Amazon Redshift JDBC Data Connector

Installation and Configuration Guide

Version 1.2.54

April 7, 2021

Contact Us

For support, check the Amazon Redshift Forum at https://forums.aws.amazon.com/forum.jspa?forumID=155 or open a support case using the AWS Support Center at https://aws.amazon.com/support

About This Guide

Purpose

The *Amazon Redshift JDBC Data Connector Installation and Configuration Guide* explains how to install and configure the Amazon Redshift JDBC Data Connector on all supported platforms. The guide also provides details related to features of the connector.

Audience

The guide is intended for end users of the Amazon Redshift JDBC Connector.

Knowledge Prerequisites

To use the Amazon Redshift JDBC Connector, the following knowledge is helpful:

- Familiarity with the platform on which you are using the Amazon Redshift JDBC Connector
- Ability to use the data store to which the Amazon Redshift JDBC Connector is connecting
- An understanding of the role of JDBC technologies in connecting to a data store
- Experience creating and configuring JDBC connections
- Exposure to SQL

Document Conventions

Italics are used when referring to book and document titles.

Bold is used in procedures for graphical user interface elements that a user clicks and text that a user types.

`Monospace font` indicates commands, source code or contents of text files.

> **Note:**
>
> A text box with a pencil icon indicates a short note appended to a paragraph.

! Important:

A text box with an exclamation mark indicates an important comment related to the preceding paragraph.

Table of Contents

About the Amazon Redshift JDBC Connector

The Amazon Redshift JDBC Connector enables Business Intelligence (BI), analytics, and reporting on data that is stored in Amazon Redshift databases. The connector complies with the JDBC 4.2 data standard.

JDBC is one of the most established and widely supported APIs for connecting to and working with databases. At the heart of the technology is the JDBC connector, which connects an application to the database.

This guide is suitable for users who want to access data residing within Redshift from their desktop environment. Application developers might also find the information helpful. Refer to your application for details on connecting via JDBC.

System Requirements

Each machine where you use the Amazon Redshift JDBC Connector must have Java Runtime Environment (JRE) 8.0 installed.

Amazon Redshift JDBC Connector Files

The Amazon Redshift JDBC Connector is delivered in a ZIP archive named `Amazon_RedshiftJDBC42_[Version].zip`, where *[Version]* is the version number of the connector.

The archive contains the connector supporting the JDBC API version indicated in the archive name, as well as release notes and third-party license information.

Installing and Using the Amazon Redshift JDBC Connector

To install the Amazon Redshift JDBC Connector on your machine, extract the files from the ZIP archive to the directory of your choice.

To access a Redshift data store using the Amazon Redshift JDBC Connector, you need to configure the following:

- The list of connector library files (see Referencing the JDBC Connector Libraries on page 11)
- The `Driver` or `DataSource` class (see Registering the Connector Class on page 12)
- The connection URL for the connector (see Building the Connection URL on page 13)

The Amazon Redshift JDBC Connector provides read-write access to Redshift data.

Referencing the JDBC Connector Libraries

Before you use the Amazon Redshift JDBC Connector, the JDBC application or Java code that you are using to connect to your data must be able to access the connector JAR files. In the application or code, specify all the JAR files that you extracted from the ZIP archive.

Using the Connector in a JDBC Application

Most JDBC applications provide a set of configuration options for adding a list of connector library files. Use the provided options to include all the JAR files from the ZIP archive as part of the connector configuration in the application. For more information, see the documentation for your JDBC application.

Using the Connector in Java Code

You must include all the connector library files in the class path. This is the path that the Java Runtime Environment searches for classes and other resource files. For more information, see "Setting the Class Path" in the appropriate Java SE Documentation.

- For Windows: http://docs.oracle.com/javase/8/docs/technotes/tools/windows/classpath.html
- For Linux and Solaris: http://docs.oracle.com/javase/8/docs/technotes/tools/unix/classpath.html

Registering the Connector Class

Before connecting to your data, you must register the appropriate class for your application.

The following classes are used to connect the Amazon Redshift JDBC Connector to Redshift data stores:

- The `Driver` classes extend `java.sql.Driver`.
- The `DataSource` classes extend `javax.sql.DataSource` and `javax.sql.ConnectionPoolDataSource`.

The connector supports the following fully-qualified class names (FQCNs) that are independent of the JDBC version:

- `com.amazon.redshift.jdbc.Driver`
- `com.amazon.redshift.jdbc.DataSource`

The following sample code shows how to use the `DriverManager` class to establish a connection for JDBC 4.2:

```
private static Connection connectViaDM() throws Exception
{
   Connection connection = null;
   connection = DriverManager.getConnection(CONNECTION_URL);
   return connection;
}
```

The following sample code shows how to use the `DataSource` class to establish a connection:

```
private static Connection connectViaDS() throws Exception
{
   Connection connection = null;
   DataSource ds = new com.amazon.redshift.jdbc.DataSource
   ();
   ds.setURL(CONNECTION_URL);
   connection = ds.getConnection();
   return connection;
}
```

Building the Connection URL

Use the connection URL to supply connection information to the data store that you are accessing. The following is the format of the connection URL for the Amazon Redshift JDBC Connector, where *[Host]* the endpoint of the Redshift server and *[Port]* is the number of the TCP port that the server uses to listen for client requests:

```
jdbc:redshift://[Host]:[Port]
```

Note:

By default, Redshift uses port 5439.

You can specify optional settings such as the schema to use or any of the connection properties supported by the connector. For a list of the properties available in the connector, see Connector Configuration Options on page 26.

The following is the format of a connection URL that specifies some optional settings:

```
jdbc:redshift://[Host]:[Port]/[Schema];[Property1]=[Value];
[Property2]=[Value];...
```

For example, to connect to port 9000 on a Redshift server in the us-west-1 region on AWS, access the schema named Default, and authenticate the connection using a user name and password, you would use the following connection URL:

```
jdbc:redshift://redshift.company.us-west-
1.redshift.amazonaws.com:9000/Default;UID=amazon;PWD=amazon
```

Important:

Do not duplicate properties in the connection URL.

Configuring Authentication and SSL

Configure the Amazon Redshift JDBC Connector to authenticate your connection according to the security requirements of the Redshift server that you are connecting to.

You must always provide your Redshift user name and password to authenticate the connection. Depending on whether SSL is enabled and required on the server, you might also need to configure the connector to connect through SSL or use one-way SSL authentication so that the client (the connector itself) verifies the identity of the server.

You provide the configuration information to the connector in the connection URL. For more information about the syntax of the connection URL, see Building the Connection URL on page 13.

Note:

In this documentation, "SSL" indicates both TLS (Transport Layer Security) and SSL (Secure Sockets Layer). The connector supports industry-standard versions of TLS/SSL.

Using User Name and Password Only

If the server you are connecting to does not use SSL, then you only need to provide your user name and password to authenticate the connection.

To configure authentication using your user name and password only:

1. Set the `UID` property to your user name for accessing the Redshift server.
2. Set the `PWD` property to the password corresponding to your user name.

Using SSL without Identity Verification

If the server you are connecting to uses SSL but does not require identity verification, then you can configure the connector to use a non-validating SSL factory.

To configure an SSL connection without identity verification:

1. Set the `UID` property to your user name for accessing the Redshift server.
2. Set the `PWD` property to the password corresponding to your user name.
3. Set the `SSL` property to `true`.

4. Set the `SSLFactory` property to `com.amazon.redshift.ssl.NonValidatingFactory`.

Using One-Way SSL Authentication

If the server you are connecting to uses SSL and has a certificate, then you can configure the connector to verify the identity of the server using one-way authentication.

One-way authentication requires a signed, trusted SSL certificate for verifying the identity of the server. You can configure the connector to use a specific certificate or access a TrustStore that contains the appropriate certificate. If you do not specify a certificate or TrustStore, then the connector uses the default Java TrustStore (typically either `jssecacerts` or `cacerts`).

To configure one-way SSL authentication:

1. Set the `UID` property to your user name for accessing the Redshift server.
2. Set the `PWD` property to the password corresponding to your user name.
3. Set the `SSL` property to `true`.
4. Set the `SSLRootCert` property to the location of your root CA certificate.
5. If you are not using one of the default Java TrustStores, then do one of the following:
   - To specify a server certificate, set the `SSLRootCert` property to the full path of the certificate.
   - Or, to specify a TrustStore, do the following:
     a. Use the keytool program to add the server certificate to the TrustStore that you want to use
     b. Specify the TrustStore and password to use when starting the Java application using the connector. For example:

```
-Djavax.net.ssl.trustStore=[TrustStoreName]
-Djavax.net.ssl.trustStorePassword=
[TrustStorePassword]
```

6. Choose one:
   - To validate the certificate, set the `SSLMode` property to `verify-ca`.
   - Or, to validate the certificate and verify the host name in the certificate, set the `SSLMode` property to `verify-full`.

Configuring IAM Authentication

If you are connecting to a Redshift server using IAM authentication, set the following properties as part of your data source connection string.

For more information on IAM Authentication, see http://docs.aws.amazon.com/redshift/latest/mgmt/redshift-iam-authentication-access-control.html.

To use IAM Authentication, use one of the following connection string formats:

Connection String	Description
`jdbc:redshift:iam://[host]:[port]/[db]`	A regular connection string. The connector infers the ClusterID and Region from the host.
`jdbc:redshift:iam://[cluster-id]:[region]/[db]`	The connector retrieves host information, given the ClusterID and Region.
`jdbc:redshift:iam://[host]/[db]`	The connector defaults to port 5439, and infers ClusterID and Region from the host.

Profiles

If you are using IAM authentication, you have the option to specify any additional required or optional connection properties under a profile name. This enables you to avoid putting certain information directly in the connection string. You specify the profile name in your connection string using the `Profile` property.

Profiles can be added to the AWS Credentials file. The default location for this file is: `~/.aws/credentials`

You can change the default value by setting the path in the following environment variable: AWS_CREDENTIAL_PROFILES_FILE

For more information about profiles see: http://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/credentials.html

Instance Profile Credentials

If you are running an application on an EC2 instance that is associated with an IAM role, you can connect using the instance profile credentials.

To do this, use one of the IAM connection string formats in the preceding table, and set the `dbuser` connection property to the Redshift user name that you are connecting as.

For more information about instance profiles see: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html.

Credential Providers

The connector also supports credential provider plugins from the following services:

- AD FS
- Azure AD
- Okta
- PingFederate

If you use one of these services, the connection URL needs to specify the following properties:

- **Plugin_Name**: The fully-qualified class path for your credentials provider plugin class.
- **IdP_Host**: The host for the service you are using to authenticate into Redshift.
- **IdP_Port**: The port that the host for the authentication service listens at. Not required for Okta.
- **User**: The user name for the idp_host server.
- **Password**: The password associated with the idp_host user name.
- **DbUser**: The Redshift user name you are connecting as.
- **SSL_Insecure**: Indicates whether the IDP server certificate should be verified.
- **Client_ID**: The client ID associated with the user name in the Azure AD portal. Only used for Azure AD.
- **Client_Secret**: The client secret associated with the client ID in the Azure AD portal. Only used for Azure AD.
- **IdP_Tenant**: The Azure AD tenant ID for your Redshift application. Only used for Azure AD.
- **App_ID**: The Okta app ID for your Redshift application. Only used for Okta.
- **App_Name**: The optional Okta app name for your Redshift application. Only used for Okta.
- **Partner_SPID**: The optional partner SPID (service provider ID) value. Only used for PingFederate.

If you are using a browser plugin for one of these services, the connection URL can also include:

- **Login_URL**: The URL for the resource on the identity provider's website when using the SAML or Azure AD services through a browser plugin. Required if you are using a browser plugin.

- **Listen_Port**: The port that the connector uses to get the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.
- **IdP_Response_Timeout**: The amount of time, in seconds, that the connector waits for the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

For information on additional connection string properties, see Connector Configuration Options on page 26.

Configuring TCP Keepalives

By default, the Amazon Redshift JDBC Connector is configured to use TCP keepalives to prevent connections from timing out. You can specify when the connector starts sending keepalive packets or disable the feature by setting the relevant properties in the connection URL. For more information about the syntax of the connection URL, see Building the Connection URL on page 13.

To specify when the connector sends keepalive packets:

- Set the `TCPKeepAliveMinutes` property to the number of minutes of inactivity before the connector starts sending TCP keepalive packets.

To disable TCP keepalives:

- Set the `TCPKeepAlive` property to `FALSE`.

Configuring Logging

To help troubleshoot issues, you can enable logging in the connector.

! Important:

Only enable logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.

The settings for logging apply to every connection that uses the Amazon Redshift JDBC Connector, so make sure to disable the feature after you are done using it.

In the connection URL, set the `LogLevel` key to enable logging at the desired level of detail. The following table lists the logging levels provided by the Amazon Redshift JDBC Connector, in order from least verbose to most verbose.

LogLevel Value	Description
0	Disable all logging.
1	Log severe error events that lead the connector to abort.
2	Log error events that might allow the connector to continue running.
3	Log events that might result in an error if action is not taken.
4	Log general information that describes the progress of the connector.
5	Log detailed information that is useful for debugging the connector.
6	Log all connector activity.

To enable logging:

1. Set the `LogLevel` property to the desired level of information to include in log files.
2. Set the `LogPath` property to the full path to the folder where you want to save log files. To make sure that the connection URL is compatible with all JDBC applications, escape the backslashes (`\`) in your file path by typing another backslash.

For example, the following connection URL enables logging level 3 and saves the log files in the `C:\temp` folder:

```
jdbc:redshift://redshift.company.us-west-
1.redshift.amazonaws.com:9000/Default;LogLevel=3;LogPath
=C:\temp
```

3. To make sure that the new settings take effect, restart your JDBC application and reconnect to the server.

The Amazon Redshift JDBC Connector produces the following log files in the location specified in the `LogPath` property:

- A `RedshiftJDBC_driver.log` file that logs connector activity that is not specific to a connection.
- A `RedshiftJDBC_connection_[Number].log` and `RedshiftJDBC_connection_ext_[Number].log` file for each connection made to the database, where *[Number]* is a number that identifies each log file. These files log connector activity that is specific to the connection.

If the `LogPath` value is invalid, then the connector sends the logged information to the standard output stream (`System.out`).

To disable logging:

1. Set the `LogLevel` property to `0`.
2. To make sure that the new setting takes effect, restart your JDBC application and reconnect to the server.

Features

More information is provided on the following features of the Amazon Redshift JDBC Connector:

Data Types

The Amazon Redshift JDBC Connector supports many common data formats, converting between Redshift, SQL, and Java data types.

The following table lists the supported data type mappings.

Redshift Type	SQL Type	Java Type
BIGINT	SQL_BIGINT	Long
BOOLEAN	SQL_BIT	Boolean
CHAR	SQL_CHAR	String
DATE	SQL_TYPE_DATE	java.sql.Date
DECIMAL	SQL_NUMERIC	BigDecimal
DOUBLE PRECISION	SQL_DOUBLE	Double
GEOMETRY	SQL_ LONGVARBINARY	byte[]
INTEGER	SQL_INTEGER	Integer
OID	SQL_BIGINT	Long
REAL	SQL_REAL	Float

Redshift Type	SQL Type	Java Type
SMALLINT	SQL_SMALLINT	Short
SUPER	SQL_LONGVARCHAR	String
TEXT	SQL_VARCHAR	String
TIME	SQL_TYPE_TIME	java.sql.Time
TIMETZ	SQL_TYPE_TIME	java.sql.Time
TIMESTAMP	SQL_TYPE_ TIMESTAMP	java.sql.Timestamp
TIMESTAMPTZ	SQL_TYPE_ TIMESTAMP	java.sql.Timestamp
VARCHAR	SQL_VARCHAR	String

TCP Keepalives

By default, the Amazon Redshift JDBC Connector is configured to use TCP keepalives to verify the status of a connection and prevent it from timing out. After you connect to a Redshift server, the connector automatically sends keepalive packets to the server. If the server does not respond, then the connector returns an indication that the connection is broken.

For information about configuring settings for TCP keepalives, see Configuring TCP Keepalives on page 19.

Prepared Statement Support

The Amazon Redshift JDBC Connector supports prepared statements. You can use prepared statements to improve the performance of parameterized queries that need to be executed multiple times during the same connection.

A prepared statement is a SQL statement that is compiled on the server side but not executed immediately. The compiled statement is stored on the server as a `PreparedStatement` object until you close the object or the connection. While that object exists, you can execute the prepared statement as many times as needed using different parameter values, without having to compile the statement again. This reduced overhead enables the set of queries to be executed more quickly.

Note:

For more information about prepared statements, see "Using Prepared Statements" in the *JDBC Basics* tutorial from Oracle: https://docs.oracle.com/javase/tutorial/jdbc/basics/prepared.html.

You can prepare a statement that contains multiple queries. For example, the following prepared statement contains two INSERT queries:

```
PreparedStatement pstmt = conn.prepareStatement("INSERT INTO
MyTable VALUES (1, 'abc'); INSERT INTO CompanyTable VALUES
(1, 'abc');");
```

However, be aware that these queries cannot depend on the results of other queries that are specified within the same prepared statement. Because queries are not executed during the prepare step, the results have not been returned yet, and therefore are not available to other queries that are part of the same prepared statement.

For example, the following prepared statement, which creates a table and then inserts values into that newly-created table, is not allowed:

```
PreparedStatement pstmt = conn.prepareStatement("CREATE
TABLE MyTable(col1 int, col2 varchar); INSERT INTO myTable
VALUES (1, 'abc');");
```

If you try to prepare this statement, the server returns an error stating that the destination table (myTable) does not exist yet. The CREATE query must be executed before the INSERT query can be prepared.

Note:

This process for handling multiple queries in a prepared statement is consistent with established standards.

Catalog Function Support

The Amazon Redshift JDBC Connector supports catalog functions such as `getSchemas`, `getTables`, and `getColumns` to return data in result sets.

However, be aware that the connector's behavior differs from the standard JDBC specifications when using an empty string in `catalog`, `schemaPattern`, `tableNamePattern`, or `columnNamePattern` for any catalog functions.

For example, when an empty string is used in `schemaPattern`, it is supposed to retrieve data without a schema. The connector treats empty strings as NULL, meaning

that the schema name is not used to narrow the search. Therefore, the same results are returned whether an empty string or NULL is used.

Security and Authentication

To protect data from unauthorized access, Redshift data stores require all connections to be authenticated using user credentials. Some data stores also require connections to be made over the Secure Sockets Layer (SSL) protocol, either with or without one-way authentication.

The Amazon Redshift JDBC Connector provides full support for these authentication protocols. For detailed configuration instructions, see Configuring Authentication and SSL on page 14.

The SSL version that the connector supports depends on the JVM version that you are using. For information about the SSL versions that are supported by each version of Java, see "Diagnosing TLS, SSL, and HTTPS" on the Java Platform Group Product Management Blog: https://blogs.oracle.com/java-platform-group/entry/diagnosing_tls_ssl_and_https.

Note:

The SSL version used for the connection is the highest version that is supported by both the connector and the server, which is determined at connection time.

Connector Configuration Options

Connector Configuration Options lists and describes the properties that you can use to configure the behavior of the Amazon Redshift JDBC Connector.

You can set configuration properties using the connection URL. For more information, see Building the Connection URL on page 13.

AccessKeyID

Default Value	Data Type	Required
None	String	No

Description

The IAM access key for the IAM user or role. This can usually by found by looking at an existing connection string or user profile. If this is specified, then `SecretAccessKey` must also be specified.

AllowDBUserOverride

Default Value	Data Type	Required
`0`	String	No

Description

This option specifies whether the connector uses the DbUser value from the SAML assertion or the value that is specified in the `DbUser` connection property in the connection URL.

- `1`: The connector uses the DbUser value from the SAML assertion.

  If the SAML assertion does not specify a DbUser, the connector uses the value specified in the `DbUser` connection property. If the connection property also does not specify a value, the connector uses the value specified in the connection profile.
- `0`: The connector uses the DbUser value specified in the `DbUser` connection property.

If the `DbUser` connection property does not specify a value, the connector uses the value specified in the connection profile. If the connection profile also does not specify a value, the connector uses the value from the SAML assertion.

App_ID

Default Value	Data Type	Required
None	String	Yes, if authenticating through the Okta service.

Description

The Okta-provided unique ID associated with your Redshift application.

App_Name

Default Value	Data Type	Required
None	String	No

Description

The name of the Okta application that you use to authenticate the connection to Redshift.

AuthMech

Default Value	Data Type	Required
`DISABLE`	String	No

Description

The authentication mechanism to use. Set the property to one of the following values:

- `DISABLE`: The connector connects to the server without using TLS/SSL.

- `ALLOW`: First, the connector attempts to connect to the server without using TLS/SSL. If the connection fails, then the connector will attempt to connect through TLS/SSL.
- `REQUIRE`: The connector connects to the server through TLS/SSL.
- `PREFER`: First, the connector attempts to connect to the server through TLS/SSL. If the connection fails, then the connector will attempt to connect without using TLS/SSL.

Note:

Alternatively, you can configure the `SSL` property. For more information, see SSL on page 41.

AutoCreate

Default Value	Data Type	Required
`false`	String	No

Description

This option specifies whether the connector causes a new user to be created when the specified user does not exist.

- `true`: If the user specified by either DbUser or UID does not exist, a new user with that name is created.
- `false`: The connector does not cause new users to be created. If the specified user does not exist, the authentication fails.

BlockingRowsMode

Default Value	Data Type	Required
`0`	Integer	No

Description

The number of rows at a time that the connector holds in memory when returning query results. After one set of rows is discarded, the next set is loaded in its place.

When this property is set to `0`, the connector returns the entire query result into memory.

Client_ID

Default Value	Data Type	Required
None	String	Yes, if authenticating through the Azure AD service.

Description

The Client ID to use when authenticating the connection using the Azure AD service.

Client_Secret

Default Value	Data Type	Required
None	String	Yes, if authenticating through the Azure AD service.

Description

The Client Secret to use when authenticating the connection using the Azure AD service.

ClusterID

Default Value	Data Type	Required
None	String	No

Description

The name of the Redshift cluster you want to connect to.

DatabaseMetadataCurrentDbOnly

Default Value	Data Type	Required
1	Boolean	No

Description

This option specifies whether the connector returns metadata from multiple databases and clusters.

- 1: The connector only returns metadata from the current database.
- 0: The connector returns metadata across multiple Redshift databases and clusters (only if the server used to connect supports it).

DbGroups

Default Value	Data Type	Required
None	String	No

Description

A comma-separated list of existing database group names that the DbUser joins for the current session. If not specified, defaults to PUBLIC.

DbGroupsFilter

Default Value	Data Type	Required
None	String	No

Description

The regular expression you can specify to filter DbGroups that are received from the SAML response to Redshift when using Azure, Browser Azure, and Browser SAML authentication types.

DbUser

Default Value	Data Type	Required
None	String	No

Description

The user ID to use with your Redshift account. You can use an ID that does not currently exist if you have enabled the `AutoCreate` property.

DisableIsValidQuery

Default Value	Data Type	Required
`False`	Boolean	No

Description

This option specifies whether the connector submits a new database query when using the Connection.isValid() method to determine whether the database connection is active.

- `true`: The connector does not submit a query when using Connection.isValid() to determine whether the database connection is active. This may cause the connector to incorrectly identify the database connection as active if the database server has shut down unexpectedly.
- `false`: The connector submits a query when using Connection.isValid() to determine whether the database connection is active.

FilterLevel

Default Value	Data Type	Required
`NOTICE`	String	No

Description

The minimum severity level that an error or notice message from the server must be at before the client will process it. The following values are possible, in order from lowest

to highest severity:

- DEBUG
- INFO
- NOTICE
- WARNING
- LOG
- ERROR

ForceLowercase

Default Value	Data Type	Required
false	Boolean	No

Description

This option specifies whether the connector lowercases all DbGroups sent from the identity provider to Redshift when using SSO authentication

- true: The connector lowercases all DbGroups that are sent from the identity provider.
- false: The connector does not alter DbGroups.

IAMDuration

Default Value	Data Type	Required
900	Integer	No

Description

The length of time, in seconds, until the temporary IAM credentials expire. Minimum value 900, maximum value 3600.

IdP_Host

Default Value	Data Type	Required
None	String	No

Description

The IdP (identity provider) host you are using to authenticate into Redshift. This can be specified in either the connection string or in a profile.

IdP_Port

Default Value	Data Type	Required
None	String	No

Description

The port used by an IdP (identity provider). This can be specified in either the connection string or in a profile.

IdP_Tenant

Default Value	Data Type	Required
None	String	Yes, if authenticating through the Azure AD service.

Description

The Azure AD tenant ID for your Redshift application.

IdP_Response_Timeout

Default Value	Data Type	Required
120	Integer	No

Description

The amount of time, in seconds, that the connector waits for the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

Listen_Port

Default Value	Data Type	Required
7890	Integer	No

Description

The port that the connector uses to get the SAML response from the identity provider when using the SAML or Azure AD services through a browser plugin.

Login_URL

Default Value	Data Type	Required
None	String	Yes, if authenticating with the SAML or Azure AD services through a browser plugin.

Description

The URL for the resource on the identity provider's website when using the SAML or Azure AD services through a browser plugin.

LoginTimeout

Default Value	Data Type	Required
0	Integer	No

Description

The number of seconds to wait before timing out when connecting and authenticating to the server. If establishing the connection takes longer than this threshold, then the connection is aborted.

When this property is set to 0, connections do not time out.

loginToRp

Default Value	Data Type	Required
urn:amazon:webservices	String	No

Description

The relying party trust you want to use for the AD FS authentication type.

LogLevel

Default Value	Data Type	Required
None	Integer or String	No

Description

Use this property to enable or disable logging in the connector and to specify the amount of detail included in log files.

! Important:

Only enable logging long enough to capture an issue. Logging decreases performance and can consume a large quantity of disk space.

Set the property to one of the following numbers:

- `0`: Disable all logging.
- `1`: Enable logging on the FATAL level, which logs very severe error events that will lead the connector to abort.
- `2`: Enable logging on the ERROR level, which logs error events that might still allow the connector to continue running.
- `3`: Enable logging on the WARNING level, which logs events that might result in an error if action is not taken.
- `4`: Enable logging on the INFO level, which logs general information that describes the progress of the connector.
- `5`: Enable logging on the DEBUG level, which logs detailed information that is useful for debugging the connector.
- `6`: Enable logging on the TRACE level, which logs all connector activity.

When logging is enabled, the connector produces the following log files in the location specified in the `LogPath` property:

- A `RedshiftJDBC_driver.log` file that logs connector activity that is not specific to a connection.
- A `RedshiftJDBC_connection_[Number].log` and `RedshiftJDBC_connection_ext_[Number].log` file for each connection made to the database, where *[Number]* is a number that identifies each log file. These files log connector activity that is specific to the connection.

If the `LogPath` value is invalid, then the connector sends the logged information to the standard output stream (`System.out`).

LogPath

Default Value	Data Type	Required
The current working directory	String	No

Description

The full path to the folder where the connector saves log files when logging is enabled.

Note:

To make sure that the connection URL is compatible with all JDBC applications, it is recommended that you escape the backslashes (\) in your file path by typing another backslash.

OpenSourceSubProtocolOverride

Default Value	Data Type	Required
FALSE	String	No

Description

This property specifies whether the Amazon Redshift JDBC Connector can be loaded into the same class path as the open-source PostgreSQL driver.

- TRUE: Both connectors can be loaded into the same class path.
- FALSE: The connectors cannot be loaded into the same class path.

Partner_SPID

Default Value	Data Type	Required
None	String	No

Description

The partner SPID (service provider ID) value to use when authenticating the connection using the PingFederate service.

Password

Default Value	Data Type	Required
None	String	No.

Description

When connecting using IAM authentication through an IDP, this is the password for the IDP_Host server. When using standard authentication this can be used for the Redshiftdatabase password instead of PWD.

Plugin_Name

Default Value	Data Type	Required
None	String	No

Description

The fully-qualified class name to implement a specific credentials provider plugin.

The following provider options are supported:

- `AdfsCredentialsProvider` (Active Directory Federation Service)
- `AzureCredentialsProvider` (Microsoft Azure Active Directory (AD) Service)
- `BrowserAzureCredentialsProvider` (Browser Microsoft Azure Active Directory (AD) Service)
- `BrowserSamlCredentialsProvider` (Browser SAML for SAML services such as Okta or Ping)
- `OktaCredentialsProvider` (Okta Service)
- `PingCredentialsProvider` (PingFederate Service)

Preferred_Role

Default Value	Data Type	Required
None	String	No

Description

The IAM role you want to assume during the connection to Redshift.

Profile

Default Value	Data Type	Required
None	String	No.

Description

The name of the profile to use, containing any additional connection properties not specified in the connection string. Used for IAM authentication.

PWD

Default Value	Data Type	Required
None	String	Yes

Description

The password corresponding to the Redshift user name that you provided using the property UID on page 45.

QueryGroup

Default Value	Data Type	Required
None	String	No

Description

The label to apply to all queries made from the Redshift connection.

ReadOnly

Default Value	Data Type	Required
`false`	Boolean	No

Description

This property specifies whether the connector is in read-only mode.

- `true`: The connection is in read-only mode, and cannot write to the data store.
- `false`: The connection is not in read-only mode, and can write to the data store.

SecretAccessKey

Default Value	Data Type	Required
None	String	No

Description

The IAM access key for the user or role. If this is specified, then `AccessKeyID` must also be specified.

SelectorProvider

Default Value	Data Type	Required
None	String	No

Description

The fully-qualified class path for a class that extends `java.nio.channels.spi.SelectorProvider` and opens the SocketChannel. Using a custom SelectorProvider enables you to customize the SocketChannel that the connector uses.

SelectorProviderArg

Default Value	Data Type	Required
None	String	No

Description

A string argument to pass to the constructor of the class indicated by the `SelectorProvider` property.

SessionToken

Default Value	Data Type	Required
None	String	No

Description

The temporary IAM session token associated with the IAM role you are using to authenticate.

SocketTimeout

Default Value	Data Type	Required
`0`	Integer	No

Description

The number of seconds to wait during socket read operations before timing out. If the operation takes longer than this threshold, then the connection is closed.

When this property is set to `0`, the connection does not time out.

SSL

Default Value	Data Type	Required
`TRUE`	String	No

Description

Use this property to enable or disable SSL for the connection. The following values are possible:

- `TRUE`: The connector connects to the server through SSL.
- `FALSE`: The connector connects to the server without using SSL. This option is not supported with IAM authentication.

Note:

Alternatively, you can configure the `AuthMech` property. For more information, see AuthMech on page 27.

SSL_Insecure

Default Value	Data Type	Required
`true`	String	No

Description

This property indicates whether the IDP hosts server certificate should be verified.

`true`: The connector does not check the authenticity of the IDP server certificate.

`false`: The connector checks the authenticity of the IDP server certificate.

SSLCert

Default Value	Data Type	Required
None	String	Yes, if `SSLKey` is specified.

Description

The full path of a `.pem` or `.crt` file containing additional trusted CA certificates for verifying the Redshift Server instance when using SSL.

SSLFactory

Default Value	Data Type	Required
None	String	No

Description

The SSL factory to use when connecting to the server through TLS/SSL without using a server certificate. The following values are possible:

- `org.postgresql.ssl.NonValidatingFactory`
- `com.amazon.redshift.ssl.NonValidatingFactory`

SSLKey

Default Value	Data Type	Required
None	String	Yes, if `SSLCert` is specified.

Description

The full path of a `.der` file containing the PKCS8 key file for verifying the certificates specified in `SSLCert`.

SSLMode

Default Value	Data Type	Required
`verify-ca`	String	No

Description

Use this property to specify how the connector validates certificates when TLS/SSL is enabled. The following values are possible:

- `verify-ca`: The connector verifies that the certificate comes from a trusted certificate authority (CA).
- `verify-full`: The connector verifies that the certificate comes from a trusted CA and that the host name in the certificate matches the host name specified in the connection URL.

SSLPassword

Default Value	Data Type	Required
0	String	Yes, if SSLKey is specified and the key file is encrypted.

Description

The password for the encrypted key file specified in SSLKey.

SSLRootCert

Default Value	Data Type	Required
None	String	No

Description

The full path of a .pem or .crt file containing the root CA certificate for verifying the Redshift Server instance when using SSL.

TCPKeepAlive

Default Value	Data Type	Required
TRUE	String	No

Description

Use this property to enable or disable TCP keepalives. The following values are possible:

- TRUE: The connector uses TCP keepalives to prevent connections from timing out.
- FALSE: The connector does not use TCP keepalives.

TCPKeepAliveMinutes

Default Value	Data Type	Required
5	Integer	No

Description

The number of minutes of inactivity before the connector starts sending TCP keepalive packets.

UID

Default Value	Data Type	Required
None	String	Yes

Description

The user name that you use to access the database.

UnknownLength

Default Value	Data Type	Required
None	String	No

Description

The length and precision that the connector reports for data types of unknown length when the `DataBaseMetaData.getColumns()` method is called.

User

Default Value	Data Type	Required
None	String	No.

Description

When connecting using IAM authentication through an IDP, this is the user name for the idp_host server. When using standard authentication this can be used for the Redshiftdatabase user name.

Contact Us

For support, check the Amazon Redshift Forum at https://forums.aws.amazon.com/forum.jspa?forumID=155 or open a support case using the AWS Support Center at https://aws.amazon.com/support

Third-Party Trademarks

Simba, the Simba logo, SimbaEngine, SimbaEngine C/S, SimbaExpress and SimbaLib are registered trademarks of Simba Technologies Inc.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

All other trademarks are trademarks of their respective owners.