



Informatica® Data Integration - Free & PayGo

# Snowflake Data Cloud Connector

© Copyright Informatica LLC 2019, 2023

This software and documentation are provided only under a separate license agreement containing restrictions on use and disclosure. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica LLC.

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation is subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License.

Informatica, Informatica Cloud, Informatica Intelligent Cloud Services, PowerCenter, PowerExchange, and the Informatica logo are trademarks or registered trademarks of Informatica LLC in the United States and many jurisdictions throughout the world. A current list of Informatica trademarks is available on the web at <https://www.informatica.com/trademarks.html>. Other company and product names may be trade names or trademarks of their respective owners.

Portions of this software and/or documentation are subject to copyright held by third parties. Required third party notices are included with the product.

The information in this documentation is subject to change without notice. If you find any problems in this documentation, report them to us at [infa\\_documentation@informatica.com](mailto:infa_documentation@informatica.com).

Informatica products are warranted according to the terms and conditions of the agreements under which they are provided. INFORMATICA PROVIDES THE INFORMATION IN THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Publication Date: 2023-04-04

# Table of Contents

<b>Preface .....</b>	<b>6</b>
Informatica Resources. ....	6
Informatica Documentation. ....	6
Informatica Intelligent Cloud Services web site. ....	6
Informatica Intelligent Cloud Services Communities. ....	6
Informatica Intelligent Cloud Services Marketplace. ....	6
Data Integration connector documentation. ....	7
Informatica Knowledge Base. ....	7
Informatica Intelligent Cloud Services Trust Center. ....	7
Informatica Global Customer Support. ....	7
 <b>Chapter 1: Introduction to Snowflake Data Cloud Connector.....</b>	<b>8</b>
Snowflake Data Cloud Connector assets. ....	8
 <b>Chapter 2: Getting started with Snowflake Data Cloud Connector .....</b>	<b>9</b>
Verify permissions. ....	9
Prepare for authentication. ....	10
Configure OAuth 2.0 authorization code authentication. ....	10
Configure key pair authentication. ....	10
Configure proxy settings. ....	11
Configure private connectivity to Snowflake. ....	11
 <b>Chapter 3: Connections for Snowflake Data Cloud.....</b>	<b>12</b>
Snowflake Data Cloud connection properties. ....	12
Standard authentication. ....	12
OAuth 2.0 authorization code authentication. ....	13
Key pair authentication. ....	15
Additional JDBC connection parameters. ....	16
 <b>Chapter 4: Mappings for Snowflake Data Cloud.....</b>	<b>18</b>
Snowflake Data Cloud transformations. ....	18
SQL transformation. ....	19
Handling dynamic schemas. ....	20
Rules and guidelines for dynamic schema handling. ....	21
Mapping example. ....	21
 <b>Chapter 5: Sources for Snowflake Data Cloud.....</b>	<b>24</b>
Source properties for Snowflake Data Cloud. ....	24
Source objects and operations. ....	26
Overriding SQL. ....	27

<b>Chapter 6: Targets for Snowflake Data Cloud.....</b>	<b>28</b>
Target properties for Snowflake Data Cloud. . . . .	28
Target objects and operations. . . . .	30
Specify a target. . . . .	31
Override the update operation. . . . .	32
Optimize the .csv file size. . . . .	32
Configure load properties in mappings. . . . .	33
Capturing changed data from CDC sources. . . . .	34
Configuring a mapping task to read from a CDC source. . . . .	35
Step 1. Configure the source. . . . .	36
Step 2. Configure the target. . . . .	36
Step 3. Configure the mapping task. . . . .	36
Viewing job statistics. . . . .	37
Rules and guidelines for Snowflake Data Cloud target transformations. . . . .	38
 <b>Chapter 7: Lookups for Snowflake Data Cloud .....</b>	 <b>39</b>
Lookup properties for Snowflake Data Cloud. . . . .	39
Parameterization. . . . .	40
Multiple match restrictions. . . . .	41
Enable lookup caching. . . . .	41
Log uncached lookup queries. . . . .	41
 <b>Chapter 8: Pushdown optimization.....</b>	 <b>43</b>
Pushdown optimization types. . . . .	43
Pushdown optimization scenarios. . . . .	44
Prepare for pushdown optimization using a Snowflake Data Cloud connection. . . . .	44
Access data files in a Google Cloud Storage bucket. . . . .	45
Access data files in a Microsoft Azure Data Lake Storage Gen2 container. . . . .	45
Previewing pushdown optimization. . . . .	47
Configuring pushdown optimization. . . . .	47
Context based optimization for multiple targets. . . . .	47
Understanding an SCD type 2 merge mapping. . . . .	48
Pushing logic to a database with different schemas. . . . .	49
Configuring cross-schema pushdown optimization. . . . .	49
Pushing logic to different databases. . . . .	49
Configuring cross-database pushdown optimization. . . . .	50
Cancelling a pushdown optimization task. . . . .	50
Clean stop a pushdown optimization job. . . . .	51
 <b>Chapter 9: Pushdown optimization using a Snowflake Data Cloud connection.....</b>	 <b>52</b>
Pushdown compatibility for Snowflake Data Cloud connection. . . . .	53

Functions with Snowflake Data Cloud. . . . .	53
Operators with Snowflake Data Cloud. . . . .	54
Variables with Snowflake Data Cloud. . . . .	55
Transformations with Snowflake Data Cloud. . . . .	55
Aggregator transformation. . . . .	55
Expression transformation. . . . .	56
Lookup transformation. . . . .	56
Router transformation. . . . .	57
Sequence Generator transformation. . . . .	57
SQL transformation. . . . .	58
Union transformation. . . . .	59
Update Strategy transformation. . . . .	59
Features. . . . .	59
Snowflake Data Cloud sources, targets, and lookups. . . . .	59
Amazon S3 V2 source. . . . .	62
Google Cloud Storage V2 source. . . . .	63
Microsoft Azure Data Lake Storage Gen2 source. . . . .	63
Creating temporary view for source overrides. . . . .	64
Configuring target copy command options. . . . .	64
Verify the pushdown query in the session log. . . . .	64
<b>Appendix A: Data type reference. . . . .</b>	<b>66</b>
Snowflake Data Cloud and transformation data types. . . . .	66
Semi-structured data types and transformation data types. . . . .	67
Rules and guidelines for data types. . . . .	68
<b>Appendix B: Additional runtime configurations. . . . .</b>	<b>69</b>
Configure JVM memory requirements. . . . .	69
Configure bulk processing. . . . .	69
Configure logging properties. . . . .	70
Configure the temp directory for a mapping. . . . .	70
Optimize the staging performance for a mapping. . . . .	71
<b>Index. . . . .</b>	<b>73</b>

# Preface

Use *Snowflake Data Cloud Connector* to learn how to read from or write to Snowflake. Learn to create a connection, develop and run mappings, mapping tasks, and data transfer tasks in Data Integration. Learn how to push down the transformation logic for processing to the Snowflake database.

## Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

### Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit <https://docs.informatica.com>.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at [infa\\_documentation@informatica.com](mailto:infa_documentation@informatica.com).

### Informatica Intelligent Cloud Services web site

You can access the Informatica Intelligent Cloud Services web site at <http://www.informatica.com/cloud>. This site contains information about Informatica Cloud integration services.

### Informatica Intelligent Cloud Services Communities

Use the Informatica Intelligent Cloud Services Community to discuss and resolve technical issues. You can also find technical tips, documentation updates, and answers to frequently asked questions.

Access the Informatica Intelligent Cloud Services Community at:

<https://network.informatica.com/community/informatica-network/products/cloud-integration>

Developers can learn more and share tips at the Cloud Developer community:

<https://network.informatica.com/community/informatica-network/products/cloud-integration/cloud-developers>

### Informatica Intelligent Cloud Services Marketplace

Visit the Informatica Marketplace to try and buy Data Integration Connectors, templates, and mapplets:

<https://marketplace.informatica.com/>

## Data Integration connector documentation

You can access documentation for Data Integration Connectors at the Documentation Portal. To explore the Documentation Portal, visit <https://docs.informatica.com>.

## Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit <https://search.informatica.com>. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at [KB\\_Feedback@informatica.com](mailto:KB_Feedback@informatica.com).

## Informatica Intelligent Cloud Services Trust Center

The Informatica Intelligent Cloud Services Trust Center provides information about Informatica security policies and real-time system availability.

You can access the trust center at <https://www.informatica.com/trust-center.html>.

Subscribe to the Informatica Intelligent Cloud Services Trust Center to receive upgrade, maintenance, and incident notifications. The [Informatica Intelligent Cloud Services Status](#) page displays the production status of all the Informatica cloud products. All maintenance updates are posted to this page, and during an outage, it will have the most current information. To ensure you are notified of updates and outages, you can subscribe to receive updates for a single component or all Informatica Intelligent Cloud Services components. Subscribing to all components is the best way to be certain you never miss an update.

To subscribe, go to <https://status.informatica.com/> and click **SUBSCRIBE TO UPDATES**. You can then choose to receive notifications sent as emails, SMS text messages, webhooks, RSS feeds, or any combination of the four.

## Informatica Global Customer Support

You can contact a Customer Support Center by telephone or online.

For online support, click **Submit Support Request** in Informatica Intelligent Cloud Services. You can also use Online Support to log a case. Online Support requires a login. You can request a login at <https://network.informatica.com/welcome>.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at <https://www.informatica.com/services-and-training/support-services/contact-us.html>.

## CHAPTER 1

# Introduction to Snowflake Data Cloud Connector

You can use Snowflake Data Cloud Connector to securely read data from or write data to Snowflake tables and views. You can also read from Snowflake external tables, hybrid tables, and materialized views.

Additionally, you can read data from other applications, databases, and flat files and write data to Snowflake. You can also use Snowflake Data Cloud Connector to read data from and write data to Snowflake that is enabled for staging data in Azure, Amazon, Google Cloud Platform, or Snowflake GovCloud.

When you use Snowflake Data Cloud Connector, you can create a Snowflake Data Cloud connection and use the connection in Data Integration mappings and tasks.

When you run a Snowflake Data Cloud mapping or task, the Secure Agent writes data to Snowflake based on the workflow and Snowflake Data Cloud connection configuration.

## Snowflake Data Cloud Connector assets

Create assets in Data Integration to integrate data using Snowflake Data Cloud Connector.

When you use Snowflake Data Cloud Connector, you can include the following Data Integration assets:

- Data transfer task
- Mapping
- Mapping task

For more information about configuring assets and transformations, see *Mappings*, *Transformations*, and *Tasks* in the Data Integration documentation.



## CHAPTER 2

# Getting started with Snowflake Data Cloud Connector

Before you create a connection and run assets with Snowflake Data Cloud Connector, complete the prerequisites.

Depending on your requirement, perform the following tasks:

- Verify if you have the applicable permissions to perform operations in Snowflake.
- Get the authorization details based on the type of the authentication that you want to configure in the connection.
- If your organization uses an outgoing proxy server to connect to the internet, configure the Secure Agent to use the proxy server.
- Ensure that ports 443 and 80 in Snowflake are open for access before you connect with the Snowflake Data Cloud connection.

## Verify permissions

Permissions define the level of access for the operations that you can perform in Snowflake.

The following table lists the permissions that you require in the Snowflake account:

Privileges	Object Type
Database	Usage
Schema	Usage, Create Table, Create View, Create Procedure, Create Sequence
Table	All
Sequence	All
Stored Procedure	All

For more information, see [Access Control Privileges](#) in the Snowflake documentation.

# Prepare for authentication

You can configure standard and KeyPair authentication types to access Snowflake.

Before you configure authentication, complete the prerequisites. Get the account name, warehouse, and role details from the Snowflake account.

Keep the following details handy based on the authentication type that you want to use:

- **Standard authentication.** Requires the Snowflake account user name and password.
- **KeyPair authentication.** Requires the public and private key pair, along with the Snowflake account name.
- **Authorization code authentication.** Requires at a minimum the client ID, authorization URL, access token URL, and the access token. You must first create an authorization integration in Snowflake and then get the authorization details.

When you use authorization code authentication, you can also use external OAuth such as PingFederate.

For detailed authentication configurations, see the Snowflake documentation.

## Configure OAuth 2.0 authorization code authentication

To use authorization code, you must first register the Informatica redirect URL in Security Integration. Security Integration is a type of integration that enables clients that support OAuth to redirect users to an authorization page and generate access tokens, and optionally, refresh tokens to access Snowflake.

Register the Informatica redirect URL in Security Integration:

```
https://<Informatica cloud hosting facility for your organization>/ma/proxy/oauthcallback
```

If the access token expires, Informatica redirect URL, which is outside the customer firewall, tries to connect to the endpoint and retrieves a new access token.

For more information about how to get the authorization details, see the Snowflake documentation.

## Configure key pair authentication

Before you use key pair authentication, generate the public and private key pair using OpenSSL. The key pair authentication method requires a 2048-bit RSA key pair. Specify the path to the private key file and password in the connection properties to access Snowflake.

### Configuring key pair authentication for mappings

Before you use key pair authentication to connect to Snowflake from Cloud Data Integration, you must have the SecurityAdmin role or higher, and then perform the following tasks:

1. From the OpenSSL command line, generate a private key:
  - To generate an unencrypted private key, run the following command from the command line, and provide a passphrase when prompted:

```
$ openssl genrsa 2048 | openssl pkcs8 -topk8 -inform PEM -out rsa_key.p8 -nocrypt
```
  - To generate an encrypted private key, run the following command from the command line, and provide a passphrase when prompted:

```
$ openssl genrsa 2048 | openssl pkcs8 -topk8 -inform PEM -out rsa_key.p8
```

The passphrase is used to encrypt the private key file while connecting to Snowflake.

2. Generate the public key. Run the following command and specify the encrypted private key located in file, for example, `rsa_key.p8`:

```
openssl rsa -in rsa_key.p8 -pubout -out rsa_key.pub
```

3. Copy the public and private key files in a directory that the Secure Agent can access. For example, `C:\Program Files\Informatica Cloud Secure Agent\apps\Data_Integration_Server\data\snowflake\rsa_key.p8`

You require the path details when you configure the Snowflake connection.

4. In Snowflake, assign the public key to the Snowflake user using the ALTER USER command:

```
alter user <user> set rsa_public_key='<content of the public key after removing the header and footer lines>';
```

For example, `alter user jsmith set rsa_public_key='MIIXBIjABCdef...';`

For more information about configuring a key pair authentication for Snowflake, see the Snowflake documentation.

## Configure proxy settings

If your organization uses an outgoing proxy server to connect to the internet, the agent connects to Informatica Intelligent Cloud Services through the proxy server.

You can configure the Secure Agent to use the proxy server on Windows and Linux. You can use the unauthenticated or authenticated proxy server.

To configure the proxy settings for the Secure Agent, use one of the following methods:

- Configure the Secure Agent through the Secure Agent Manager on Windows or shell command on Linux. For instructions, see the topic "Configure the proxy settings on Windows" or "Configure the proxy settings on Linux," in *Getting Started* in the Data Integration documentation.
- Configure the JVM options for the DTM in the Secure Agent properties. For instructions, see the Knowledge Base article [000185646](#).
- Configure the proxy server properties in the additional JDBC URL parameters in the Snowflake connection. For more information, see ["Additional JDBC connection parameters" on page 16](#).

Contact your network administrator for the correct proxy settings.

## Configure private connectivity to Snowflake

You can access Snowflake using AWS or Azure Private Link endpoints.

The AWS or Azure Private Link setup ensures that the connection to Snowflake uses the AWS or Azure internal network and does not take place over the public Internet.

To connect to the Snowflake account over the private AWS network, see [AWS Private Link and Snowflake](#).

To connect to the Snowflake account over the private Azure network, see [Azure Private Link and Snowflake](#).

## CHAPTER 3

# Connections for Snowflake Data Cloud

Create a Snowflake Data Cloud connection to securely read data from or write data to Snowflake. You can use a Snowflake Data Cloud connection to specify sources, targets, and lookups in mappings and mapping tasks. You can also use the Snowflake connection in an SQL transformation.

## Snowflake Data Cloud connection properties

When you set up a Snowflake Data Cloud connection, configure the connection properties.

You can use the following authentication methods to connect to Snowflake:

- **Standard.** Uses Snowflake account user name and password credentials to connect to Snowflake.
- **Authorization Code.** Uses the OAuth 2.0 protocol with Authorization Code grant type to connect to Snowflake. Authorization Code allows authorized access to Snowflake without sharing or storing your login credentials.
- **KeyPair.** Uses the private key file and private key file password, along with the existing Snowflake account user name to connect to Snowflake.

You create a Snowflake Data Cloud connection on the Connections page. You can then use the connection when you read from or write data to Snowflake.

## Standard authentication

When you set up a Snowflake Data Cloud connection, configure the connection properties.

The following table describes the Snowflake Data Cloud connection properties for the Standard authentication mode:

Property	Description
Connection Name	Name of the connection. Each connection name must be unique within the organization. Connection names can contain alphanumeric characters, spaces, and the following special characters: _ . + -, Maximum length is 255 characters.
Description	Description of the connection. Maximum length is 4000 characters.

Property	Description
Type	The Snowflake Data Cloud connection type.
Runtime Environment	The name of the runtime environment where you want to run the tasks. You can specify a Secure Agent or a Hosted Agent.
Authentication	The authentication method that the connector must use to log in to Snowflake. Select <b>Standard</b> . Default is <b>Standard</b> .
Username	The user name to connect to the Snowflake account.
Password	The password to connect to the Snowflake account.
Account	<p>The name of the Snowflake account.</p> <p>For example, if the Snowflake URL is <code>https://&lt;123abc&gt;.us-east-2.aws.snowflakecomputing.com/console/login#/,</code> your account name is the first segment in the URL before <code>snowflakecomputing.com</code>. Here, <code>123abc.us-east-2.aws</code> is your account name.</p> <p>If you use the Snowsight URL, for example, <code>https://app.snowflake.com/us-east-2.aws/&lt;123abc&gt;/dashboard,</code> your account name is <code>123abc.us-east-2.aws</code></p> <p><b>Note:</b> Ensure that the account name doesn't contain underscores. To use an alias name, contact Snowflake Customer Support.</p>
Warehouse	The Snowflake warehouse name.
Role	The Snowflake role assigned to the user.
Additional JDBC URL Parameters	<p>The additional JDBC connection parameters.</p> <p>Enter one or more JDBC connection parameters in the following format:</p> <pre>&lt;param1&gt;=&lt;value&gt;&amp;&lt;param2&gt;=&lt;value&gt;&amp;&lt;param3&gt;=&lt;value&gt;...</pre> <p>For example:</p> <pre>user=jon&amp;warehouse=mywh&amp;db=mydb&amp;schema=public</pre> <p><b>Important:</b> Ensure that there is no space before and after the equal sign (=) when you add the parameters.</p>

## OAuth 2.0 authorization code authentication

The following table describes the Snowflake Data Cloud connection properties for an OAuth 2.0 - AuthorizationCode type connection:

Property	Description
Connection Name	<p>Name of the connection.</p> <p>Each connection name must be unique within the organization. Connection names can contain alphanumeric characters, spaces, and the following special characters: <code>_ . + -</code>, Maximum length is 255 characters.</p>
Description	Description of the connection. Maximum length is 4000 characters.
Type	The Snowflake Data Cloud connection type.

Property	Description
Runtime Environment	The name of the runtime environment where you want to run the tasks. You can specify a Secure Agent or a Hosted Agent.
Authentication	The authentication method that Snowflake Data Cloud Connector must use to log in to Snowflake. Select <b>AuthorizationCode</b> .
Account	The name of the Snowflake account. For example, if the Snowflake URL is <code>https://&lt;123abc&gt;.us-east-2.aws.snowflakecomputing.com/console/login#</code> , your account name is the first segment in the URL before <code>snowflakecomputing.com</code> . Here, <code>123abc.us-east-2.aws</code> is your account name. If you use the Snowsight URL, for example, <code>https://app.snowflake.com/us-east-2.aws/&lt;123abc&gt;/dashboard</code> , your account name is <code>123abc.us-east-2.aws</code> <b>Note:</b> Ensure that the account name doesn't contain underscores. To use an alias name, contact Snowflake Customer Support.
Warehouse	The Snowflake warehouse name.
Additional JDBC URL Parameters	The additional JDBC connection parameters. Enter one or more JDBC connection parameters in the following format:  <code>&lt;param1&gt;=&lt;value&gt;&amp;&lt;param2&gt;=&lt;value&gt;&amp;&lt;param3&gt;=&lt;value&gt;...</code> For example: <code>user=jon&amp;warehouse=mywh&amp;db=mydb&amp;schema=public</code> <b>Important:</b> Ensure that there is no space before and after the equal sign (=) when you add the parameters.
Authorization URL	The Snowflake server endpoint that is used to authorize the user request. The authorization URL is <code>https://&lt;account name&gt;.snowflakecomputing.com/oauth/authorize</code> , where <code>&lt;account name&gt;</code> specifies the full name of your account provided by Snowflake. For example, <code>https://&lt;abc&gt;.snowflakecomputing.com/oauth/authorize</code> <b>Note:</b> If the account name contains underscores, use the alias name. You can also use the Authorization Code grant type that supports the authorization server in a Virtual Private Cloud network.
Access Token URL	The Snowflake access token endpoint that is used to exchange the authorization code for an access token. The access token URL is <code>https://&lt;account name&gt;.snowflakecomputing.com/oauth/token-request</code> , where <code>&lt;account name&gt;</code> specifies the full name of your account provided by Snowflake. For example, <code>https://&lt;abc&gt;.snowflakecomputing.com/oauth/token-request</code> <b>Note:</b> If the account name contains underscores, use the alias name.
Client ID	Client ID of your application that Snowflake provides during the registration process.
Client Secret	Client secret of your application.
Scope	Determines the access control if the API endpoint has defined custom scopes. Enter space-separated scope attributes. For example, specify <code>session:role:CQA_GCP</code> as the scope to override the value of the default user role. The value must be one of the roles assigned in Security Integration.

Property	Description
Access Token Parameters	<p>Additional parameters to use with the access token URL.</p> <p>Define the parameters in the JSON format.</p> <p>For example, define the following parameters:</p> <pre>[{"Name": "code_verifier", "Value": "5PMddu6Zcg6Tc4sbg"}]</pre>
Authorization Code Parameters	<p>Additional parameters to use with the authorization token URL.</p> <p>Define the parameters in the JSON format.</p> <p>For example, define the following parameters:</p> <pre>[{"Name": "code_challenge", "Value": "Ikr-vv52th0UeVRi4"}, {"Name": "code_challenge_method", "Value": "S256"}]</pre>
Access Token	<p>The access token value.</p> <p>Enter the populated access token value, or click <b>Generate Token</b> to populate the access token value.</p>
Generate Token	Generates the access token and refresh token based on the OAuth attributes you specified.
Refresh Token	<p>The refresh token value.</p> <p>Enter the populated refresh token value, or click <b>Generate Token</b> to populate the refresh token value. If the access token is not valid or expires, the agent fetches a new access token with the help of the refresh token.</p> <p><b>Note:</b> If the refresh token expires, provide a valid refresh token or regenerate a new refresh token by clicking <b>Generate Token</b>.</p>

## Key pair authentication

The following table describes the Snowflake Data Cloud connection properties for the KeyPair authentication type connection:

Connection property	Description
Connection Name	<p>Name of the connection.</p> <p>Each connection name must be unique within the organization. Connection names can contain alphanumeric characters, spaces, and the following special characters: _ . + - ,</p> <p>Maximum length is 255 characters.</p>
Description	Description of the connection. Maximum length is 4000 characters.
Type	The Snowflake Data Cloud connection type.
Runtime Environment	<p>The name of the runtime environment where you want to run the tasks.</p> <p>You can specify a Secure Agent or a Hosted Agent.</p>
Authentication	<p>The authentication method to log in to Snowflake.</p> <p>Select <b>KeyPair</b>.</p>
Username	The user name to connect to the Snowflake account.

Connection property	Description
Account	<p>The name of the Snowflake account.</p> <p>For example, if the Snowflake URL is <code>https://&lt;123abc&gt;.us-east-2.aws.snowflakecomputing.com/console/login#/,</code> your account name is the first segment in the URL before <code>snowflakecomputing.com</code>. Here, <code>123abc.us-east-2.aws</code> is your account name.</p> <p>If you use the Snowsight URL, for example, <code>https://app.snowflake.com/us-east-2.aws/&lt;123abc&gt;/dashboard,</code> your account name is <code>123abc.us-east-2.aws</code>.</p> <p><b>Note:</b> Ensure that the account name doesn't contain underscores. To use an alias name, contact Snowflake Customer Support.</p>
Warehouse	The Snowflake warehouse name.
Additional JDBC URL Parameters	<p>Optional. The additional JDBC connection parameters.</p> <p>Enter one or more JDBC connection parameters in the following format:</p> <pre>&lt;param1&gt;=&lt;value&gt;&amp;&lt;param2&gt;=&lt;value&gt;&amp;&lt;param3&gt;=&lt;value&gt;...</pre> <p>For example:</p> <pre>user=jon&amp;warehouse=mywh&amp;db=mydb&amp;schema=public</pre> <p><b>Important:</b> Ensure that there is no space before and after the equal sign (=) when you add the parameters.</p>
Private Key File	<p>Path to the private key file, including the private key file name, that the Secure Agent uses to access Snowflake.</p> <p><b>Note:</b> Verify that the keystore is FIPS-certified.</p>
Private Key Password	Password for the private key file.

## Additional JDBC connection parameters

You can configure additional options in the JDBC URL parameters field in the Snowflake Data Cloud connection.

You can configure the following properties as additional JDBC URL parameters:

- To override the database and schema name used to create temporary tables in Snowflake, enter the database and schema name in the following format:

```
ProcessConnDB=<DB name>&ProcessConnSchema=<schema_name>
```

- To view only the specified database and schema while importing a Snowflake table, enter the database and schema name in the following format:

```
db=<database_name>&schema=<schema_name>
```

- To read UDF string and numeric data from Snowflake, enter the database and schema where the UDF is created in Snowflake in the following format:

```
db=<database_name>&schema=<schema_name>
```

- To access Snowflake through Okta SSO authentication, enter the web-based IdP implementing SAML 2.0 protocol in the following format:

```
authenticator=https://<Your_Okta_Account_Name>.okta.com
```



**Note:** Microsoft ADFS is not applicable.

For more information about configuring Okta authentication, see [Configuring Snowflake to use federated authentication](#).

- To load data from Google Cloud Storage or Microsoft Azure Data Lake Storage Gen2 to Snowflake for pushdown optimization, enter the Cloud Storage Integration name created for these in Snowflake in the following format:

```
storage_integration=<Storage Integration name>
```

The storage integration name is case-sensitive. For example, if the storage integration name you created for Google Cloud Storage or Microsoft Azure Data Lake Storage Gen2 in Snowflake is *STORAGE\_INT*, you must specify the same integration name:

```
storage_integration=STORAGE_INT
```

- To connect to Snowflake using the proxy server, enter the following parameters:

```
useProxy=true&  
proxyHost=<proxy host IP address>&  
proxyPort=<proxy server port number>&  
proxyUser=<proxy server user name>&  
proxyPassword=<proxy server password>
```

- To ignore double quotes in the table and treat all tables as case-insensitive, enter the following parameter:

```
QUOTED_IDENTIFIERS_IGNORE_CASE=true
```

When you set this property in the connection to true, Snowflake ignores the double quotes in the table and treats all tables as case-insensitive. After you set this property to true, you cannot query existing case-sensitive tables using the same connection as Snowflake ignores the double quotes in the query. You must create a new connection to fetch any existing case-sensitive tables.

**Note:** This property is not applicable when you create a new target at run time.

## CHAPTER 4

# Mappings for Snowflake Data Cloud

When you configure a mapping, you describe the flow of data from the source to the target.

A mapping defines reusable data flow logic that you can use in mapping tasks.

When you create a mapping, you define the Source, Target, and Lookup transformations to represent a Snowflake Data Cloud object. Use the Mapping Designer in Data Integration to add the Source, Target, or Lookup transformations in the mapping canvas and configure the Snowflake Data Cloud source, target, and lookup properties.

You can use Monitor to monitor the jobs.

## Snowflake Data Cloud transformations

Add transformations to the mapping to represent the operations that you want to perform on data.

You can add the following transformations and define the transformation details in a mapping:

### **Source transformation**

Add a Source transformation to read data from Snowflake or other data sources. You can read from a single or from multiple Snowflake tables. You can also use a query or parameter as the source type to read from Snowflake.

You can use one or more Source transformations in a mapping. When you use a single transformation to read from multiple tables, you can use a join. If you use two Source transformations in a mapping, use a Joiner transformation to join the data.

### **Target transformation**

Add a Target transformation to write data to Snowflake. You can use one or more Target transformations in a mapping. When you configure a target transformation, you can use a single Snowflake target object. You can use a new or an existing Snowflake target object. You can also use parameters to define the target connection and target object when you run the mapping task.

### **Lookup transformation**

Add a Lookup transformation to retrieve data based on a specified lookup condition. The lookup object can be a single object or query. You can also parameterize the lookup objects and connections.

For more information about the objects and the properties that you can configure for each of these transformations in a mapping, see the [Chapter 5, "Sources for Snowflake Data Cloud" on page 24](#), [Chapter 6,](#)

[“Targets for Snowflake Data Cloud” on page 28](#), and [Chapter 7, “Lookups for Snowflake Data Cloud ” on page 39](#) chapters.

You can also add other transformations to the mapping to transform the data. For more information about configuring transformations, see *Transformations* in the Data Integration documentation.

For more information about configuring transformations, see *Transformations* in the Data Integration documentation.

## SQL transformation

You can configure an SQL transformation in a Snowflake Data Cloud mapping to process SQL queries and stored procedures in Snowflake.

When you add an SQL transformation to the mapping, on the **SQL** tab, you define the database connection and the type of SQL that the transformation processes.

You can choose to use a parameterized connection in a SQL transformation. You can also override the values defined in a parameter file at runtime. To use a parameterized connection in an SQL transformation, first create an SQL transformation in a mapping that uses a valid connection. Then, parameterize the connection in the SQL transformation. You can also use an SQL transformation to read from Java or SQL user-defined functions (UDF) in Snowflake.

You can use an SQL transformation to process the following types of SQL statements:

### Stored procedure

You can configure an SQL transformation to call a stored procedure in Snowflake. The stored procedure name is case-sensitive. You can select the stored procedure from the database, or enter the exact name of the stored procedure to call in the SQL transformation. The stored procedure must exist in the Snowflake database before you create the SQL transformation.

When you specify the Snowflake database, schema, and procedure name in the advanced SQL properties, the agent considers the properties specified in the stored procedure first, followed by the advanced source properties, then the additional JDBC URL parameters in the connection, and finally the source object metadata.

If you add a new stored procedure to the database when the mapping is open, the new stored procedure does not appear in the list of available stored procedures. To refresh the list, close and reopen the mapping.

### SQL Query

You can configure an SQL transformation to process an entered query that you define in the SQL editor. Do not use more than one SQL query in an SQL transformation.

The SQL transformation processes the query and returns the rows. The SQL transformation also returns any errors that occur from the underlying database or if there is an error in the user syntax.

For more information about SQL queries and stored procedures, see *Transformations* in the Data Integration documentation.

### Rules and guidelines

You can use a SQL transformation with the following restrictions:

- You cannot use the saved query type in an SQL transformation.

- Mappings fail when you specify schema override results in multiple stored procedures that contain the same name and number of arguments.
- Mappings fail when the stored procedure contains Unicode characters.
- If NULL is returned from a stored procedure, a warning appears in the user interface and the session log. However, the mapping continues to process the rows.
- The runtime processing ignores the following properties in an SQL transformation:
  - In-out and input parameters
  - Advanced properties
  - Auto commit
  - Transformation scope
  - Stop on error

## Handling dynamic schemas

When you add a mapping to a mapping task, you can choose how Data Integration handles changes in the data object schemas. To refresh the schema every time the task runs, you can enable dynamic schema handling in the task.

A schema change includes one or more of the following changes to the data object:

- Fields added, deleted, or renamed.
- Fields updated for data type, precision, or scale.

Configure schema change handling on the **Schedule** page when you configure the task. You can configure asynchronous or dynamic schema change handling.

When you configure dynamic schema change handling, you can choose from the following options to refresh the schema:

### **Alter and apply changes**

Data Integration applies the following changes from the source schema to the target schema:

- New fields. Alters the target schema and adds the new fields from the source.
- Renamed fields. Adds renamed fields as new columns in the target.
- Data type and precision updates. Applies these changes to the target. Updates to the scale are not applicable.
- Deleted fields. Ignores deleted fields.

### **Don't apply DDL changes**

Data Integration does not apply the schema changes to the target.

### **Drop current and recreate**

Drops the existing target table and then recreates the target table at runtime using all the incoming metadata fields from the source.

## Rules and guidelines for dynamic schema handling

Consider the following rules and guidelines when you enable dynamic schema change handling:

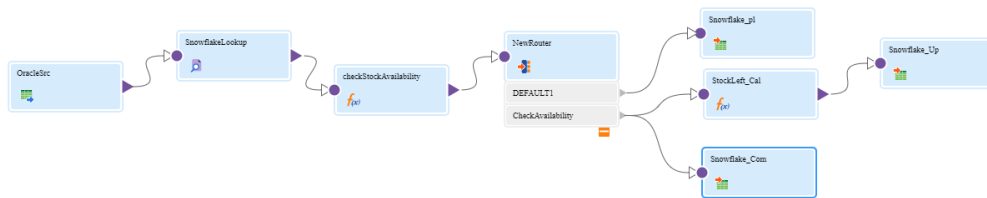
- Do not include an override for the target or source from the advanced properties. If the mapping is enabled for dynamic schema handling and contains a source or target override, the mapping fails with the following error: `Exception: class java.lang.NullPointerException occurred in writeMetadata`. However, you can specify an SQL override in mappings enabled for dynamic schema handling.

## Mapping example

An enterprise application uses the Oracle database to store product transaction details. You want to check the availability of stocks based on completed and pending transactions from the source data. You want to integrate the available stocks and transaction details to Snowflake for further analysis.

### Mapping

The following image shows the Snowflake Data Cloud mapping for this use case:



You create a mapping to read the product transaction details from an Oracle source and apply a lookup condition on the `PRODUCTDET` table in Snowflake which stores details of product and its availability. Based on the availability and requirement, you write the transactions to the `PENDINGTRANSACTION` and `COMPLETEDTRANSACTION` tables in Snowflake and update the `INSTOCK` field in the `PRODUCTDET` table for the completed transactions.

You use the following transformations in the Snowflake Data Cloud mapping:

### Source transformation

To read the product transaction details from an Oracle source, include an Oracle connection in the Source transformation to connect to Oracle. The source object for the mapping task is an `OracleSrc` table from Oracle.

The following image shows the transaction details stored in the `OracleSrc` table that you want to read:

transactionID	CustomerID	productID	quantity	OrderPlacedOn
Tran511	CUST21	P45	100	2016-04-05
Tran512	CUST22	P46	200	2016-07-05
Tran513	CUST23	P47	20	2016-07-25
Tran514	CUST24	P47	100	2016-10-25
Tran515	CUST25	P45	1000	2016-12-02
Tran517	CUST27	P46	5000	2017-01-02
Tran516	CUST26	P48	60	2017-01-02
Tran518	CUST28	P49	60	2017-01-03
Tran519	CUST29	P50	700	2017-03-13
Tran520	CUST30	P47	750	2017-03-14

## Lookup transformation

The lookup object for the mapping task is PRODUCTDET table in Snowflake, which has details of the product and its availability. Apply the lookup condition on the PRODUCTDET table in Snowflake which stores details of product and its availability based on the product ID.

The following image shows the data stored in the PRODUCTDET table:

Data Preview			
Connection: snowflake_CQA		Object: PRODUCTDET	
PRODUCTID	INSTOCK	PRODUCTDET	PRICE
p45	900	2.5" 80GB IDE Laptop Har...	1968
p46	10000	Laptop Internal CD/DVD R...	1229
p47	5000	New HP ProBook 430 G3 ...	5289
p48	50	New HP ProBook 430 G3 ...	9594
p49	20	Dell Inspiron 15R N5110 B...	1699
p50	800	HP 15-be016TU 15.6-inch...	27490

## Expression transformation

The Expression transformation uses an expression to calculate the stock availability.

## Router transformation

The Router transformation filters data based on the availability of stocks and redirects completed transactions, pending transactions, and product details to the appropriate target tables.

## Target transformation

The mapping task has the following target objects to write the completed transactions, pending transactions, and product details:

### COMPLETEDTRANSACTION

The COMPLETEDTRANSACTION table includes the TRANSACTIONID, PRODUCTID, QUANTITY, ORDERPLACEDON, and ORDERCOMPLETEDON fields.

The following image shows the data stored in the COMPLETEDTRANSACTION table:

Connection: snowflake_CQA		Object: COMPLETEDTRANSACTION		
TRANSACTIONID	PRODUCTID	QUANTITY	ORDERPLACEDON	ORDERCOMPLETEDON
Tran511	P45	100	2016-04-05 00:00:00.0	2016-04-05 00:00:00.0
Tran512	P46	200	2016-07-05 00:00:00.0	2016-07-05 00:00:00.0
Tran513	P47	20	2016-07-25 00:00:00.0	2016-07-25 00:00:00.0
Tran514	P47	100	2016-10-25 00:00:00.0	2016-10-25 00:00:00.0
Tran517	P46	5000	2017-01-02 00:00:00.0	2017-01-02 00:00:00.0
Tran519	P50	700	2017-03-13 00:00:00.0	2017-03-13 00:00:00.0
Tran520	P47	750	2017-03-14 00:00:00.0	2017-03-14 00:00:00.0

### PENDINGTRANSACTION

The PENDINGTRANSACTION table includes the PRODUCTID, TRANSACTIONID, REQUIREDQUANTITY, and ORDERPLACEDON fields.

The following image shows the data stored in the PENDINGTRANSACTION table:

Connection: snowflake_CQA		Object: PENDINGTRANSACTION	
PRODUCTID	TRANSACTIONID	REQUIREDQUANTITY	ORDERPLACEDON
P45	Tran515	1000	2016-12-02 00:00:00.0
P48	Tran516	60	2017-01-02 00:00:00.0
P49	Tran518	60	2017-01-03 00:00:00.0

## PRODUCTDET

The PRODUCTDET table includes the PRODUCTID, INSTOCK, PRODUCTDET, and PRICE fields. Based on the completed transactions, the INSTOCK field is updated.

The following image shows the data stored in the PRODUCTDET table:

Connection: snowflake_CQA		Object: PRODUCTDET	
PRODUCTID	INSTOCK	PRODUCTDET	PRICE
P48	50	New HP ProBook 430 G3 ...	9594
P49	20	Dell Inspiron 15R N5110 B...	1699
P45	800	2.5" 80GB IDE Laptop Har...	1968
P46	4800	Laptop Internal CD/DVD R...	1229
P47	4130	New HP ProBook 430 G3 ...	5289
P50	100	HP 15-be016TU 15.6-inch...	27490

When you run the mapping, the agent reads the transaction details from source, fetches fields from the lookup, and based on the conditions applied write the available quantity and transaction details to the target tables.

## CHAPTER 5

# Sources for Snowflake Data Cloud

Add a Source transformation to extract data from a source. When you add a Source transformation to a mapping, you define the source connection, source objects, and source properties related to the Snowflake Data Cloud connection type.

## Source properties for Snowflake Data Cloud

In a mapping, you can configure a Source transformation to represent a Snowflake Data Cloud source.

The following table describes the Snowflake Data Cloud source properties that you can configure in a Source transformation:

Property	Description
Connection	<p>Name of the source connection.</p> <p>You can select an existing connection, create a new connection, or define parameter values for the source connection property.</p> <p><b>Note:</b> You can switch between a non-parameterized and a parameterized Snowflake Data Cloud connection. The advanced property values are retained during the switch.</p> <p>If you want to overwrite the source connection properties at runtime, select the <b>Allow parameter to be overridden at run time</b> option.</p> <p>Specify the parameter file directory and name in the advanced session properties.</p>
Source Type	<p>Type of the source object.</p> <p>Select Single Object, Multiple Objects, Query, or Parameter.</p>
Parameter	<p>A parameter file where you define values that you want to update without having to edit the task. Select an existing parameter for the source object, or click <b>New Parameter</b> to define a new parameter for the source object.</p> <p>The <b>Parameter</b> property appears only if you select parameter as the source type.</p> <p>If you want to overwrite the parameter at runtime, select the <b>Allow parameter to be overridden at run time</b> option when you create a parameter. When the task runs, it uses the parameters from the file that you specify in the task advanced session properties.</p>
Object	<p>The source object for the task.</p> <p>Select the source object for a single source. When you select the multiple source option, you can add multiple source objects and configure a relationship between them.</p>



Property	Description
Filter	Filters records based on the filter condition. You can specify a simple filter or an advanced filter.
Sort	Sorts records based on the conditions you specify. You can specify the following sort conditions: <ul style="list-style-type: none"> <li>- Not parameterized. Select the fields and type of sorting to use.</li> <li>- Parameterized. Use a parameter to specify the sort option.</li> </ul>

The following table describes the advanced properties that you can configure in a Source transformation:

Advanced Property	Description
Database	Overrides the database specified in the connection.
Schema	Overrides the schema specified in the connection.
Warehouse	Overrides the Snowflake warehouse name specified in the connection.
Role	Overrides the Snowflake role assigned to user you specified in the connection. The warehouse name in the mapping overrides the warehouse name you specify in the connection. Even though you provide an incorrect warehouse name in the connection properties, the connection is successful. However, before you run the mapping, ensure that you specify the correct warehouse name in the mapping properties.
Pre SQL	The pre-SQL command to run on the Snowflake source table before the agent reads the data. For example, if you want to update records in the database before you read the records from the table, specify a pre-SQL statement. The query must include a fully qualified table name. You can specify multiple pre-SQL commands, each separated with a semicolon.
Post SQL	The post-SQL command to run on the Snowflake table after the agent completes the read operation. For example, if you want to delete some records after the latest records are loaded, specify a post-SQL statement. The query must include a fully qualified table name. You can specify multiple post-SQL commands, each separated with a semicolon.
Table Name	Overrides the table name of the imported Snowflake source table.
SQL Override	The SQL statement to override the default query used to read data from the Snowflake source.
Tracing Level	Determines the amount of detail that appears in the log file. You can select Terse, Normal, Verbose Initialization, or Verbose Data. Default value is Normal.

# Source objects and operations

In a Source transformation, you can use a single object, multiple objects, query, or parameter as the source type to read data from Snowflake.

Some restrictions apply with certain source objects.

## Parameter source type

When you parameterize the source object and connection and enable the **Allow parameter to be overridden at run time** option in a transformation, you cannot override the object name using the fully qualified name such as `db.schema.tablename`.

You can pass the `db=<dbname>&schema<schemaname>` values in the **Additional JDBC URL Parameters** field in the Snowflake Data Cloud connection.

## Multiple source type

You can use a single Source transformation to read from multiple Snowflake tables within the same database. To read from multiple Snowflake sources, you can create multiple Source transformations and then use a Joiner transformation to join the sources.

To read from multiple tables using a single Source transformation, select multiple object as the source type and then configure a join to combine the tables. You can either add related objects with PK-FK relationships that are already defined or you can define a relationship condition to join the tables. To set your own conditions to define the relationship between the tables, select **Advanced Relationship** from the **Related Objects Actions** menu, and then define the relationship. When you configure a join expression, select the fields and define a join query syntax. You must specify only the condition and not the type of join in the query. The condition you specify in the text box for the expression is appended to the join condition.

When you specify a join condition in the advanced relationship to join the tables, you cannot override the database and schema names from the connection. You need to manually change the database and schema name in the advanced relationship condition. If the condition includes columns with a fully qualified name such as `db.schema.tablename`, do not configure an override. Delete the fully qualified database and schema names from the advanced relationship condition and then run the mapping.

## Restrictions for the multiple source type

A multiple source type has the following restrictions:

- A mapping fails when you read data from multiple tables joined using related objects and the tables and column names have case-sensitive columns.
- A mapping configured with a join for one or more tables that have the same column names fails.
- A mapping that reads from multiple Snowflake objects that do not belong to the same database and schema fails.
- A mapping configured with a join that reads from multiple tables fails if you specify an override to the table, schema, or database in the Snowflake source advanced properties.

## Query source type

When you use a custom SQL query to import Snowflake tables, specify the Snowflake database and schema name in the custom SQL query. If you do not specify the database and schema name, the agent considers the database and schema name specified in the connection properties. The table name in the query that reads from Snowflake must be a fully qualified. When you use a custom query to call a stored procedure, ensure that the role has access to the database and schema.

You can use a query source type with the following restrictions:

- Do not configure an override for the database and schema names in the connection.
- The following operations are not applicable for the query source type:
  - Filter and sort options.
  - Advanced properties, except for pre-SQL and post-SQL statements.

### General restrictions for the source

The Source transformation has the following general restrictions:

- You cannot read or write Snowflake table and column names that contain double quotes. The mapping fails with the following error: `SQL compilation error`
- You cannot use system variables in filters.

## Overriding SQL

When you override the object, database, schema, or role, you must follow some guidelines.

If you specify an SQL override query to override the custom query used for importing the metadata from Snowflake tables, specify a fully qualified table name.

When you specify both an SQL override and overrides to the database, schema, warehouse, or role from the advanced source properties, consider the following guidelines for the override combinations:

### Override the SQL and role

If you override the SQL and the role of a Snowflake object, verify that the override role has access to the table selected in the mapping.

To override the SQL and the role, you can perform one of the following tasks:

- Grant the overriding role with the same access permissions as the role used for the Snowflake object that you selected in the mapping.
- Specify an override to the table from the advanced properties. The specified override table is used in the design time and the SQL override takes precedence at runtime.

### Override the SQL, database, schema, and role

If you select a Snowflake object in a mapping and you specify both an SQL override and also override the database, schema, and role from the advanced properties, and the table does not exist in the overriding database, the mapping fails with a `table1 not found error`.

Specify a valid table name that corresponds to the overriding database and schema name.

### SQL override ports

The SQL override ports must be the same as the connected ports in the field mapping.

## CHAPTER 6

# Targets for Snowflake Data Cloud

Add a Target transformation to write data to a target. When you add a Target transformation to a mapping, you define the target connection, target objects, and target properties related to the Snowflake Data Cloud connection type.

## Target properties for Snowflake Data Cloud

The following table describes the Snowflake Data Cloud target properties that you can configure in a Target transformation:

Property	Description
Connection	<p>Name of the target connection.</p> <p>You can select an existing connection, create a new connection, or define parameter values for the target connection property.</p> <p><b>Note:</b> You can switch between a non-parameterized and a parameterized Snowflake Data Cloud connection. When you switch between the connections, the advanced property values are retained.</p> <p>To overwrite the target connection properties at runtime, select the <b>Allow parameter to be overridden at run time</b> option.</p>
Target Type	<p>Type of target object. Select <b>Single Object</b> or <b>Parameter</b>.</p>
Parameter	<p>A parameter file where you define values that you want to update without having to edit the task. Select an existing parameter for the target object or click <b>New Parameter</b> to define a new parameter for the target object.</p> <p>The <b>Parameter</b> property appears only if you select parameter as the target type.</p> <p>If you want to overwrite the target object at runtime, select the <b>Allow parameter to be overridden at run time</b> option. When the task runs, it uses the parameters from the file that you specify in the task advanced session properties.</p>
Object	<p>The target object for the task. Select the target object.</p> <p>You can either select an existing table or create a new table. You can write data by selecting an existing table or creating a new table in the target by using the <b>Create New at Runtime</b> option.</p>

Property	Description
Create New at Runtime	<p>Creates a Snowflake target table at runtime based on the table type and the path you specify. To create a target table at runtime, provide the following parameters:</p> <ul style="list-style-type: none"> <li>- Optional. Specify the table type as <code>table</code>.</li> <li>- In the <b>Path</b> field, specify the Snowflake database name and schema in the following format: <code>&lt;database name&gt;/&lt;schema&gt;</code></li> </ul> <p>The agent creates the target table based on the object name and the path you specify.</p> <p><b>Note:</b> You can edit the metadata of the source fields before creating the target.</p>
Use Exact Source Field Names in Target	<p>Applies to the <b>Create New at Runtime</b> option.</p> <p>Determines if you can create a target object at runtime with the exact source field names. Select to retain all the source field names in the target exactly as in the source, including any special characters. If you disable this option, special characters in the source are replaced with underscore characters in the target.</p> <p>Default is disabled.</p>
Operation	The target operation. Select Insert, Update, Upsert, Delete, or Data Driven.
Update columns	The temporary key column to update data to or delete data from a Snowflake target.
Data Driven Condition	Enables you to define expressions that flag rows for an insert, update, upsert, or delete operation.

The following table describes the advanced properties that you can configure in a Target transformation:

Advanced Property	Description
UpdateMode	<p>Loads data to the target based on the mode you specify.</p> <p>Applicable when you select the Update operation or the Data Driven operation.</p> <p>Select from one of the following modes:</p> <ul style="list-style-type: none"> <li>- Update As Update. Updates all rows flagged for update if the entries exist.</li> <li>- Update Else Insert. The agent first updates all rows flagged for update if the entries exist in the target. If the entries do not exist, the agent inserts the entries.</li> </ul>
Database	Overrides the database that you used to import the object.
Schema	Overrides the schema that you used to import the object.
Warehouse	<p>Overrides the Snowflake name specified in the connection.</p> <p>The warehouse name in the mapping overrides the warehouse name you specify in the connection.</p> <p>Even though you provide an incorrect warehouse name in the connection properties, the connection is successful. However, before you run the mapping, ensure that you specify the correct warehouse name in the mapping properties.</p>
Role	Overrides the Snowflake role assigned to the user specified in the connection.
Pre SQL	<p>The pre-SQL command to run before the agent writes to Snowflake.</p> <p>For example, if you want to assign sequence object to a primary key field of the target table before you write data to the table, specify a pre-SQL statement.</p> <p>You can specify multiple pre-SQL commands, each separated with a semicolon.</p>

Advanced Property	Description
Post SQL	The post-SQL command to run after the agent completes the write operation. For example, if you want to alter the table created by using create target option and assign constraints to the table before you write data to the table, specify a post-SQL statement. You can specify multiple post-SQL commands, each separated with a semicolon.
Truncate Target Table	Truncates the database target table before inserting new rows. Select one of the following options: - True. Truncates the target table before inserting all rows. - False. Inserts new rows without truncating the target table Default is false.
Additional Write Runtime Parameters	Specify additional runtime parameters. For example, if you want to specify the user-defined stage in the Snowflake database to upload the local staging files, specify the name of the stage location in the following format:  <code>remoteStage=REMOTE_STAGE</code>  If you want to optimize the write performance, you can choose to compress files before writing to Snowflake tables. You can set the compression parameter to On or Off, for example:  <code>Compression=On</code>  By default, compression is on. Separate multiple runtime parameters with &.
Table Name	Overrides the table name of the Snowflake target table.
Update Override	Overrides the default update query that the agent generates for the update operation with the update query.
Success File Directory	Not applicable.
Error File Directory	Not applicable.
Forward Rejected Rows	Determines whether the transformation passes rejected rows to the next transformation or drops rejected rows. By default, the agent forwards rejected rows to the next transformation.

## Target objects and operations

In a Target transformation, you can use a single object or parameter as the target type.

When you choose the target type, you can select the operation to insert, update, upsert, or delete data in a Snowflake target. You can also use the data driven operation to define expressions that flag rows for an insert, update, delete, or reject operation.

### Target operation restrictions

Some rules apply when you configure a target operation.

## Parameterized target

When you parameterize the target object and connection and enable the **Allow parameter to be overridden at run time** option in a transformation, you cannot override the object name using the fully qualified name such as `db.schema.tablename`. You must pass the `db=<dbname>&schema<schemaname>` values in the **Additional JDBC URL Parameters** field in the Snowflake Data Cloud connection.

## Update columns

You need to specify the temporary key column to update data to or delete data from a Snowflake target. If the Snowflake target does not include a primary key column, click **Add** to add a temporary key. You can select multiple columns.

If the records from the source tables contain duplicate primary keys, perform one of the following tasks in mappings to update or delete records in Snowflake:

- Before you import the target table, define multiple primary keys in the target table.
- Define more than one custom key for the target object using the Update Columns option in the advanced target properties.

## Specify a target

You can use an existing target or create a new target to write data to Snowflake. If you choose to create a new target, the agent creates the target when it runs the task.

Ensure that the path for the schema and database that you specify for the target object is in uppercase letters. Specify the Snowflake database name and schema in the following format: `<database name>/<schema name>` for the target object is in uppercase letters. If you do not enter the path, the Secure Agent considers the schema and database name you specified in the **Additional JDBC URL Parameters** field from the Snowflake Data Cloud connection properties.

Specify the TableType as `table`. The TableType property is optional.

The following image shows an example of a target object configured to create at runtime:

**Target Object**

Select an existing target object or create a new one. Any new target objects will be created when the mapping task is executed.

Target Object: ☐ Existing ☒ Create New at Runtime

Object Name:\*

TableType:

Path:

## Restrictions for a Snowflake target

The following rules apply when you configure the target:

- If the Secure Agent is installed on Windows, you cannot write data to a Snowflake target table when the table names contain the following special characters: `/\:*?"<>|`

- You can create a target at runtime with the following restrictions:
  - The table and column names are case sensitive. The Secure Agent adds double quotes to the table and column name while creating the target table at runtime. It is mandatory that you query the table name and column name in Snowflake using double quotes.
  - When you add an advanced filter condition for the source object, or when you define a relationship for multiple source objects in a Source transformation, if the condition contains the "/" separator to separate the database name, schema name, and table name, the mapping fails with an SQL compilation error. You must change the slash "/" separator to a dot (.) in the SQL query and run the mapping.
  - If the target name you specify contains special characters and you do not enable the target to use the exact source field names in the target, and the mapping also contains an Expression or Aggregator transformation, the target table is created with the special characters. The rest of fields that contain special characters that comes from the upstream transformations are replaced with underscore characters in the target.
  - When you parameterize the target object in a mapping and you create a target at runtime, the target object must have the full path in the parameter file. However, if you use an existing target object, the target object must have only the table name.

## Override the update operation

You can specify an update override to override the update query that the Secure Agent generates for the update operation.

When you configure an update override, the Secure Agent uses the query that you specify, stages the data in files, and then loads that data into a temporary table using the Snowflake's loader copy command. The data from the temporary table is then loaded to the Snowflake target table. The syntax that you specify for the update query must be supported by Snowflake.

Specify the update query in the following format:

```
UPDATE <Target table name> SET <Target table name>.<Column1> = :TU.<Column1>, <Target table name>.<Column2> = :TU.<Column2>, ... <Target table name>.<ColumnN> = :TU.<ColumnN>
FROM :TU WHERE <Target table name>.<Update Column1> = :TU.<Update Column1> AND <Target table name>.<Update Column2> = :TU.<Update Column2> AND ... <Target table name>.<Update ColumnN> = :TU.<Update ColumnN>
```

where, :TU. represents the incoming data source for the target port.

The Secure Agent replaces :TU. with a temporary table name while running the mapping and does not validate the update query.

When you configure an update override in a mapping to write to Snowflake, consider the following rules:

- Ensure that the column names for :TU matches the target table column names.
- Ensure that the column names are fully qualified names.
- Specify the update query with a valid SQL syntax because Snowflake Data Cloud Connector replaces :TU with a temporary table name and does not validate the update query.
- Do not change the order of the column in the mappings when you configure the update override option.
- The update query in the mapping must not contain unconnected fields to the target.

## Optimize the .csv file size

When you create a mapping to write to Snowflake, you can specify the size of the local staging .csv file in bytes. Specify the local staging file size property, csvFileSize, in the **Additional Write Runtime Parameters** field in the advanced Snowflake target properties. The default file size is 50 MB.



If the intended file size is 50 MB, calculate the csvFileSize value in bytes, for example 50\*1024\*1024 and then specify 52428800 as the csvFileSize. For optimal performance, the number and size of the CSV files must be in synchrony.

## Configure load properties in mappings

You can configure write properties to load data to Snowflake in the **Additional Write Runtime Parameters** field in the Snowflake Data Cloud advanced target properties of the Target transformation.

The following table lists some of the additional runtime parameters that you can specify to load data to Snowflake:

Property	Description
oneBatch	Process all data in a single batch. Type is Boolean. Default is false.
remoteStage	Specifies to use internal stage. External stage is not applicable. Type is String. Default is "~"(user stage).
onError	Specifies the action to perform when an error is encountered while loading data from a file. For example, <code>onError option ABORT_STATEMENT CONTINUE SKIP_FILE</code> Type is String. Default is CONTINUE.
compressFileByPut	Compress file by PUT. Type is Boolean. Default is false.
compressDataBeforePut	Compress data before PUT. The loader compresses the data to a gzip format before uploading the data. Type is Boolean. Default is true.
copyEmptyFieldAsEmpty	The COPY command option to set incoming empty fields as null. Type is Boolean.
enablePGZIP	Enables parallelism for the file compression. Type is Boolean. Default is true.
onError=ABORT_STATEMENT&oneBatch=true	Load the entire data in single batch and to stop the task if an error occurs. Simultaneously, validate the user-specified reject file path and write the error records to this file and to the session log. Type is onError - String or oneBatch - Boolean.

When you set the values in the additional runtime parameters field, every configured partition initializes a new loader instance and the configured values apply similarly across all the partitions.

### Example 1. Compress files

You want to compress files by using the Put command before loading data to Snowflake.

Specify the following compression option: `compressDataBeforePut=false&compressFileByPut=true`

If you specify both the options as true, Snowflake considers the `compressDataBeforePut` option.

### Example 2. Replace empty values as null

You want to replace the incoming fields with empty values as NULL while loading the data to Snowflake.

Specify the `copyEmptyFieldAsEmpty` Boolean option and set the value to true or false based on your requirement.

Consider the following scenarios before you configure the `copyEmptyFieldAsEmpty` Boolean parameter:

- If you do not configure this parameter, Null values are received as NULL, and empty values are received as Empty. This is the default behavior.
- If you set the parameter `copyEmptyFieldAsEmpty=false`, Null values are received as Null and empty values are received as Null.
- If you set the parameter `copyEmptyFieldAsEmpty=true`, Null values are received as empty, while empty values are received as empty.

### Example 3. Write data in a single batch

You want to write source data in a single batch to Snowflake. If an error is encountered while loading, you also want to capture the errors.

Specify the `onError=ABORT_STATEMENT&oneBatch=true` property based on your requirement.

While loading in a single batch, if an error occurs, the Secure Agent checks for the specified reject file name, runs the COPY command, validates the reject file, and then passes the file name to capture the errors, if any.

## Capturing changed data from CDC sources

You can use Snowflake Data Cloud Connector to write changed data from a CDC source such as Oracle CDC and Oracle CDC V2 and write the changed data to a Snowflake target.

When you configure a mapping, add the CDC sources and then run the associated mapping task to write the changed data to Snowflake. If you define a column as required in the Snowflake target table, map a column in the CDC source to the required column in the Snowflake target in the mapping before you run the task.

When the mapping task processes the changed data from a CDC source, Snowflake Data Cloud Connector creates a state table in Snowflake. When the changed data is received from the CDC source, Snowflake Data Cloud Connector uploads the changed data to the staging table. Then, it generates a `Job_Id` and writes the `Job_Id` to the state table along with the restart information.

The connector then merges the stage table with the actual target table in Snowflake. Each time you run the mapping task, Snowflake Data Cloud Connector creates the state table, if it does not exist, to store the state information.

Snowflake Data Cloud Connector uses the following naming convention for the tables:

- State table name: `<MappingTaskID>_<Instance(s)>`
- Staging table name: `<MappingTaskID>_<TargetInstanceName>`

## Restrictions

Consider the following restrictions when you capture changed data from CDC sources:

### Mapping

You can use a Snowflake target in a CDC mapping to write data from CDC sources. You cannot use a Snowflake source or lookup in a CDC mapping.

### staging optimization property

The optimization property that you set for staging data in the DTM of the Secure Agent is not applicable. If you run a mapping with both CDC and staging property enabled, the mapping runs successfully. However, staging optimization is disabled and you can view a message logged in the session logs.

### Recovery initialization error

When you run a CDC mapping to write data from a CDC source to a Snowflake target created at runtime, you might encounter the following error:

```
Error occurred while initializing CCI State Handler  
com.informatica.cci.runtime.internal.utils.impl.CExceptionImpl: Internal error: Recovery  
Init failed
```

To avoid this error, you must have the grant permissions to create a table in Snowflake.

# Configuring a mapping task to read from a CDC source

You can use Snowflake Data Cloud Connector to capture changed data from a CDC source and write the changed data to a Snowflake target.

Add the CDC source in the mapping, and then run the associated mapping task to write the changed data to the Snowflake target. You can also configure multiple pipelines in a single mapping to write the captured changed data to a Snowflake target.

When you configure a mapping to write changed data from a CDC source to Snowflake, you can configure the following advanced properties in the Snowflake Target transformation:

- Database
- Schema
- Warehouse
- Role
- Pre SQL
- Post SQL
- Truncate target table
- Table name
- Update override

## Step 1. Configure the source

Configure a Source transformation to read from a CDC source such as Oracle CDC and Oracle CDC V2.

1. In the Source transformation, specify a name and description in the general properties.
2. In the **Source** tab, select any configured CDC connection and specify the required source properties.

## Step 2. Configure the target

Configure a Target transformation to write changed data from a CDC source to Snowflake.

You can only configure a single Snowflake Data Cloud target transformation in a mapping to write changed data from a CDC source.

1. On the **Target** tab, perform the following steps to configure the target properties:
  - a. In the **Connection** field, select the Snowflake Data Cloud connection.
  - b. In the **Target Type** field, select the type of the target object.
  - c. In the **Object** field, select the required target object.
  - d. In the **Operation** field, select **Insert** or **Data Driven**.

**Note:** Update, upsert, and delete target operations are not applicable. Ensure that the target tables do not have a primary key defined in Snowflake.
  - e. If you select **Data Driven Condition**, specify the **DD\_INSERT** condition.

**Note:** Ensure that the target tables do not have a primary key defined in Snowflake.
  - f. Keep the **Update Column** field empty.

It is recommended that the source object contains a primary key.
  - g. Configure the applicable advanced target properties for the CDC mode.
2. On the **Field Mapping** tab, map the incoming fields to the target fields. You can manually map an incoming field to a target field or automatically map fields based on the field names.

If you define a column as required in the Snowflake target table, map a column in the CDC source to the required column in the Snowflake target in the mapping.

## Step 3. Configure the mapping task

After you create a mapping, add the mapping to a mapping task, and configure the advanced properties. Run the associated mapping task to write the changed data to Snowflake.

1. From the **Actions** menu, click **New Mapping Task**.

The **New Mapping Task** page appears.
2. In the **Definition** tab, enter the task name and select the configured mapping.
3. In the **CDC Runtime** tab, specify the required properties for the selected CDC source.

For more information about the **CDC Runtime** properties, see the source properties for the selected CDC source.
4. In the **Schedule** tab, add the following properties in the **Advanced Session Properties** section:
  - a. In the **Commit on End of File** field, select the value of the property as **No**.
  - b. In the **Commit Type** field, select the value of the property as **Source**.
  - c. In the **Recovery Strategy** field, select the value of the property as **Resume from last checkpoint**.

5. Click **Save > Run** the mapping task.

Alternatively, you can create a schedule that runs the mapping task on a recurring basis without manual intervention. You can define the schedule to minimize the time between mapping task runs. In **Monitor**, you can monitor the status of the logs after you run the task.

To improve performance, specify a higher commit interval for the Maximum Rows Per Commit property in the CDC Runtime page in the mapping task wizard. However, in case of failure, recovery takes more time for a higher commit interval.

## Viewing job statistics

You can view statistics for each job to see the number of rows processed and the job state.

The job can have one of the following states:

- Success. The Secure Agent applied all rows of insert, update, and delete operations.
- Warning. The Secure Agent rejected one or more rows. The **Rows Processed** field in the **My Jobs** page reflects the total number of rows that the Secure Agent processed.
- Failed. The job did not complete because it encountered errors.

The following image shows the **My Jobs** page that shows the details of the state and the number of processed rows of a Snowflake Data Cloud job:

Instance Name	Location	Subtasks	Start Time	End Time	Rows Processed	State
m_update_noreject-1	snowflake		Jan 28, 2019, 7:07 AM	Jan 28, 2019, 7:09 AM	40	Success
m_update_noreject-1	snowflake		Jan 28, 2019, 6:49 AM	Jan 28, 2019, 6:50 AM	40	Failed
m_rowstats_insert_reject-2	snowflake		Jan 28, 2019, 6:46 AM	Jan 28, 2019, 6:48 AM	40	Warning
m_rowstats_insert_reject-1	snowflake		Jan 28, 2019, 6:39 AM	Jan 28, 2019, 6:40 AM	40	Warning

To view how many among the processed rows were a success and how many resulted in an error, select the specific instance name and view the **Results** section. You can view the number of success rows and error rows.

The following image shows the details of the Snowflake Data Cloud task:

Job Properties

Task Name:

m\_rowstats\_insert\_reject

Instance ID:

1

Task Type:

Mapping

Started By:

snowflake\_mig through UI

Start Time:

Jan 28, 2019 6:39:05 AM

End Time:

Jan 28, 2019 6:40:20 AM

Duration:

1 minute, 15 seconds

Runtime Environment:

ADAPGAHER001

Secure Agent:

ADAPGAHER001

Results

State:

Warning

Success Rows:

39

Error Rows:

1

Session Log:

Download Session Log

Individual Source/Target Results

Name	Success Rows	Error Rows	Error Message	Actions
<div><div></div><div>Source</div></div>	40	0		
<div><div></div><div>TSTATS_BATCHROW_TGT1</div></div>	39	1		

You can download the session log to get details of the number of output rows, affected rows, applied rows, and rejected rows.

You might also encounter the following scenarios of target statistics for Snowflake Data Cloud write operations:

### Constraint violation

In insert, update, or delete operation scenarios where the Secure Agent rejects rows due to a constraint violation, a warning appears in the **Job Properties** page. You can download the session log to view the target statistics.

**Can't find a match**

In update or delete operation scenarios where the Secure Agent does not find a match for some records, that number does not reflect in the **My Jobs** page and the session log. For example, if there are 5 input rows and the Secure Agent updates only 4 target rows, the status of the number of processed rows stills reflects as 5. This issue occurs when Snowflake does not return an error message for rejected rows.

**Non-unique match**

In update or delete operation scenarios where the Secure Agent updates or deletes more rows because of a non-unique match, that actual number of updated or deleted records does not reflect both in the **My Jobs** page and in the session log. For example, if there were 5 input records and the Secure Agent updated 10 target rows, the **My Jobs** page reflects only 5 processed rows.

**Success rows not updated**

The number of success rows for the target object in the **Job Properties** page is not updated and remains the same as the number of incoming rows. For example, while writing 5 records to the target, if two records are rejected, the number of success rows still reflects as 5.

## Rules and guidelines for Snowflake Data Cloud target transformations

The following rules and guidelines apply for Snowflake Data Cloud target transformations:

- If some records are rejected while writing large amounts of data to Snowflake, the rejected file might not display some of the rejected records even though the statistics of rejected records appear correctly in the session logs.

## CHAPTER 7

# Lookups for Snowflake Data Cloud

Add a Lookup transformation to retrieve data based on a specified lookup condition. When you add a Lookup transformation to a mapping, you define the lookup connection, lookup objects, and lookup properties related to Snowflake.

You can look up Snowflake data values based on a condition that you configure. In the Lookup transformation, select the lookup connection and object. Then, define the lookup condition and the outcome for multiple matches.

The mapping queries the lookup source based on the lookup fields and the defined lookup condition. The lookup operation returns the result to the Lookup transformation, which then passes the results downstream.

You can configure the following lookups:

- **Connected.** You can use a cached or uncached connected lookup for mappings. You can also use a dynamic lookup cache to keep the lookup cache synchronized with the target.
- **Unconnected.** You can use a cached lookup. You need to supply input values for an unconnected Lookup transformation from a :LKP expression in a transformation that uses an Expression transformation.

For more information about Lookup transformation, see *Transformations* in the Data Integration documentation.

## Lookup properties for Snowflake Data Cloud

The following table describes the Snowflake Data Cloud lookup object properties that you can configure in a Lookup transformation:

Property	Description
Connection	Name of the lookup connection. You can select an existing connection, create a new connection, or define parameter values for the lookup connection property. <b>Note:</b> You can switch between a non-parameterized and a parameterized Snowflake Data Cloud connection. The advanced property values are retained during the switch. If you want to overwrite the lookup connection properties at runtime, select the <b>Allow parameter to be overridden at run time</b> option.
Source Type	Type of the source object. Select Single Object, or Parameter.

Property	Description
Parameter	<p>A parameter file where you define values that you want to update without having to edit the task. Select an existing parameter for the lookup object or click <b>New Parameter</b> to define a new parameter for the lookup object.</p> <p>The <b>Parameter</b> property appears only if you select parameter as the lookup type.</p> <p>If you want to overwrite the parameter at runtime, select the <b>Allow parameter to be overridden at run time</b> option.</p> <p>When the task runs, it uses the parameters from the file that you specify in the task advanced session properties.</p>
Lookup Object	Name of the lookup object for the mapping.
Filter	Not applicable.
Sort	Not applicable.

The following table describes the Snowflake Data Cloud lookup object advanced properties that you can configure in a Lookup transformation:

Advanced Property	Description
Database	Overrides the database specified in the connection.
Schema	Overrides the schema specified in the connection.
Warehouse	<p>Overrides the Snowflake warehouse name specified in the connection.</p> <p>The warehouse name in the mapping overrides the warehouse name you specify in the connection. Even though you provide an incorrect warehouse name in the connection properties, the connection is successful. However, before you run the mapping, ensure that you specify the correct warehouse name in the mapping properties.</p>
Role	Overrides the Snowflake role assigned to the user specified in the connection.
Pre SQL	Not applicable.
Post SQL	Not applicable.

## Parameterization

When you parameterize lookup objects and connections, you must follow some rules to override the object name.

If you enable the **Allow parameter to be overridden at run time** option in a transformation, you cannot override the object name using the fully qualified name such as `db.schema.tablename`.

You must pass the `db=<dbname>&schema<schemaname>` values in the **Additional JDBC URL Parameters** field in the Snowflake Data Cloud connection.



## Multiple match restrictions

When you configure a lookup, you define the behavior when a lookup condition returns more than one match. You can return all rows, any row, the first row, the last row, or an error.

The following configurations have multiple match policy restrictions:

- If the multiplicity option is set to Report error, the task does not fail.
- When you configure a lookup override without caching, the return first row and return last row multiple matches options are not applicable.
- Mappings configured for an uncached lookup and multiple matches fails when the Snowflake source contains case-sensitive tables and column names.

## Enable lookup caching

When you configure a Lookup transformation in a mapping, you can cache the lookup data during the runtime session.

When you select **Lookup Caching Enabled**, Data Integration queries the lookup source once and caches the values for use during the session, which can improve performance. You can specify the directory to store the cached lookup. You can configure dynamic and persistent lookup caches.

For information about lookup caching, see the chapter "Lookup transformations" in *Transformations* in the Data Integration documentation.

Dynamic lookup cache has the following restrictions:

- When you configure dynamic lookup cache, set the **On Multiple Matches** property to **Report Error**. To reset the property, change the dynamic lookup to a static lookup, change the property, and then change the static lookup to a dynamic lookup. Ensure that all the lookup fields are mapped.
- A lookup condition in dynamic lookups can use only an equal operator.
- Pushdown optimization is not applicable.
- You cannot parameterize a Lookup transformation enabled to use dynamic lookup cache.

## Log uncached lookup queries

When you enable caching, Data Integration queries the lookup source once and caches the values for use during the session. When you disable caching, each time a row passes into the transformation, a SELECT statement gets the lookup values.

Data Integration logs the uncached lookup queries for a connected and unconnected lookup in the session log. Enable the **Lookup Caching Enabled** property in the mapping, and then enable the verbose mode in the Snowflake Data Cloud mapping task.

The following image shows the selected verbose mode in the mapping task:

The image shows a configuration window for a mapping task, specifically the 'Schedule' tab. The interface includes the following elements:

- Tabs:** '1 Definition' and '2 Schedule' (active).
- Maximum Number of Log Files:** A text input field containing '10' with a help icon.
- Schema Change Handling:** Two radio button options: 'Asynchronous (Default)' (selected) and 'Dynamic'.
- Parameter File Location:** A section header.
- Download Parameter File Template:** A button with a help icon.
- Local:** A radio button option (selected).
- Parameter File Directory:** A text input field with a help icon.
- Parameter File Name:** A text input field with a help icon.
- Cloud Hosted:** A radio button option.
- Execution Mode:** A section header.
- Mode:** Two radio button options: 'Standard' and 'Verbose' (selected).

The 'Execution Mode' section, including the 'Verbose' radio button, is highlighted with a red rectangular box.

When you run the mapping, Data Integration logs the uncached lookup queries in the session logs.

## CHAPTER 8

# Pushdown optimization

You can use pushdown optimization to push the transformation logic to the Snowflake database.

When you run a task configured for pushdown optimization, Data Integration converts the transformation logic to an SQL query or a Snowflake command. Data Integration sends the query to the database, and the database runs the query. The amount of transformation logic that Data Integration pushes to the database depends on the database, the transformation logic, and the mapping configuration. Data Integration processes all transformation logic that it cannot push to a database.

Configure pushdown optimization for a mapping in the tasks properties. Full pushdown optimization is enabled by default in mapping tasks.

## Pushdown optimization types

When you apply pushdown optimization, Data Integration pushes the transformation logic to the source or target database based on the optimization type you specify in the task properties. Data Integration translates the transformation logic into SQL queries or Snowflake commands to the Snowflake database based on the connection you selected.

The source or target database executes the SQL queries or Snowflake commands to process the transformations. The amount of transformation logic you can push to the database depends on the database, transformation logic, and mapping and session configuration. Data Integration processes all the transformation logic that it cannot push to a database.

You can configure the following types of pushdown optimization in mappings:

### Full

Data Integration pushes down as much transformation logic as possible to process in the target database.

For mappings that read from and write to Snowflake, Data Integration analyses all the transformations from the source to the target. If all the transformations are compatible in the target, it pushes the entire mapping logic to the target. If it cannot push the entire mapping logic to the target, Data Integration first pushes as much transformation logic to the source database and then pushes as much transformation logic as possible to the target database.

**Note:** If the source and target Snowflake accounts are separate and reside in different regions but are hosted on the same cloud platform, you can configure full pushdown optimization. However, ensure that the Snowflake account user and role of the target Snowflake account has access to the Snowflake source account.

## Source

Data Integration pushes down as much as the transformation logic as possible to process in the source database.

# Pushdown optimization scenarios

You can configure pushdown optimization using the Snowflake Data Cloud Connector in mappings.

You can configure pushdown optimization for the following scenarios:

Source and target endpoints	Description
Snowflake source Snowflake target	Reads from and writes to Snowflake tables using the Snowflake Data Cloud connection. You can also read from Snowflake external tables, views, and materialized views. Pushdown optimization type is Source or Full.
Amazon S3 source Snowflake target	Reads from Amazon S3 using an Amazon S3 V2 connection and writes to Snowflake using a Snowflake Data Cloud connection. Pushdown optimization type is Full.
Google Cloud Storage source Snowflake target	Reads from Google Cloud Storage using a Google Cloud Storage V2 connection and writes to Snowflake using a Snowflake Data Cloud connection. Pushdown optimization type is Full.
Microsoft Azure Data Lake Storage Gen2 source Snowflake target	Reads from Microsoft Azure Data Lake Storage Gen2 using a Microsoft Azure Data Lake Storage Gen2 V2 connection and writes to Snowflake using a Snowflake Data Cloud connection. Pushdown optimization type is Full.
You can use the Secure Agent or Hosted Agent to push mapping logic to the database.	

**Note:** You can also configure pushdown for a mapping that uses a Snowflake ODBC connection to read from and write to Snowflake. Informatica recommends that you use the Snowflake Data Cloud connection in mappings to configure pushdown optimization. If you cannot push specific transformation logic using the Snowflake Data Cloud connection, you can explore configuring pushdown optimization using the Snowflake ODBC connection. The Snowflake ODBC connection uses the Snowflake ODBC 64-bit drivers on Windows and Linux systems. For more information, see the How-To Library article, [Configuring pushdown optimization for Snowflake using the ODBC Connector](#).

# Prepare for pushdown optimization using a Snowflake Data Cloud connection

Before you can configure pushdown optimization for a Snowflake Data Cloud mapping to load data from Google Cloud Storage or Microsoft Azure Data Lake Storage Gen2 to Snowflake, you need to meet certain requirements.

## Access data files in a Google Cloud Storage bucket

Before you configure pushdown optimization to load data from Google Cloud Storage to Snowflake, create a Cloud Storage Integration object in Snowflake.

The Storage Integration contains the details of the enabled Google Cloud Storage buckets from which you want to read data. Snowflake Data Cloud Connector creates a temporary external stage that uses the Cloud Storage Integration you created.

After you create the Cloud Storage Integration in Snowflake, specify the Cloud Storage Integration name in the Snowflake Data Cloud connection properties. Specify the name in the **Additional JDBC URL Parameters** field. The Storage Integration value is case-sensitive.

### Configuring storage integration for Google Cloud Storage

Create a storage integration to allow Snowflake to read data from the Google Cloud Storage bucket.

You can refer to the [Snowflake](#) documentation to perform the following steps:

1. Create a Cloud Storage Integration in Snowflake.
2. Retrieve the Cloud Storage Service account for your Snowflake account.
3. Grant the service account permissions to access the bucket objects.
  - a. Create a custom IAM role.
  - b. Assign the custom role to the Cloud Storage Service account.
4. Grant permissions for the role to create an external stage.

The role must have the CREATE STAGE privilege on the schema and the USAGE privilege on the storage integration.

For example, run the following commands to grant these privileges:

```
grant create stage on schema public to role myrole;  
grant usage on integration gcs_int to role myrole;
```

## Access data files in a Microsoft Azure Data Lake Storage Gen2 container

Before you configure pushdown optimization to load data from Microsoft Azure Data Lake Storage Gen2 to Snowflake, create a Cloud Storage Integration object in Snowflake.

The Storage Integration contains the details of the enabled Microsoft Azure Data Lake Storage Gen2 container from which you want to read data. The Snowflake Data Cloud Connector creates a temporary external stage that uses the Cloud Storage Integration you created.

After you create the Cloud Storage Integration in Snowflake, specify the Cloud Storage Integration name in the Snowflake Data Cloud connection properties. Specify the name in the **Additional JDBC URL Parameters** field. The Storage Integration value is case-sensitive.

### Configuring storage integration for Microsoft Azure Data Lake Storage Gen2

Create a storage integration to allow Snowflake to read data from the Microsoft Azure Data Lake Storage Gen2 container.

You can refer to the [Snowflake](#) documentation to perform the following steps:

1. Create a Cloud Storage Integration in Snowflake.

2. Retrieve the Cloud Storage Service account for your Snowflake account. See [“Granting access to the storage locations” on page 46](#)
3. Grant the service account permissions to access the bucket objects.
  - a. Create a custom IAM role.
  - b. Assign the custom role to the Cloud Storage Service account.
4. Grant permissions for the role to create an external stage.  
 The role must have the CREATE STAGE privilege on the schema and the USAGE privilege on the storage integration.

For example, run the following commands to grant these privileges:

```
grant create stage on schema public to role myrole;
grant usage on integration adls_int to role myrole;
```

## Granting access to the storage locations

Grant the Snowflake service principal access to the Azure Services storage accounts.

1. Run the DESCRIBE INTEGRATION command to retrieve the following consent URL: `desc storage integration <integration_name>;`  
 where `integration_name` is the name of the integration you created.  
 The URL in the `AZURE_CONSENT_URL` column has the following format:  

```
https://login.microsoftonline.com/<tenant_id>/oauth2/authorize?
client_id=<snowflake_application_id>
```

 Copy the value in the `AZURE_MULTI_TENANT_APP_NAME` column. This is the name of the Snowflake client application created for your account. You need this information to grant this application the required permissions to get an access token for the storage locations.
2. In a web browser, navigate to the URL in the `AZURE_CONSENT_URL` URL column.  
 The page displays a Microsoft permissions request page.
3. Click **Accept**.  
 This allows the Azure service principal created for your Snowflake account to obtain an access token on any resource inside your tenant. The access token is generated successfully only if you grant the service principal the appropriate permissions on the container.
4. Log into the Microsoft Azure portal.
5. Navigate to **Azure Services > Storage Accounts**, and then click the name of the storage account for which you want to grant the Snowflake service principal access to.
6. Click **Access Control (IAM) > Add Role Assignment**.
7. Select the required role to grant to the Snowflake service principal:
  - Storage Blob Data Reader: Grants read access only. You can load data from files staged in the storage account.
  - Storage Blob Data Contributor: Grants read and write access. You can load data from or unload data to files staged in the storage account.
8. Search for the Snowflake service principal.  
 This is the identity in the `AZURE_MULTI_TENANT_APP_NAME` property in the `DESC STORAGE INTEGRATION` output in Step 1. It might take an hour or longer for Azure to create the Snowflake service principal requested through the Microsoft request page. If the service principal is not available immediately, it is recommended that you wait for an hour or two and then search again. If you delete the service principal, the storage integration stops working.

9. Click **Save**.

The role assignments might take up to five minutes to take affect.

## Previewing pushdown optimization

Before you can run a mapping task configured for pushdown optimization, you can preview if pushdown optimization is possible when you create the mapping. You can preview pushdown optimization from the **Pushdown Optimization** panel in the Mapping Designer.

After you select the required pushdown optimization options and run the preview, Data Integration creates and runs a temporary pushdown preview mapping task. When the job completes, Data Integration displays the SQL queries to be executed and any warnings in the **Pushdown Optimization** panel. The warning messages help you understand which transformations in the configured mapping are not applicable for pushdown optimization. If pushdown optimization fails, Data Integration lists any queries generated up to the point of failure. You can edit the mapping and fix the required transformations before you run the mapping for pushdown optimization.

You can also view the temporary job created under **My Jobs** and download the session log to view the queries generated.

For more information about how to preview pushdown optimization, see the topic "Pushdown optimization preview" in *Mappings* in the Data Integration documentation.

## Configuring pushdown optimization

To optimize a mapping, add the mapping to a task, and then configure pushdown optimization in the mapping task. Full pushdown optimization is enabled by default in mapping tasks.

1. Create a mapping task.
2. In the **Pushdown Optimization** section on the **Schedule** tab, set the pushdown optimization value to **Full** or **To Source**.

When you run the mapping task, the transformation logic is pushed to the configured database. To verify that the mapping was optimized, you can check the session log for the job. In Monitor, view the log for jobs.

## Context based optimization for multiple targets

When you configure a mapping enabled for full pushdown optimization to write to multiple Snowflake targets using a Snowflake Data Cloud connection, you can specify the optimization context for multi-insert and slowly changing dimension type 2 merge scenarios.

You can specify the **Optimization context type** on the **Schedule** tab in the task properties. Based on the optimization context that you specify, Data Integration combines queries issued from multiple targets and constructs a single query for pushdown optimization and the task is optimized.

You can enable the following optimization modes in the task properties based on the target operations you specify in the mapping:

#### Multi-insert

Enable this mode when you insert data from a Snowflake source to multiple Snowflake targets. Data Integration combines the queries generated for each of the targets and issues a single query.

#### SCD Type 2 merge

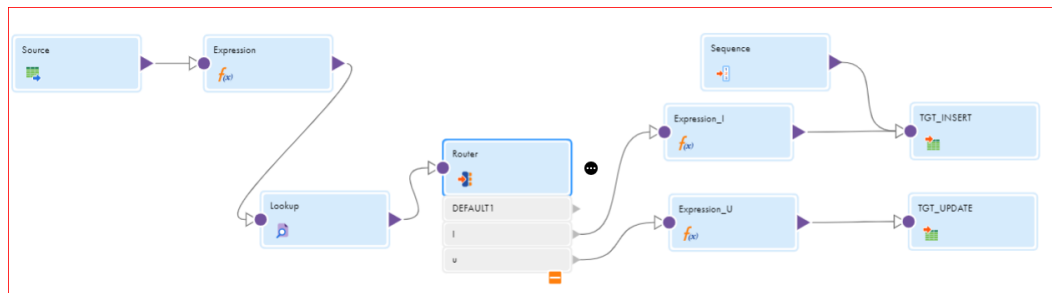
Enable this mode when you write to two Snowflake targets, where you use one target to insert data and the other target to update data. Data Integration combines the queries for both the targets and issues a Merge query.

Default is None.

## Understanding an SCD type 2 merge mapping

The SCD Type 2 merge mapping uses a Snowflake source and two target transformations that write to the same Snowflake table. One target transformation updates the table while the other transformation inserts data to the Snowflake table

The following image shows a mapping that writes slowly changing dimension data to a Snowflake target table:



Add lookup and expression transformations to compare source data against the existing target data. You enter the lookup conditions and source columns that you want the Data Integration to compare against the existing target.

For each source row without a matching primary key in the target, the Expression transformation marks the new row. For each source row with a matching primary key in the target, the Expression compares user-defined source and target columns. If those columns do not match, the Expression marks the row changed. The mapping then splits into two data flows.

The first data flow uses the Router transformation to pass only new rows to the Expression transformation. The Expression transformation inserts new rows to the target. A Sequence Generator creates a primary key for each row. The Expression transformation increases the increment between keys by 1,000 and creates a version number of 0 for each new row.

In the second data flow, the Router transformation passes only changed rows to pass to the Expression transformation. The Expression transformation inserts changed rows to the target. The Expression transformation increments both the key and the version number by one.

### Restrictions

You cannot use a filter, joiner, and a custom SQL query in an SCD type 2 merge mapping.



# Pushing logic to a database with different schemas

You can use cross-schema pushdown optimization for a mapping task to read from or write data to Snowflake objects associated with different schemas within the same Snowflake database.

To use cross-schema pushdown optimization, create two connections and specify the schema in each connection. Ensure that the schema in the source connection is different from the schema in the target connection, but both the schemas must belong to the same database.

## Configuring cross-schema pushdown optimization

To push processing logic for different schemas within a Snowflake database, configure cross-schema pushdown optimization in the mapping task.

1. Create a mapping task.
  - a. Select the configured mapping.
  - b. In the **Pushdown Optimization** section on the **Schedule** tab, set the pushdown optimization value to **Full**.
2. To enable cross-schema pushdown optimization, select **Enable cross-schema pushdown optimization**. Default is enabled.

The following image shows where you can configure the **Enable cross-schema pushdown optimization** property:

The screenshot displays the configuration interface for a mapping task. It features two main sections: 'Pushdown Optimization' and 'Advanced Session Properties'. In the 'Pushdown Optimization' section, the 'Pushdown Optimization' dropdown is set to 'Full'. Below this are two unchecked checkboxes: 'Create Temporary View' and 'Create Temporary Sequence'. The 'Advanced Session Properties' section includes an 'Add' button and a message stating 'There are no session properties.' At the bottom, there are two checkboxes: 'Enable cross-schema pushdown optimization' (which is checked) and 'Allow the mapping task to be executed simultaneously.' (which is unchecked).

3. Save the task and click **Finish**.

# Pushing logic to different databases

You can enable cross-database pushdown optimization to run queries on data spread across multiple Snowflake databases.

You can configure cross-database pushdown optimization in the mapping task. Ensure that the Snowflake source and target transformations in the mapping uses two different Snowflake Data Cloud connections, or Snowflake ODBC connections.

## Configuring cross-database pushdown optimization

To push processing logic across different Snowflake databases, configure cross-database pushdown optimization in the mapping task.

1. Create a mapping task.
  - a. Select the configured mapping.
  - b. In the **Pushdown Optimization** section on the **Schedule** tab, set the pushdown optimization value to **Full**.
2. To enable push down across databases, in the **Advanced Options** on the **Schedule** tab, select **Allow Pushdown across Databases** as the **Session Property Value** and set the **Session Property Value** to **Yes**.

The following image shows where you can configure the cross-database pushdown optimization property:

The screenshot displays the configuration window for a mapping task. The 'Pushdown Optimization' section at the top shows a dropdown menu set to 'Full'. Below this are two unchecked checkboxes: 'Create Temporary View' and 'Create Temporary Sequence'. The 'Advanced Session Properties' section contains an 'Add' button and a table with two rows. The first row has 'Allow Pushdown across Databases' as the 'Session Property Name' and 'Yes' as the 'Session Property Value'. The second row has 'Custom Properties' as the 'Session Property Name' and 'AddQuotesAlwaysPDO=Yes' as the 'Session Property Value'. Below the table are two more unchecked checkboxes: 'Enable cross-schema pushdown optimization' and 'Allow the mapping task to be executed simultaneously'. At the bottom are 'Save', '< Back', 'Next >', 'Finish', and 'Cancel' buttons.

## Cancelling a pushdown optimization task

When you configure full pushdown optimization for a task that reads from and writes to Snowflake, you can determine how Data Integration handles the job when pushdown optimization does not work.

You can set the task to fail or run without pushdown optimization. You can use this functionality only when you configure full pushdown optimization for a task that runs using the Snowflake Data Cloud connection in the Source and Target transformations.

1. In the **Pushdown Optimization** section on the **Schedule** tab, set the pushdown optimization value to **Full** for the selected mapping.
2. To determine what action Data Integration must perform when pushdown optimization does not occur, enable or disable the **If pushdown optimization is not possible, cancel the task** option based on your requirement:
  - a. Enable to cancel the task when pushdown does not occur.
  - b. Disable to run the task without pushdown optimization.  
Default is disabled.

# Clean stop a pushdown optimization job

When a task enabled for pushdown optimization is running, you can clean stop the job to terminate all the issued statements and processes spawned by the job.

Use the **Clean Stop** option on the My Jobs page in Data Integration and the All Jobs and Running Jobs page in Monitor.

See the following exceptions before you clean stop a pushdown optimization task:

- When you clean stop a task enabled for source pushdown optimization that reads from or writes to Snowflake and the target or source properties in the mapping contains pre-SQL or post-SQL statements, the job continues to run the target post-SQL query even though the select query is terminated.
- When you run a mapping configured to create a new target at runtime and clean stop the job immediately, Data Integration creates the target table even if the job is terminated.

## CHAPTER 9

# Pushdown optimization using a Snowflake Data Cloud connection

You can use Snowflake Data Cloud connection to configure pushdown optimization for a mapping task. Pushdown optimization enhances the task performance.

When you run a task configured for pushdown optimization, the task converts the transformation logic to Snowflake queries. The task sends the queries to Snowflake and the mapping logic is processed in the Snowflake database.

You can configure pushdown optimization for a mapping in the following scenarios:

### **Snowflake to Snowflake**

Read from and write to Snowflake using a Snowflake Data Cloud connection.

### **Amazon S3 to Snowflake**

Read from Amazon S3 using an Amazon S3 V2 connection in the Source transformation and write to Snowflake using a Snowflake Data Cloud connection in the Target transformation.

### **Microsoft Azure Data Lake Storage Gen2 to Snowflake**

Read from Microsoft Azure Data Lake Storage Gen2 using a Microsoft Azure Data Lake Storage Gen2 connection in the Source transformation and write to Snowflake using a Snowflake Data Cloud connection in the Target transformation.

### **Google Cloud Storage to Snowflake**

Read from Google Cloud Storage using a Google Cloud Storage connection in the Source transformation and write to Snowflake using a Snowflake Data Cloud connection in the Target transformation.

### **Example**

You work for healthcare solutions and your organization provides healthcare technology to pharmacies and pharmacy chains. You enable pharmacies to process prescriptions, store and provide access to healthcare records, and improve patient outcomes. Your organization stores its data in Google Cloud Storage.

The management wants to create a patient-centric pharmacy management system. The organization plans to leverage the warehouse infrastructure of Snowflake and load all its data to Snowflake so that they can make operational, financial, and clinical decisions with ease.

To load data from a Google Cloud Storage object to Snowflake, you must use ETL and ELT with the required transformations that support the data warehouse model.

Use a Google Cloud Storage V2 connection to read data from a Google Cloud Storage bucket and a Snowflake Data Cloud connection to write to a Snowflake target. Configure full pushdown optimization in the mapping to optimize the performance. The Google Cloud Storage source data is uploaded to the Snowflake stage using the PUT command. The Snowflake COPY commands are used to convert the transformations to

the corresponding SQL functions and expressions while loading the data to Snowflake. Pushdown optimization enhances the performance of the task and reduces the cost involved.

## Pushdown compatibility for Snowflake Data Cloud connection

You can configure the task to push transformations, variables, functions, and operators to the database.

When you use pushdown optimization, the Secure Agent converts the expression in the transformation by determining equivalent operators, variables, and functions in the database. If there is no equivalent operator, variable, and function, the Secure Agent processes the transformation logic.

### Functions with Snowflake Data Cloud

When you use pushdown optimization, Data Integration converts the expression in the transformation by determining equivalent functions in the database. If there is no equivalent function, Data Integration processes the transformation logic.

The tables summarize the availability of pushdown functions that you can push to Snowflake using full pushdown optimization:

Function	Function	Function
ABS()	IS_SPACES	SIN()
ASCII()	LAST_DAY()	SINH()
ADD_TO_DATE()	LENGTH()	SQRT()
AVG()	LN()	STDDEV()*
CEIL()	LOG()	SUBSTR()
CHR()	LOWER()	SUM()
CONCAT()	LPAD()	SYSDATE()
COS()	LTRIM()	SYSTIMESTAMP()
COSH()	MAX()	TAN()
COUNT()	MAKE_DATE_TIME	TANH()
DATE_COMPARE()	MEDIAN()	TO_BIGINT
DATE_DIFF()	MIN()	TO_CHAR(DATE)
DECODE()	MOD()	TO_CHAR(NUMBER)
EXP()	POWER()	TO_DATE()

Function	Function	Function
FLOOR()	REG_EXTRACT()	TO_DECIMAL()
GET_DATE_PART()	REG_MATCH()	TO_FLOAT()
IIF()	REG_REPLACE	TO_INTEGER()
IN()	REPLACECHR()	TRUNC(DATE)
INITCAP()	REPLACESTR()	TRUNC(NUMBER)
INSTR()	ROUND(NUMBER)	UPPER()
IS_DATE	RPAD()	MD5()
IS_NUMBER	RTRIM()	VARIANCE()
ISNULL()	SIGN()	-
*You cannot pass the filter condition argument in the STDDEV() function.		

**Note:** If you specify a function that is not supported for Snowflake full pushdown optimization, the task runs either with partial pushdown optimization or without full pushdown optimization.

## Operators with Snowflake Data Cloud

When you use pushdown optimization, Data Integration converts the expression in the transformation by determining equivalent operators in the database. If there is no equivalent operator, Data Integration processes the transformation logic.

The tables lists the operators that you can push to Snowflake:

Operator	Operator
+	>=
-	<=
*	!=
/	AND
%	OR
	NOT
>	IS NULL
<	IS NOT NULL
=	

## Variables with Snowflake Data Cloud

You can use full pushdown to push the `SESSSTARTTIME` variable to the Snowflake database.

## Transformations with Snowflake Data Cloud

When you configure pushdown optimization, Data Integration tries to push the configured transformation to Snowflake.

You can use full pushdown to push the following transformations to Snowflake:

- Aggregator
- Expression
- Filter
- Joiner
- Lookup
- Normalizer
- Rank
- Router
- Sequence Generator
- SQL
- Sorter
- Union
- Update Strategy

**Note:** Router transformation is applicable only for source pushdown optimization.

### Aggregator transformation

You can configure full pushdown optimization to push an Aggregator transformation to process in Snowflake.

#### Aggregate calculations

You can perform the following aggregate calculations:

- AVG
- COUNT
- MAX
- MIN
- MEDIAN
- SUM
- VARIANCE

#### Incoming ports

When you configure an Aggregator transformation and the incoming port is not used in an aggregate function or in a group by field in mappings, the `ANY_VALUE()` function is used for columns that are not part of the

group by or the aggregator function. In this case, the output is not deterministic as the ANY\_VALUE() function returns any value from the port.

## Expression transformation

You can configure full pushdown optimization to push an Expression transformation to process in Snowflake.

You can add an Expression transformation to each of the sources in the mapping, followed by a join downstream in the mapping. Additionally, you can add multiple Expression transformations that branch out from a transformation and then branch in into a transformation downstream in the mapping.

When you configure an Expression transformation, consider the following rules to include variables in the expression:

- You cannot use variables where you are using the value assigned while processing a previous row for calculations in the current row. If you do, the mapping runs without pushdown optimization.
- The variables can be nested, but you cannot refer to a variable before it is defined in the expression. If the variables are not defined in that order, the mapping runs without pushdown optimization.

For example,

```
var: AGEPLUS2 = AGEPLUS1 + 1
var: AGEPLUS1 = AGE + 1
out: NEXTAGE = AGEPLUS2 + 1
```

Here, AGE + 1 is defined later. AGEPLUS2 in the first variable refers to AGEPLUS1 and remains unresolved.

To resolve this, specify the variables in the following order:

```
var: AGEPLUS1 = AGE + 1
var: AGEPLUS2 = AGEPLUS1 + 1
out: NEXTAGE = AGEPLUS2 + 1
```

- The variables cannot have an expression that is cyclic or refers to itself:

For example,

```
var: AGEPLUS1 = AGEPLUS2 + 1
var: AGEPLUS2 = AGEPLUS1 + 1
out: NEXTAGE = AGEPLUS2
```

Here, AGEPLUS1 refers to AGEPLUS2 and remains unresolved.

## Lookup transformation

You can configure full pushdown optimization to push a Lookup transformation to process in Snowflake. You can push both a connected and an unconnected lookup.

When the mapping contains an unconnected lookup, you can also nest the unconnected lookup function with other expression functions. For example, :LKP.U\_LOOKUP(Upper(argument1), argument)

### Lookup objects

Consider the following rules when you configure lookups:

- You can configure a lookup for Snowflake when the Source transformation uses the following sources:
  - Amazon S3
  - Google Cloud Storage
  - Microsoft Azure Data Lake Storage Gen2
  - Snowflake source

**Note:** You can configure a lookup for an Amazon S3, Google Cloud Storage, or Microsoft Azure Data Lake Storage Gen2 object only when the Source transformation uses the corresponding Amazon S3, Google Cloud Storage, or Microsoft Azure Data Lake Storage Gen2 source.



## Unconnected lookup

When you configure an unconnected Lookup transformation, consider the following rules:

- Do not configure an expression for an output received from an unconnected lookup.
- In a mapping with a Snowflake Data Cloud source and target where some of input fields from the unconnected lookup transformation is not mapped to the Snowflake target object, the select query includes all the unmapped fields.

## Multiple matches behavior in a connected and unconnected lookup

If you enable pushdown optimization for a mapping that contains a connected or unconnected lookup, you must follow these guidelines:

- In an unconnected lookup, ensure that you always select the **Multiple Matches** option to **Report Error**. When you look up data and the lookup condition finds multiple matches, all the matching rows are selected and the task runs with pushdown optimization. If you enabled **Multiple Matches** to any option other than **Report Error**, the mapping runs without pushdown optimization.
- In a connected lookup, you can set the **Multiple Matches** option to **Return all rows** or **Report Error**. When you set the **Multiple Matches** option to **Report Error**, you can set the `Lkp_apdo_allow_report_error` custom flag in the task advanced session properties to determine how Data Integration handles multiple matches:
  - When you set the property to Yes and if there are multiple matches in the data, the multiple match policy is ignored and the job runs successfully with pushdown optimization.
  - When you do not set the property, and if there are multiple matches in the data, Data Integration considers the policy and displays a warning message. Pushdown optimization is ignored and the task fails.

## FileName port

When you configure a lookup for an Amazon S3 source in a mapping that contains an Amazon S3 source and Snowflake target, remove the filename port from both the Amazon S3 source and lookup object. The FileName port is not applicable.

## Lookup query object

When you use a lookup object as a query in a Lookup transformation in a mapping to lookup data in Snowflake, specify the database and schema in the advanced lookup properties or in the additional JDBC URL parameters in the Snowflake Data Cloud connection.

## Router transformation

You can configure source pushdown optimization to push a Router transformation to the database for processing.

When you configure a Router transformation, connect or map only one output group to the target transformation.

## Sequence Generator transformation

When you configure a Sequence Generator transformation, you can connect the NEXTVAL port to single or multiple ports in the transformations that follow the Sequence Generator transformation.

You can push a Sequence Generator transformation with the following restrictions:

- You cannot use a shared sequence in a Sequence Generator transformation.

- You can add only an Expression or a Target transformation after a Sequence Generator transformation in a mapping.

## SQL transformation

You can use an SQL transformation only to push certain functions and shared sequence.

### Use functions to run queries

You can include functions in an entered query in an SQL transformation and run queries with the Snowflake target endpoint.

You must use only the SELECT clause SQL statement to push a function. Specify the column name in the select query or function. Do not push functions using statements such as "SELECT \* FROM TABLE".

You can use the following functions in an entered query:

- UUID\_STRING
- RANDOM
- RANDSTR
- SIGN
- CURRENT\_REGION
- CURRENT\_ACCOUNT
- CURRENT\_ROLE
- CURRENT\_USER
- CURRENT\_DATABASE
- CURRENT\_SCHEMA
- DAYNAME
- SPLIT
- SPLIT\_PART

To use the CURRENT\_ROLE, CURRENT\_DATABASE, and CURRENT\_SCHEMA functions in an SQL transformation, ensure to provide the database, role, and schema name in the additional JDBC parameters field in the Snowflake Data Cloud connection. If you do not specify the values in the connection, Data Integration inserts null to the target.

### Reuse shared sequence

You can push a mapping with a shared sequence defined in an SQL transformation to a Snowflake endpoint. Data Integration writes the data in the same sequence to the target as in the Snowflake source.

Get the shared sequence from Snowflake and define the sequence in an entered query in an SQL transformation.

Specify the shared sequence in the entered query in the following syntax: Select

```
<Snowflake_schema_name>.<Snowflake_database_name>.<sequence_name>.NEXTVAL
```

### User defined functions

You can configure a custom query in an SQL transformation to read from Java or SQL user-defined functions (UDF) in Snowflake. You cannot read UDFs that have newline characters in the UDF name.

## Union transformation

You can push a Union transformation with the following restrictions:

- The Source transformation in the mapping must only include Snowflake source objects.
- A mapping runs without pushdown optimization when the source is Amazon S3, Google Cloud Storage, or Microsoft Azure Data Lake Storage Gen2.

## Update Strategy transformation

You cannot use an Update Strategy transformation.

You can instead use the update and upsert operations in the Target transformation to write to Snowflake.

## Features

You can configure pushdown optimization for a mapping that reads from the following sources and writes to a Snowflake target:

- Snowflake source
- Amazon S3 source
- Google Cloud Storage source
- Microsoft Azure Data Lake Storage Gen2 source

When you configure a mapping, some parameters are not supported for a mapping enabled for pushdown optimization. You can refer to the list of parameters that each source supports.

## Snowflake Data Cloud sources, targets, and lookups

When you configure pushdown optimization, refer to this list of supported Snowflake Data Cloud properties in the Source, Target, and Lookup transformations.

### Source properties

You can configure the following properties in a Snowflake source transformation:

- Source connection - Parameter, Allow parameter to be overridden at run time
- Source type - Single object, multiple objects, query, and parameter. You can also use a parameter file to override the Snowflake source connections and objects in a mapping from the mapping task.  
**Note:** When you use the query source type to read from Snowflake, you can choose to retain the field metadata and save the mapping. Even if you edit the query and run the mapping, the field metadata specified at design time is retained.
- Allow parameter to be overridden at run time.
- Query options - Filter and Join. You can use both simple and advanced filter conditions. You can use join to join related objects based on existing relationships or you can create an advanced relationship.
- Database override
- Schema override
- Warehouse override

- Role override
- Table name override
- SQL override
- Tracing level

The following source properties are not applicable:

- Query options - Sort

## Target properties

You can add multiple Snowflake targets in a mapping. The target can be the same Snowflake target table added multiple times or different Snowflake target tables.

You can configure the following properties in a Snowflake target transformation:

- Target connection - Parameter, Allow parameter to be overridden at run time.
- Target type - Single object, parameter. You can also use a parameter file to override the Snowflake target connections and objects in a mapping from the mapping task.
- Allow parameter to be overridden at run time
- Target object - Existing target, Create new at runtime.
- Operation - Insert, update, upsert, delete, or data driven
- Database override
- Schema override
- Warehouse override
- Role override
- Pre-SQL and Post-SQL
- Table name override
- Truncate Target Table
- Additional Write Runtime Parameters

The following target properties are not applicable:

- Update Mode
- Batch row size
- Number of local staging files
- Rejected File Path
- Update Override
- Forward Rejected Rows

## Lookup properties

When you enable pushdown optimization, you can configure the following properties for Snowflake connected and unconnected lookups:

- Lookup connection - Parameter, Allow parameter to be overridden at run time
- Source type - Single object, query, parameter. You can also use a parameter file to override the Snowflake lookup connections and objects in a mapping from the mapping task.
- Multiple matches - Report Error

- Database override
- Schema override
- Warehouse override
- Role override
- Table name override
- SQL override
- Tracing level

The following lookup properties are not applicable:

- Pre SQL
- Post SQL

## Guidelines for mappings

Consider the following guidelines when you configure mappings:

- For a target created at runtime, ensure that the Snowflake source does not contain records with the Time data type.
- When you configure filters, consider the following guidelines:
  - If a mapping contains a Filter transformation and also a filter in the Source transformation, the mapping consolidates the filter conditions from both these transformations to filter the records. However, it is recommended that you use only one of these filters at a time in a mapping.
  - You cannot use system variables in filters.
  - You cannot apply a filter for query and multiple source objects.
  - When you configure an IS\_date function in an Expression transformation, specify the format for this function. Else, the mapping populates incorrect data.
  - When you configure two Sequence Generator transformations to write to two Snowflake targets, and the sequence objects have the same sequence name in the custom properties, data populates incorrectly.
- For mappings that read from and write to Snowflake, consider the following guidelines:
  - You cannot use a query to read from stored procedures.
  - Even if you decrease the precision of the Snowflake String data type in a Source transformation to write to a Snowflake table, the mapping passes without truncating the data.
  - You can configure pre-SQL and post-SQL in mappings enabled for source pushdown optimization that read from and write to Snowflake. Pre-SQL and post-SQL are not applicable for mappings enabled with full pushdown optimization.
  - When you configure a mapping for source or partial pushdown optimization, do not connect the Source transformation to more than one transformation in the mapping. However, in a mapping enabled with full pushdown optimization, the Source transformation can branch out to multiple transformations in the mapping pipeline.
  - You can configure a custom query in a Source transformation to read from Java or SQL user-defined functions (UDF) in Snowflake.
  - When the mapping runs with full or source pushdown optimization, some of the queries in the session log are not aliased correctly. The alias for simple queries reflects properly.
  - A mapping fails to read data from multiple tables joined using related objects, where the tables and column names have case-sensitive, special, and unicode characters.

- A mapping that reads from multiple Snowflake objects that do not belong to the same database and schema fails.
- When you use the `is_number` function, the data populated for some values such as `inf`, `inf` and `NaN` in Snowflake differs with and without pushdown optimization applied.
- When you use the `IS_NUMBER` function in a transformation and the input data contains `d` or `D`, for example, in formats such as `+3.45d+32` or `+3.45D-32`, the function returns `False` or `0`.
- When you use the `IS_DATE` function in a transformation, do not use the `J`, `MM/DD/YYYY SSSSS`, `MM/DD/Y`, and `MM/DD/RR` formats.
- Mappings that read from or write to Snowflake with multibyte characters in the table or column names might fail. Before you configure a mapping to read from or write data with multibyte characters, set the `-DdisablePDOAdvancedAliasing` property in the JVM options in the Secure Agent properties.
- When you pass columns with Null values in a Normalizer transformation, Null values are not written to the target.

## Amazon S3 V2 source

The mapping supports the following properties for an Amazon S3 V2 connection:

- Access Key
- Secret Key

The mapping supports the following properties for an Amazon S3 V2 source:

- Source connection parameter
- Source Type - Single, parameter
- Format - Delimited, Avro, ORC, Parquet, and JSON
- Source Type - File and directory
- Folder Path
- File Name
- Compression Format. - Gzip

A mapping enabled for pushdown optimization that reads from an Amazon S3 V2 source and writes to a Snowflake target has some restrictions.

### Authentication

When you read multiple Avro files using an Amazon S3 connection enabled for IAM authentication, specify the right access key and the secret key in the Amazon S3 connection. For more information, see the help for Amazon S3 V2 Connector.

### Create a new target at runtime

A mapping that creates a new target at runtime has the following restrictions:

- To write data from file data types such as Avro, ORC, or Parquet from Amazon S3 to Snowflake, you must delete the **Filename** field.
- Mappings fails with a casting error when the table name contains Unicode characters.

### Data types

A mapping has the following restrictions for certain data types:

- You cannot write Avro files that contain special characters.

- You cannot write data that contains the Binary data type.
- You cannot read data in JSON format that contains special characters. For more information about using identifiers, see [Identifiers Syntax](#) in the Snowflake documentation.
- If you specify any escape character for the S3 file format, the escape character defaults to backslash.
- ORC files with year 1523 is loaded incorrectly as 1524.
- When you write data with the Time data types from a Parquet file from Amazon S3 to Snowflake, the value of the time differs in the target.
- The precision of JSON data must not exceed the precision of the Snowflake target table.
- If the Amazon S3 source type is a directory and you enable wildcard characters for the directory, the mapping fails. A warning message appears stating that wildcard characters read are not supported with pushdown optimization.

For information on how to configure the supported properties, see the Amazon S3 V2 Connector documentation.

## Google Cloud Storage V2 source

The mapping supports the following properties for a Google Cloud Storage V2 source connection:

- Source connection, connection parameter
- Source Type - Single, parameter
- Format - Delimited, Avro, Parquet, and JSON
- Google Cloud Storage Path
- Source File Name
- Is Directory

For information on how to configure the supported properties, see the Google Cloud Storage V2 Connector documentation.

## Microsoft Azure Data Lake Storage Gen2 source

The mapping supports the following properties for an Microsoft Azure Data Lake Storage Gen2 source connection:

- Account Name
- File System Name

The mapping supports the following properties for a Microsoft Azure Data Lake Storage Gen2 source:

- Source connection, connection parameter
- Source Type - Single, parameter
- Format - Delimited, Avro, Parquet, JSON, and ORC
- Formatting Options
- Filesystem Name Override
- Source Type - File, Directory
- Directory Override - Absolute path; Relative path
- File Name Override - Source object
- Compression Format - GZip to read flat files

- Tracing Level

For information on how to configure the supported properties, see the Microsoft Azure Data Lake Storage Gen2 Connector documentation.

## Creating temporary view for source overrides

To push a custom query or an SQL override to Snowflake, the task must create a temporary view based on the provided custom query and then generates the pushdown query with the temporary view.

Before you configure a custom query as the source object or you configure an SQL override, perform the following task:

In the **Pushdown Optimization** section on the **Schedule** tab, select **Create Temporary View**.

**Note:** If you do not set the **Create Temporary View** property, the mapping runs without pushdown optimization.

## Configuring target copy command options

You can specify the copy command options to load data from Amazon S3 to Snowflake.

Specify the options in the **Additional Write Runtime Parameters** field in the Snowflake Data Cloud advanced target properties of the Target transformation.

When you specify multiple copy command options, separate each option with an ampersand &.

**Note:** The copy option *MATCH\_BY\_COLUMN\_NAME* is not applicable.

For more information about the supported copy command options, see the Snowflake documentation at the following website: <https://docs.snowflake.com/en/sql-reference/sql/copy-into-table.html>

## Verify the pushdown query in the session log

To verify that the pushdown optimization was applied while running the mapping, you can check the session log for the job. In Monitor, view the log for jobs.

Check the queries in the session logs to verify if the mapping applied pushdown optimization.

For example, the following query is generated in the session log for a mapping enabled with full pushdown optimization:

```
39 OPT_63306 [2020-08-06 08:57:23.875] [Full Pushdown optimization is supported between the source and the target system.].
40 OPT_63306 [2020-08-06 08:57:23.875] [Validating mapping for Pushdown Optimization.].
41 OPT_63309 [2020-08-06 08:57:23.875] Pushdown Optimization was successfully enabled.

86 MAPPING> SNOWFLAKECLOUDDATAWAREHOUSE_10000 [2020-08-06 08:57:40.523] [INFO] Inserting data into Snowflake target table : INSERT INTO
"DB_PC_AUTO"."SCHEMA_PC_AUTO"."TEST1_TGT"("F1_INT") SELECT (SINH((t0."F1_INT":NUMBER(38,0))::DOUBLE)::DOUBLE)::DOUBLE FROM "CQA"."CQA_SCHEMA"."TEST1_SRC"
AS t0
```



In the example, the generated SQL includes both the `Insert Into` and `Select` queries pushed down to the database as a single statement.

The session log provides the pushdown status. You can check the details to troubleshoot the error.

For example, the session log shows the following error details in the query:

```
OPT_63306 [2020-09-10 14:09:05.716] [Source level filter is not supported for pushdown optimization.].  
OPT_63307 [2020-09-10 14:09:05.716] Pushdown optimization failed at Analyze phase.  
OPT_63310 [2020-09-10 14:09:05.716] Failed to enable Pushdown Optimization.
```

When you do not enable pushdown optimization in a mapping, separate select and insert statements are generated for the read and write operations:

```
READER_1_1_1>SNOWFLAKECLOUDDATAWAREHOUSE_1000 [2020-09-10 14:09:29.4781] [INFO]  
The Snowflake Connector uses the following SQL query to read data: SELECT "DEPTID",  
"DEPTNAME" FROM "DEPT" WHERE  
( 'DEPT"."DEPTID" >=103) ORDER BY "DEPT"."DEPTID" desc
```

## APPENDIX A

# Data type reference

Cloud Data Integration uses the following data types in Snowflake Data Cloud mappings and mapping tasks:

- Snowflake native data types appear in the source and target transformations when you choose to edit metadata for the fields.
- Transformation data types. Set of data types that appear in the transformations. These are internal data types based on ANSI SQL-92 generic data types, which the Secure Agent uses to move data across platforms. They appear in all transformations in a mapping.

When the Secure Agent reads source data, it converts the native data types to the comparable transformation data types before transforming the data. When the Secure Agent writes to a target, it converts the transformation data types to the comparable native data types.

## Snowflake Data Cloud and transformation data types

The following table lists the Snowflake data types that Data Integration supports and the corresponding transformation data types:

Snowflake Data Cloud Data Type	Transformation Data Type	Range and Description
Binary (Varbinary)	binary	Maximum value: 8,388,60 Default value is 8,388,60.
Boolean	string	A Boolean attribute.
Date	datetime	Date and time values.
Float (Double, Double precision, Real, Float, Float4, Float8)	double	Floating point numbers with double-precision (64 bit). Maximum value: 1.7976931348623158e+307 Minimum value: -1.79769313486231E+307
Number (Decimal, Numeric)	decimal	Number with 28-bit precision and scale.
NUMBER (Int, Integer, Bigint, Smallint, Tinyint, Byteint)	decimal	Number with 28-bit precision and scale as 0. Maximum value: 9.9999999999999E+27 Minimum value: -9.9999999999999E+26
Time	datetime	Date and time values.

Snowflake Data Cloud Data Type	Transformation Data Type	Range and Description
Timestamp_LTZ	datetime	Date and time values.
Timestamp_NTZ (Timestamp_NTZ, datetime)	datetime	Date and time values.
Timestamp_TZ	datetime	Date and time values.
Varchar (Text, Char, Character, String)	string	Maximum value: 16,777,216 Default value is 16,777,216.

## Semi-structured data types and transformation data types

Snowflake uses the Variant data type to store and represent semi-structured data. The Variant data type can store values of any other type, including object and array, up to a maximum size of 16 MB uncompressed.

Array, object, and variant from the Snowflake source are mapped to the String data type in Cloud Data Integration. While writing to the target, these strings can be written as Array, Object, or Variant columns to the Snowflake target. The strings that you write to Snowflake must be in a serialization format, just as they appear after the read operation.

When you use the Create New at Runtime option to write the Variant data type from a source to the Snowflake target, Data Integration writes Variant as Varchar to the target. You can edit the field mapping to map Varchar to Variant before you write to the target. In a completely parameterized mapping, you cannot edit the target metadata from the default Varchar data type to variant.

The following table lists the semi-structured data types that you can read from Snowflake and the corresponding transformation data types that these map to in Cloud Data Integration:

Snowflake Data Type	Transformation Data Type	Description and Range
Array	String	16,777,216
Object	String	16,777,216
Variant	String	16,777,216

**Note:** The default size to read or write semi-structured data types is set to 65536 bytes. To increase the limit, add the following parameter and set the required value in the **Additional JDBC URL Parameters** field of the Snowflake Data Cloud connection properties: `semiStructuredDTPrecision=<size>`

# Rules and guidelines for data types

When you read or write data, some differences in processing and configuration apply for certain data types.

## Mappings

Consider the following rules and guidelines for mappings:

- You can read or write data of Binary data type that is in Hexadecimal format.
- The agent reads or writes the maximum float value `1.7976931348623158e+308` as infinity.
- You can use the following formats to specify filter values of the Datetime data type:
  - YYYY-MM-DD HH24:MI:SS
  - YYYY/MM/DD HH24:MI:SS
  - MM/DD/YYYY HH24:MI:SS
- If a Snowflake Cloud Data lookup object contains fields with the String data type of maximum or default precision and the row size exceeds the maximum row size, the task fails.
- The performance of a write operation slows down if the data contains the Date fields.
- A task that captures changed data from a CDC source fails when the Snowflake target contains a repeated column of the Record data type.
- When you handle dynamic schemas in mappings, the following updates are not applicable:
  - Schema updates to the Timestamp and Date data types.
  - Schema updates that involve a decrease in the precision of the Varchar data type.

## APPENDIX B

# Additional runtime configurations

You can configure additional attributes for some of the Snowflake Data Cloud tasks in the Secure Agent properties.

You can enable bulk processing of records, set logging for large records, configure a local staging directory, optimize data staging, and improve memory requirements for read operations.

## Configure JVM memory requirements

Specify the Java heap space memory as `-Xmx256m` in the Secure Agent properties to avoid the JDBC driver internal errors in mappings and mapping tasks that read data from Snowflake:

This update does not apply for mappings configured with pushdown optimization using the Snowflake ODBC connection.

Perform the following steps to configure the JVM memory:

1. In Administrator, select the Secure Agent listed on the **Runtime Environments** tab.
2. Click **Edit**.
3. In the **System Configuration Details** section, select **Data Integration Service** as the service and **DTM** as the type.
4. Edit the **JVMOption1** property, and enter `-Xmx256m`.
5. Click **Save**.

## Configure bulk processing

You can enable bulk processing to write large amounts of data to Snowflake. Bulk processing utilizes minimal number of API calls and the performance of the write operation is optimized.

To enable bulk processing, specify the property `-DENABLE_WRITER_BULK_PROCESSING=true` in the Secure Agent properties:

Perform the following steps to configure bulk processing before you run a mapping:

1. In Administrator, select the Secure Agent listed on the **Runtime Environments** tab.
2. Click **Edit**.

3. In the **System Configuration Details** section, select **Data Integration Server** as the service and **DTM** as the type.
4. Edit the JVM option, and enter `-DENABLE_WRITER_BULK_PROCESSING=true`.
5. Click **Save**.

**Note:** This update does not apply for mapping tasks configured with pushdown optimization using the Snowflake ODBC connection or the Snowflake Data Cloud connection.

## Configure logging properties

Error messages might appear in the session log from Snowflake in some scenarios.

To avoid this error, perform the following tasks:

1. Navigate to the following location:  
`<Secure Agent installation directory>\apps\jdk\1.8.0_Zulu<version>\jre\lib`
2. Set the `logging.properties` field to include `java.util.logging.ConsoleHandler.level = WARNING`

If you do not set this logging property, the following errors might appear in the session logs:

```
net.snowflake.client.jdbc.internal.apache.http.impl.execchain.RetryExec execute
INFO: I/O exception (net.snowflake.client.jdbc.internal.apache.http.NoHttpResponseException)
caught when processing request to {s}...> The target server failed to respond.
```

## Configure the temp directory for a mapping

The Secure Agent creates the local staging files in a default temp directory. You can configure a different directory to store the local staging files.

To configure a different directory for the local staging files, perform the following steps:

1. In Administrator, click **Runtime Environments**.  
The Runtime Environments page appears.
2. Select the Secure Agent for which you want to set the custom configuration property.
3. Click **Edit Secure Agent** icon corresponding to the Secure Agent you want to edit in **Actions**.  
The Edit Secure Agent page appears.
4. Select the **Service** as **Data Integration Server** in the **System Configuration Details** section.
5. Select the **Type** as **DTM** in the **System Configuration Details** section.
6. Set the JVM option to `-Djava.io.tmpdir=E:\Snowflake\temp`.
7. Click **Save**.
8. Restart the Secure Agent.

# Optimize the staging performance for a mapping

Data Integration, by default, creates a flat file locally in a temporary folder to stage the data before reading from or writing to Snowflake. You can set Data Integration to optimize the staging performance.

If you do not set the staging property, Data Integration performs staging without the optimized settings, which might impact the performance of the task.

## Improve the staging performance for a read operation

You cannot configure a custom query to call a stored procedure from Snowflake. When you run a query to call a stored procedure, the task fails.

## Improve the staging performance for a write operation

Consider the following rules when you enable the staging property for the write operation:

- If you run a mapping enabled for pushdown optimization, the mapping runs without pushdown optimization.
- If the data contains timestamp data types with time zone, the job runs without staging the data in the local flat file.
- If the mapping contains Oracle CDC as the source and Snowflake as the target, the job runs without staging the data in the local flat file.
- You can determine the behavior of writing data with empty and null values to the target when you enable the staging property and set the `copyEmptyFieldAsEmpty` in the **Additional Write Runtime Parameters** field in the Target transformation properties.

The table describes the behavior of empty and null values when you set these properties:

Additional Write Runtime Parameters	Enable staging optimization	Disable staging optimization
<code>copyEmptyFieldAsEmpty=FALSE</code>	Empty -> NULL NULL -> NULL	Empty -> NULL NULL -> NULL
<code>copyEmptyFieldAsEmpty=TRUE</code>	Empty -> Empty NULL -> Empty	Empty -> Empty NULL -> Empty
Property not set	Empty -> Empty NULL -> NULL	Empty -> Empty NULL -> NULL

## Enabling Snowflake Data Cloud Connector to optimize the staging performance

Set the following staging property in the agent properties based on the operation you want to configure:

- Read operation: `INFA_DTM_RDR_STAGING_ENABLED_CONNECTORS`
- Write operation. `INFA_DTM_STAGING_ENABLED_CONNECTORS`

Perform the following tasks to set the staging property for the Tomcat in the Secure Agent properties:

1. In Administrator, click **Runtime Environments**.
2. Edit the Secure Agent for which you want to set the property.
3. In the **System Configuration Details** section, select the **Service** as **Data Integration Server** and the type as **Tomcat**.

4. Set the value of the Tomcat property to the plugin ID of Snowflake Data Cloud Connector. You can find the plugin ID in the manifest file located in the following directory: <Secure Agent installation directory>/downloads/<Snowflake package>/CCIManifest

The following image shows the property set for the Secure Agent for the read and write operations:

Tomcat	INFA_DTM_STAGING_ENABLED_CONNECTORS	305501
Tomcat	INFA_DTM_RDR_STAGING_ENABLED_CONNECTORS	305501

When you run the mapping, the flat file is created in the following directory in your machine: C:\Windows\Temp\snowflake\stage\<Snowflake\_Target.txt>

When you run the mapping, the flat file is created in the following directory in your machine:

- Read operation: C:\Windows\Temp\StagingReader\<Source\_Name><data\_.csv>
- Write operation: C:\Windows\Temp\snowflake\stage\<Snowflake\_Target.txt>

You can check the session logs.

- Read operation: If the staging is done through the flat file successfully, Data Integration logs the following message in the session log: Staging mode is enabled to read data.
- Write operation: You can check the session logs. If the staging is done through the flat file successfully, Data Integration logs the following message in the session log: The INFA\_DTM\_STAGING is successfully enabled to use the flat file to create local staging files.



# INDEX

## A

authentication  
    OAuth 2.0 authorization code [13](#), [15](#)

## C

cache  
    enable lookup cache [41](#)  
Cloud Application Integration community  
    URL [6](#)  
Cloud Developer community  
    URL [6](#)  
connections  
    Snowflake Data Cloud [12](#)  
connector overview [8](#)

## D

Data Integration community  
    URL [6](#)  
data types  
    native data types [66](#), [67](#)  
    overview [66](#)  
    transformation data types [66](#), [67](#)  
dynamic schema handling [20](#)

## F

filter [24](#)  
flat file  
    staging data [71](#)

## I

Informatica Global Customer Support  
    contact information [7](#)  
Informatica Intelligent Cloud Services  
    web site [6](#)

## L

local staging files  
    directory configuration [70](#)  
    JVM option [70](#)  
lookup  
    database [39](#)  
    multiple matches [39](#)  
    role [39](#)  
    schema [39](#)  
    uncached [41](#)

lookup (*continued*)  
    warehouse [39](#)  
lookup caches  
    dynamic [41](#)  
Lookup transformation  
    lookup caching [41](#)

## M

maintenance outages [7](#)  
mappings  
    database [24](#)  
    example [21](#)  
    lookup overview [39](#)  
    lookup properties [39](#)  
    Post-SQL [24](#)  
    Pre-SQL [24](#)  
    role [24](#)  
    schema [24](#)  
    source properties [24](#)  
    warehouse [24](#)

## N

not parameterized sort [24](#)

## P

parameterized sort [24](#)  
pushdown optimization  
    functions [53](#), [54](#)  
    transformations [53–55](#)  
Pushdown optimization  
    preview [47](#)  
Pushdown optimization preview [47](#)

## S

Snowflake Cloud Data Warehouse V2 connection  
    configuration [50](#)  
    pushdown optimization overview [43](#)  
Snowflake Data Cloud  
    assets [8](#)  
        authentication  
        standard [12](#)  
    connection properties [12](#)  
    connections [12](#)  
    connector [8](#)  
    lookup [8](#)  
    mapping example [21](#)  
    pushdown [52](#)  
    source [8](#)

- Snowflake Data Cloud (*continued*)
  - target [8](#)
  - transformations [8](#)
- Snowflake data Cloud connection
  - configuration [47](#)
- Snowflake Data Cloud connection
  - configuration [47](#), [49](#)
- Snowflake Data Cloud connector
  - rules and guidelines [68](#)
- Snowflake ODBC connection
  - configuration [49](#), [50](#)
  - pushdown optimization overview [43](#)
- sort [24](#)
- SQL transformations
  - configuration [19](#)
- staging data
  - flat file [71](#)
- status
  - Informatica Intelligent Cloud Services [7](#)

- system status [7](#)

## T

- transformations
  - pushdown optimization [55](#)
- trust site
  - description [7](#)

## U

- upgrade notifications [7](#)

## W

- web site [6](#)