

HTTP
POST https://login.microsoftonline.com/common/GetCredentialType?mkt=en-CA HTTP/1.1
Host: login.microsoftonline.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: application/json
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://login.microsoftonline.com/f8e32097-5c65-485f-9eb7-f40149462461/oauth/v2.0/authorize?client_id=98d037d4-60df-4cfe-900a-73f0dd16c8e4&response_type=code&redirect_uri=https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback&state=eyJjb25uZWN0aW9u3VyaSI6ICzbn932mxha2U6y9hbGF0aW9uLWFsYXRp25wcm9zZXJ2LnNub3dmbGFrZWNvblxBlDluZy5jb206NDQzLz93YXJlaG91c2U9UFNFQ09NUFVURV9XSC2hdXRoZw50aNhwdG9yPW9hdXRoIwgImRzX2lkjogMjysICJ0aW1c3rbhXai0iaMjAyMy0wMy0wMyMto0Mzo0Ny4yNz03OTk1f0==&scope=https://alation.com/suraj/SESSION:ROLE-ANY
Host: alation.okta.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://login.microsoftonline.com/
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Here we can see the username

```
GET https://alation.okta.com/api/v1/flows/359/exb8i6n5PJa46357feso?wsfed/passive?client-request-id=8fe1b664-5ed3-44d4-9d14-3220fe1348&c_WS-FED
GET https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexkb8u8n15PJa46357%2Fssos%2Fwsfed%2Fpassive%3Fclient-request-id%2
GET https://login.okta.com/discovery/frame.html
GET https://alation.okta.com/auth/services/devicefingerprint
POST https://alation.okta.com/api/v1/internal/device/hnonce

HTTP Parameters
GET
client-request-id: 8fe1b664-5ed3-44d4-9d14-32296f61348c
wa: wsigin1
wt: wsigIn1
wtrealm: federation:MicrosoftOnline
wtctx: LoginOptions=3&estradirect=2&estrarequest=rQO1ARAAtVI7bNNQPK2btobKfIRggNTA0IVKjy_yTtUifStIMFxINPUjiWRknx-XZsvsu0Ms5q1IVJBGjy1dmJd61QjBzTGReg1bDFU6opJILEMtdi3v3ffg6C7lmwLcbidicxECCsBySMx2bhAgsaA6wLJowSzaxXhh_rz1_P2D1
9tV1ps5cEKS1xd1rX1xR6I9y2Ygkh1Se8EPqg0JQ53XWt0PhbT1iyw3q3iOKQs0GK1XjycV92V6sVc64717f3813XpCmYg8iM8b64HgEi6lXg3v1cpUR1r1Ud9htnc2Axggms1Rhr1scFaQcYJM60
dq1uhraof6wmf6QNkXmungr2jlbzURX5Jtc0G7YrsNAGU25WtrIFtdaq7lgagnWWz_b1r1WFy0GV7nqLn-6kg2JqjNxk7hb73W17MaH_R_mng1QVvYaxtm98pjPQHjXSHVmxtUDUvwzbhVktMK2Nvuelke3sst_FH6t60TPye-PE8Pb4S2QdkA1kpBRX5b0S2ka8dnhak54TG6q6D3NL_55guwut1i1Q8hdWflendtjv1QuzT887Tyrc1clrTUfz22zMoM8TT2ewtPoIrNx9tfPr2v1vYKr
g8TxzOUpLM1726iGoG9gM41IDY2d0Rc0w-w7-nas2XMoasxt780wwwciJc174pkfco2V1cPq_FwrtA1Mn0_i_C_UzifvvPq8Y7q
cbext:
username: suraj.lal@alation.com
mkts:
lc:
```

Step2.2 SSO page2

Now Actual User Authentication is Started (here OKTA does)

In our case MFA is configured at OKTA

So it ask for Verify Push notification too

	https://login.microsoftonline.com/tenantid/oauth2/v2.0/authorize?response_type=code&client_id=clientid&redirect_uri=https://alation.okta.com/api/v1/internal/device/fingerprint	WS-FED
GET	https://alation.okta.com/appoffline365exxbrru6lSPxJa46357/ssowefed/passive?client-request-id=8fe1b664-5ed3-44d4-9d14-3229bf61348cb	
GET	https://alation.okta.com/login/login.htm?fromURI=%2Fapp%2Foffice365%2Fexbxru6lSPxJa46357%2Fssowefed%2Fpassive%3Fclient-request-id%3	
GET	https://alation.okta.com/discovery/iframe.html	
GET	https://alation.okta.com/api/v1/internal/device/fingerprint	
POST	https://alation.okta.com/api/v1/internal/device/nonce	
GET	https://alation.okta.com/login/getImage?username=suraj.lal%40alation.com	
GET	https://alation.okta.com/auth/services/device/fingerprint	
POST	https://alation.okta.com/api/v1/internal/device/nonce	
POST	https://alation.okta.com/api/v1/internal/device/fingerprint	
POST	https://alation.okta.com/api/v1/auth/factors/otp1mdpxmPckHqH357/verify?autoPush=false&rememberDevice=false	

Step2.3 Alation Gets Code for OAUTH call

Once User is Authenticated Successfully , OKTA sends info back to AZure AD in POST call with SAML response in WSFED Call

This info is redirected to Alation call back with Code value that Alation requested

We See OKTA make POST Call

Referer: https://alation.okta.com/

POST https://login.microsoftonline.com/login.srf HTTP/1.1

Redirect Location is Location: https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0.A

Line	https://okt-static.okta.com/assets/img/icons/navicons/navicon-5x52_9bd5db62f5d02f79302d69f61c26.png	WS-FED SAML
POST https://login.microsoftonline.com/login.srf		

HTTP	Parameters	SAML	Summary
POST	https://login.microsoftonline.com/login.srf HTTP/1.1 ✓		
Host:	login.microsoftonline.com		
User-Agent:	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0		
Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8		
Accept-Language:	en-Ca,en-US;q=0.7,en;q=0.3		
Accept-Encoding:	gzip, deflate, br		
Content-Type:	application/x-www-form-urlencoded		
Content-Length:	7347		
Origin:	https://alation.okta.com		
DNT:	1		
Connection:	keep-alive		
Referer:	https://alation.okta.com/	✓	
Cookie:	buide=0.RcAlaypj-GVX0iet_0BSUVYkjd030JjfYPS5MkApz8N0WyoQAAA.AQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--EbucpzYIFGTLBszsIGX3ETxdM4U02zKBDKUf6xEGr0itycav-DliuLDMVSHykdFEM-FbJ7xpDjggPgaAA; x-ms-gateway-slice=estssf; stsservicececookiestsfd; brcap=0; clrc=%2219420%22%3a%22VTMwOz2m%22%2c%22H7mY5pwH%22%}		
Upgrade-Insecure-Requests:	1		
Sec-Fetch-Dest:	document		
Sec-Fetch-User:	navigate		
Sec-Fetch-Site:	cross-site		
HTTP/1.1 302 Found			
Cache-Control:	no-store, no-cache		
Pragma:	no-cache		
Content-Type:	text/html; charset=utf-8		
Content-Encoding:	gzip		
Expires:	-1		
Location:	https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0.ARcAlaypj-GVX0iet_0BSUVYkjd030JjfYPS5MkApz8N0WyoQAAA.AQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--EbucpzYIFGTLBszsIGX3ETxdM4U02zKBDKUf6xEGr0itycav-DliuLDMVSHykdFEM-FbJ7xpDjggPgaAA; x-ms-gateway-slice=estssf; stsservicececookiestsfd; brcap=0; clrc=%2219420%22%3a%22VTMwOz2m%22%2c%22H7mY5pwH%22%}		
HTTP/1.1 200 OK			
Cache-Control:	no-store, no-cache		
Pragma:	no-cache		
Content-Type:	text/html; charset=utf-8		
Content-Encoding:	gzip		
Expires:	-1		
Location:	https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0.ARcAlaypj-GVX0iet_0BSUVYkjd030JjfYPS5MkApz8N0WyoQAAA.AQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--DLA3VO7rdqdgJ7Wevr2TyIrevrZAMz1ntdDDCT2vCwlvTxuK1zJE5mlCv0g81C3vJVN04fsF5zCa3zs0z0Edn9Urzlz4udXYJFa_e37BhvXF26sUwawq8pVgogaA; fpc=AhBzJW_r150goa-WirpfkickyHAQQAABhNlsOAAA; esctx=PAQABAAEAAA--EbucpzYIFGTLBszsIGX3ETxdM4U02zKBDKUf6xEGr0itycav-DliuLDMVSHykdFEM-FbJ7xpDjggPgaAA; x-ms-gateway-slice=estssf; stsservicececookiestsfd; brcap=0; clrc=%2219420%22%3a%22VTMwOz2m%22%2c%22H7mY5pwH%22%}		

Line	https://okt-static.okta.com/assets/img/icons/navicons/navicon-5x52_9bd5db62f5d02f79302d69f61c26.png	WS-FED SAML
POST https://login.microsoftonline.com/login.srf		

HTTP	Parameters	SAML	Summary
POST	https://login.microsoftonline.com/login.srf		
wresult:	<?xml version="1.0" encoding="UTF-8"?><wst:RequestSecurityTokenResponse xmlns:wst="http://docs.oasis-open.org/wss/ws-trust/200512"><wst:TokenType>wsu:Created</wst:TokenType><wsu:Created>https://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</wsu:Created><wsu:Expires>2023-03-03T21:41:53.740Z</wsu:Expires><wsu:Issuer>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd</wsu:Issuer>		

Line	https://okt-static.okta.com/assets/img/icons/navicons/navicon-5x52_9bd5db62f5d02f79302d69f61c26.png	WS-FED SAML
POST https://login.microsoftonline.com/login.srf		

HTTP	Parameters	SAML	Summary
POST	https://login.microsoftonline.com/login.srf		
<saml1:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">>suraj.lal@alation.com</saml1:AttributeValue>			

In summary we can see UPN from OKTA to Azure

POST https://login.microsoftonline.com/login.srf WS-FED SAML

HTTP	Parameters	SAML	Summary
			RequestSecurityTokenResponse
			TokenType urn:oasis:names:tc:SAML:1.0:assertion
			Lifetime - Created 2023-03-03T22:14:53.740Z
			Lifetime - Expired 2023-03-03T23:14:53.740Z
			SAML 1.0 Assertion
			AssertionID aidizvH1Q2lUrRuhMmFFIz-Q
			MajorVersion 1
			MinorVersion 1
			IssueInstant 2023-03-03T22:14:53.740Z
			Issuer alation.coneekhrBu0u15PxJa46357
			Audience urn:federation:MicrosoftOnline
			Subject Ouao61qk7j0Wfb5vY357
			SAML 1.0 AttributeStatement
			Subject Ouao61qk7j0Wfb5vY357
			UPN suraj.lal@alation.com
			ImmutableID Ouao61qk7j0Wfb5vY357
			Embedded certificates
			Certificate 0 Download

Step 3:Compose uses code to get token

Using the Code value Alation compose make internal call for Token

	WS-FED SAML
POST https://login.microsoftonline.com/login.srf	
GET https://suraj-ha-sandbox-r53.alationproser.com/api/datasource_auth/callback?code=0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY	
GET https://suraj-ha-sandbox-r53.alationproser.com/api/datasource_auth/callback?code=0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY	
POST https://suraj-ha-sandbox-r53.alationproser.com/ajax/connector/rdb_connect/	
POST https://suraj-ha-sandbox-r53.alationproser.com/metrics/collect/	
HTTP Parameters	
GET code: 0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY0xXAAA_AgBAAIAAAD--DLA3V070rddgJg7WevrAgDs_wdI9p-M3UkVc9818HdLd2ZypZ28L7D15xp0ky8m-PMty2Z0d1rf-N3vrx3boD0YTtf10Bw48Dcw9A8817JG_XjPrGe17z1GSwspABCfRuInDGW-SFw-j73t73hA73t05G710D-zrfdB9p9OdebcycXkbdl8wjd1XkteeC61_AKs29XDXBDQb0DC002Ha5q1fvG91fKrbdJXcdkjwSSzImxsKY2Pkm7ohn0ql1u7meT8H34b8q25cDj5Tl4bJt7pTt8h72L9e7uR3Qj0uWzqBn4w74u0s12L9281QXG933QqoA_YC3Tx1dur-mv_x01uKqW1Rq14zPQm8_F59nMsF5CLc4c1b2-advuw-vrdx31at1NhJc9M1Qh9MnxLzB8z1-ecrl-PdbrLrb0d4xVL28u4xNRXjpb_a9Y9vHllacot7tVDP1rkh6p2w4Pbm1m55874ctnPyqxz790YoFkC141ly1b9m8Mr7TMoq1gtbVvbewURRHV11CmW7j4PF4X829586a78l1436055306874f418q8x1t7ba2nbd0518h6m4VQDppdAbh8h924yXTSPU_o48QjLqWqf7989ilaau11Kmd1s0o3tCduh3PAJ2PxHjwsQxJ00W1tbAIHOKIwm-dL0EpzP5418Q0baU0x2zGnOp2g-uV-kgfFjeV27vgyEz3XVzdxaJ1L15r5HFWUQdfIDY71j1goyxvtaoWf7fUfjRoKXYV7vDnIMkHf1mCr71WQh6yv_Ks95tmrzhmwPaQZ	
POST code: session_state: 28831d0d-c155-4178-be0a-8ff272c28aba5#	
HTTP Parameters	
GET https://suraj-ha-sandbox-r53.alationproser.com/api/datasource_auth/callback?code=0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY	
GET https://suraj-ha-sandbox-r53.alationproser.com/api/datasource_auth/callback?code=0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY	
POST https://suraj-ha-sandbox-r53.alationproser.com/ajax/connector/rdb_connect/	
POST https://suraj-ha-sandbox-r53.alationproser.com/metrics/collect/	
HTTP Parameters	
GET a4uOpuxqC9xP3apvF0W8_F5mMefdCl4b12-advuOv-BknlScD7p9ab_p9vKvlLscD7f09plrh6p2w6a8tFp1m955874ctnPyqxz790YoFkC141ly1b9m8Mr7TMoq1gtbVvbewURRHV11CmW7j4PF4X829586a78l1436055306874f418q8x1t7ba2nbd0518h6m4VQDppdAbh8h924yXTSPU_o48QjLqWqf7989ilaau11Kmd1s0o3tCduh3PAJ2PxHjwsQxJ00W1tbAIHOKIwm-dL0EpzP5418Q0baU0x2zGnOp2g-uV-kgfFjeV27vgyEz3XVzdxaJ1L15r5HFWUQdfIDY71j1goyxvtaoWf7fUfjRoKXYV7vDnIMkHf1mCr71WQh6yv_Ks95tmrzhmwPaQZ	
POST eyj3b25usWNOa9pw3XvaS161C1zbm93z25chz256y9hbfGp0a9uWf1PmRzXp025vcm9zzX2j1LnNbubdN0Pfr2NWvhXB1gd1u7v5jb206NDQzLz93YXJlC2U5UFRf09NUFVURV9XSC2hdXr0z50aWhndg9yW99hdXr0iwyImzrX21k1JjogMjySICJ0a11c3RhbxAl1aMjyAyMy0WMyAyM7to0Nzy4yNz030Tkif0==	
session_state: 28831d0d-c155-4178-be0a-8ff272c28aba5#	

Query id

We see the query id we are running in the HTTP request <https://suraj-ha-sandbox-r53.alationproser.com/compose/query/28/>

The screenshot shows the Alation Compose interface. The top bar has tabs for "Compose | Alation" and "Untitled Query | Alation Compose". The main area is titled "Untitled Query". A red oval highlights the URL in the browser: <https://suraj-ha-sandbox-r53.alationproser.com/compose/query/28/>. Below it, another red oval highlights the "Settings" tab. The "Connection Settings" section shows a connection named "_EXTERNAL_A..._". The preview pane shows the result of the query:

122 requests received (45 hidden)

We See

Encrypted token received

csrmiddlewaretoken: dYCjz1P9H1kGVQydgCGuPlKtBZhfojOVvgeZ031Vkf6fvnHG25qJuOKzZ4JTwNVVm

title	https://suraj-ha-sandbox-r53.alationproser.com/api/datasource_auth/callback?code=0.ARcAlyDj-GVcXoIet_QBSUYKydQ30JyfP5MkApz8N0WY
id	compose-28
cmd	db_connect
args	[{"26": "snowflake://alation-alationproser.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH", "authenticator": "oauth", "suraj.lal@alation.com": "", "SHARED_NEW"]}

Step4 with token query snowflake

We see GET call to Snowflake via query with Token in request

As

X-CSRFToken: dYCjzIP9H1kGVQydCGuPlKtiBZfhojOVvgeZ031VkfvnHG25qJuOKz4JTvwNVVm

```
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0.ARcAlyDj-GVcXUiEt_QBSUYKydQ30JrYp5MkApz8N0WyoO
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0.ARcAlyDj-GVcXUiEt_QBSUYKydQ30JrYp5MkApz8N0WyoO
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/db_connect/
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/query_update_settings/
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/query_set_output_dir/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials/?db_uri=snowflake;%2F%2Falation-alationproserv.snowflakecomputing.com:443
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
PUT https://suraj-ha-sandbox-r53.alationproserv.com/api/preferred_connection/22/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/preferred_connection/?ds_id=26&user_id=1
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
```

HTTP Parameters

```
GET db_uri: snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth
user: 1
ds_id: 26
```

```
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/query_set_output_dir/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials/?db_uri=snowflake;%2F%2Falation-alationproserv.snowflakecomputing.com:443
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
PUT https://suraj-ha-sandbox-r53.alationproserv.com/api/preferred_connection/22/
C GET https://suraj-ha-sandbox-r53.alationproserv.com/api/preferred_connection/?ds_id=26&user_id=1
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
```

HTTP Parameters

```
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials/?db_uri=snowflake;%2F%2Falation-alationproserv.snowflakecomputing.com:443?Fwarehouse=PS_COMPUTE_WH%26authenticator=OAuth&user=1&ds_id=26
Host: suraj-ha-sandbox-r53.alationproserv.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:109.0) Gecko/20100101 Firefox/110.0
Accept: application/json, text/plain, */*
Accept-Language: en-CA,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
```

X-CSRFToken: dYCjzIP9H1kGVQydCGuPlKtiBZfhojOVvgeZ031VkfvnHG25qJuOKz4JTvwNVVm

DNT: 1

Connection: keep-alive

Referer: https://suraj-ha-sandbox-r53.alationproserv.com/compose/query/28/

Cookie: sessionid=a33j6yxhnlieg5boz63y57weyyjjh2;

tokenid=26; alationproserv_snowflake=ad0ygCTkd4f2HkSrUlrf4PKWsxEF7j0bNlNlnt1zGouEaYqaUEpGthr

Sec-Fetch-Dest: empty

Sec-Fetch-Mode: cors

Sec-Fetch-Site: same-origin

HTTP/2.0 200 OK

date: Fri, 03 Mar 2023 21:46:57 GMT

After Connection success if we make a run Query we see POST call recorded with Token

```
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/query_try_fetch/
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/db_get_current_schema/
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
C GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collect/
```

HTTP Parameters

```
POST
carmaiddleswaretoken: dYCjzIP9H1kGVQydCGuPlKtiBZfhojOVvgeZ031VkfvnHG25qJuOKz4JTvwNVVm
sandbox_id: compose-28
ds_id: 26
cmd: query_try_fetch
args: ["2023-03-03T21:43:18.412Z#b3c78981aa3578","snowflake://alation-alationproserv.snowflakecomputing.com:443
/?warehouse=PS_COMPUTE_WH&authenticator=oauth",2,0,0,false,true]
```

```
POST https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connector/db_reconnect/
POST https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials/?db_uri=snowflake;%2F%2Falation-alationproserv.snowflakecomputing.com:443
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
```

```
HTTP Parameters
POST
carmaiddleswaretoken: dYCjzIP9H1kGVQydCGuPlKtiBZfhojOVvgeZ031VkfvnHG25qJuOKz4JTvwNVVm
query_id: 28
args: ["use role alation_ocf"]
query_id: 28
args: [{"query_id": "28", "session_id": "2023-03-03T21:43:18.412Z#b3c78981aa3578", "uri": "snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth", "schema_name": "CITIBIKE.PUBLIC", "run_index": 1, "stmt_index": 0, "stmt": "use role alation_ocf"}, {"run_variabls": {}, "export_file_format": "csv"}]
```

Step5: Re Connect if broken

If we run query we get error

Connection Settings **All Edits Saved**

Extension: (SAML-tracer) - SAML-tracer

HTTP

POST carmaiddleswaretoken: dYCjzIP9H1kGVQydCGuPlKtiBZfhojOVvgeZ031VkfvnHG25qJuOKz4JTvwNVVm

query_id: 28

args: [{"query_id": "28", "session_id": "2023-03-03T21:43:18.412Z#b3c78981aa3578", "uri": "snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth", "schema_name": "CITIBIKE.PUBLIC", "run_index": 1, "stmt_index": 0, "stmt": "use role alation_ocf"}, {"run_variabls": {}, "export_file_format": "csv"}]

Results **Previews** **History** **Description**

< > ... △ Mar 3, 2023 7... ✓ Mar 3, 2023 5... ✓ Mar 3, 2023 5... △ Query Execution Failed, took unknown seconds.

=SQL

Database connection was not established.

If you suspect any problem with the database connection, you may try [Suggestions are below and take note of them](#). In the meantime, you can run queries without suggestions.

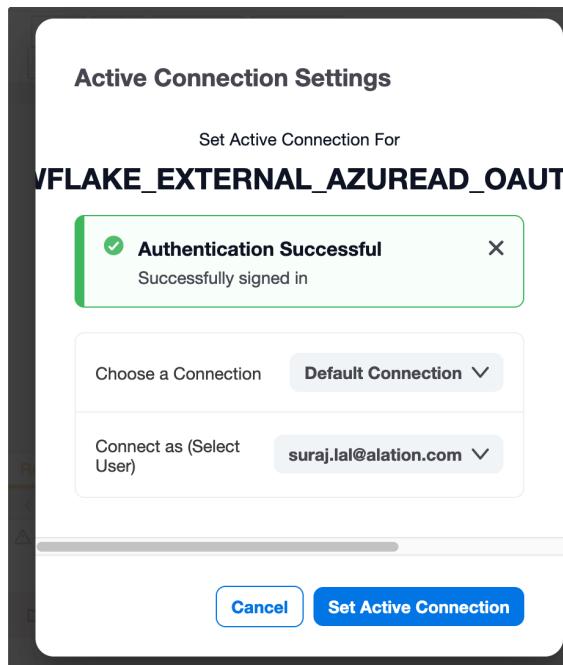
Untitled Query against OCF_SNOWFLAKE_EXTERNAL_/_

Connection Settings All Edits Saved



retry

Once we retry to connect, Since SSO token is available on browser it connected:



```

POST https://suraj-ha-sandbox-r53.alationproserv.com/ajax/connectordb_get_current_schema/
POST https://suraj-ha-sandbox-r53.alationproserv.com/metrics/collection
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5MkA
GET https://login.microsoftonline.com/98e32097-5c55-485f-9eb7-f461149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cf6-900a-73fd1f6de4&sp=
GET https://login.microsoftonline.com/98e32097-5c55-485f-9eb7-f461149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cf6-900a-73fd1f6de4&sp=
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5Mk/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5Mk/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials?db_uri=snowflake%2F%2Falation-alationproserv.snowflakecomputing.com:421
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/

```

Flow Chart :

Code request, code, token(Backend call hidden from Browser) ,Snowflake query , result

Code request

We see Authorize call initiated with code request

```

GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5MkA
GET https://login.microsoftonline.com/98e32097-5c55-485f-9eb7-f461149462461/oauth2/v2.0/authorize?client_id=98d037d4-60df-4cf6-900a-73fd1f6de4&sp=
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5MkA
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/datasource_auth/oauth/callback?code=0_ARCaLyDj-GVcX0let_QBSUYkYdQ30JyJYP5Mk/
GET https://suraj-ha-sandbox-r53.alationproserv.com/api/database_credentials?db_uri=snowflake%2F%2Falation-alationproserv.snowflakecomputing.com:421
GET https://suraj-ha-sandbox-r53.alationproserv.com/version/

```

Code and token

In the next call we have the code so we initiated the token call

```

HTTP Parameters
GET
code=0.ARcAlyDj-GVcX0iet_QBSUYkYdQ30JjFYp5MKapz8N0NyOQXAAA.AqABAAIAAAD-
DLA3V07Qrdggj7ewAgfWuN9P96yvM954tFVZQ0nyI0W4C0yealhthc77mfk11lH-Mp9_N-
Qas_Kr-1V5TjLACRpuG1QVQFXCE_Nrr-JWXYF0M_1x099-1pxg-1pxg-Yt0c7o-1pxg-1pxg-1pxg-
2pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-
OrT3y9yX9TbH2N1LyLGH-X9YF0M_excgxDvNGGdg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-1pxg-
zBVKIRfeot5iy2n-c52zNn-0SL0q7x1m9y7R-WVYt7M9Gw-
eyJzb2s5uNN0aw9u3v3yaS16ICzbm932xhba26l9hbGf0aW91LwFeX9Ypb25cm9z8X2LnNu3dmbGFr2WNvbXB1dGlu2y5jb206NDQzLz93XJla91c2
U9UFUfq9NUFUURVYSX2hdRoEWS0aNhhdg9yP9W9dxRoIw1qImRzX2lKj9mJySICJ0A1lMjAyMy0wMyMyMzozMtozHy42Mjc0MTAI
FO==
```

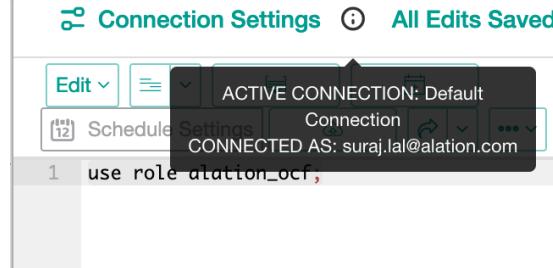
session_state: 28831dd-c155-4178-be0a-8ff27c28aba

Snowflake call

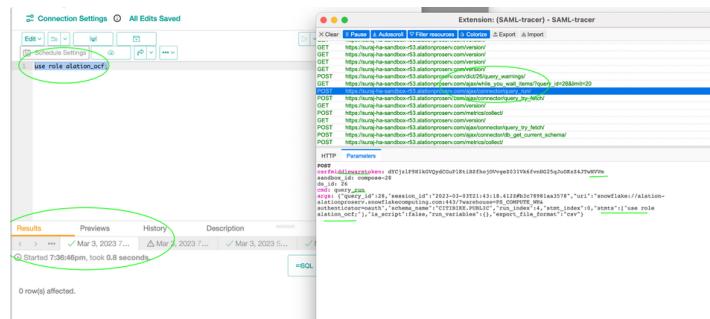
```

HTTP Parameters
GET
db_uri: snowflake://alation-alationproserv.snowflakecomputing.com:443/?warehouse=PS_COMPUTE_WH&authenticator=oauth
user: l
ds_id: 26
t
c

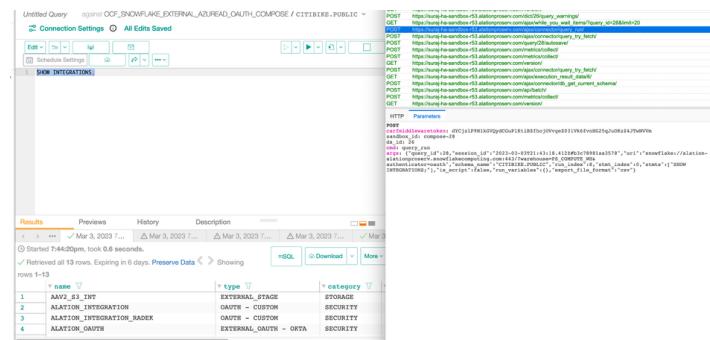
```



Now lets run the Query again



RAN NEW QUERY:query_run



Results:execution_result_data

Then we see execution_result_data call success

GET https://suraj-ha-sandbox-r53.alationproserv.com/ajax/execution_result_data/6

The screenshot shows a browser window with two tabs open. The left tab is titled "SHOW_INTEGRATIONS" and displays a table of results. The right tab shows a network trace with several requests listed:

- HTTP GET https://eu-west-1.snowflakecomputing.com/api/v1/authn/login?query_by_oauth2
- HTTP POST https://eu-west-1.snowflakecomputing.com/api/v1/authn/login?query_by_oauth2
- HTTP POST https://eu-west-1.snowflakecomputing.com/api/v1/authn/logout
- HTTP GET https://eu-west-1.snowflakecomputing.com/api/v1/authn/logout

The table in the left tab has columns: name, type, and category. The data is as follows:

name	type	category
ALATION_OCF	ACCOUNTADMIN	SECURITY
ALATION_INTEGRATION	OAUTH - CUSTOM	SECURITY
ALATION_INTEGRATION_RADER	OAUTH - CUSTOM	SECURITY
ALATION_OAUTHS	EXTERNAL_OAUTH - OCF	SECURITY

Try Switching Roles with Built-in OAuth

Switch using JDBC URL

Default Role here ALATION

The screenshot shows a JDBC connection settings page. The connection is named "Untitled Query" against the database "OCF-Snowflake Test / BAT, PUBLIC". The "Connection Settings" tab is selected. The SQL query is:

```
SELECT CURRENT_ROLE();
```

The results table shows one row:

CURRENT_ROLE()
ALATION

Adding new role

ADD role=role_name in JDBC url

Add new URI Connection

Name

URI

Cancel Save

Validate User is mapped with the role

Granted Roles

SURAJLAL has been granted 4 roles

NAME ↑

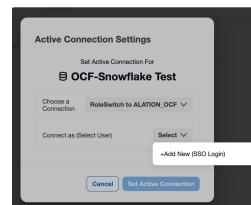
ACCOUNTADMIN

ALATION

ALATION_OCF

ALATION_PS_READ_ONLY

Test



Allow OAUTH



Connection Settings All Edits Saved

ACTIVE CONNECTION: RoleSwitch to ALATION_OCF
CONNECTED AS: SURAJLAL

Validate current role

Untitled Query against OCF-Snowflake Test / BAT.PUBLIC Connection Settings All Edits Saved

Edit Format Embed Insert Date

ACTIVE CONNECTION: RoleSwitch to ALATION_OCF CONNECTED AS: SURAJLAL

1 SELECT CURRENT_ROLE;

Results Previews History Description

< > *** Mar 20, 2023 ... Mar 20, 2023 ...

Started 9:24:11am, took 0.6 seconds.

Retrieved all 1 rows. Expiring in 6 days. Preserve Data Showing rows 1–1

CURRENT_ROLE()

1 ALATION_OCF

Try Switching Roles with External OAuth

https://alationcorp.atlassian.net/wiki/spaces/ProServ/pages/12611060432 Can't find link

Before Connection

Tried creating a JDBC URL with role parameter

After successful authentication