How-To Library

Informatica

Configuring pushdown optimization for Amazon Redshift using the ODBC Connector

Abstract

This article explains how to configure pushdown optimization for a mapping that uses the ODBC connection type to read from or write to Amazon Redshift. This article also explains the pushdown optimization functions, transformations, operators, and data types that you can push down to Amazon Redshift using the Amazon Redshift ODBC connection type.

Supported Versions

- Informatica Cloud® Data Integration

Table of Contents

Overview

When you enable pushdown optimization for a task, the task pushes down the entire transformation logic as SQL queries to the underlying database for processing and the database runs the query. Since the entire transformation logic is encapsulated in an SQL query and there is no back-and-forth data movement between the database and the Informatica engine, the task is processed faster.

It is important to know what your primary use case is before you configure pushdown optimization. If your use case is to process data in the cloud for a mapping that reads from a Cloud Data Warehouse or Cloud Data Lake and writes to a Cloud Data Warehouse target such as Amazon Redshift, you can use pushdown optimization to push the processing logic to the warehouse or data lake using the applicable native connector as the source and Redshift as the target. The native connectors offer a wide range of capabilities that you can explore. For more information about configuring pushdown optimization and the supported functionalities using Amazon Redshift, see the help for Amazon Redshift Connector.

However, if you are using your on-premises data warehouse or any ODBC-based targets, ODBC based pushdown using the ODBC Connector is the better choice. You do not need any separate license to use ODBC pushdown. In this article, we'll configure pushdown optimization for a task that uses ODBC Connector to push transformation logic to process in the Amazon Redshift source and target databases. The ODBC connection must use the Redshift subtype in the connection.

You can set the pushdown optimization for the ODBC connection type that uses Amazon ODBC Redshift drivers to enhance the mapping performance. You must create a data source name in the ODBC datasource administrator. We will show you how to optimize an ODBC task to read from or write to Amazon Redshift. You can configure source or full pushdown for the task. Perform the following steps to optimize an Amazon Redshift ODBC task:

- Configure the Amazon Redshift ODBC driver.
- Create an ODBC connection with the ODBC subtype as Redshift and configure the Amazon Redshift database properties to which you want to connect.
- Create a mapping that uses the configured ODBC connection and enable the task for pushdown optimization.

After you create an Amazon Redshift ODBC connection, select the value of the **Pushdown Optimization** property as **Full** or **To Source** in the advanced session properties. You can check the session log to verify that the pushdown optimization has taken place.

When you run the task enabled with pushdown optimization, the task converts the transformation logic to an SQL query. The task sends the query to the database, and the database uses the database resources to run the query. You can also create a temporary view, temporary sequence, and push logic across databases or schemas. For information about the advanced session properties that you can use with pushdown optimization using an Amazon Redshift ODBC connection, see the topic "Advanced Session Properties" in Tasks in the Data Integration help.

Note: Amazon Redshift does not support upsert operations in a full pushdown optimization.

Prepare for pushdown optimization using an Amazon Redshift ODBC connection

Before you can configure pushdown optimization using an Amazon Redshift ODBC connection, you need to configure the Amazon Redshift ODBC driver based on your operating system. Amazon Redshift supports Redshift ODBC drivers on Windows and Linux systems. Install the Amazon Redshift ODBC 64-bit driver based on your system requirement.

Note: Informatica certifies Amazon Redshift ODBC driver version, `AmazonRedshiftODBC-64-bit-1.4.8.1000-1.x86_64`, to use for pushdown optimization.

Configure the Amazon Redshift ODBC driver on Windows

Before you establish an ODBC connection to connect to Amazon Redshift on Windows, you must configure the ODBC connection.

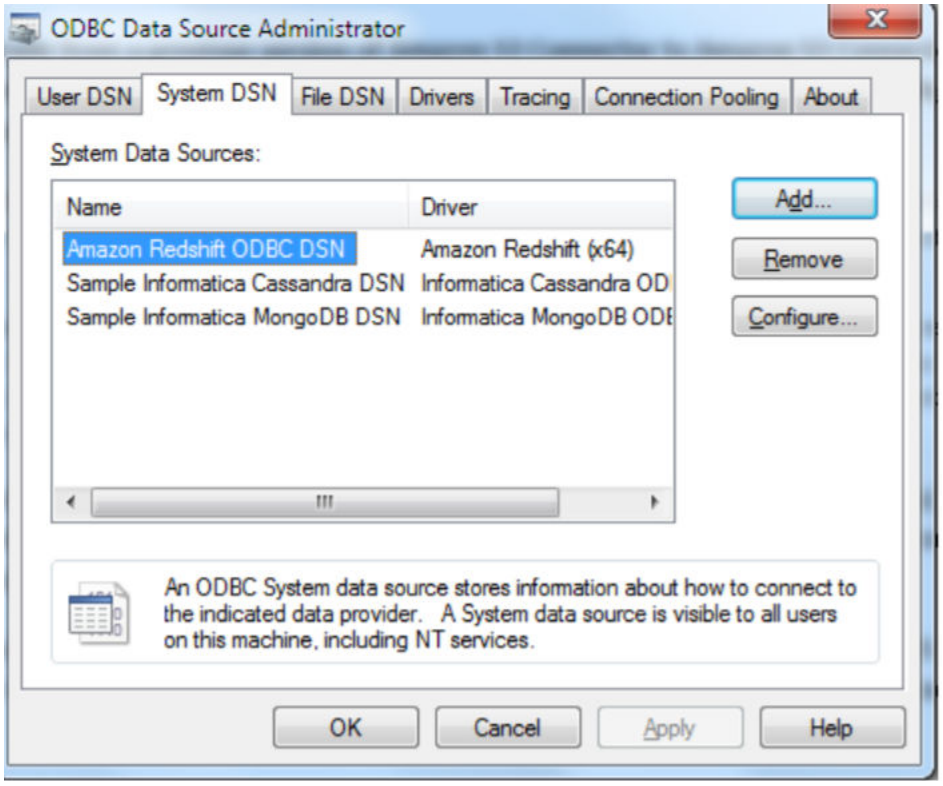
Before you establish an ODBC connection to connect to Amazon Redshift on Windows, configure the ODBC driver.

1. Download the Amazon Redshift ODBC drivers from the AWS website.

   You must download the Amazon Redshift ODBC 64-bit driver.
2. Install the Amazon Redshift ODBC drivers on the machine where the Secure Agent is installed.
3. Open the folder in which ODBC data source file is installed.
4. Run the `odbcad32.exe` file.

   The **ODBC Data Source Administrator** dialog box appears.
5. Click **System DSN**.

The **System DSN** tab appears. The following image shows the **System DSN** tab on the **ODBC Data Source Administrator** dialog box:



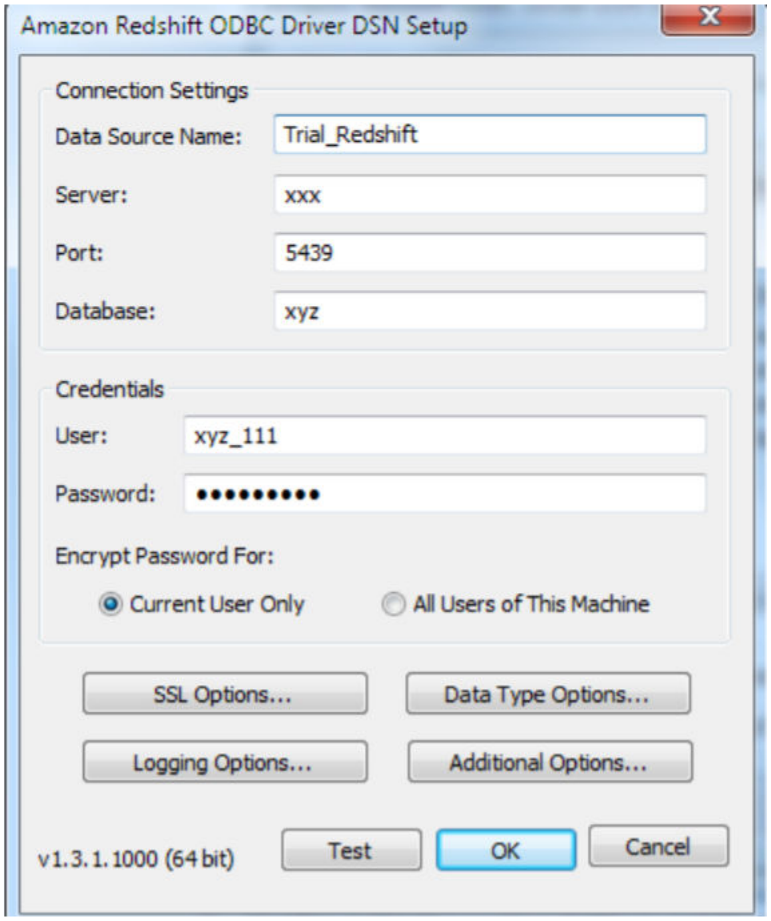
System Data Sources:

Name	Driver
Amazon Redshift ODBC DSN	Amazon Redshift (x64)
Sample Informatica Cassandra DSN	Informatica Cassandra OD
Sample Informatica MongoDB DSN	Informatica MongoDB ODI

An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.

6. Click **Configure**.

The **Amazon Redshift ODBC Driver DSN Setup** dialog box displays. The following image shows the **Amazon Redshift ODBC Driver DSN Setup** dialog box where you can configure the **Connection Settings** and **Credentials** section:



Amazon Redshift ODBC Driver DSN Setup

Connection Settings

Data Source Name: Trial_Redshift

Server: xxx

Port: 5439

Database: xyz

Credentials

User: xyz_111

Password: ●●●●●●●●●●

Encrypt Password For:

[✓ Current User Only] All Users of This Machine

7. Specify the following connection properties:

Property	Description
Data Source Name	Name of the data source.
Server	Location of the Amazon Redshift server.
Port	Port number of the Amazon Redshift server.
Database	Name of the Amazon Redshift database.

Note: You must specify the **Server**, **Port**, and **Database** values from the JDBC URL.

8. Specify the following credentials in the **Credentials** section:

Property	**Description**
User	User name to access the Amazon Redshift database.
Password	Password for the Amazon Redshift database.
Encrypt Password For	Encrypts the password for the following users: - **Current User Only** - **All Users of This Machine** Default is **Current User Only**.

9. Click **Test** to test the connection in the **Amazon Redshift ODBC Driver DSN Setup** box.
10. Click **OK**.

The Amazon Redshift ODBC connection is configured successfully on Windows.

After you configure the Amazon Redshift ODBC connection, you must create an ODBC connection to connect to Amazon Redshift.

For more information about how to create an ODBC connection to connect to Amazon Redshift, see "Create an Amazon Redshift ODBC connection" on page 7

Configure the Amazon Redshift ODBC driver on Linux

Before you establish an ODBC connection to connect to Amazon Redshift on Linux, you must configure the ODBC connection.

1. Download the Amazon Redshift ODBC drivers from the AWS website.

   You must download the Amazon Redshift ODBC 64-bit driver.
2. Install the Amazon Redshift ODBC drivers on the machine where the Secure Agent is installed.
3. Configure the `odbc.ini` file properties in the following format:

```
[ODBC Data Sources]
driver_name=dsn_name

[dsn_name]
Driver=path/driver_file

Host=cluster_endpoint
Port=port_number
Database=database_name
```

4. Specify the following properties in the `odbc.ini` file:

Property	**Description**
ODBC Data Sources	Name of the data source.
Driver	Location of the Amazon Redshift ODBC driver file.
Host	Location of the Amazon Redshift host.

Property	Description
Port	Port number of the Amazon Redshift server.
Database	Name of the Amazon Redshift database.

Note: You must specify the **Host**, **Port**, and **Database** values from the JDBC URL.

5. Set the following environment variables for the operating system:

Variable	Description
LD_LIBRARY_PATH	Directory where the Amazon Redshift ODBC driver is installed.
ODBCINI	Directory that contains the odbc.ini file.
ODBCHROME	ODBC installation directory.

6. Run the following command to export the `odbc.ini` file.

```
Export ODBCINI=/<odbc.ini file path>/odbc.ini
```

7. Restart the Secure Agent.

   The Amazon Redshift ODBC connection on Linux is configured successfully.

After you configure the Amazon Redshift ODBC connection, you must create an ODBC connection to connect to Amazon Redshift.
For more information about how to create an ODBC connection to connect to Amazon Redshift, see "Create an Amazon Redshift ODBC connection" on page 7

Create an Amazon Redshift ODBC connection

You must create an ODBC connection to connect to Amazon Redshift after you configure the ODBC connection.

Perform the following steps to create an Amazon Redshift ODBC connection on the **Connections** page:

1. Configure the following connection details in the **Connection Details** section:

Property	Description
Connection Name	Name of the ODBC connection. For example, Redshift ODBC.
Description	Description of the connection.
Type	Type of the connection. Select the type of the connection as **ODBC**.

2. Configure the following connection details in the **Connection Properties** section:

Property	**Description**
Runtime Environment	Runtime environment that contains the Secure Agent you can use to access the system.
User Name	User name to log in to Amazon Redshift.
Password	Password to log in to Amazon Redshift.
Data Source Name	Enter the name of the ODBC data source name that you created for the Amazon Redshift database.
Schema	Name of the Amazon Redshift schema.
Code Page	The code page of the database server or flat file defined in the connection.
ODBC Subtype	Enter the value of the **ODBC Subtype** field as **Redshift**.
Driver Manger for Linux	The driver that the Amazon Redshift ODBC driver manager sends database calls to.

You can use the configured connection in a mapping. You can then enable pushdown optimization on the Schedule tab in the mapping task.

Configuring pushdown optimization

Perform the following steps to configure pushdown optimization in an Amazon Redshift ODBC mapping task:

1. Create a mapping to read from or write to Amazon Redshift:
   a. Use the Amazon Redshift ODBC connection in the Source transformation.
   b. Use the Amazon Redshift ODBC connection in the Target transformation.
2. 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** or **To Source**.
   c. Save the task and click **Finish**.

When you run the mapping task, the transformation logic is pushed to the Amazon Redshift database. To verify that the pushdown optimization has taken place, you can check the session log for the job. In Monitor, view the log for jobs.

Cross-Schema pushdown optimization

You can configure cross-schema pushdown optimization for a mapping task that uses a Amazon Redshift ODBC connection to read or write data to Amazon Redshift objects of different schemas in the same database.

To use cross-schema pushdown optimization, create Amazon Redshift ODBC connections and specify the schema for the source and target connections. The source and target schemas must be different but must belong to the same database. Configure pushdown optimization for the mapping task and enable cross-schema pushdown optimization in the advanced session properties. By default, the **Enable cross-schema pushdown optimization** check box is selected.

Configure cross-schema optimization for an Amazon Redshift mapping task

Perform the following steps to configure cross-schema pushdown optimization for an Amazon Redshift V2 mapping task:

1. Create Amazon Redshift ODBC source and target connections, each defined with a different schema.

   For example,

   - Create a `rs_odbc1` Amazon Redshift ODBC connection and specify `CQA_SCHEMA1` schema in the connection properties.
   - Create a `rs_odbc2` Amazon Redshift ODBC connection and specify `CQA_SCHEMA2` schema in the connection properties.
2. Create an Amazon Redshift V2 mapping.

   For example, create a `m_rs_pdo_crossSchema` Amazon Redshift V2 mapping.
3. Add a Source transformation. Include an Amazon Redshift V2 source object and connection to read data using the schema specified in the connection.

   For example, add a Source transformation. Include an Amazon Redshift V2 source object and connection `rs_odbc1` to read data using `CQA_SCHEMA1`.
4. Add a Target transformation. Include an Amazon Redshift V2 target object and connection to write data using the schema specified in the connection.

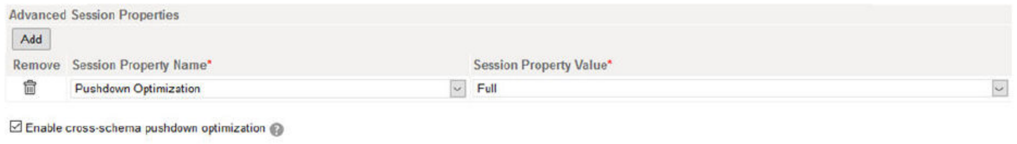
   For example, add a Target transformation. Include an Amazon Redshift V2 target object and connection `rs_odbc2` to write data using `CQA_SCHEMA2`.
5. Create an Amazon Redshift V2 mapping task, and perform the following tasks:
   a. Select the configured Amazon Redshift V2 mapping.

      For example, select the `m_rs_pdo_crossSchema` Amazon Redshift V2 mapping.

   b. In the **Advanced Options** on the **Schedule** tab, add **Pushdown Optimization** and set the value to **Full**.

   c. Select **Enable cross-schema pushdown optimization**.

      The following image shows the configured **Enable cross-schema pushdown optimization** property:



Advanced Session Properties

Remove	Session Property Name*	Session Property Value*
	Pushdown Optimization	Full

☑ Enable cross-schema pushdown optimization

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

   When you run the mapping task, the Secure Agent reads data from the Amazon Redshift source object associated with the `CQA_SCHEMA1` schema and writes data to the Amazon Redshift V2 target object associated with `CQA_SCHEMA2` schema.

Pushdown compatibility for Amazon Redshift ODBC connection

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

When you use pushdown optimization, Data Integration 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, Data Integration processes the transformation logic.

Functions and operators with Amazon Redshift ODBC connection

The following table displays the functions that you can push to the Amazon Redshift database by using source-side or full pushdown optimization.

Columns marked with an X indicate that the function can be pushed to the Amazon Redshift database by using source-side or full pushdown optimization. Columns marked with S indicate that the function can be pushed to the Amazon Redshift database only by using source-side pushdown optimization. Columns marked with a dash (-) symbol indicate that the function cannot be pushed to the database.

Function	Pushdown	Function	Pushdown	Function	Pushdown
ABORT()	-	INSTR()	X	REG_REPLACE	-
ABS()	X	IS_DATE()	-	REPLACECHR()	-
ADD_TO_DATE()	X	IS_NUMBER()	-	REPLACESTR()	-
AES_DECRYPT()	-	IS_SPACES()	-	REVERSE()	-
AES_ENCRYPT()	-	ISNULL()	S	ROUND(DATE)	-
ASCII()	-	LAST()	-	ROUND(NUMBER)	X
AVG()	S	LAST_DAY()	X	RPAD()	X
CEIL()	X	LEAST()	-	RTRIM()	X
CHOOSE()	-	LENGTH()	X	SET_DATE_PART()	-
CHRCODE()	-	LN()	X	SIGN()	X
COMPRESS()	-	LOG()	-	SIN()	X
CONCAT()	X	LOOKUP	-	SINH()	-
COS()	X	LOWER()	X	SOUNDEX()	-
COSH()	-	LPAD()	X	SQRT()	X
COUNT()	S	LTRIM()	X	STDDEV()	S
CRC32()	-	MAKE_DATE_TIME()	-	SUBSTR()	X
CUME()	-	MAX()	S	SUM()	S
DATE_COMPARE()	X	MD5()	X	SYSTIMESTAMP()	S
DATE_DIFF()	X	MEDIAN()	-	TAN()	S
DECODE()	X	METAPHONE()	-	TANH()	-
DECODE_BASE64()	-	MIN()	S	TO_BIGINT	X
DECOMPRESS()	-	MOD()	S	TO_CHAR(DATE)	S

Function	Pushdown	Function	Pushdown	Function	Pushdown
ENCODE_BASE64()	-	MOVINGAVG()	-	TO_CHAR(NUMBER)	X
EXP()	X	MOVINGSUM()	-	TO_DATE()	X
FIRST()	-	NPER()	-	TO_DECIMAL()	X
FLOOR()	X	PERCENTILE()	-	TO_FLOAT()	X
FV()	-	PMT()	-	TO_INTEGER()	X
GET_DATE_PART()	X	POWER()	X	TRUNC(DATE)	S
GREATEST()	-	PV()	-	TRUNC(NUMBER)	S
IIF()	X	RAND()	-	UPPER()	X
IN()	S	RATE()	-	VARIANCE()	S
INDEXOF()	-	REG_EXTRACT()	-		
INITCAP()	X	REG_MATCH()	-		

Amazon Redshift database supports the following operators:

```
+ - * / % || > = >= <= != AND OR NOT ^=
```

Restrictions

When you push functions to Amazon Redshift, adhere to the following guidelines:

- To push TRUNC(DATE) to Amazon Redshift, you must define the date and format arguments. Otherwise, the agent does not push the function to Amazon Redshift .
- The aggregator functions for Amazon Redshift accept only one argument, a field set for the aggregator function. The filter condition argument is ignored. In addition, verify that all fields mapped to the target are listed in the GROUP BY clause.
- To push TO_DATE() to Amazon Redshift, you must define the string and format arguments.
- To push TO_CHAR() to Amazon Redshift, you must define the date and format arguments.
- Do not specify a format for SYSTIMESTAMP() to push the SYSTIMESTAMP to Amazon Redshift. The Amazon Redshift database returns the complete time stamp.
- To push INSTR() to Amazon Redshift, you must only define string, search_value, and start arguments. Amazon Redshift does not support occurrence and comparison_type arguments.
- The flag argument is ignored when you push TO_BIGINT and TO_INTEGER to Amazon Redshift.
- The CaseFlag argument is ignored when you push IN() to Amazon Redshift.
- If you use the NS format as part of the ADD_TO_DATE() function, the agent does not push the function to Amazon Redshift.
- If you use any of the following formats as part of the TO_CHAR() and TO_DATE() functions, the agent does not push the function to Amazon Redshift:

  - - NS

  - - SSSS

- - SSSSS

- - RR

- To push TRUNC(DATE), GET_DATE_PART(), and DATE_DIFF() to Amazon Redshift, you must use the following formats:

  - - D

  - - DDD

  - - HH24

  - - MI

  - - MM

  - - MS

  - - SS

  - - US

  - - YYYY

- When you push the DATE_DIFF() function to Amazon Redshift using a Redshift ODBC connection, the Secure Agent incorrectly returns the difference values. If the result is positive, the Secure Agent returns negative values and if the result is negative, the positive value is returned.

Transformations with Amazon Redshift ODBC connection

The following table lists the transformations that you can push using source of full pushdown:

Transformations	Pushdown
Aggregator	Source, Full
Expression	Source, Full
Filter	Source, Full
Joiner	Source, Full
Sorter	Source, Full
Union	Source, Full
Router	Full

Note: The Update Override ODBC advanced target property is not applicable when you use an ODBC connection to connect to Amazon Redshift.

Lookup transformation

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

See the following restrictions for multiple matches when you configure a connected and unconnected Lookup transformation:

Connected lookup

If you use a connected Lookup transformation, you must select the **Return All Rows** multiple matches option in the lookup object properties. If you select any other option other than **Return All Rows**, the pushdown query is not generated.

Unconnected lookup

For pushdown optimization to work with an unconnected lookup, select the **Report error** multiple matches option. Additionally, you must enable the **Create Temporary View** property in the session properties of the mapping task. If there are multiple matches in the data, the Secure Agent processes the records, but does not log an error when it finds multiple matches. If there are multiple matches in the data, the Secure Agent processes the records, but does not log an error when it finds multiple matches.

Source transformation

You can push down a custom query or an SQL override to Amazon Redshift.

Before you run a task that contains a custom query as the source object or you configure an SQL override, you must set the **Create Temporary View** session property in the mapping task properties.

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

Perform the following task to set the property:

1. In the mapping task, navigate to the **Advanced Session Properties** on the **Schedule** tab.
2. From the **Session Property Name** list, select **Create Temporary View** and set the **Session Property Value** to **Yes**.
3. Click **Finish**.

Mappings enabled for pushdown optimization has the following restrictions for sources:

- When you select an Amazon Redshift ODBC connection as source and select an external table as a source object, the data preview fails.
- You must include all the unique column fields in the SQL query.
- When you use a custom query or an SQL override in a Source transformation, you must have the permissions to access and perform operations in the Amazon Redshift source and target tables.
- When you configure an SQL override, make sure that you include all the connected fields in the SQL query.

Author

Divya Ramesh