
AWS Schema Conversion Tool

User Guide

Version 1.0



AWS Schema Conversion Tool: User Guide

Copyright © 2017 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What Is the AWS Schema Conversion Tool?	1
Converting Your Schema	1
Additional Features	2
Feedback	2
About the User Interface	4
The Project Window	4
Source Schema Filter	5
Keyboard Shortcuts	6
Related Topics	6
Related Topics	6
Installing and Updating	7
Installing the AWS Schema Conversion Tool	7
Installing the Required Database Drivers	8
Installing JDBC Drivers on Linux	9
Storing Driver Paths in the Global Settings	10
Updating the AWS Schema Conversion Tool	11
Related Topics	11
Getting Started	12
Before You Begin	12
Starting SCT	13
Creating a Project	13
Connecting to Your Source Database	14
Connecting to Your Target Database	14
Converting Your Schema	15
Required Database Privileges	16
Source: Amazon Redshift	16
Source: Greenplum	16
Source: Microsoft SQL Server	17
Source: Microsoft SQL Server Data Warehouse	17
Source: MySQL	17
Source: Netezza	17
Source: Oracle	18
Source: Oracle Data Warehouse	18
Source: PostgreSQL	18
Source: Teradata	18
Source: Vertica	18
Target: Amazon Aurora (MySQL)	19
Target: Amazon Aurora (PostgreSQL)	19
Target: Amazon Redshift	19
Target: Microsoft SQL Server	20
Target: MySQL	20
Target: Oracle	20
Target: PostgreSQL	23
Related Topics	23
Getting Started Converting Database Schema	24
Before You Begin	24
Converting Your Schema	24
The Database Migration Assessment Report	26
Applying the Converted Schema to Your Target DB Instance	30
Related Topics	31
Getting Started Converting Data Warehouse Schema	32
Before You Begin	32
Choosing Optimization Strategies and Rules	32
Collecting or Uploading Statistics	33

Converting Your Schema	35
Managing and Customizing Keys	36
The Database Migration Assessment Report	37
Applying the Converted Schema to Your Target Database	41
Related Topics	42
Connecting to Your Source Database	43
Connecting to an Amazon Redshift Source Database	43
Related Topics	45
Connecting to a Greenplum Source Database	45
Related Topics	47
Connecting to a Microsoft SQL Server Source Database	47
Related Topics	49
Connecting to a Microsoft SQL Server Data Warehouse Source Database	49
Related Topics	51
Connecting to a MySQL Source Database	51
Related Topics	53
Connecting to a Netezza Source Database	53
Related Topics	55
Connecting to an Oracle Source Database	55
Related Topics	58
Connecting to an Oracle Data Warehouse Source Database	58
Related Topics	61
Connecting to a PostgreSQL Source Database	61
Related Topics	63
Connecting to a Teradata Source Database	63
Related Topics	65
Connecting to a Vertica Source Database	65
Related Topics	67
Working with Databases	68
Converting Database Schema	69
Creating Mapping Rules	70
Converting Your Schema	72
Creating and Using the Assessment Report	76
Handling Manual Conversions	80
Updating and Refreshing Your Converted Schema	81
Saving and Applying Your Schema	82
The Extension Pack and AWS Services for Databases	85
Using AWS Services to Emulate Database Functionality	85
Before You Begin	85
Applying the Extension Pack	86
Working with Data Warehouses	87
Converting Data Warehouse Schema	88
Choosing Optimization Strategies and Rules	90
Collecting or Uploading Statistics	91
Creating Mapping Rules	92
Converting Your Schema	94
Managing and Customizing Keys	98
Creating and Using the Assessment Report	99
Handling Manual Conversions	103
Updating and Refreshing Your Converted Schema	104
Saving and Applying Your Schema	105
The Extension Pack and Python Libraries for Data Warehouses	109
Using AWS Services to Upload Custom Python Libraries	109
Before You Begin	109
Applying the Extension Pack	110
Related Topics	110
Optimizing Amazon Redshift	111

Before You Begin	111
Optimizing Your Amazon Redshift Database	111
Related Topics	112
Working with AWS DMS	113
Before You Begin	113
Credentials	113
Creating a Task	113
Running and Monitoring a Task	114
Related Topics	114
Using Data Extraction Agents	115
Prerequisite Settings	116
S3 Settings	116
Security Settings	116
Related Topics	117
Installing, Configuring, and Starting Agents	117
Installing Agents	118
Configuring Agents	119
Starting Agents	120
Related Topics	120
Registering Agents	121
Related Topics	122
Data Extraction Filters	122
Related Topics	123
Data Extraction Tasks	123
Related Topics	125
Data Extraction Task Output	125
Related Topics	126
Best Practices and Troubleshooting	126
Related Topics	127
Converting Application SQL	128
Before You Begin	128
Overview of Converting Application SQL	128
Creating Application Conversion Projects	129
Analyzing and Converting Your SQL Code	132
Creating and Using the Assessment Report	133
Creating an Application Assessment Report	133
Editing and Saving Your Converted SQL Code	134
Storing AWS Profiles	135
Storing AWS Credentials	135
Setting the Default Profile for a Project	137
Related Topics	137
Best Practices	138
General Memory Management and Performance Options	138
Configuring Additional Memory	138
Related Topics	138
Troubleshooting	140
Cannot load objects from an Oracle source database	140
Related Topics	140
Reference	141
Microsoft SQL Server to MySQL Supported Schema Conversion	141
Statements	141
Procedures	145
Flow Control	145
Functions	146
Operators	153
Transactions	156
Data Types	157

Data Definition Language (DDL)	159
Cursors	160
Related Topics	161
Microsoft SQL Server to PostgreSQL Supported Schema Conversion	161
Data Definition Language (DDL)	161
Data Manipulation Language (DML)	164
Data Types	169
Database Mail	171
Functions	172
Operators	182
Other	184
Service Broker	185
SQL Server Agent	186
SQL Server Backup	189
T-SQL	189
Microsoft SQL Server to PostgreSQL Conversion Issue Reference	191
MySQL to PostgreSQL Supported Schema Conversion	215
DDL	215
DML	217
Functions	220
Data Types	229
MySQL to PostgreSQL Conversion Reference	233
Oracle to MySQL Supported Schema Conversion	245
Statements	246
Procedures	250
Flow Control	251
Packages	251
Functions	251
Operators	261
Data Types	264
Data Definition Language (DDL)	266
Cursors	270
Hints	270
Exceptions	273
Related Topics	274
Oracle to PostgreSQL Supported Schema Conversion	274
DDL	274
DML	277
PLSQL	298
Data Types	303
Oracle to PostgreSQL Conversion Reference	304
PostgreSQL to MySQL Supported Schema Conversion	332
DDL	333
DML	334
PL/pgSQL	343
PostgreSQL to MySQL Conversion Reference	350
Related Topics	384
Document History	385

What Is the AWS Schema Conversion Tool?

You can use the AWS Schema Conversion Tool (AWS SCT) to convert your existing database schema from one database engine to another. You can convert relational OLTP schema, or data warehouse schema. Your converted schema is suitable for an Amazon Relational Database Service (Amazon RDS) MySQL DB instance, an Amazon Aurora DB cluster, an Amazon RDS PostgreSQL DB instance, or an Amazon Redshift cluster.

AWS SCT supports the following OLTP conversions.

Source Database	Target Database on Amazon RDS
Microsoft SQL Server (version 2008 and later)	Amazon Aurora (MySQL or PostgreSQL), Microsoft SQL Server, MySQL, PostgreSQL
MySQL (version 5.5 and later)	Amazon Aurora (PostgreSQL), MySQL, PostgreSQL You can migrate schema and data from MySQL to an Amazon Aurora (MySQL) DB cluster without using AWS SCT. For more information, see Migrating Data to an Amazon Aurora DB Cluster .
Oracle (version 10.2 and later)	Amazon Aurora (MySQL or PostgreSQL), MySQL, Oracle, PostgreSQL
PostgreSQL (version 9.1 and later)	Amazon Aurora (MySQL), MySQL, PostgreSQL

AWS SCT supports the following data warehouse conversions.

Source Database	Target Database on Amazon Redshift
Greenplum Database (version 4.3 and later)	Amazon Redshift
Microsoft SQL Server (version 2008 and later)	Amazon Redshift
Netezza (version 7.0.3 and later)	Amazon Redshift
Oracle (version 10 and later)	Amazon Redshift
Teradata (version 13 and later)	Amazon Redshift
Vertica (version 7.2.2 and later)	Amazon Redshift

Converting Your Schema

AWS SCT provides a project-based user interface to automatically convert the database schema of your source database into a format compatible with your target Amazon RDS instance. If schema from your

source database can't be converted automatically, AWS SCT provides guidance on how you can create equivalent schema in your target Amazon RDS database.

For information about how to install AWS SCT, see [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#).

To get started with AWS SCT, and create your first project, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

To start converting your schema, see [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool \(p. 69\)](#) or [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#).

Additional Features of the AWS Schema Conversion Tool

In addition to converting your existing database schema from one database engine to another, AWS SCT has some additional features that help you move your data and applications to the cloud:

- You can use data extraction agents to extract data from your data warehouse to prepare to migrate it to Amazon Redshift. To manage the data extraction agents, you can use AWS SCT. For more information, see [Using Data Extraction Agents \(p. 115\)](#).
- You can use AWS SCT to create AWS DMS endpoints and tasks. You can run and monitor these tasks from AWS SCT. For more information, see [Working with the AWS Database Migration Service Using the AWS Schema Conversion Tool \(p. 113\)](#).
- In some cases, database features can't be converted to equivalent Amazon RDS or Amazon Redshift features. The AWS SCT extension pack wizard can help you install AWS Lambda functions and Python libraries to emulate the features that can't be converted. For more information, see [The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases \(p. 85\)](#) and [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses \(p. 109\)](#).
- You can use AWS SCT to optimize your existing Amazon Redshift database. AWS SCT recommends sort keys and distribution keys to optimize your database. For more information, see [Optimizing Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 111\)](#).
- You can use AWS SCT to copy your existing on-premises database schema to an Amazon RDS DB instance running the same engine. You can use this feature to analyze potential cost savings of moving to the cloud and of changing your license type.
- You can use AWS SCT to convert SQL in your C++, C#, Java, or other application code. You can view, analyze, edit, and save the converted SQL code. For more information, see [Converting Application SQL by Using the AWS Schema Conversion Tool \(p. 128\)](#).

Providing Customer Feedback

You can provide feedback about the AWS Schema Conversion Tool. You can file a bug report, you can submit a feature request, or you can provide general information.

To provide feedback about AWS SCT.

1. Start the AWS Schema Conversion Tool.
2. Open the **Help** menu and then choose **Leave Feedback**. The **Leave Feedback** dialog box appears.
3. For **Area**, choose **Information**, **Bug report**, or **Feature request**.
4. For **Source database**, choose your source database. Choose **Any** if your feedback is not specific to a particular database.
5. For **Target database**, choose your target database. Choose **Any** if your feedback is not specific to a particular database.
6. For **Title**, type a title for your feedback.
7. For **Message**, type your feedback.
8. Choose **Send** to submit your feedback.

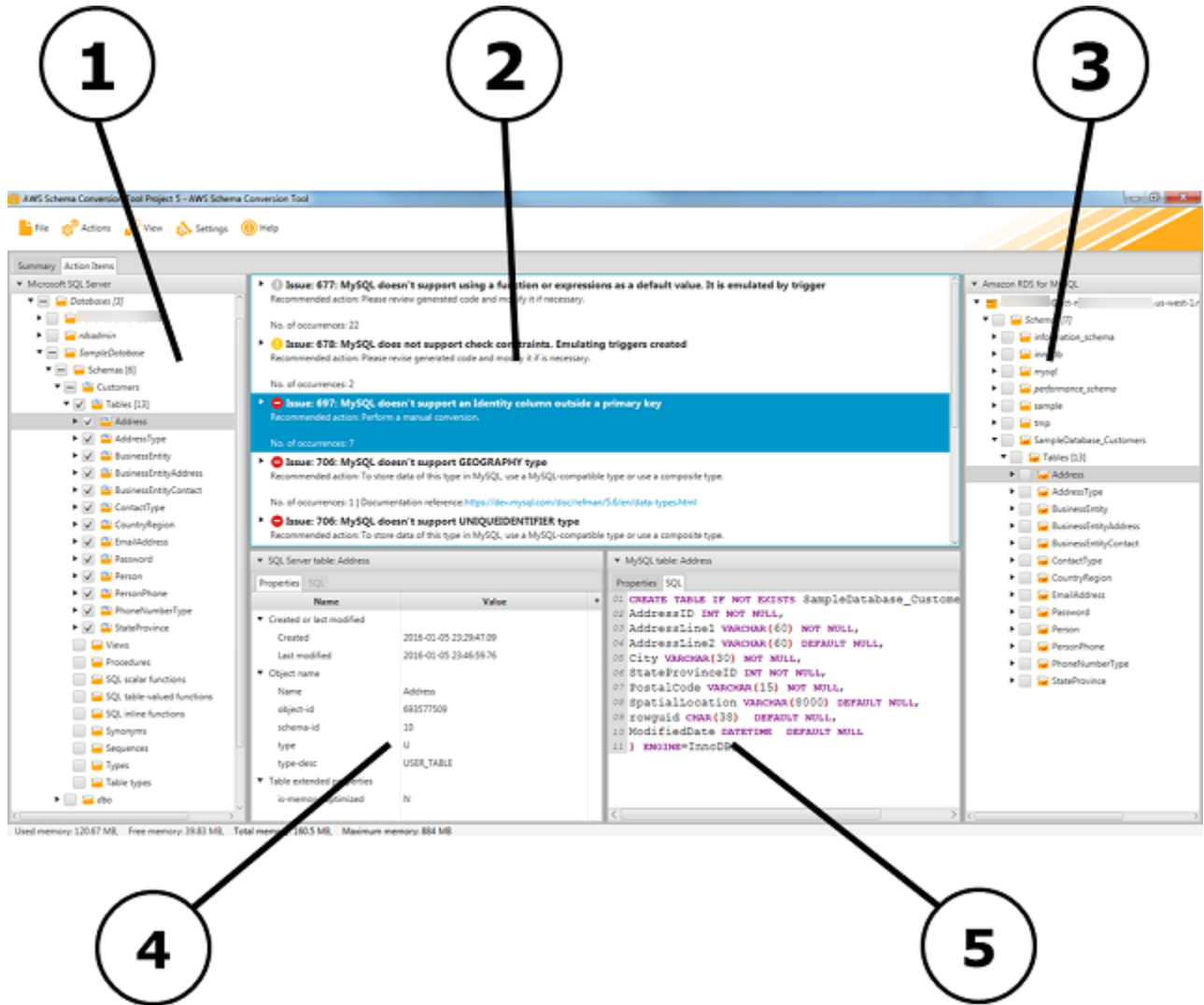
About the AWS Schema Conversion Tool User Interface

The following sections help you work with the AWS Schema Conversion Tool (AWS SCT) user interface.

The Project Window

The illustration following is what you see in the AWS Schema Conversion Tool when you create a schema migration project, and then convert a schema.

1. In the left panel, the schema from your source database is presented in a tree view. Your database schema is "lazy loaded." In other words, when you select an item from the tree view, AWS SCT gets and displays the current schema from your source database.
2. In the top middle panel, action items appear for schema elements from the source database engine that couldn't be converted automatically to the target database engine.
3. In the right panel, the schema from your target DB instance is presented in a tree view. Your database schema is "lazy loaded." That is, at the point when you select an item from the tree view, AWS SCT gets and displays the current schema from your target database.



4. In the lower left panel, when you choose a schema element, properties describing the source schema element and the SQL command to create that element in the source database are displayed.
5. In the lower right panel, when you choose a schema element, properties describing the target schema element and the SQL command to create that element in the target database are displayed. You can edit this SQL command and save the updated command with your project.

Source Schema Filter

Your source schema appears in a tree view in the left panel. If your source schema is very large, you might have a hard time locating specific objects in the tree. You can filter the objects that appear in the tree.

To filter the source schema, at the top right corner of the source schema panel, choose the filter icon as shown following.



When you filter the schema that appears in the tree, you don't change the objects that are converted when you convert your schema. The filter only changes what you see in the tree.

Keyboard Shortcuts for the AWS Schema Conversion Tool

The following are the keyboard shortcuts that you can use with the AWS Schema Conversion Tool (AWS SCT).

Keyboard Shortcut	Description
Ctrl+N	Create a new project.
Ctrl+O	Open an existing project.
Ctrl+S	Save an open project.
Ctrl+W	Create a new project by using the wizard.
Ctrl+L	Connect to the source database.
Ctrl+R	Connect to the target database.

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Related Topics

- [About the AWS Schema Conversion Tool User Interface \(p. 4\)](#)
- [Working with Databases \(p. 68\)](#)
- [Working with Data Warehouses \(p. 87\)](#)
- [Best Practices for the AWS Schema Conversion Tool \(p. 138\)](#)

Installing and Updating the AWS Schema Conversion Tool

The AWS Schema Conversion Tool (AWS SCT) is a standalone application that provides a project-based user interface. AWS SCT is available for Fedora Linux, macOS, Microsoft Windows, and Ubuntu Linux version 15.04. AWS SCT is supported only on 64-bit operating systems. AWS SCT also installs the Java Runtime Environment (JRE) version 8u45.

Installing the AWS Schema Conversion Tool

To install the AWS Schema Conversion Tool

1. Download the .zip file that contains the AWS SCT installer, using the link for your operating system, shown following:
 - [Fedora Linux download](#)
 - [macOS download](#)
 - [Microsoft Windows download](#)
 - [Ubuntu Linux version 15.04 download](#)
2. Extract the AWS SCT installer file for your operating system, shown following.

Operating System	File Name
Fedora Linux	aws-schema-conversion-tool-1.0. <i>build-number</i> .x86_64.rpm
macOS	AWS Schema Conversion Tool-1.0. <i>build-number</i> .dmg
Microsoft Windows	AWS Schema Conversion Tool-1.0. <i>build-number</i> .msi
Ubuntu Linux	aws-schema-conversion-tool-1.0. <i>build-number</i> .deb

3. Run the AWS SCT installer file extracted in the previous step. Use the instructions for your operating system, shown following.

Operating System	Install Instructions
Fedora Linux	Run the following command in the folder that you downloaded the file to: <code>sudo yum install aws-schema-conversion-tool-1.0.<i>build-number</i>.x86_64.rpm</code>
macOS	In Finder , open AWS Schema Conversion Tool-1.0. <i>build-number</i> .dmg. Drag AWS Schema Conversion Tool-1.0. <i>build-number</i> .dmg to the Applications folder.

Operating System	Install Instructions
Microsoft Windows	Double-click the file to run the installer.
Ubuntu Linux	Run the following command in the folder that you downloaded the file to: <pre>sudo dpkg -i aws-schema-conversion-tool-1.0.<i>build-number</i>.deb</pre>

4. Install the Java Database Connectivity (JDBC) drivers for your source and target database engines. For instructions and download links, see [Installing the Required Database Drivers \(p. 8\)](#).

Installing the Required Database Drivers

For the AWS Schema Conversion Tool to work correctly, you must install the JDBC drivers for your source and target database engines.

After you download the drivers, you give the location of the driver files. For more information, see [Storing Driver Paths in the Global Settings \(p. 10\)](#).

You can download the database drivers from the following locations.

Important

Install the latest version of the driver available. The versions in the table following are example version numbers.

Database Engine	Drivers	Download Location
Amazon Aurora (MySQL compatible)	mysql-connector-java-5.1.6.jar	https://www.mysql.com/products/connector/
Amazon Aurora (PostgreSQL compatible)	postgresql-9.4-1204-jdbc42.jar	https://jdbc.postgresql.org/download.html
Amazon Redshift	RedshiftJDBC41-1.1.10.1010.jar	http://docs.aws.amazon.com/redshift/latest/mgmt/configure-jdbc-connection.html
Greenplum Database	postgresql-9.4-1204-jdbc42.jar	https://jdbc.postgresql.org/
Microsoft SQL Server	sqljdbc4.jar	https://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774
MySQL	mysql-connector-java-5.1.6.jar	https://www.mysql.com/products/connector/
Netezza	nzjdbc.jar Use the client tools software. Install driver version 7.2.1, which is backwards compatible with data warehouse version 7.2.0.	http://www.ibm.com/support/knowledgecenter/SSULQD_7.2.1/com.ibm.nz.datacon.doc/c_datacon_plg_overview.html

Database Engine	Drivers	Download Location
Oracle	ojdbc7.jar Driver versions 7 and later are supported.	http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html
PostgreSQL	postgresql-9.4-1204-jdbc42.jar	https://jdbc.postgresql.org/download.html
Teradata	terajdbc4.jar tdgssconfig.jar	https://downloads.teradata.com/download/connectivity/jdbc-driver
Vertica	vertica-jdbc-7.2.3-0_all Driver versions 7.2.0 and later are supported.	https://my.vertica.com/download/vertica/client-drivers/

Installing JDBC Drivers on Linux

You can use the following steps to install the JDBC drivers on your Linux system for use with AWS SCT.

To install JDBC drivers on your Linux system

1. Create a directory to store the JDBC drivers in.

```
PROMPT>sudo mkdir -p /usr/local/jdbc-drivers
```

2. Install the JDBC driver for your database engine using the commands shown following.

Database Engine	Installation Commands
Amazon Aurora (MySQL compatible)	PROMPT> cd /usr/local/jdbc-drivers PROMPT> sudo tar xzvf /tmp/mysql-connector-java-X.X.X.tar.gz
Amazon Aurora (PostgreSQL compatible)	PROMPT> cd /usr/local/jdbc-drivers PROMPT> sudo cp -a /tmp/postgresql-X.X.X.jre7.tar .
Microsoft SQL Server	PROMPT> cd /usr/local/jdbc-drivers PROMPT> sudo tar xzvf /tmp/sqljdbc_X.X.X_enu.tar.gz
MySQL	PROMPT> cd /usr/local/jdbc-drivers PROMPT> sudo tar xzvf /tmp/mysql-connector-java-X.X.X.tar.gz
Oracle	PROMPT> cd /usr/local/jdbc-drivers PROMPT> sudo mkdir oracle-jdbc PROMPT> cd oracle-jdbc PROMPT> sudo cp -a /tmp/ojdbc7.jar .
PostgreSQL	PROMPT> cd /usr/local/jdbc-drivers

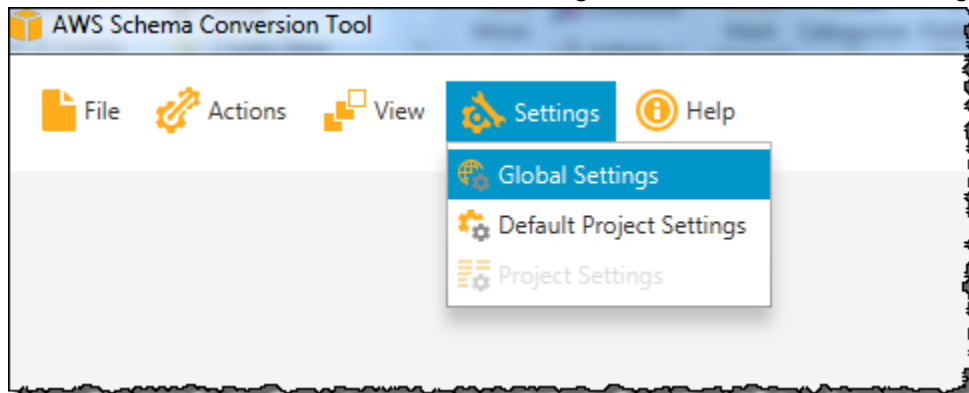
Database Engine	Installation Commands
	PROMPT> sudo cp -a /tmp/postgresql-X.X.X.jre7.tar .

Storing Driver Paths in the Global Settings

After you have downloaded and installed the required JDBC drivers, you can set the location of the drivers globally in the AWS SCT settings. If you don't set the location of the drivers globally, the application asks you for the location of the drivers when you connect to a database.

To update the driver file locations

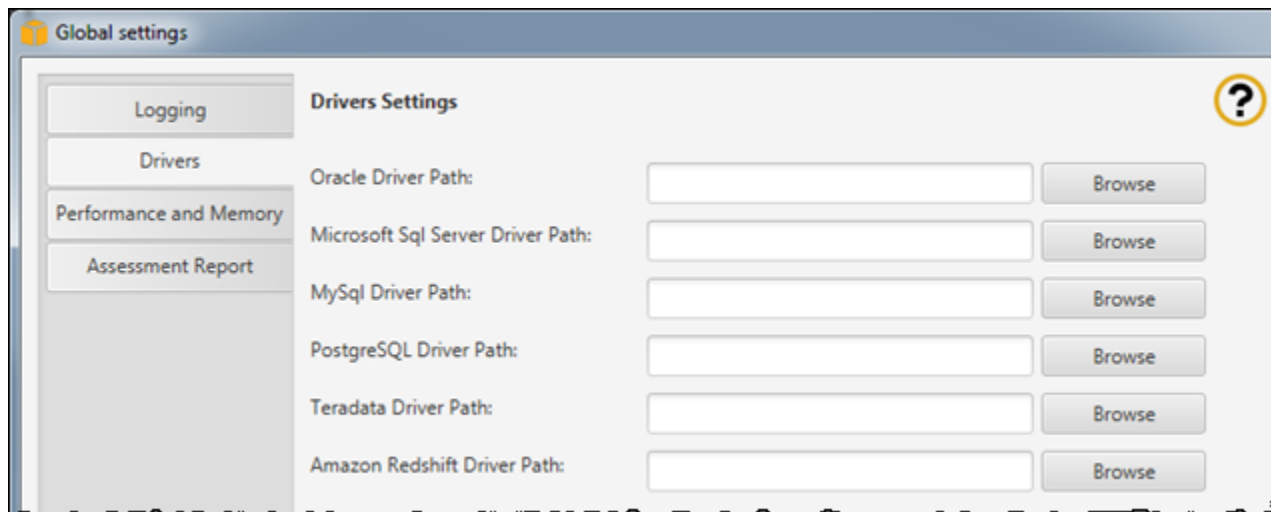
1. In the AWS Schema Conversion Tool, choose **Settings**, and then choose **Global Settings**.



2. For **Global settings**, choose **Drivers**. Add the file path to the JDBC driver for your source database engine and your target Amazon RDS DB instance database engine.

Note

For Teradata, you specify two drivers separated by a semicolon.



3. When you are finished adding the driver paths, choose **OK**.

Updating the AWS Schema Conversion Tool

To check whether new version of AWS SCT is available, choose **Help**, and then choose **Check for Updates**. If there is a newer version of AWS SCT available, you are prompted to download and install the newer version.

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)

Getting Started with the AWS Schema Conversion Tool

Following, you can find procedures that help you get started using the AWS Schema Conversion Tool (AWS SCT). AWS SCT provides a project-based user interface. Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

Before You Begin

Before you complete the procedures in this topic, you must first do the following:

- Install the AWS Schema Conversion Tool. For more information, see [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#).
- Create your target Amazon Relational Database Service (Amazon RDS) DB instance or Amazon Redshift database, if it doesn't already exist. For more information, see the following documentation.

Database Engine	Relevant Documentation
Amazon Aurora	Creating an Amazon Aurora DB Cluster Connecting to an Amazon Aurora DB Cluster
Amazon Redshift	Getting Started with Amazon Redshift
Microsoft SQL Server	Creating a DB Instance Running the Microsoft SQL Server Database Engine Connecting to a DB Instance Running the Microsoft SQL Server Database Engine
MySQL	Creating a DB Instance Running the MySQL Database Engine Connecting to a DB Instance Running the MySQL Database Engine
Oracle	Creating a DB Instance Running the Oracle Database Engine Connecting to a DB Instance Running the Oracle Database Engine
PostgreSQL	Creating a DB Instance Running the PostgreSQL Database Engine Connecting to a DB Instance Running the PostgreSQL Database Engine

- Verify that you have sufficient privileges for the source and target databases so that you can run AWS SCT. For more information, see [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#).

Starting the AWS Schema Conversion Tool

To start the AWS Schema Conversion Tool, use the instructions for your operating system shown following.

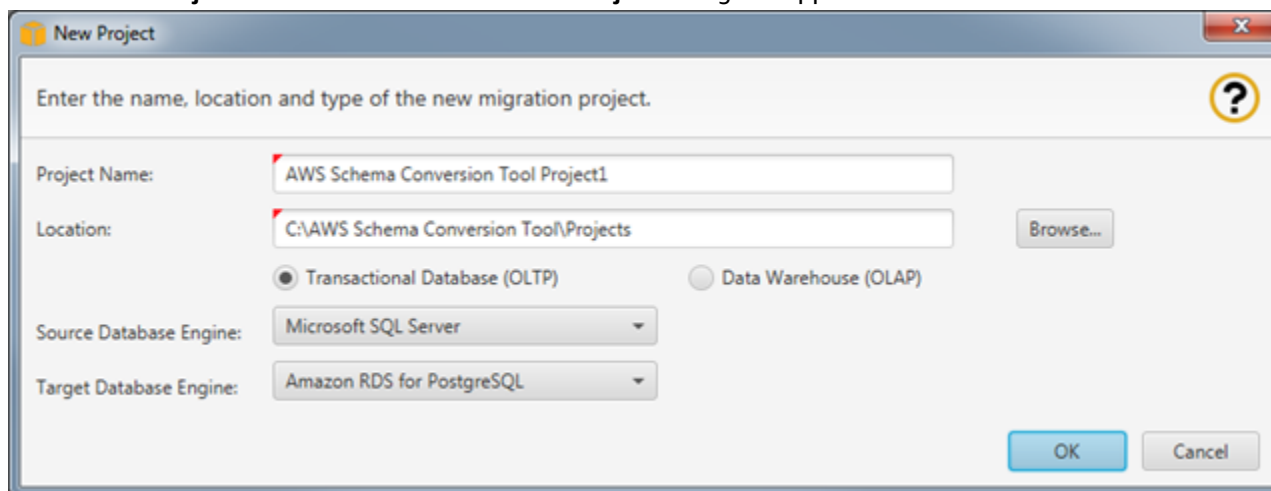
Operating System	Instructions
Fedora Linux	Run the following command: /opt/AWSSchemaConversionTool/AWSSchemaConversionTool
Microsoft Windows	Double-click the icon for the application.
Ubuntu Linux	Run the following command: /opt/AWSSchemaConversionTool/AWSSchemaConversionTool

Creating an AWS Schema Conversion Tool Project

The following procedure shows you how to create an AWS Schema Conversion Tool project.

To create your project

1. Start the AWS Schema Conversion Tool.
2. Choose **New Project** from the **File** menu. The **New Project** dialog box appears.



3. Add the following preliminary project information.

For This Parameter	Do This
Project Name	Type a name for your project, which is stored locally on your computer.
Location	Type the location for your local project file.
	Choose Transactional Database (OLTP) or Data Warehouse (OLAP) .

For This Parameter	Do This
Source DB Engine	(OLTP) Choose Microsoft SQL Server, MySQL, Oracle, or PostgreSQL. (OLAP) Choose Amazon Redshift, Greenplum, Microsoft SQL Server DW, Netezza, Oracle DW, Teradata, or Vertica.
Target DB Engine	(OLTP) Choose Amazon Aurora (MySQL compatible), Amazon Aurora (PostgreSQL compatible), Amazon RDS for Microsoft SQL Server, Amazon RDS for MySQL, Amazon RDS for Oracle, or Amazon RDS for PostgreSQL. (OLAP) Choose Amazon Redshift.

4. Choose **OK** to create your AWS SCT project.

Connecting to Your Source Database

The following topics show you how to connect to your source database. Choose the topic appropriate for your source database.

- [Connecting to an Amazon Redshift Source Database \(p. 43\)](#)
- [Connecting to a Greenplum Source Database \(p. 45\)](#)
- [Connecting to a Microsoft SQL Server Source Database \(p. 47\)](#)
- [Connecting to a Microsoft SQL Server Data Warehouse Source Database \(p. 49\)](#)
- [Connecting to a MySQL Source Database \(p. 51\)](#)
- [Connecting to a Netezza Source Database \(p. 53\)](#)
- [Connecting to an Oracle Source Database \(p. 55\)](#)
- [Connecting to an Oracle Data Warehouse Source Database \(p. 58\)](#)
- [Connecting to a PostgreSQL Source Database \(p. 61\)](#)
- [Connecting to a Teradata Source Database \(p. 63\)](#)
- [Connecting to a Vertica Source Database \(p. 65\)](#)

Connecting to Your Target Database

The following procedure shows you how to connect to your target database.

To connect to your target database

1. Start the AWS Schema Conversion Tool.
2. Choose **Connect to *target***, where *target* indicates the database engine for your target DB instance.
3. Add the following information to connect to your target Amazon RDS DB instance.

For This Parameter	Do This
Server name	Type the DNS name of your target DB instance.
Server port	Type the port used to connect to your target DB instance.

For This Parameter	Do This
User name and Password	Type the user name and password to connect to your target DB instance. Note AWS SCT uses the password to connect to your target database only when you create your project or choose the Connect to <i>target</i> option in a project, where <i>target</i> is your target database. To guard against exposing the password for your target database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your target database as needed.
Use SSL	Select this option if you want to use SSL to connect to your database. Provide additional information, as appropriate, on the SSL tab.

4. Choose **Test Connection** to verify that you can successfully connect to your target database.
5. Choose **OK** to connect to your target DB instance.

Converting Your Schema

After you get started by using the procedures in this topic, you can continue working with AWS SCT with transactional databases and data warehouses in the following topics:

- [Getting Started Converting Database Schema with the AWS Schema Conversion Tool \(p. 24\)](#)
- [Getting Started Converting Data Warehouse Schema with the AWS Schema Conversion Tool \(p. 32\)](#)

Required Database Privileges for Using the AWS Schema Conversion Tool

When you use the AWS Schema Conversion Tool (AWS SCT) to convert your database schema, you need certain privileges for the source and target databases. Following, you can find lists of these privileges.

- Source: Amazon Redshift (p. 16)
- Source: Greenplum (p. 16)
- Source: Microsoft SQL Server (p. 17)
- Source: Microsoft SQL Server Data Warehouse (p. 17)
- Source: MySQL (p. 17)
- Source: Netezza (p. 17)
- Source: Oracle (p. 18)
- Source: Oracle Data Warehouse (p. 18)
- Source: PostgreSQL (p. 18)
- Source: Teradata (p. 18)
- Source: Vertica (p. 18)

- Target: Amazon Aurora (MySQL) (p. 19)
- Target: Amazon Aurora (PostgreSQL) (p. 19)
- Target: Amazon Redshift (p. 19)
- Target: Microsoft SQL Server (p. 20)
- Target: MySQL (p. 20)
- Target: Oracle (p. 20)
- Target: PostgreSQL (p. 23)

Privileges for Amazon Redshift as a Source Database

The privileges required for Amazon Redshift as a source are listed following:

- USAGE ON SCHEMA *<schema_name>*
- SELECT ON ALL TABLES IN SCHEMA *<schema_name>*
- SELECT ON PG_CATALOG.PG_STATISTIC
- SELECT ON SVV_TABLE_INFO
- SELECT ON TABLE STV_BLOCKLIST
- SELECT ON TABLE STV_TBL_PERM

Privileges for Greenplum as a Source Database

The privileges required for Greenplum as a source are listed following:

- CONNECT ON DATABASE *<database_name>*
- USAGE ON SCHEMA *<schema_name>*

Privileges for Microsoft SQL Server as a Source Database

The privileges required for Microsoft SQL Server as a source are listed following:

- VIEW DEFINITION
- VIEW DATABASE STATE

Repeat the grant for each database whose schema you are converting.

Privileges for Microsoft SQL Server Data Warehouse as a Source Database

The privileges required for Microsoft SQL Server data warehouse as a source are listed following:

- VIEW DEFINITION
- VIEW DATABASE STATE
- SELECT ON SCHEMA :: *<schema_name>*

Repeat the grant for each database whose schema you are converting.

In addition, grant the following, and run the grant on the master database:

- VIEW SERVER STATE

Privileges for MySQL as a Source Database

The privileges required for MySQL as a source are listed following:

- SELECT ON *.*
- SELECT ON mysql.proc
- SHOW VIEW ON *.*

Privileges for Netezza as a Source Database

The privileges required for Netezza as a source are listed following:

- SELECT ON SYSTEM.DEFINITION_SCHEMA.SYSTEM VIEW
- SELECT ON SYSTEM.DEFINITION_SCHEMA.SYSTEM TABLE
- SELECT ON SYSTEM.DEFINITION_SCHEMA.MANAGEMENT TABLE
- LIST ON *<database_name>*
- LIST ON *<database_name>*.ALL.TABLE
- LIST ON *<database_name>*.ALL.EXTERNAL TABLE
- LIST ON *<database_name>*.ALL.VIEW
- LIST ON *<database_name>*.ALL.MATERIALIZED VIEW
- LIST ON *<database_name>*.ALL.PROCEDURE

- LIST ON `<database_name>.ALL.SEQUENCE`
- LIST ON `<database_name>.ALL.FUNCTION`
- LIST ON `<database_name>.ALL.AGGREGATE`

Privileges for Oracle as a Source Database

The privileges required for Oracle as a source are listed following:

- connect
- select_catalog_role
- select any dictionary

Privileges for Oracle Data Warehouse as a Source Database

The privileges required for Oracle Data Warehouse as a source are listed following:

- connect
- select_catalog_role
- select any dictionary

Privileges for PostgreSQL as a Source Database

The privileges required for PostgreSQL as a source are listed following:

- CONNECT ON DATABASE `<database_name>`
- USAGE ON SCHEMA `<database_name>`
- SELECT ON ALL TABLES IN SCHEMA `<database_name>`
- SELECT ON ALL SEQUENCES IN SCHEMA `<database_name>`

Privileges for Teradata as a Source Database

The privileges required for Teradata as a source are listed following:

- SELECT ON DBC

Privileges for Vertica as a Source Database

The privileges required for Vertica as a source are listed following:

- USAGE ON SCHEMA `<schema_name>`
- USAGE ON SCHEMA PUBLIC
- GRANT SELECT ON ALL TABLES IN SCHEMA `<schema_name>`
- SELECT ON ALL SEQUENCES IN SCHEMA `<schema_name>`
- EXECUTE ON ALL FUNCTIONS IN SCHEMA `<schema_name>`
- EXECUTE ON PROCEDURE `<schema_name.procedure_name(procedure_signature)>`

Privileges for Amazon Aurora (MySQL) as a Target Database

The privileges required for Amazon Aurora (MySQL) as a target are listed following:

- CREATE ON *.*
- ALTER ON *.*
- DROP ON *.*
- INDEX ON *.*
- REFERENCES ON *.*
- SELECT ON *.*
- CREATE VIEW ON *.*
- SHOW VIEW ON *.*
- TRIGGER ON *.*
- CREATE ROUTINE ON *.*
- ALTER ROUTINE ON *.*
- EXECUTE ON *.*
- SELECT ON mysql.proc

If your source database is Microsoft SQL Server, grant the additional privilege INSERT,UPDATE ON AWS_SQLSERVER_EXT.*

If your source database is Oracle, grant the additional privilege INSERT,UPDATE ON AWS_ORACLE_EXT.*

If your source database is PostgreSQL, grant the additional privilege INSERT,UPDATE ON AWS_POSTGRESQL_EXT.*

Privileges for Amazon Aurora (PostgreSQL) as a Target Database

The privileges required for Amazon Aurora (PostgreSQL) as a target are explained following.

If the new schema doesn't exist yet, grant the privilege CREATE ON DATABASE *<database_name>*

If the new schema already exists, grant the privilege INSERT ON ALL TABLES IN SCHEMA *<schema_name>*

In PostgreSQL, only the owner of a schema or a superuser can drop a schema. The owner can drop the schema, and all contained objects, even if the owner doesn't own all of the contained objects.

Privileges for Amazon Redshift as a Target Database

The privileges required for Amazon Redshift as a target are listed following:

- CREATE ON DATABASE *<database_name>*
- USAGE ON LANGUAGE plpythonu
- SELECT ON ALL TABLES IN SCHEMA pg_catalog

In Amazon Redshift, only the owner of a schema or a superuser can drop a schema. The owner can drop the schema, and all contained objects, even if the owner doesn't own all of the contained objects.

Privileges for Microsoft SQL Server as a Target Database

The privileges required for Microsoft SQL Server as a target are listed following:

- CREATE SCHEMA
- CREATE TABLE
- CREATE VIEW
- CREATE TYPE
- CREATE DEFAULT
- CREATE FUNCTION
- CREATE PROCEDURE
- CREATE ASSEMBLY
- CREATE AGGREGATE
- CREATE FULLTEXT CATALOG

Privileges for MySQL as a Target Database

The privileges required for MySQL as a target are listed following:

- CREATE ON *.*
- ALTER ON *.*
- DROP ON *.*
- INDEX ON *.*
- REFERENCES ON *.*
- SELECT ON *.*
- CREATE VIEW ON *.*
- SHOW VIEW ON *.*
- TRIGGER ON *.*
- CREATE ROUTINE ON *.*
- ALTER ROUTINE ON *.*
- EXECUTE ON *.*
- SELECT ON mysql.proc

If your source database is Microsoft SQL Server, grant the additional privilege INSERT,UPDATE ON AWS_SQLSERVER_EXT.*

If your source database is Oracle, grant the additional privilege INSERT,UPDATE ON AWS_ORACLE_EXT.*

If your source database is PostgreSQL, grant the additional privilege INSERT,UPDATE ON AWS_POSTGRESQL_EXT.*

Privileges for Oracle as a Target Database

The privileges required for Oracle as a target are listed following:

- SELECT_CATALOG_ROLE
- RESOURCE

- CONNECT
- alter user `<username>` quota unlimited on USERS;
- alter user `<username>` quota unlimited on IDX;
- alter user `<username>` quota unlimited on ARCH;
- alter user `<username>` quota unlimited on ARCH_IDX;
- DROP ANY CUBE BUILD PROCESS
- ALTER ANY CUBE
- CREATE ANY CUBE DIMENSION
- CREATE ANY ASSEMBLY
- ALTER ANY RULE
- SELECT ANY DICTIONARY
- ALTER ANY DIMENSION
- CREATE ANY DIMENSION
- ALTER ANY TYPE
- DROP ANY TRIGGER
- CREATE ANY VIEW
- ALTER ANY CUBE BUILD PROCESS
- CREATE ANY CREDENTIAL
- DROP ANY CUBE DIMENSION
- DROP ANY ASSEMBLY
- DROP ANY PROCEDURE
- ALTER ANY PROCEDURE
- ALTER ANY SQL TRANSLATION PROFILE
- DROP ANY MEASURE FOLDER
- CREATE ANY MEASURE FOLDER
- DROP ANY CUBE
- DROP ANY MINING MODEL
- CREATE ANY MINING MODEL
- DROP ANY EDITION
- CREATE ANY EVALUATION CONTEXT
- DROP ANY DIMENSION
- ALTER ANY INDEXTYPE
- DROP ANY TYPE
- CREATE ANY PROCEDURE
- CREATE ANY SQL TRANSLATION PROFILE
- CREATE ANY CUBE
- COMMENT ANY MINING MODEL
- ALTER ANY MINING MODEL
- DROP ANY SQL PROFILE
- CREATE ANY JOB
- DROP ANY EVALUATION CONTEXT
- ALTER ANY EVALUATION CONTEXT
- CREATE ANY INDEXTYPE
- CREATE ANY OPERATOR
- CREATE ANY TRIGGER
- DROP ANY ROLE

- DROP ANY SEQUENCE
- DROP ANY CLUSTER
- DROP ANY SQL TRANSLATION PROFILE
- ALTER ANY ASSEMBLY
- CREATE ANY RULE SET
- ALTER ANY OUTLINE
- UNDER ANY TYPE
- CREATE ANY TYPE
- DROP ANY MATERIALIZED VIEW
- ALTER ANY ROLE
- DROP ANY VIEW
- ALTER ANY INDEX
- COMMENT ANY TABLE
- CREATE ANY TABLE
- CREATE USER
- DROP ANY RULE SET
- CREATE ANY CONTEXT
- DROP ANY INDEXTYPE
- ALTER ANY OPERATOR
- CREATE ANY MATERIALIZED VIEW
- ALTER ANY SEQUENCE
- DROP ANY SYNONYM
- CREATE ANY SYNONYM
- DROP USER
- ALTER ANY MEASURE FOLDER
- ALTER ANY EDITION
- DROP ANY RULE
- CREATE ANY RULE
- ALTER ANY RULE SET
- CREATE ANY OUTLINE
- UNDER ANY TABLE
- UNDER ANY VIEW
- DROP ANY DIRECTORY
- ALTER ANY CLUSTER
- CREATE ANY CLUSTER
- ALTER ANY TABLE
- CREATE ANY CUBE BUILD PROCESS
- ALTER ANY CUBE DIMENSION
- CREATE ANY EDITION
- CREATE ANY SQL PROFILE
- ALTER ANY SQL PROFILE
- DROP ANY OUTLINE
- DROP ANY CONTEXT
- DROP ANY OPERATOR
- DROP ANY LIBRARY
- ALTER ANY LIBRARY

- CREATE ANY LIBRARY
- ALTER ANY MATERIALIZED VIEW
- ALTER ANY TRIGGER
- CREATE ANY SEQUENCE
- DROP ANY INDEX
- CREATE ANY INDEX
- DROP ANY TABLE

Privileges for PostgreSQL as a Target Database

The privileges required for PostgreSQL as a target are explained following.

If the new schema doesn't exist yet, grant the privilege CREATE ON DATABASE *<database_name>*

If the new schema already exists, grant the privilege INSERT ON ALL TABLES IN SCHEMA *<schema_name>*

In PostgreSQL, only the owner of a schema or a superuser can drop a schema. The owner can drop the schema, and all contained objects, even if the owner doesn't own all of the contained objects.

Related Topics

- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)

Getting Started Converting Database Schema with the AWS Schema Conversion Tool

Following, you can find information to help you convert your online transaction processing (OLTP) database schema from one database engine to another by using the AWS Schema Conversion Tool (AWS SCT). You can use the converted schema with an Amazon Relational Database Service (Amazon RDS) DB instance.

AWS SCT supports the following OLTP database conversions.

Source Database	Target Database on Amazon RDS
Microsoft SQL Server (version 2008 and later)	Amazon Aurora (MySQL or PostgreSQL), Microsoft SQL Server, MySQL, PostgreSQL
MySQL (version 5.5 and later)	Amazon Aurora (PostgreSQL), MySQL, PostgreSQL You can migrate schema and data from MySQL to an Amazon Aurora (MySQL) DB cluster without using AWS SCT. For more information, see Migrating Data to an Amazon Aurora DB Cluster .
Oracle (version 10.2 and later)	Amazon Aurora (MySQL or PostgreSQL), MySQL, Oracle, PostgreSQL
PostgreSQL (version 9.1 and later)	Amazon Aurora (MySQL), MySQL, PostgreSQL

If you want to convert a data warehouse schema, see [Getting Started Converting Data Warehouse Schema with the AWS Schema Conversion Tool \(p. 32\)](#).

Before You Begin

Before you complete the procedures in this topic, you must first do the following:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

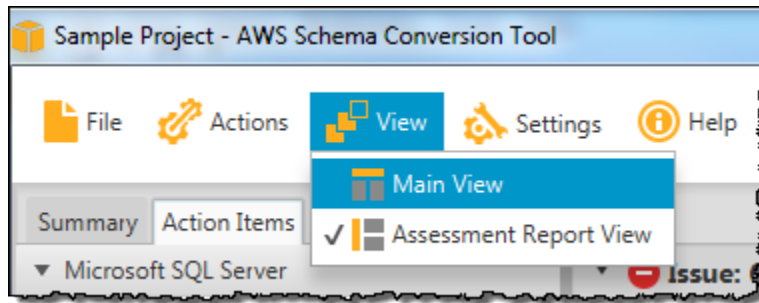
For more information, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

Converting Your Schema

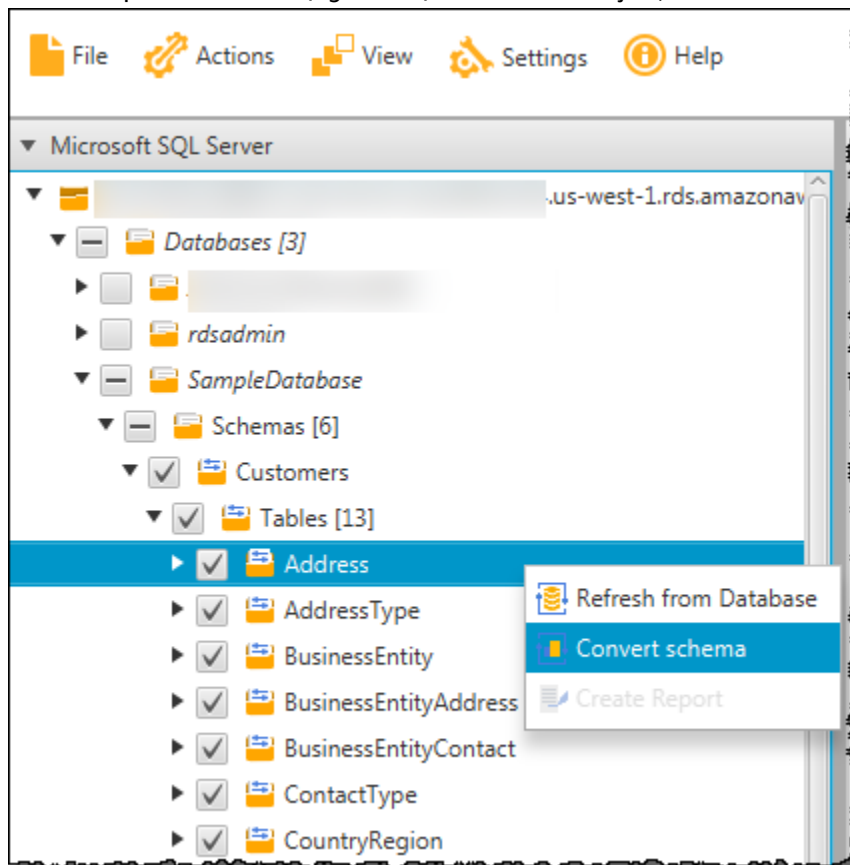
Use the following procedure to convert schema.

To convert schema

1. Choose **View**, and then choose **Main View**.



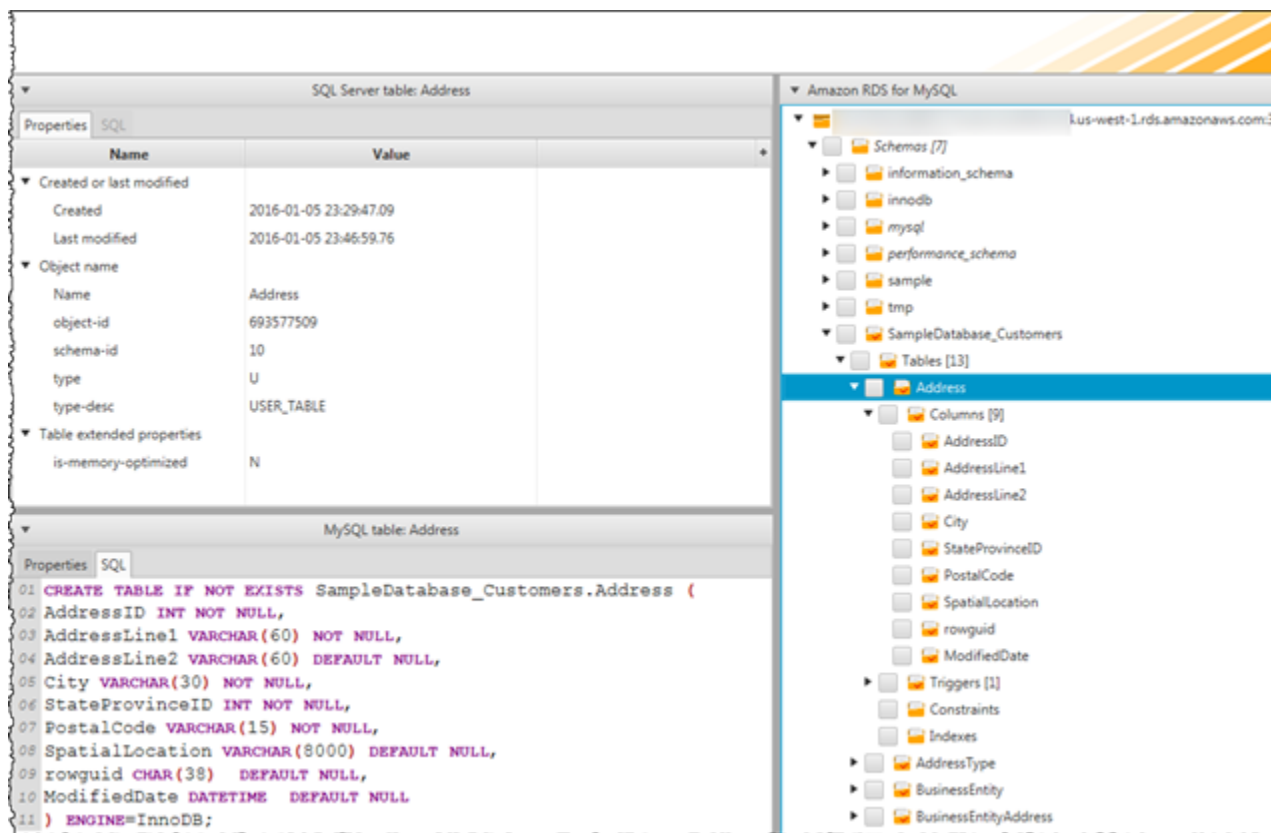
2. In the left panel that displays the schema from your source database, choose a schema object to convert. Open the context (right-click) menu for the object, and then choose **Convert schema**.



3. When AWS SCT finishes converting the schema, you can view the proposed schema in the panel on the right of your project.

At this point, no schema is applied to your target Amazon RDS DB instance. The planned schema is part of your project. If you select a converted schema item, you can see the planned schema command in the panel at lower center for your target Amazon RDS DB instance.

You can edit the schema in this window. The edited schema is stored as part of your project and is written to the target DB instance when you choose to apply your converted schema.



For more information, see [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 72\)](#).

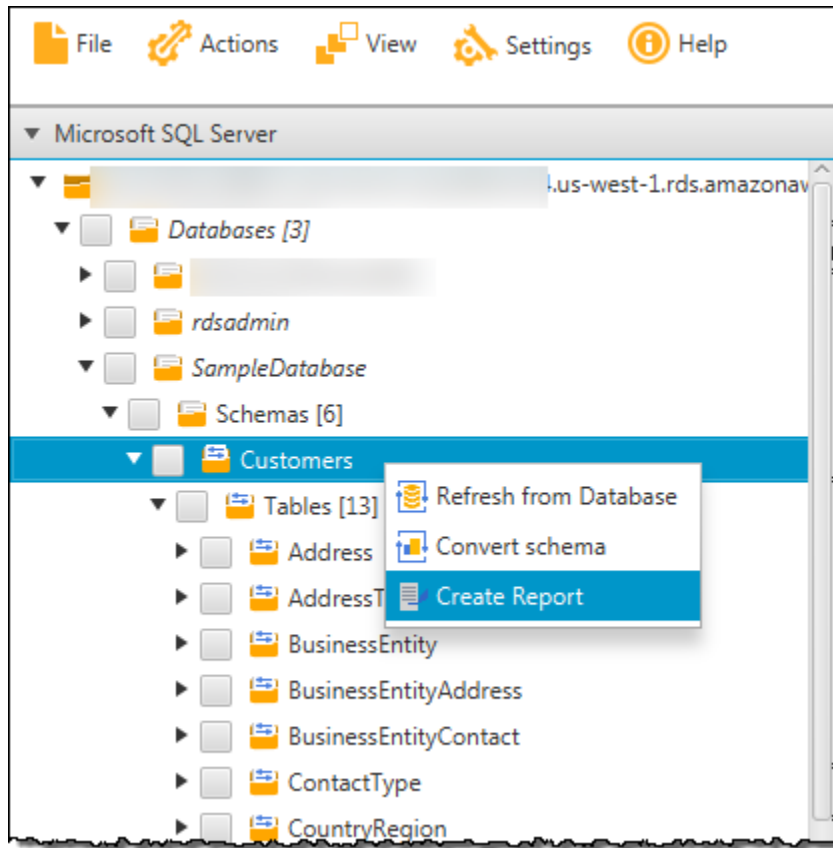
Creating and Reviewing the Database Migration Assessment Report

The *database migration assessment report* summarizes all of the action items for schema that can't be converted automatically to the engine of your target Amazon RDS DB instance. The report also includes estimates of the amount of effort that it will take to write the equivalent code for your target DB instance.

You can create (or update) a database migration assessment report in your project at any time by using the following procedure.

To create and view the database migration assessment report

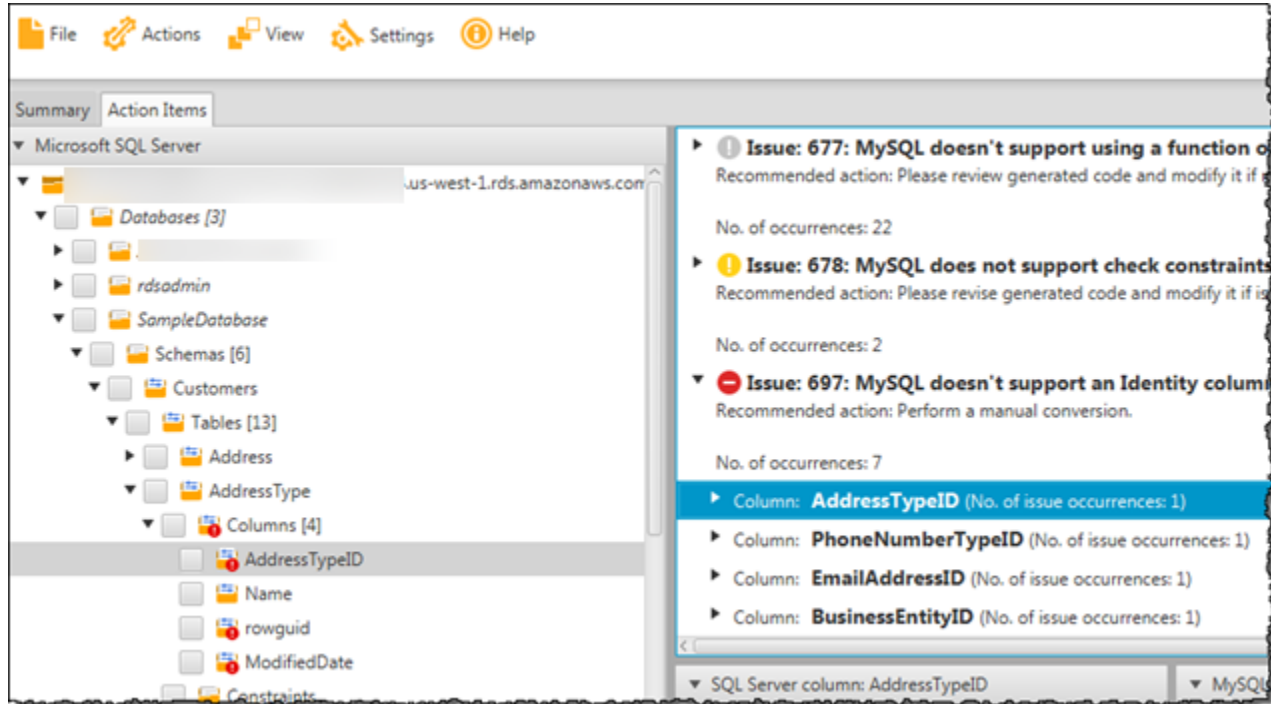
1. In the left panel that displays the schema from your source database, choose a schema object to create an assessment report for. Open the context (right-click) menu for the object, and then choose **Create Report**.



The assessment report view opens.

2. Choose the **Action Items** tab.

The **Action Items** tab displays a list of items that describe the schema that can't be converted automatically. Select one of the action items from the list. AWS SCT highlights the item from your schema that the action item applies to, as shown following.

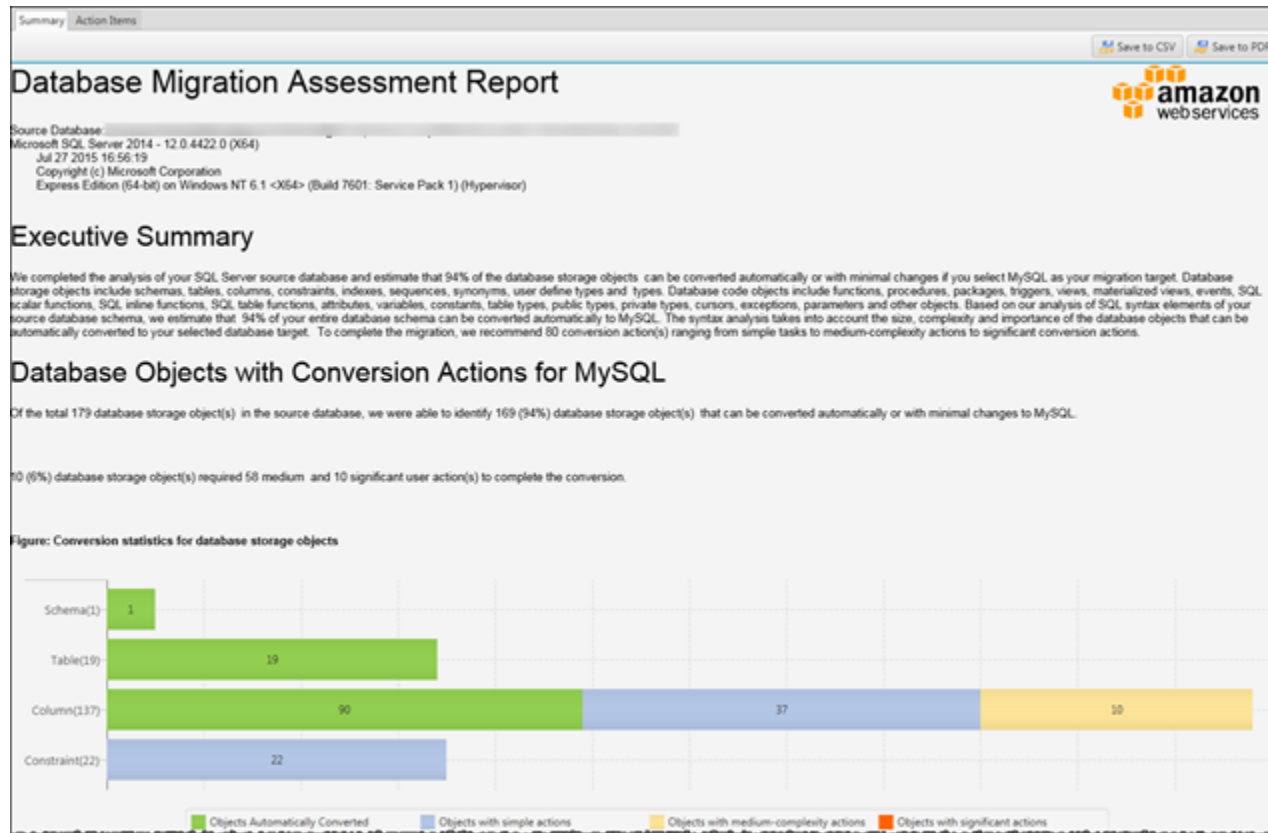


3. Choose the **Summary** tab.

The **Summary** tab displays the summary information from the database migration assessment report. It shows the number of items that were converted automatically, and the number of items that were not converted automatically. The summary also includes an estimate of the time that it will take to create schema in your target DB instance that are equivalent to those in your source database.

The section **License Evaluation and Cloud Support** contains information about moving your existing on-premises database schema to an Amazon RDS DB instance running the same engine. For example, if you want to change license types, this section of the report tells you which features from your current database should be removed.

An example of an assessment report summary is shown following.



4. Choose the **Summary** tab, and then choose **Save to PDF**. The database migration assessment report is saved as a PDF file. The PDF file contains both the summary and action item information.

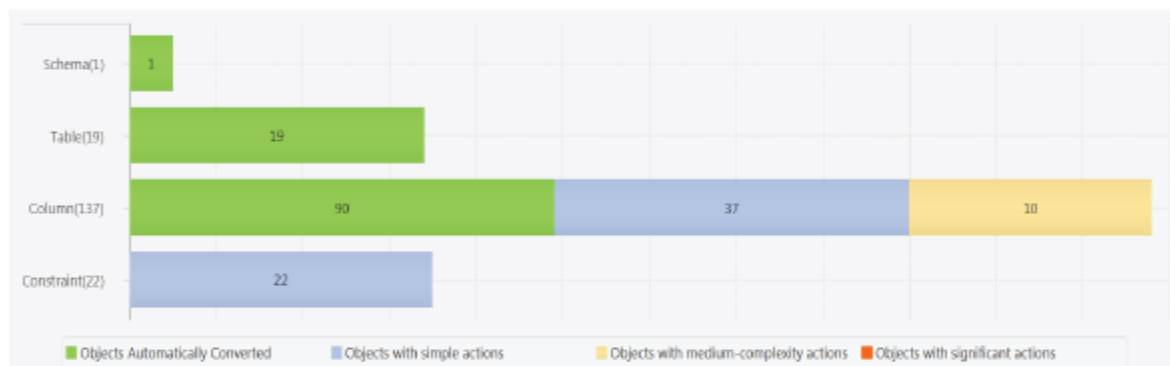
You can also choose **Save to CSV** to save the report as a comma-separated values (CSV) file. The CSV file contains only action item information.

Database Objects with Conversion Actions for MySQL

Of the total 179 database storage object(s) in the source database, we were able to identify 169 (94%) database storage object(s) that can be converted automatically or with minimal changes to MySQL.

10 (6%) database storage object(s) required 58 medium and 10 significant user action(s) to complete the conversion.

Figure: Conversion statistics for database storage objects



Detailed Recommendations for MySQL Migrations

If you choose to migrate your SQL Server database to MySQL, we recommend the following actions.

Storage Object Actions

Constraint Changes

Some changes are required to CONSTRAINTs that cannot be converted automatically. You'll need to address these issues manually.

For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool](#) (p. 76).

Applying the Converted Schema to Your Target DB Instance

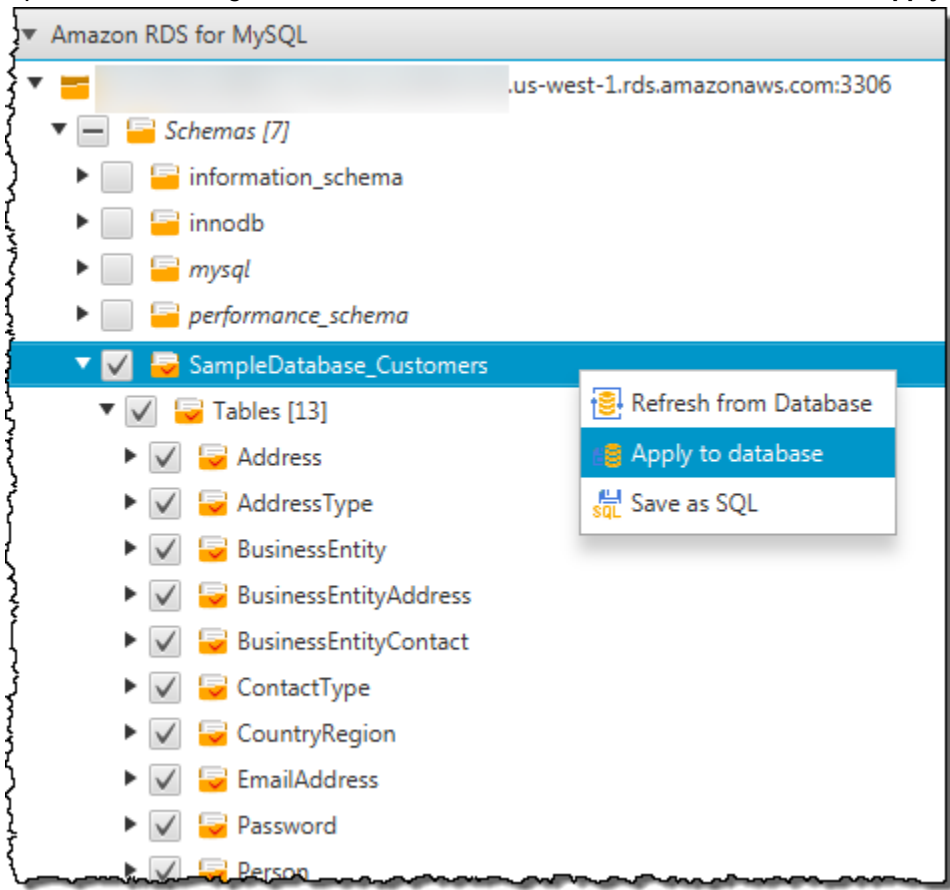
You can apply the converted database schema to your target Amazon RDS DB instance. After the schema has been applied to your target DB instance, you can update the schema based on the action items in the database migration assessment report.

Warning

This procedure overwrites the existing target schema. Be careful not to overwrite schema unintentionally. Be careful not to overwrite schema in your target DB instance that you have already modified, or you will overwrite those changes.

To apply the converted database schema to your target Amazon RDS DB instance

1. Choose the schema element in the right panel of your project that displays the planned schema for your target DB instance.
2. Open the context (right-click) menu for the schema element, and then choose **Apply to database**.



The converted schema is applied to the target DB instance.

For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 82).

Related Topics

- [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool](#) (p. 69)

Getting Started Converting Data Warehouse Schema with the AWS Schema Conversion Tool

Following, you can find information to help you convert your data warehouse schema from one database engine to another by using the AWS Schema Conversion Tool (AWS SCT). You can use the converted schema with an Amazon Redshift cluster.

AWS SCT supports the following data warehouse conversions.

Source Database	Target Database on Amazon Redshift
Greenplum Database (version 4.3 and later)	Amazon Redshift
Microsoft SQL Server (version 2008 and later)	Amazon Redshift
Netezza (version 7.0.3 and later)	Amazon Redshift
Oracle (version 10 and later)	Amazon Redshift
Teradata (version 13 and later)	Amazon Redshift
Vertica (version 7.2.2 and later)	Amazon Redshift

If you want to convert an online transaction processing (OLTP) database schema, see [Getting Started Converting Database Schema with the AWS Schema Conversion Tool \(p. 24\)](#).

Before You Begin

Before you complete the procedures in this topic, you must first do the following:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

For more information, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

Choosing Optimization Strategies and Rules

To optimize how AWS SCT converts your data warehouse schema, you can choose the strategies and rules you want the tool to use. After converting your schema, and reviewing the suggested keys, you can adjust your rules or change your strategy to get the results you want.

To choose your optimization strategies and rules

1. Choose **Settings**, and then choose **Project Settings**. The **Current project settings** dialog box appears.
2. In the left pane, choose **Optimization Strategies**. The optimization strategies appear in the right pane with the defaults selected.
3. For **Strategy Sector**, choose the optimization strategy you want to use. You can choose from the following:

- **Use metadata, ignore statistical information** – In this strategy, only information from the metadata is used for optimization decisions. For example, if there is more than one index on a source table, the source database sort order is used, and the first index becomes a distribution key.
 - **Ignore metadata, use statistical information** – In this strategy, optimization decisions about derived from statistical information only. This strategy applies only to tables and columns for which statistics are provided. For more information, see [Collecting or Uploading Statistics for the AWS Schema Conversion Tool \(p. 91\)](#).
 - **Use metadata and use statistical information** – In this strategy, both metadata and statistics are used for optimization decisions.
4. After you choose your optimization strategy, you can choose the rules you want to use. You can choose from the following:
- **Choose Distribution Key and Sort Keys using metadata**
 - **Choose fact table and appropriate dimension for collation**
 - **Analyze cardinality of indexes' columns**
 - **Find the most used tables and columns from QueryLog table**

For each rule, you can enter a weight for the sort key and a weight for the distribution key. AWS SCT uses the weights you choose when it converts your schema.

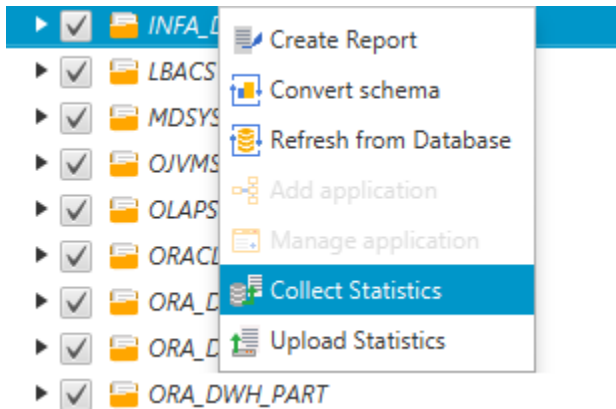
For more information, see [Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool \(p. 90\)](#).

Collecting or Uploading Statistics

To optimize how AWS SCT converts your data warehouse schema, you can provide statistics from your source database that the tool can use. You can either collect statistics directly from the database, or upload an existing statistics file.

To provide and review statistics

1. Connect to your source database. For more information, see [Connecting to Your Source Database \(p. 14\)](#).
2. Choose a schema object from the left panel of your project, and open the context (right-click) menu for the object. Choose **Collect Statistics** or **Upload Statistics** as shown following.



3. Choose a schema object from the left panel of your project, and then choose the **Statistics** tab. You can review the statistics for the object.

Table: T_PROD_SPEC			
Stats Collection Date:	2016-06-14 15:41:23	Stats Reference Co	
Stats Collection Mode:	online	Stats Row Count:	
Column Name	Stats Collection Date	Stats collection mode	Stats usage co
PART_ID	2016-06-14 15:41:23	online	
ADJUSTER_ID	2016-06-14 15:41:23	online	
SPEC_ID	2016-06-14 15:41:23	online	

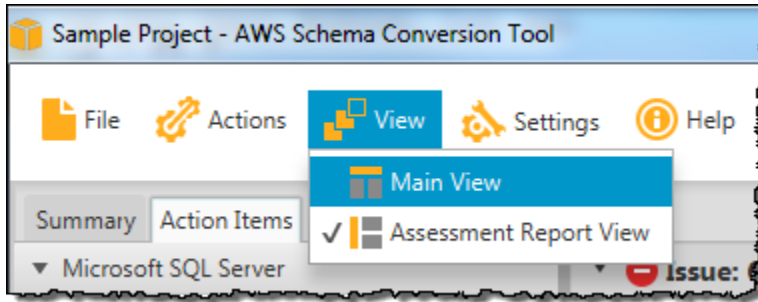
For more information, see [Collecting or Uploading Statistics for the AWS Schema Conversion Tool](#) (p. 91).

Converting Your Schema

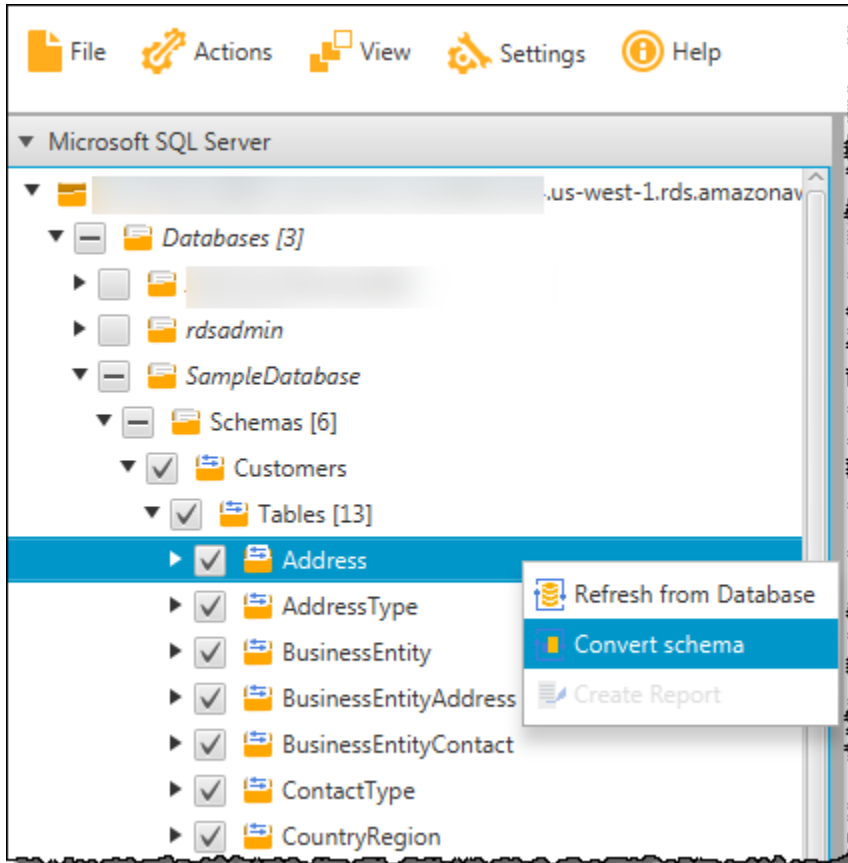
Use the following procedure to converted schema.

To convert schema

1. Choose **View**, and then choose **Main View**.



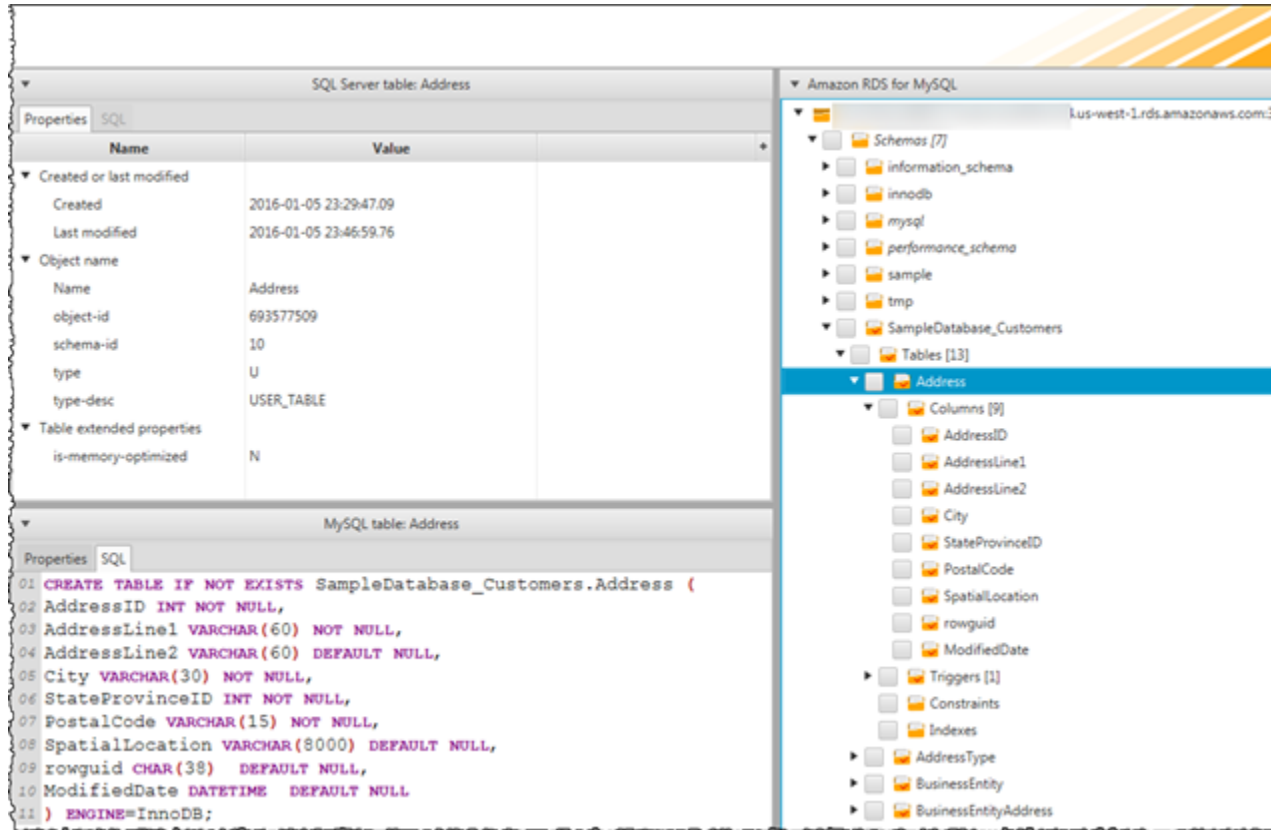
2. In the left panel that displays the schema from your source database, choose a schema object to convert. Open the context (right-click) menu for the object, and then choose **Convert schema**.



3. When AWS SCT finishes converting the schema, you can view the proposed schema in the panel on the right of your project.

At this point, no schema is applied to your target database. The planned schema is part of your project. If you select a converted schema item, you can see the planned schema command in the panel at lower center for your target database.

You can edit the schema in this window. The edited schema is stored as part of your project and is written to the target database when you choose to apply your converted schema.

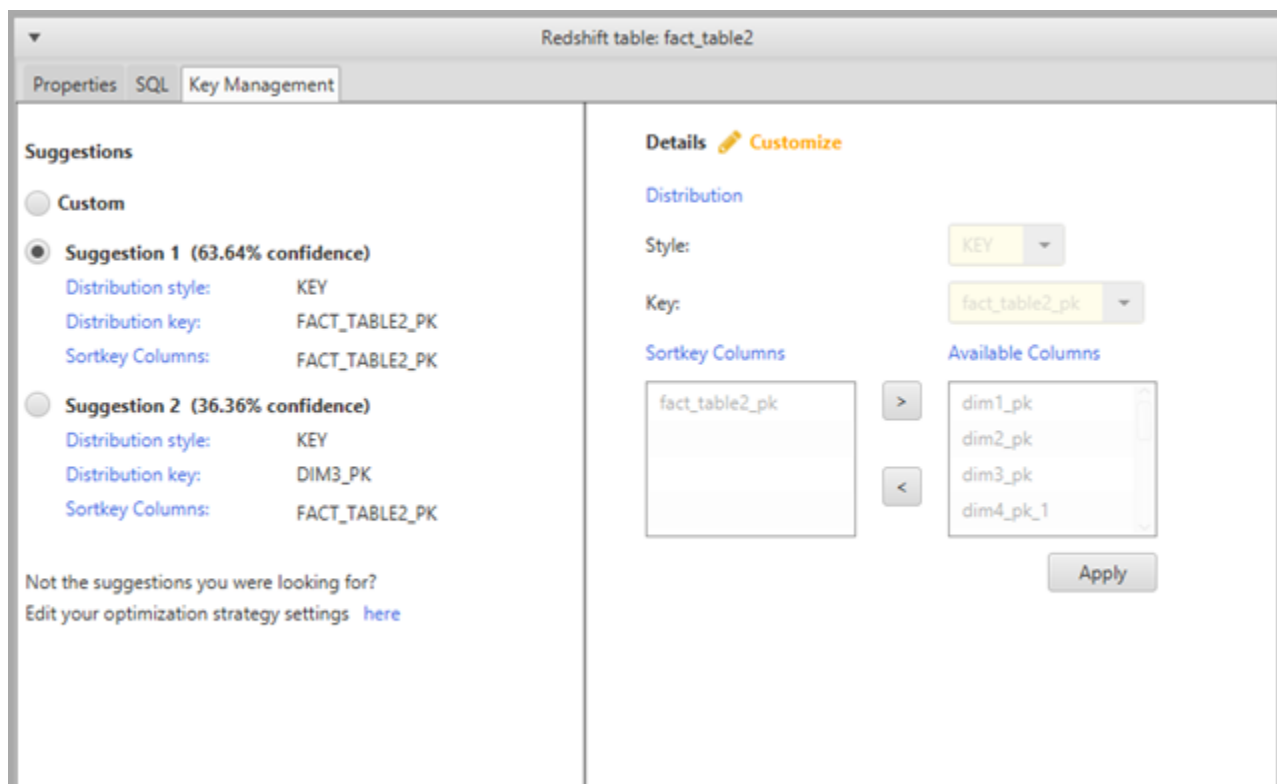


For more information, see [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 94\)](#).

Managing and Customizing Keys

After you convert your schema, you can manage and edit your keys. Key management is the heart of a data warehouse conversion.

To manage keys, select a table in your target database, and then choose the **Key Management** tab as shown following.



The left pane contains key suggestions, and includes the confidence rating for each suggestion. You can choose one of the suggestions, or you can customize the key by editing it in the right pane.

If the choices for the key don't look like what you expected, you can edit your edit your optimization strategies, and then retry the conversion. For more information, see [Choosing Optimization Strategies and Rules](#) (p. 32).

For more information, see [Managing and Customizing Keys in the AWS Schema Conversion Tool](#) (p. 98).

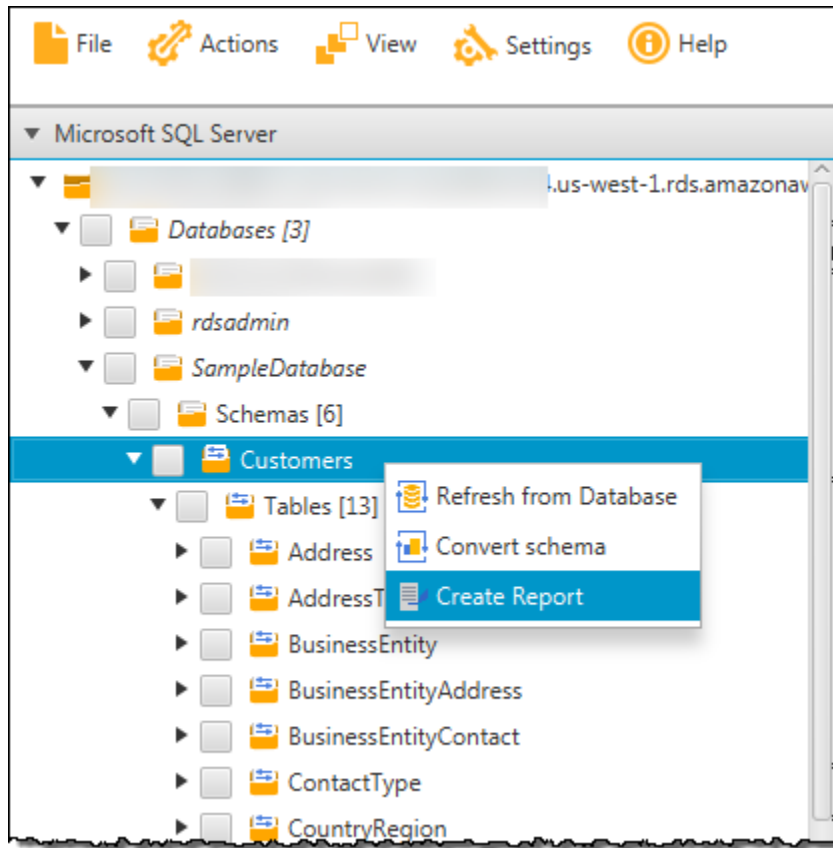
Creating and Reviewing the Database Migration Assessment Report

The *database migration assessment report* summarizes all of the action items for schema that can't be converted automatically to the engine of your target database. The report also includes estimates of the amount of effort that it will take to write the equivalent code for your target database.

You can create (or update) a database migration assessment report in your project at any time by using the following procedure.

To create and view the database migration assessment report

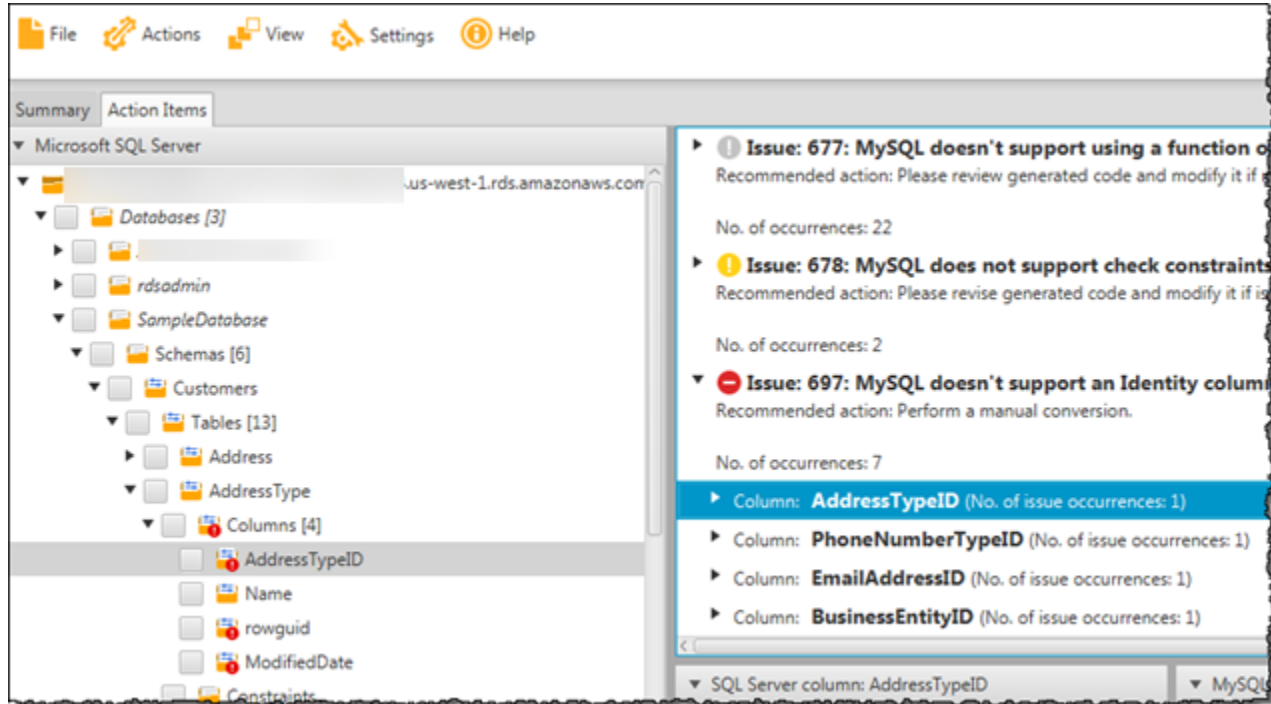
1. In the left panel that displays the schema from your source database, choose a schema object to create an assessment report for. Open the context (right-click) menu for the object, and then choose **Create Report**.



The assessment report view opens.

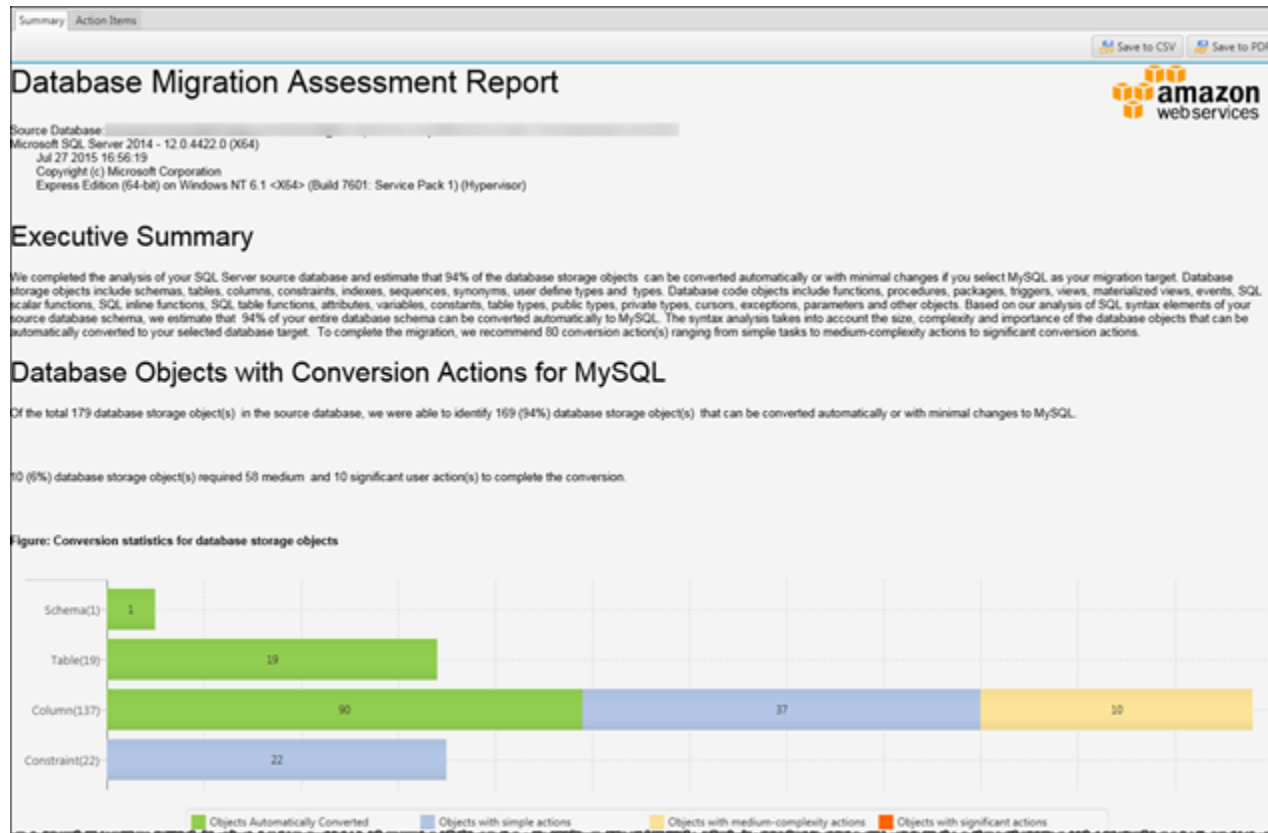
2. Choose the **Action Items** tab.

The **Action Items** tab displays a list of items that describe the schema that can't be converted automatically. Select one of the action items from the list. AWS SCT highlights the item from your schema that the action item applies to, as shown following.



3. Choose the **Summary** tab.

The **Summary** tab displays the summary information from the database migration assessment report. It shows the number of items that were converted automatically, and the number of items that were not converted automatically. The summary also includes an estimate of the time that it will take to create schema in your target database that are equivalent to those in your source database. An example is shown following.



4. Choose the **Summary** tab, and then choose **Save to PDF**. The database migration assessment report is saved as a PDF file. The PDF file contains both the summary and action item information.

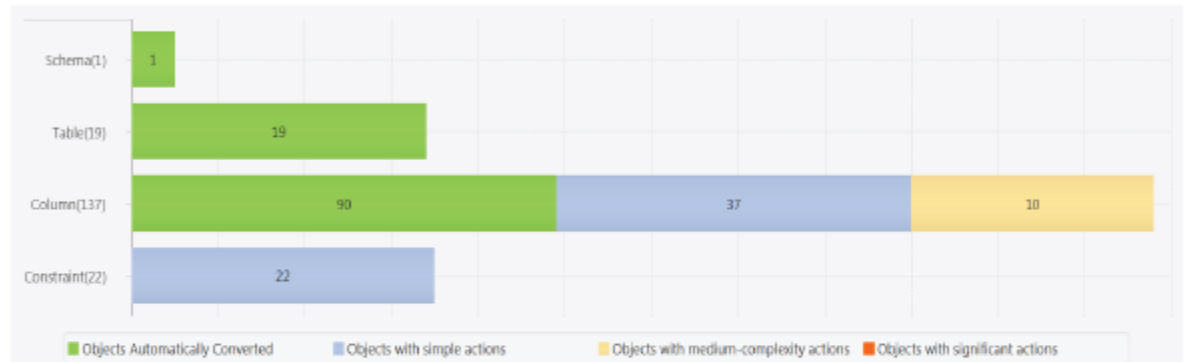
You can also choose **Save to CSV** to save the report as a comma-separated values (CSV) file. The CSV file contains only action item information.

Database Objects with Conversion Actions for MySQL

Of the total 179 database storage object(s) in the source database, we were able to identify 169 (94%) database storage object(s) that can be converted automatically or with minimal changes to MySQL.

10 (6%) database storage object(s) required 58 medium and 10 significant user action(s) to complete the conversion.

Figure: Conversion statistics for database storage objects



Detailed Recommendations for MySQL Migrations

If you choose to migrate your SQL Server database to MySQL, we recommend the following actions.

Storage Object Actions

Constraint Changes

Some changes are required to CONSTRAINTs that cannot be converted automatically. You'll need to address these issues manually.

For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool](#) (p. 99).

Applying the Converted Schema to Your Target Database

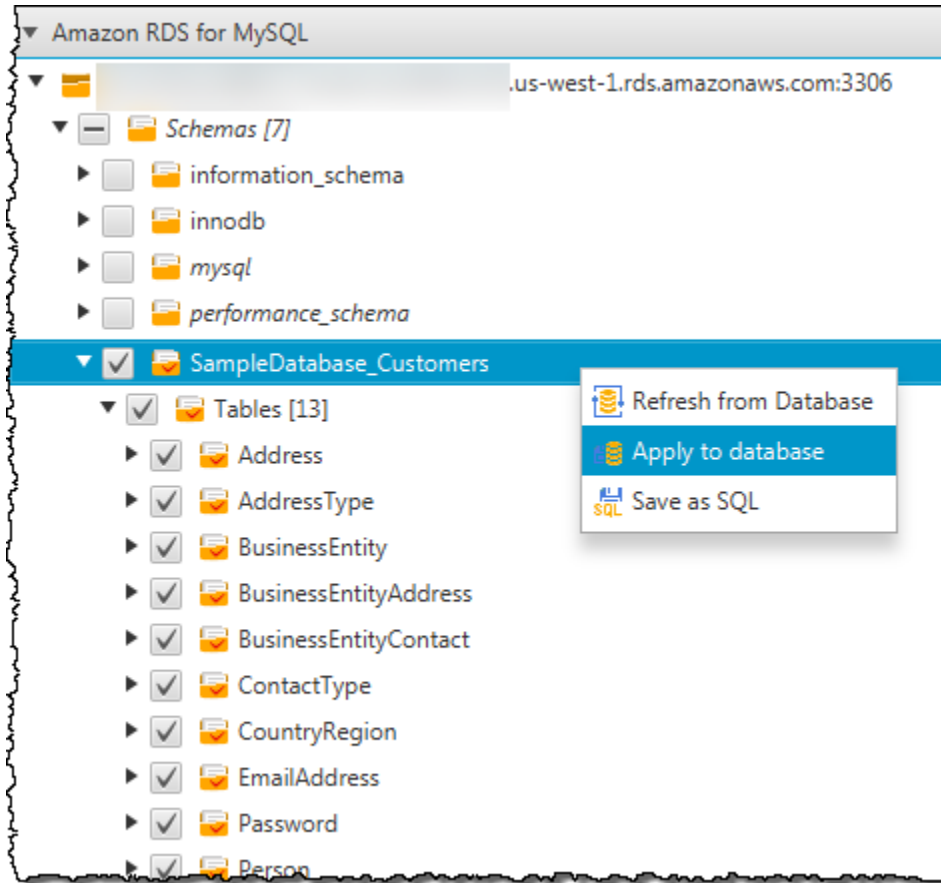
You can apply the converted database schema to your target database. After the schema has been applied to your target database, you can update the schema based on the action items in the database migration assessment report.

Warning

This procedure overwrites the existing target schema. Be careful not to overwrite schema unintentionally. Be careful not to overwrite schema in your target database that you have already modified, or you will overwrite those changes.

To apply the converted database schema to your target database

1. Choose the schema element in the right panel of your project that displays the planned schema for your target database.
2. Open the context (right-click) menu for the schema element, and then choose **Apply to database**.



The converted schema is applied to the target database.

For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 105).

Related Topics

- [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool](#) (p. 88)

Connecting to Your Source Database

Following, you can find information about how to connect to your source database. Choose the topic appropriate for your source database.

Topics

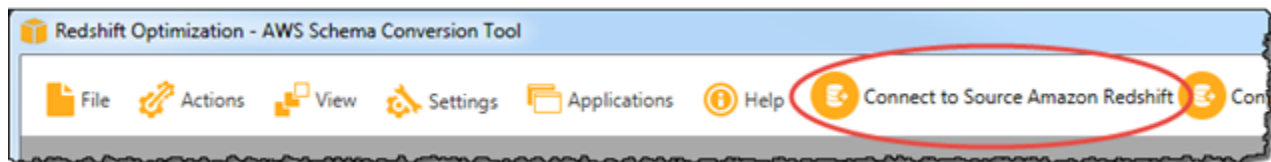
- [Connecting to an Amazon Redshift Source Database \(p. 43\)](#)
- [Connecting to a Greenplum Source Database \(p. 45\)](#)
- [Connecting to a Microsoft SQL Server Source Database \(p. 47\)](#)
- [Connecting to a Microsoft SQL Server Data Warehouse Source Database \(p. 49\)](#)
- [Connecting to a MySQL Source Database \(p. 51\)](#)
- [Connecting to a Netezza Source Database \(p. 53\)](#)
- [Connecting to an Oracle Source Database \(p. 55\)](#)
- [Connecting to an Oracle Data Warehouse Source Database \(p. 58\)](#)
- [Connecting to a PostgreSQL Source Database \(p. 61\)](#)
- [Connecting to a Teradata Source Database \(p. 63\)](#)
- [Connecting to a Vertica Source Database \(p. 65\)](#)

Connecting to an Amazon Redshift Source Database

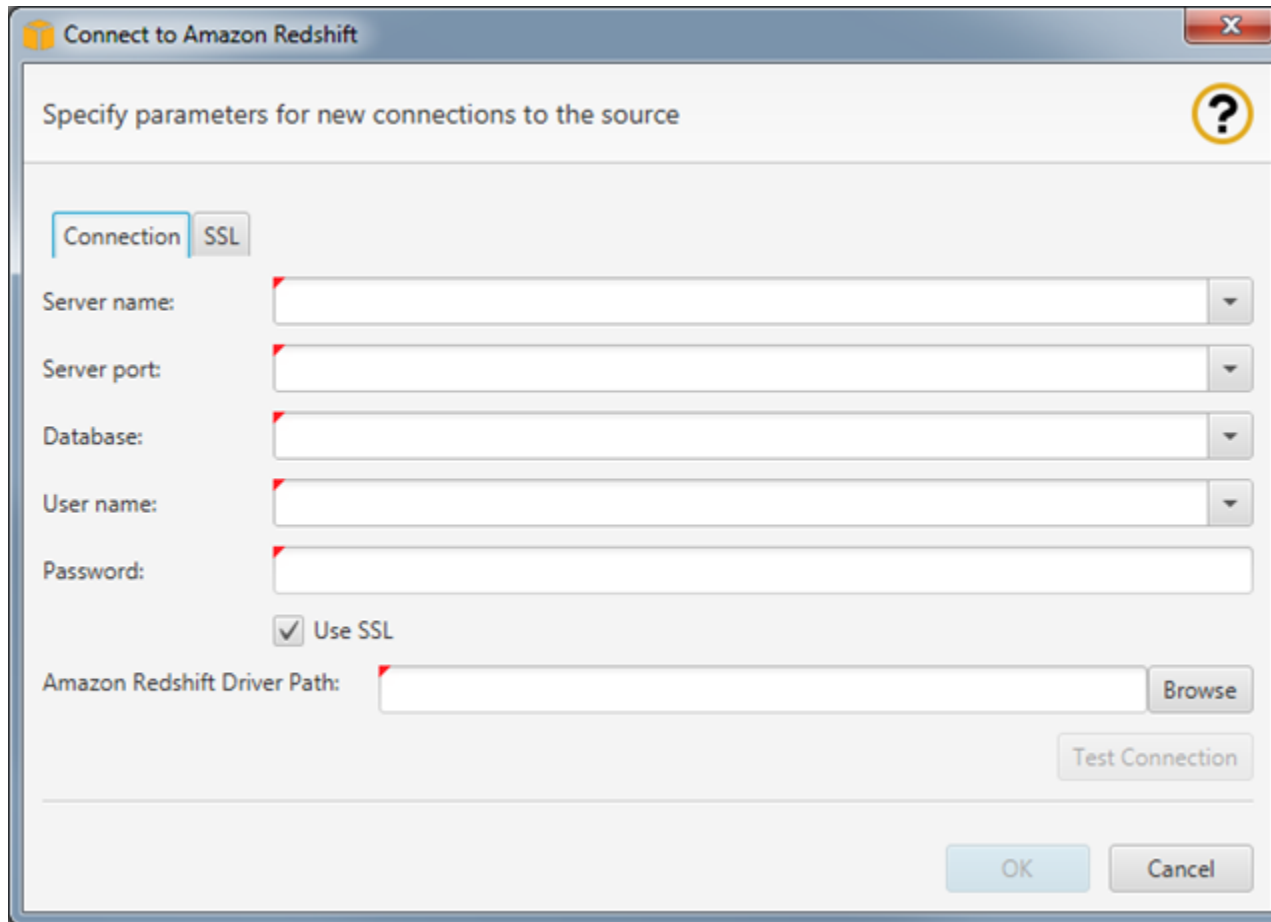
Use the following procedure to connect to your Amazon Redshift source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to an Amazon Redshift source database

1. In the AWS Schema Conversion Tool, choose **Connect to Source Amazon Redshift**.



The **Connect to Amazon Redshift** dialog box appears.



2. Provide the Amazon Redshift source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Database	Type the name of the Amazon Redshift database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to source option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.

For This Parameter	Do This
Use SSL	<p>Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • Verify Server Certificate: Select this option to verify the server certificate by using a trust store. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store. <p>For more information about SSL support for Amazon Redshift, see Configure Security Options for Connections.</p>
Amazon Redshift Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

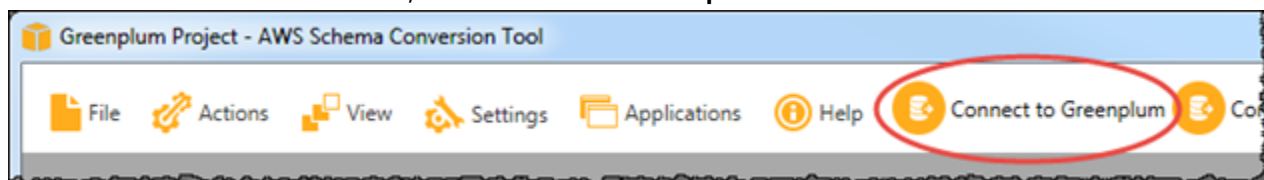
- [Required Database Privileges for Using the AWS Schema Conversion Tool](#) (p. 16)
- [Connecting to Your Target Database](#) (p. 14)

Connecting to a Greenplum Source Database

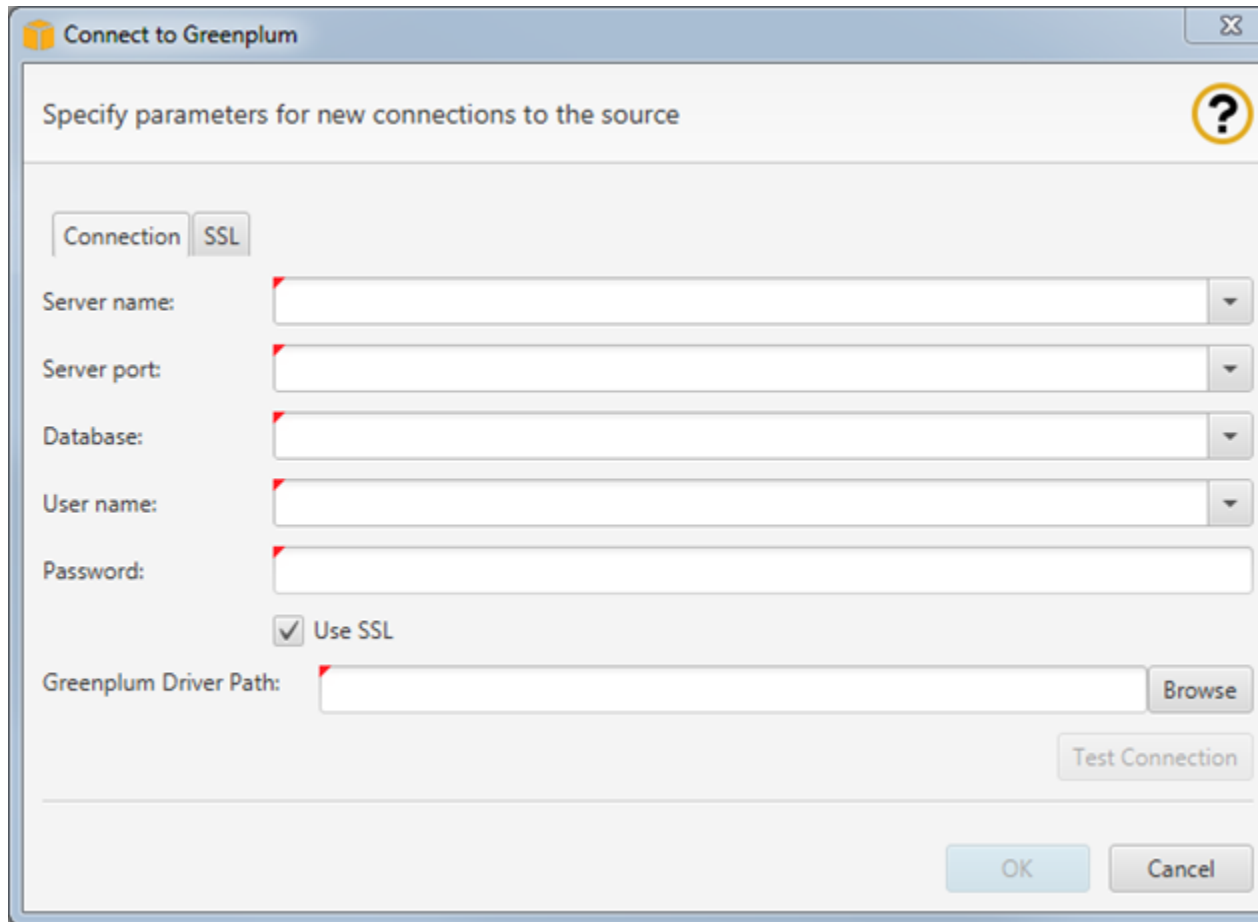
Use the following procedure to connect to your Greenplum source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Greenplum source database

1. In the AWS Schema Conversion Tool, choose **Connect to Greenplum**.



The **Connect to Greenplum** dialog box appears.



2. Provide the Greenplum source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Database	Type the name of the Greenplum database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to source option in a project, where source is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.

For This Parameter	Do This
Use SSL	<p>Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • Verify Server Certificate: Select this option to verify the server certificate by using a trust store. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store.
Greenplum Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

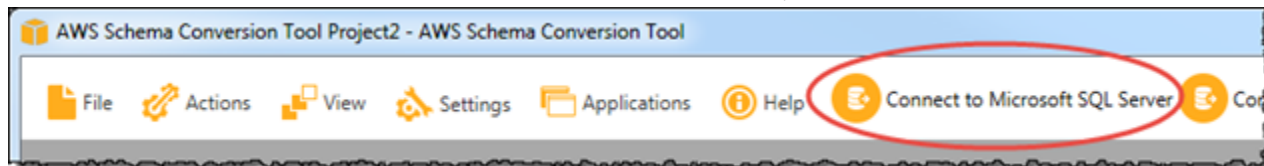
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a Microsoft SQL Server Source Database

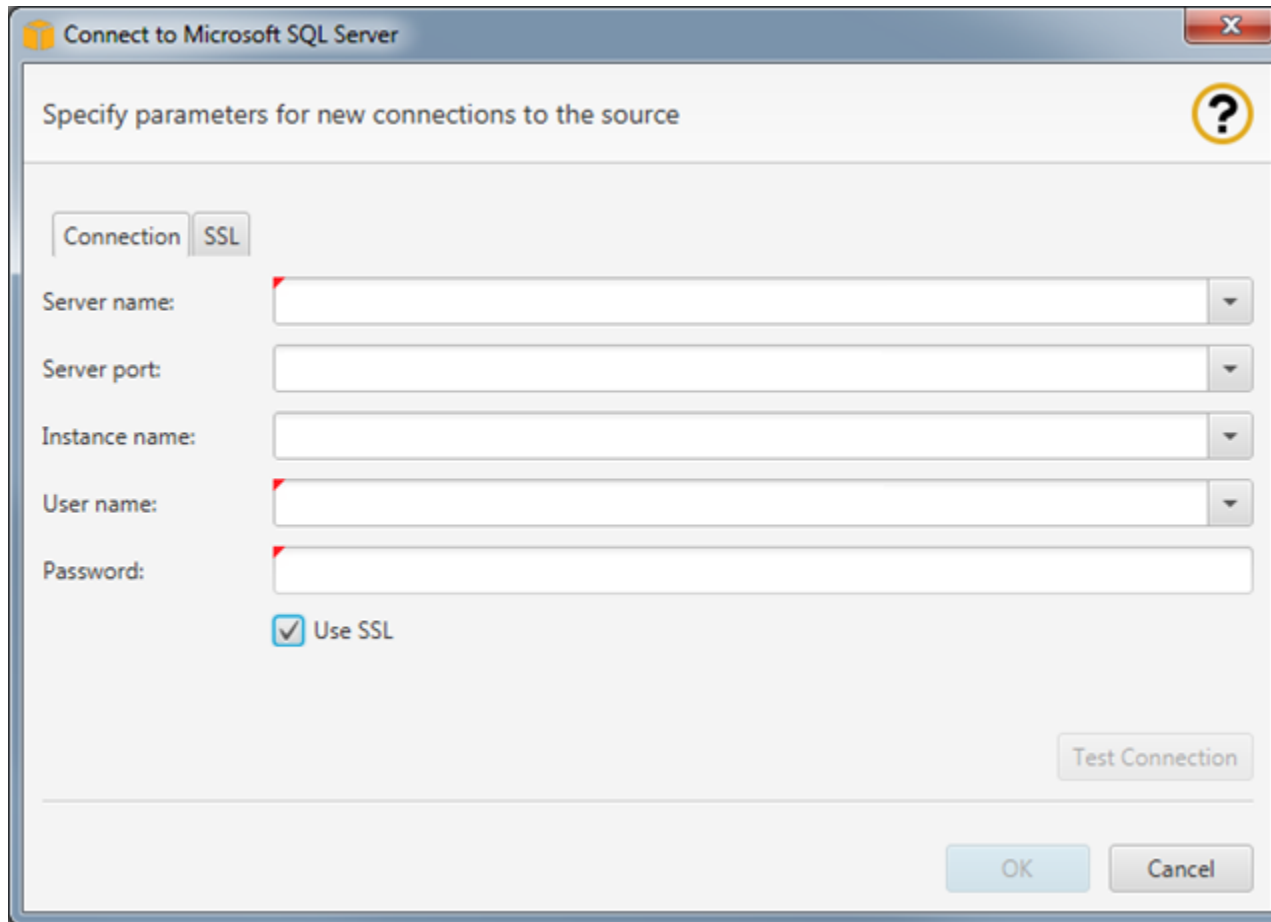
Use the following procedure to connect to your Microsoft SQL Server source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Microsoft SQL Server source database

1. In the AWS Schema Conversion Tool, choose **Connect to Microsoft SQL Server**.



The **Connect to Microsoft SQL Server** dialog box appears.



2. Provide the Microsoft SQL Server source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the Domain Name Service (DNS) name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Instance name	Type the instance name for the SQL Server database. To find the instance name, run the query <code>SELECT @@servername;</code> on your SQL Server database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS

For This Parameter	Do This
	SCT project and reopen it, you are prompted for the password to connect to your source database as needed.
Use SSL	<p>Select this option if you want to use Secure Sockets Layer (SSL) to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • Trust Server Certificate: Select this option to trust the server certificate. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store.
Sql Server Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

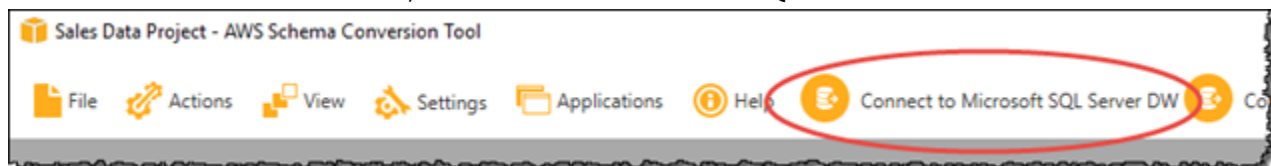
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a Microsoft SQL Server Data Warehouse Source Database

Use the following procedure to connect to your Microsoft SQL Server data warehouse source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Microsoft SQL Server data warehouse source database

1. In the AWS Schema Conversion Tool, choose **Connect to Microsoft SQL Server DW**.



The **Connect to Microsoft SQL Server DW** dialog box appears.

2. Provide the Microsoft SQL Server data warehouse source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the Domain Name Service (DNS) name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Instance name	Type the instance name for the SQL Server database. To find the instance name, run the query <code>SELECT @@servername;</code> on your SQL Server database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.
Use SSL	Select this option if you want to use Secure Sockets Layer (SSL) to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:

For This Parameter	Do This
	<ul style="list-style-type: none">• Trust Server Certificate: Select this option to trust the server certificate.• Trust Store: A trust store that you set up in the Global Settings.
Sql Server Driver Path	Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8) . If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10) .

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

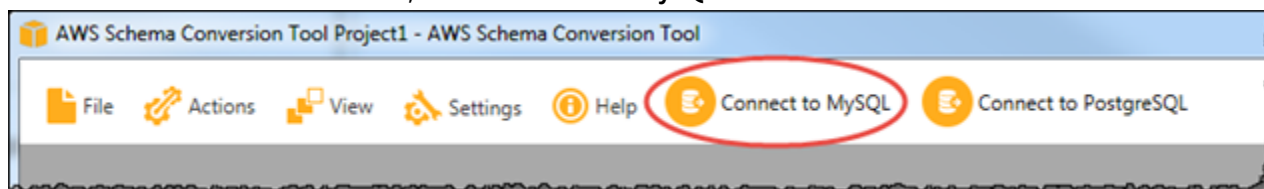
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a MySQL Source Database

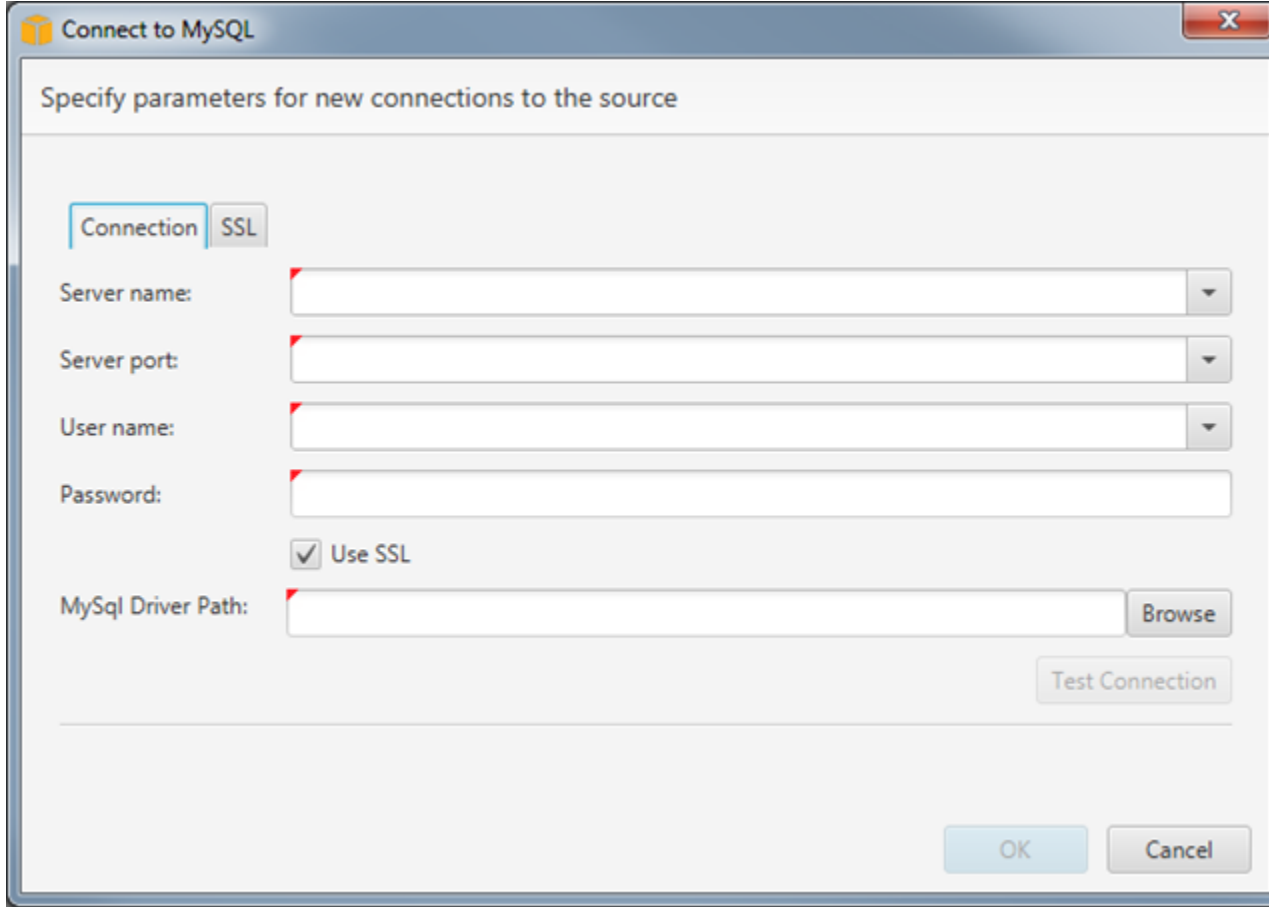
Use the following procedure to connect to your MySQL source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a MySQL source database

1. In the AWS Schema Conversion Tool, choose **Connect to MySQL**.



The **Connect to MySQL** dialog box appears.



2. Provide the MySQL source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.
Use SSL	Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:

For This Parameter	Do This
	<ul style="list-style-type: none"> • Require SSL: Select this option if you want to connect to the server only through SSL. <p>Note If you choose Require SSL, it means that if the server doesn't support SSL, you can't connect to the server. If you don't choose Require SSL and the server doesn't support SSL, you can still connect to the server without using SSL. For more information, see Using Secure Connections.</p> <ul style="list-style-type: none"> • Verify Server Certificate: Select this option to verify the server certificate by using a trust store. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store.
MySql Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

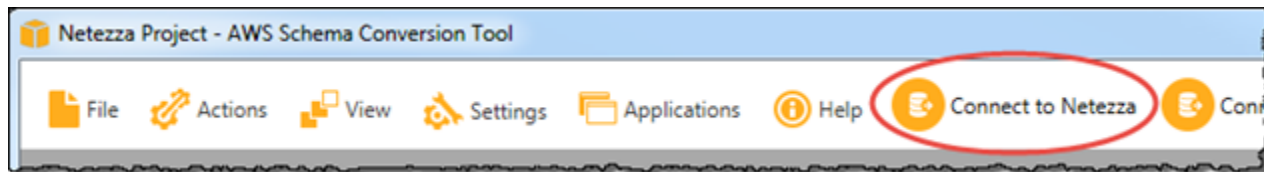
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a Netezza Source Database

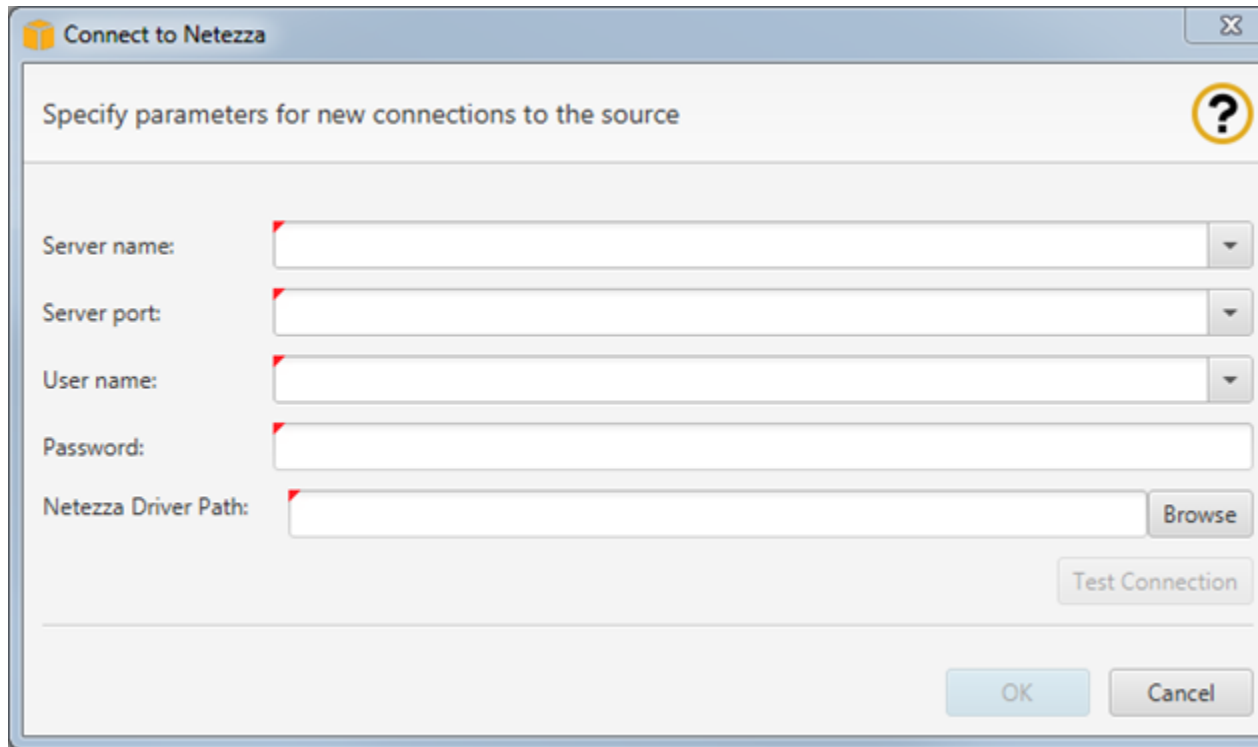
Use the following procedure to connect to your Netezza source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Netezza source database

1. In the AWS Schema Conversion Tool, choose **Connect to Netezza**.



The **Connect to Netezza** dialog box appears.



2. Provide the Netezza source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to source option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.
Netezza Driver Path	Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8) . If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10) .

3. Choose **Test Connection** to verify that you can successfully connect to your source database.

4. Choose **OK** to connect to your source database.

Related Topics

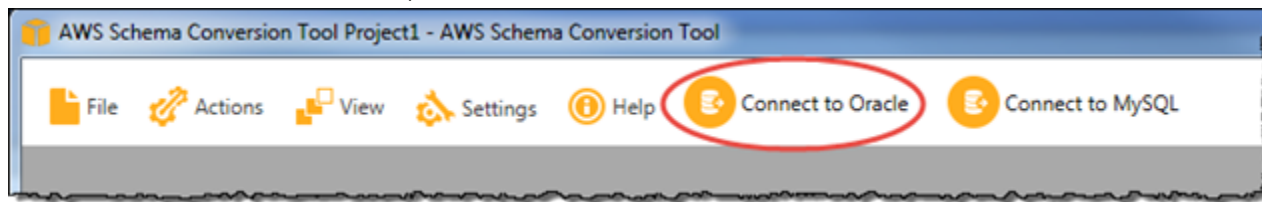
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to an Oracle Source Database

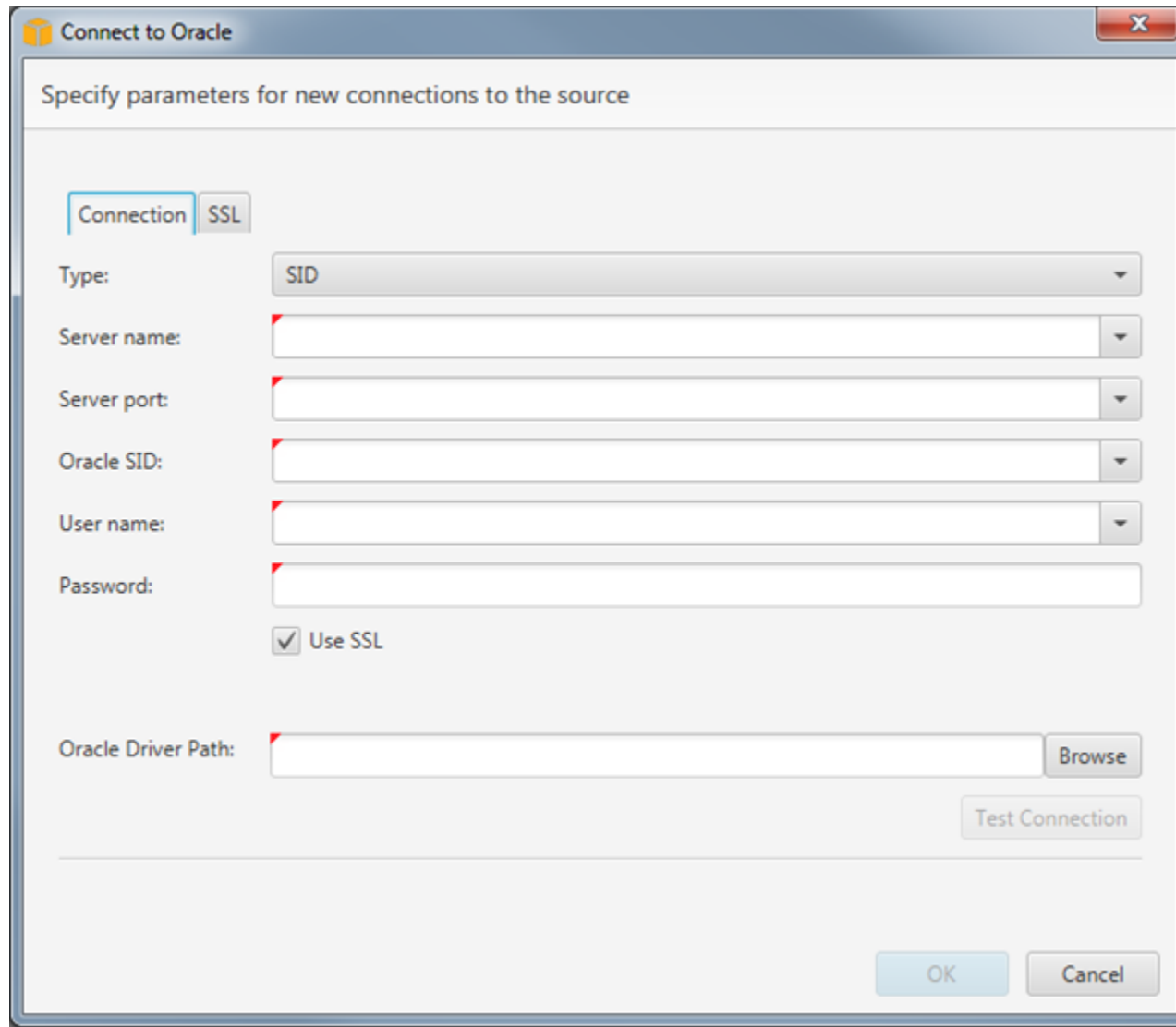
Use the following procedure to connect to your Oracle source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to an Oracle source database

1. In the AWS Schema Conversion Tool, choose **Connect to Oracle**.



The **Connect to Oracle** dialog box appears.



2. Provide the Oracle source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Type	<p>Choose the connection type to your database. Depending on your type, provide the following additional information:</p> <ul style="list-style-type: none"> • SID <ul style="list-style-type: none"> • Server name: The DNS name or IP address of your source database server. • Server port: The port used to connect to your source database server. • Oracle SID: The Oracle System ID (SID). To find the Oracle SID, submit the following query to your Oracle database: <pre>SELECT sys_context('userenv', 'instance_name') AS SID FROM dual;</pre>

For This Parameter	Do This
	<ul style="list-style-type: none"> • Service Name <ul style="list-style-type: none"> • Server name: The DNS name or IP address of your source database server. • Server port: The port used to connect to your source database server. • Service Name: The name of the Oracle service to connect to. • TNS Alias <ul style="list-style-type: none"> • TNS file path: The path to the file that contains the Transparent Network Substrate (TNS) name connection information. • TNS file path: The TNS alias from this file to use to connect to the source database. • TNS Connect Identifier <ul style="list-style-type: none"> • TNS identifier: The identifier for the registered TNS connection information.
User name and Password	<p>Type the user name and password to connect to your source database server.</p> <p>Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to source option in a project, where source is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.</p>
Use SSL	<p>Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • SSL Authentication: Select this option to use SSL authentication for the connection. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store. • Key Store: The location of a key store containing a private key and certificates. This value is required if SSL Authentication is selected and is otherwise optional. • Trust Store Password: The password for the key store. This value is required if SSL Authentication is selected and is otherwise optional.

For This Parameter	Do This
Oracle Driver Path	Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8) . If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10) .

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

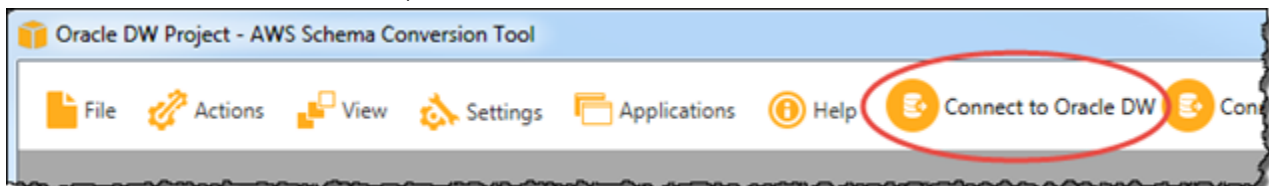
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to an Oracle Data Warehouse Source Database

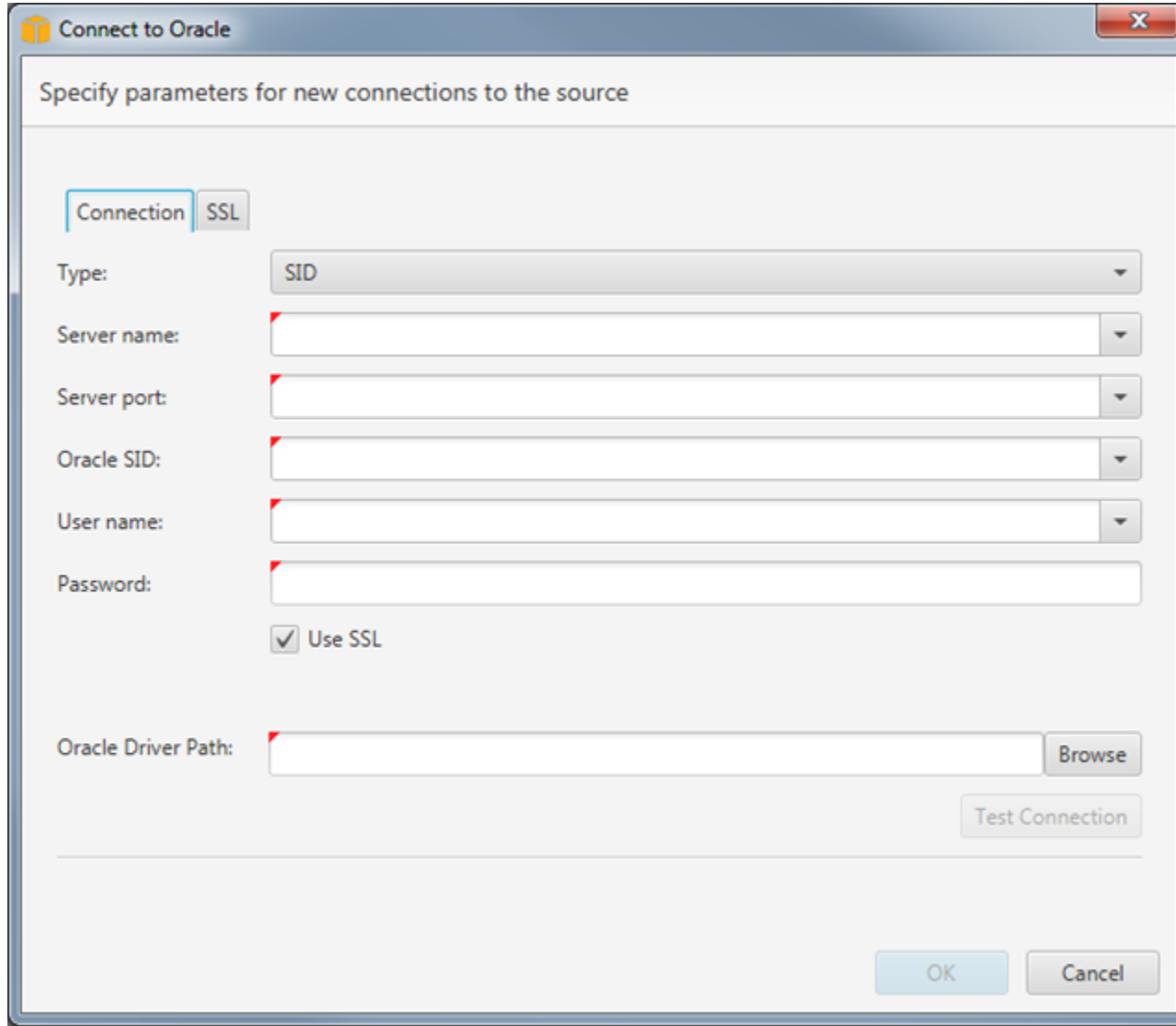
Use the following procedure to connect to your Oracle data warehouse source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to an Oracle data warehouse source database

1. In the AWS Schema Conversion Tool, choose **Connect to Oracle DW**.



The **Connect to Oracle** dialog box appears.



2. Provide the Oracle Data Warehouse source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Type	<p>Choose the connection type to your database. Depending on your type, provide the following additional information:</p> <ul style="list-style-type: none"> • SID <ul style="list-style-type: none"> • Server name: The DNS name or IP address of your source database server. • Server port: The port used to connect to your source database server. • Oracle SID: The Oracle System ID (SID). To find the Oracle SID, submit the following query to your Oracle database: <pre>SELECT sys_context('userenv', 'instance_name') AS SID FROM dual;</pre>

For This Parameter	Do This
	<ul style="list-style-type: none"> • Service Name <ul style="list-style-type: none"> • Server name: The DNS name or IP address of your source database server. • Server port: The port used to connect to your source database server. • Service Name: The name of the Oracle service to connect to. • TNS Alias <ul style="list-style-type: none"> • TNS file path: The path to the file that contains the Transparent Network Substrate (TNS) name connection information. • TNS file path: The TNS alias from this file to use to connect to the source database. • TNS Connect Identifier <ul style="list-style-type: none"> • TNS identifier: The identifier for the registered TNS connection information.
User name and Password	<p>Type the user name and password to connect to your source database server.</p> <p>Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to source option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.</p>
Use SSL	<p>Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • SSL Authentication: Select this option to use SSL authentication for the connection. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store. • Key Store: The location of a key store containing a private key and certificates. This value is required if SSL Authentication is selected and is otherwise optional. • Trust Store Password: The password for the key store. This value is required if SSL Authentication is selected and is otherwise optional.

For This Parameter	Do This
Oracle Driver Path	Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8) . If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10) .

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

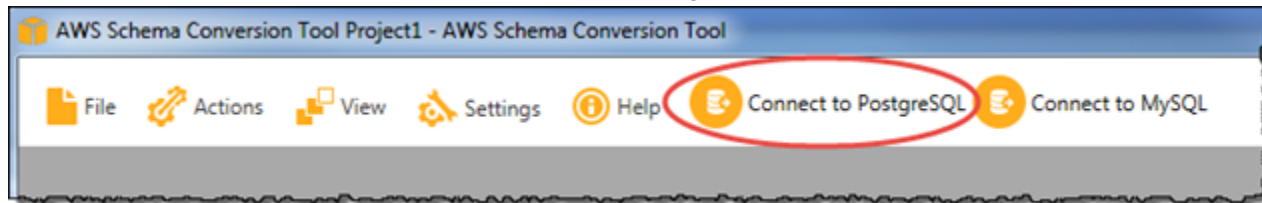
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a PostgreSQL Source Database

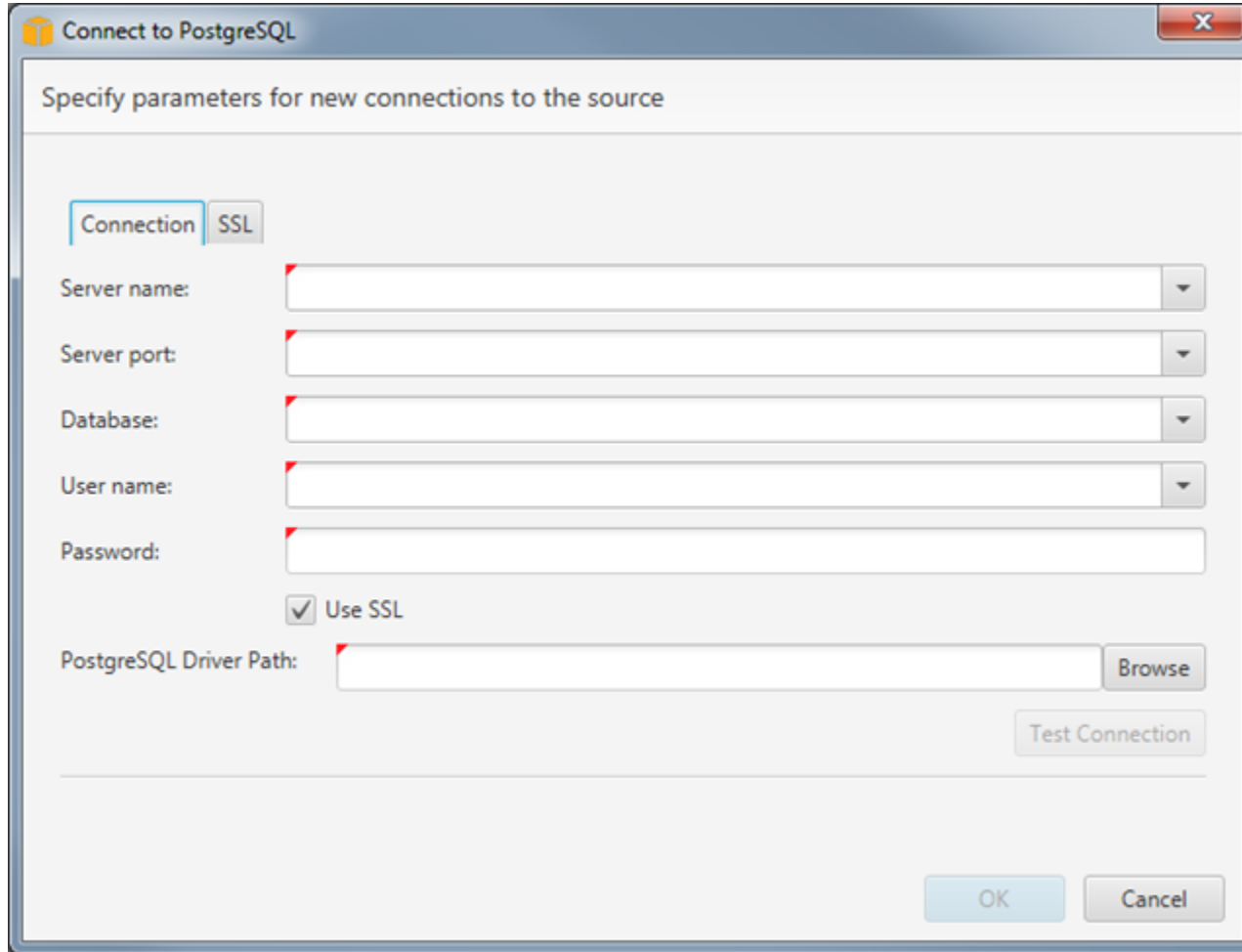
Use the following procedure to connect to your PostgreSQL source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a PostgreSQL source database

1. In the AWS Schema Conversion Tool, choose **Connect to PostgreSQL**.



The **Connect to PostgreSQL** dialog box appears.



2. Provide the PostgreSQL source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Database	Type the name of the PostgreSQL database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.

For This Parameter	Do This
Use SSL	<p>Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:</p> <ul style="list-style-type: none"> • Verify Server Certificate: Select this option to verify the server certificate by using a trust store. • Trust Store: The location of a trust store containing certificates. • Trust Store Password: The password for the trust store.
PostgreSQL Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

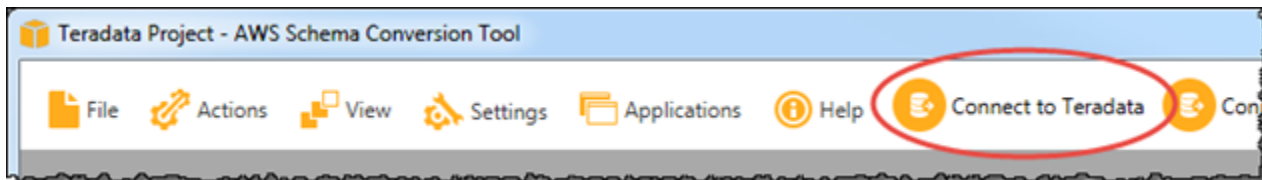
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a Teradata Source Database

Use the following procedure to connect to your Teradata source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Teradata source database

1. In the AWS Schema Conversion Tool, choose **Connect to Teradata**.



The **Connect to Teradata** dialog box appears.

2. Provide the Teradata source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Database	Type the name of the Teradata database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.

For This Parameter	Do This
Encrypt Data	Select this option if you want to encrypt data that you exchange with the database.
Teradata Driver Path	Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8) . If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10) .

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

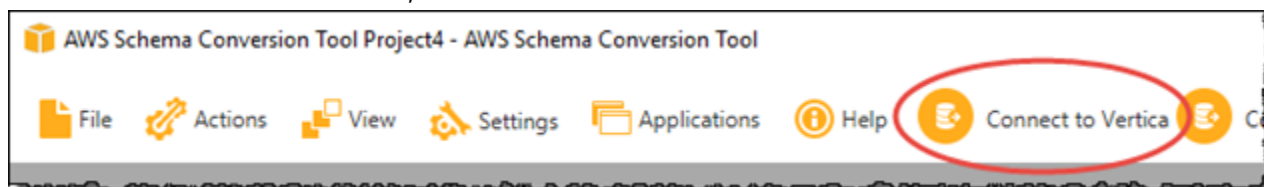
- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Connecting to a Vertica Source Database

Use the following procedure to connect to your Vertica source database with the AWS Schema Conversion Tool (AWS SCT).

To connect to a Vertica source database

1. In the AWS Schema Conversion Tool, choose **Connect to Vertica**.



The **Connect to Vertica** dialog box appears.

2. Provide the Vertica source database connection information. Use the instructions in the following table.

For This Parameter	Do This
Server name	Type the DNS name or IP address of your source database server.
Server port	Type the port used to connect to your source database server.
Database	Type the name of the Vertica database.
User name and Password	Type the user name and password to connect to your source database server. Note AWS SCT uses the password to connect to your source database only when you create your project or choose the Connect to <i>source</i> option in a project, where <i>source</i> is your source database. To guard against exposing the password for your source database, AWS SCT doesn't store the password. If you close your AWS SCT project and reopen it, you are prompted for the password to connect to your source database as needed.
Use SSL	Select this option if you want to use SSL to connect to your database. Provide the following additional information, as appropriate, on the SSL tab:

For This Parameter	Do This
	<ul style="list-style-type: none">• Trust Store: A trust store that you set up in the Global Settings.• Key Store: A key store that you set up in the Global Settings.
Vertica Driver Path	<p>Type the path to the driver to use to connect to the source database. For more information, see Installing the Required Database Drivers (p. 8).</p> <p>If you store the driver path in the global project settings, the driver path doesn't appear on the connection dialog box. For more information, see Storing Driver Paths in the Global Settings (p. 10).</p>

3. Choose **Test Connection** to verify that you can successfully connect to your source database.
4. Choose **OK** to connect to your source database.

Related Topics

- [Required Database Privileges for Using the AWS Schema Conversion Tool \(p. 16\)](#)
- [Connecting to Your Target Database \(p. 14\)](#)

Working with Databases

You can use the AWS Schema Conversion Tool (AWS SCT) to convert your existing database schema from one database engine to another. Your converted schema is suitable for Amazon Relational Database Service (Amazon RDS).

You can also use AWS SCT to copy your existing on-premises database schema to an Amazon RDS DB instance running the same engine. You can use this feature to analyze potential cost savings of moving to the cloud and of changing your license type.

In some cases, database features can't be converted to equivalent Amazon RDS features. If you host and self-manage a database on the Amazon Elastic Compute Cloud (Amazon EC2) platform, you can emulate these features by substituting AWS services for them.

Following, you can find more information about these processes.

Topics

- [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool \(p. 69\)](#)
- [The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases \(p. 85\)](#)

Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool

The AWS Schema Conversion Tool (AWS SCT) automates much of the process of converting your online transaction processing (OLTP) database schema to an Amazon Relational Database Service (Amazon RDS) MySQL DB instance, an Amazon Aurora DB cluster, or a PostgreSQL DB instance. The source and target database engines contain many different features and capabilities, and AWS SCT attempts to create an equivalent schema in your Amazon RDS DB instance wherever possible. If no direct conversion is possible, AWS SCT provides a list of possible actions for you to take.

You can also use AWS SCT to copy your existing on-premises database schema to an Amazon RDS DB instance running the same engine. You can use this feature to analyze potential cost savings of moving to the cloud.

AWS SCT supports the following OLTP conversions.

Source Database	Target Database on Amazon RDS
Microsoft SQL Server (version 2008 and later)	Amazon Aurora (MySQL or PostgreSQL), Microsoft SQL Server, MySQL, PostgreSQL
MySQL (version 5.5 and later)	Amazon Aurora (PostgreSQL), MySQL, PostgreSQL You can migrate schema and data from MySQL to an Amazon Aurora (MySQL) DB cluster without using AWS SCT. For more information, see Migrating Data to an Amazon Aurora DB Cluster .
Oracle (version 10.2 and later)	Amazon Aurora (MySQL or PostgreSQL), MySQL, Oracle, PostgreSQL
PostgreSQL (version 9.1 and later)	Amazon Aurora (MySQL), MySQL, PostgreSQL

If you want to convert a data warehouse schema, see [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#).

Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

To convert your database schema to Amazon RDS, you take the following high-level steps:

- **Create mapping rules** – Before you convert your schema with AWS SCT, you can set up rules that change the data type of columns, move objects from one schema to another, and change the names of objects.

For more information, see [Creating Mapping Rules in the AWS Schema Conversion Tool \(p. 70\)](#).

- **Convert your schema** – AWS SCT creates a local version of the converted schema for you to review, but it doesn't apply it to your target DB instance until you are ready.

For more information, see [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 72\)](#).

- **Create a database migration assessment report** – AWS SCT creates a database migration assessment report that details the schema elements that can't be converted automatically. You can use this report to identify where you need to create a schema in your Amazon RDS DB instance that is compatible with your source database.

For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool \(p. 76\)](#).

- **Decide how to handle manual conversions** – If you have schema elements that can't be converted automatically, you have 2 choices: update the source schema and then convert again, or create equivalent schema elements in your target Amazon RDS DB instance.

For more information, see [Handling Manual Conversions in the AWS Schema Conversion Tool \(p. 80\)](#).

- **Update and refresh the schema in your AWS SCT project** – You can update your AWS SCT project with the most recent schema from your source database.

For more information, see [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool \(p. 81\)](#).

- **Apply the converted schema to your target database** – When you are ready, have AWS SCT apply the converted schema in your local project to your target Amazon RDS DB instance.

For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool \(p. 82\)](#).

Creating Mapping Rules in the AWS Schema Conversion Tool

Before you convert your schema with the AWS Schema Conversion Tool (AWS SCT), you can set up rules that change the data type of columns, move objects from one schema to another, and change the names of objects. For example, if you have a set of tables in your source schema named `test_TABLE_NAME`, you can set up a rule that changes the prefix `test_` to the prefix `demo_` in the target schema.

Note

You can only create mapping rules if your source database engine and target database engine are different.

You can create mapping rules that perform the following tasks:

- Change data type
- Move objects
- Rename objects

- Prefix - add prefix, remove prefix, replace prefix
- Suffix - add suffix, remove suffix, replace suffix

You can create mapping rules for the following objects:

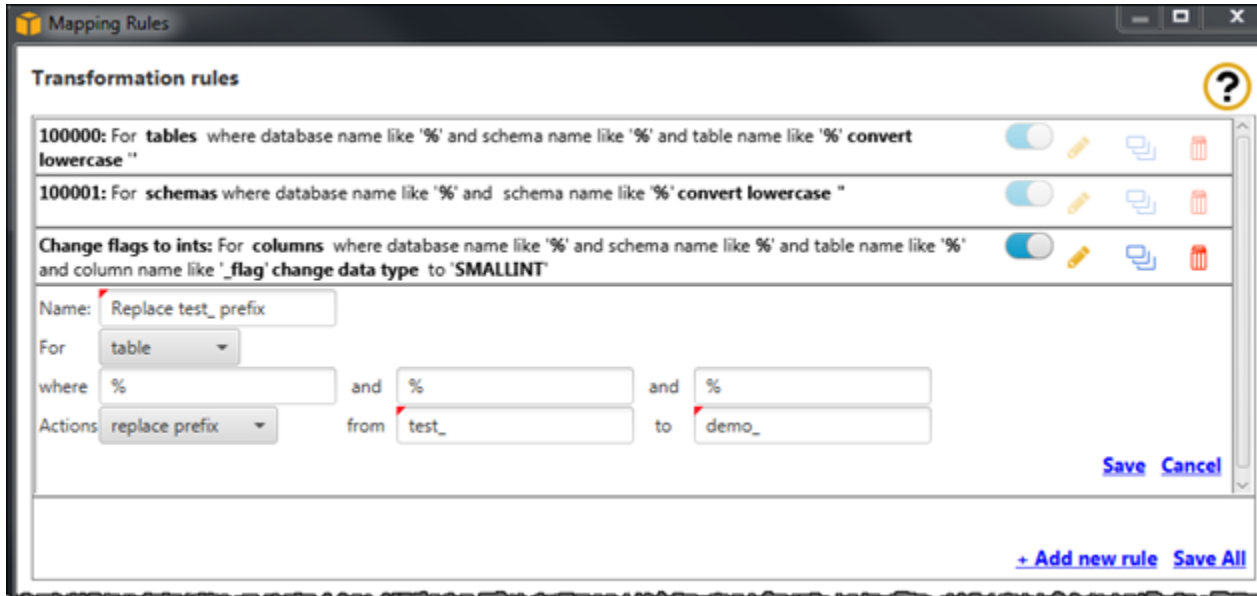
- Database
- Schema
- Table
- Column

Creating Mapping Rules

You can create mapping rules and save the rules as part of your project. With your project open, use the following procedure to create mapping rules.

To create mapping rules

1. Choose **Mapping Rules** from the **Settings** menu. The **Mapping Rules** dialog box appears.



2. Choose **Add new rule**. A new row is added to the list of rules.
3. Choose the edit icon to configure your rule.
 - a. For **Name**, type a name for your rule.
 - b. For **For**, choose the type of object that the rule applies to.
 - c. For **where**, type a filter to apply to objects before applying the mapping rule. The where clause is evaluated by using a like clause. You can enter an exact name to select one object, or you can enter a pattern to select multiple objects.

The fields available for the **where** clause are different depending on the type of the object. For example, if the object type is schema there is only one field available, for the schema name.
 - d. For **Actions**, choose the type of mapping rule you want to create.
 - e. Depending on the rule type, type one or two additional values. For example, to rename an object, type the new name of the object. To replace a prefix, type the old prefix and the new prefix.

4. After you have configured your mapping rule, choose **Save** to save your rule. You can also choose **Cancel** to cancel your changes.
5. After you are done adding, editing, and deleting rules, choose **Save All** to save all your changes.
6. Choose **Close** to close the **Mapping Rules** dialog box.

You can use the toggle icon to turn off a mapping rule without deleting it. You can use the copy icon to duplicate an existing mapping rule. You can use the delete icon to delete an existing mapping rule. To save any changes you make to your mapping rules, choose **Save All**.

Viewing Mapping Rules for Objects

After you set up your mapping rules, you can view the effect of the rules on specific objects in your schema before you convert your schema. In the source schema tree, choose the object you are interested in. In the main view, choose the **Mapping** tab. The **Mapping** tab opens and displays a list of all mapping rules that are applied to the object. You can see the name of the object in the source schema and the new name of the object in the target schema. If you have data type rules, you also see the data type of the column in the source schema and the new data type of the column in the target schema.

Exporting Mapping Rules

If you use AWS Database Migration Service (AWS DMS) to migrate your data from your source database to your target database, you can provide information about your mapping rules to AWS DMS. For more information about tasks, see [Working with AWS Database Migration Service Replication Tasks](#).

To export mapping rules

1. In the AWS Schema Conversion Tool, in the source schema tree, open the context (right-click) menu and choose **Export script for DMS**. The save dialog box opens.
2. Browse to the location where you want to save your script, and then choose **Save**. Your mapping rules are saved as a JSON script that can be consumed by AWS DMS.

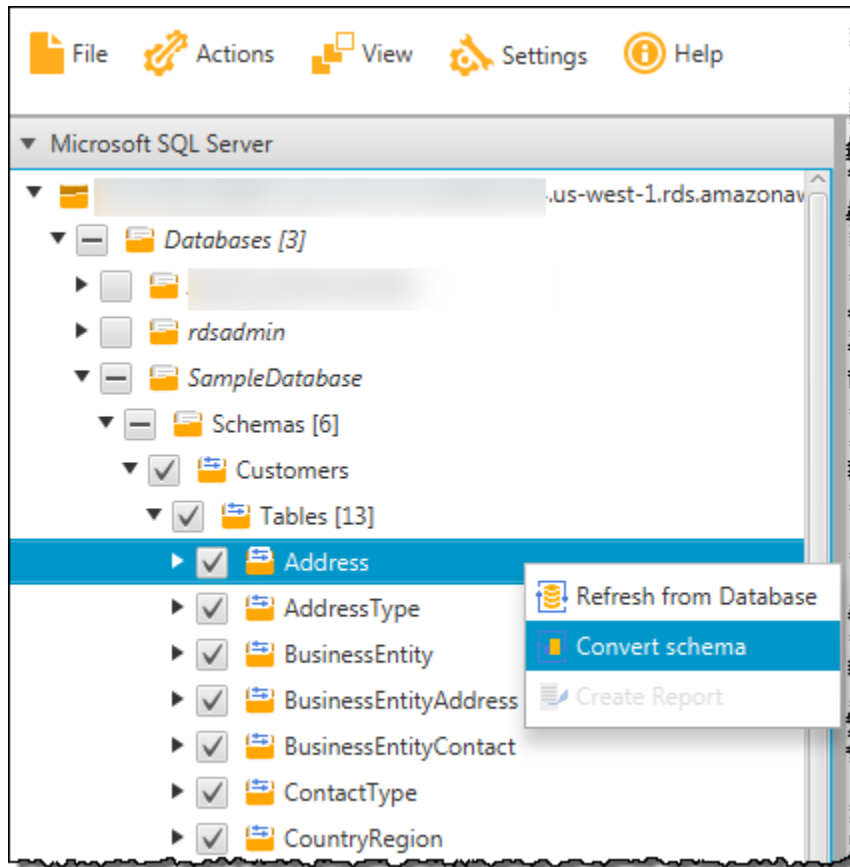
Converting Your Schema by Using the AWS Schema Conversion Tool

After you have connected your project to both your source database and your target Amazon RDS DB instance, your AWS Schema Conversion Tool (AWS SCT) project displays the schema from your source database in the left panel. The schema is presented in a tree-view format, and each node of the tree is lazy loaded. When you choose a node in the tree view, AWS SCT requests the schema information from your source database at that time.

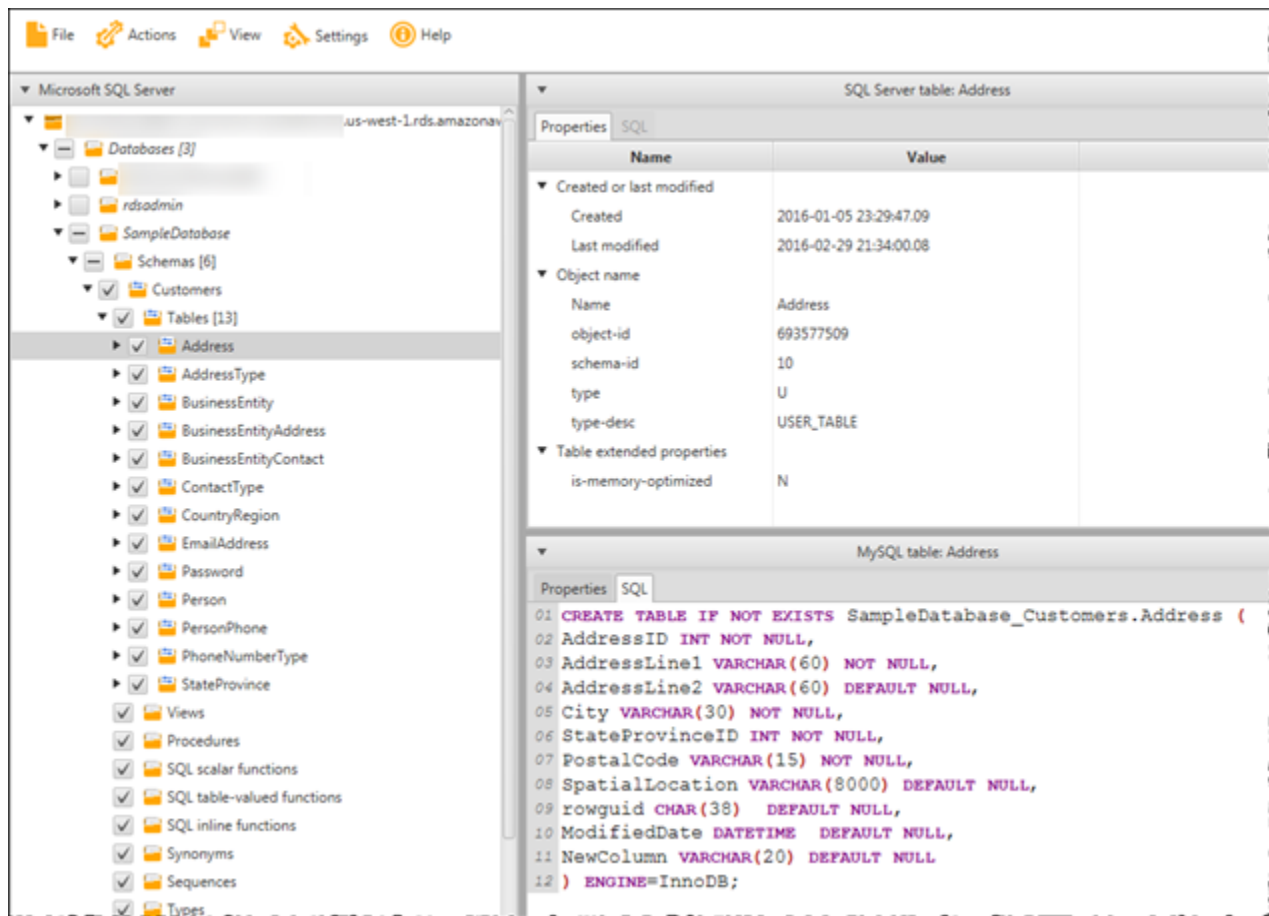
You can choose schema items from your source database and then convert the schema to equivalent schema for the DB engine of your target DB instance. You can choose any schema item from your source database to convert. If the schema item that you choose depends on a parent item, then AWS SCT also generates the schema for the parent item. For example, if you choose a column from a table to convert, then AWS SCT generates the schema for the column, the table that the column is in, and the database that the table is in.

Converting Schema

To convert schema from your source database, choose a schema object to convert from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Convert schema**, as shown following.



After you have converted the schema from your source database, you can choose schema items from the left panel of your project and view the converted schema in the center panels of your project. The lower-center panel displays the properties of and the SQL command to create the converted schema, as shown following.



After you have converted your schema, you can save your project. The schema information from your source database is saved with your project. This functionality means that you can work offline without being connected to your source database. AWS SCT connects to your source database to update the schema in your project if you choose **Refresh from Database** for your source database. For more information, see [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 81).

You can create a database migration assessment report of the items that can't be converted automatically. The assessment report is useful for identifying and resolving schema items that can't be converted automatically. For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool](#) (p. 76).

When AWS SCT generates a converted schema, it doesn't immediately apply it to the target DB instance. Instead, the converted schema is stored locally until you are ready to apply it to the target DB instance. For more information, see [Applying Your Converted Schema](#) (p. 83).

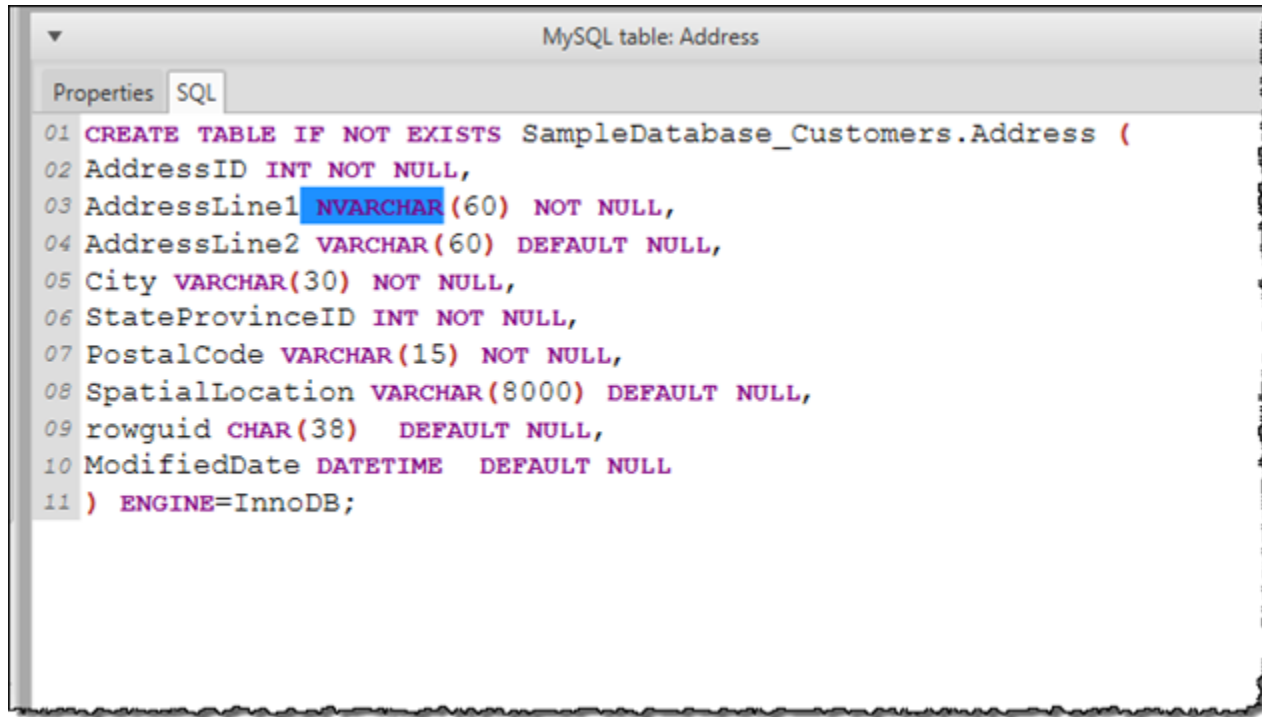
Editing Converted Schema

You can edit converted schema and save the changes as part of your project.

To edit converted schema

1. In the left panel that displays the schema from your source database, choose the schema item that you want to edit the converted schema for.

2. In the lower-center panel that displays the converted schema for the selected item, choose the **SQL** tab.
3. In the text displayed for the **SQL** tab, change the schema as needed. The schema is automatically saved with your project as you update it.



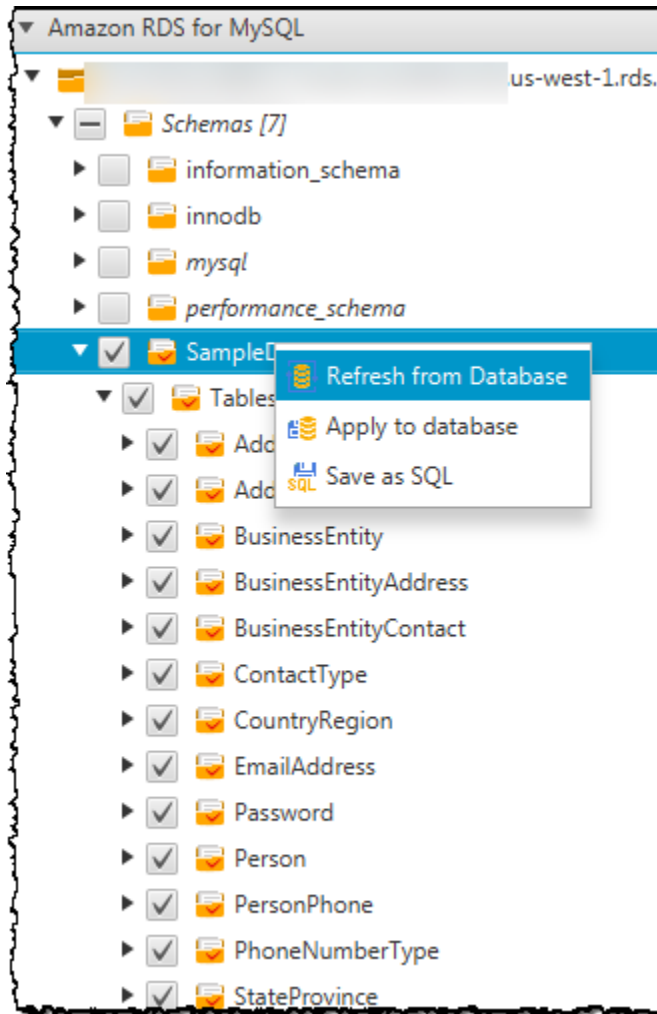
The screenshot shows a window titled "MySQL table: Address". It has two tabs: "Properties" and "SQL". The "SQL" tab is active, displaying the following SQL code:

```
01 CREATE TABLE IF NOT EXISTS SampleDatabase_Customers.Address (  
02 AddressID INT NOT NULL,  
03 AddressLine1 NVARCHAR(60) NOT NULL,  
04 AddressLine2 VARCHAR(60) DEFAULT NULL,  
05 City VARCHAR(30) NOT NULL,  
06 StateProvinceID INT NOT NULL,  
07 PostalCode VARCHAR(15) NOT NULL,  
08 SpatialLocation VARCHAR(8000) DEFAULT NULL,  
09 rowguid CHAR(38) DEFAULT NULL,  
10 ModifiedDate DATETIME DEFAULT NULL  
11 ) ENGINE=InnoDB;
```

The changes that you make to converted schema are stored with your project as you make updates. If you newly convert a schema item from your source database, and you have made updates to previously converted schema for that item, those existing updates are replaced by the newly converted schema item based on your source database.

Clearing a Converted Schema

Until you apply the schema to your target DB instance, AWS SCT only stores the converted schema locally in your project. You can clear the planned schema from your project by choosing the tree-view node for your target DB instance, and then choosing **Refresh from Database**. Because no schema has been written to your target DB instance, refreshing from the database removes the planned schema elements in your AWS SCT project to match what exists in your target DB instance.



Creating and Using the Assessment Report in the AWS Schema Conversion Tool

The AWS Schema Conversion Tool (AWS SCT) creates a *database migration assessment report* to help you convert your schema. The database migration assessment report provides important information about the conversion of the schema from your source database to your target Amazon RDS DB instance. The report summarizes all of the schema conversion tasks and details the action items for schema that can't be converted to the DB engine of your target DB instance. The report also includes estimates of the amount of effort that it will take to write the equivalent code in your target DB instance that can't be converted automatically.

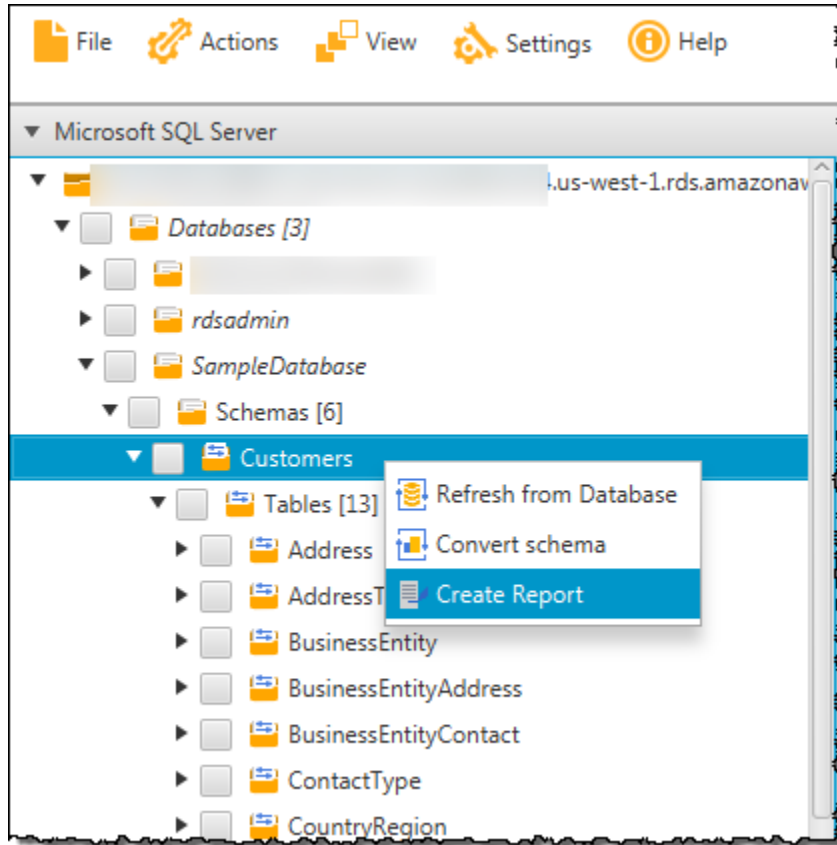
If you are using AWS SCT to copy your existing on-premises database schema to an Amazon RDS DB instance running the same engine, the report can help you analyze requirements for moving to the cloud and for changing your license type.

Creating a Database Migration Assessment Report

Use the following procedure to create a database migration assessment report.

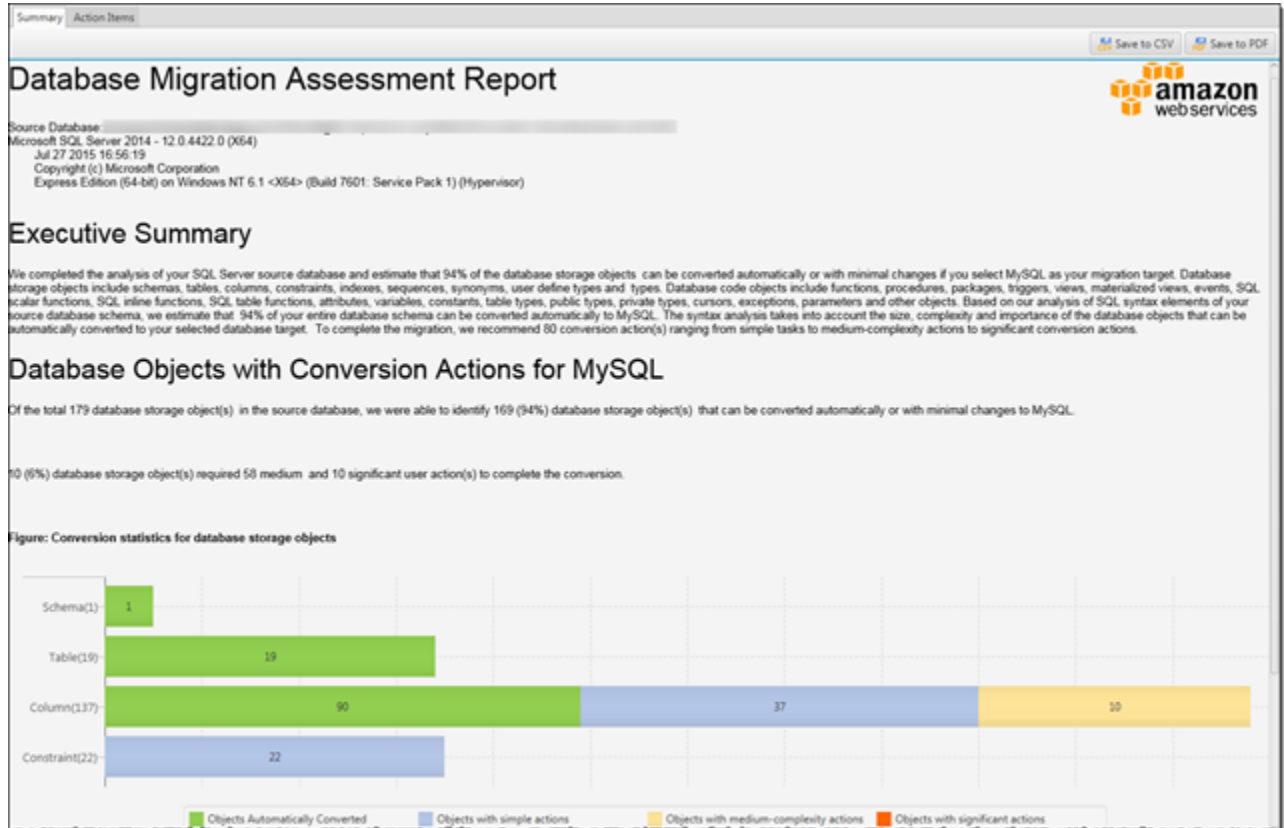
To create a database migration assessment report

1. In the left panel that displays the schema from your source database, choose a schema object to create an assessment report for.
2. Open the context (right-click) menu for the object, and then choose **Create Report**.



Assessment Report Summary

After you create an assessment report, the assessment report view opens, showing the **Summary** tab. The **Summary** tab displays the summary information from the database migration assessment report. It shows items that were converted automatically, and items that were not converted automatically.



For schema items that can't be converted automatically to the target database engine, the summary includes an estimate of the effort that it takes to create schema items in your target DB instance that are equivalent to those in your source database.

The report categorizes the estimated time to convert these schema items as follows:

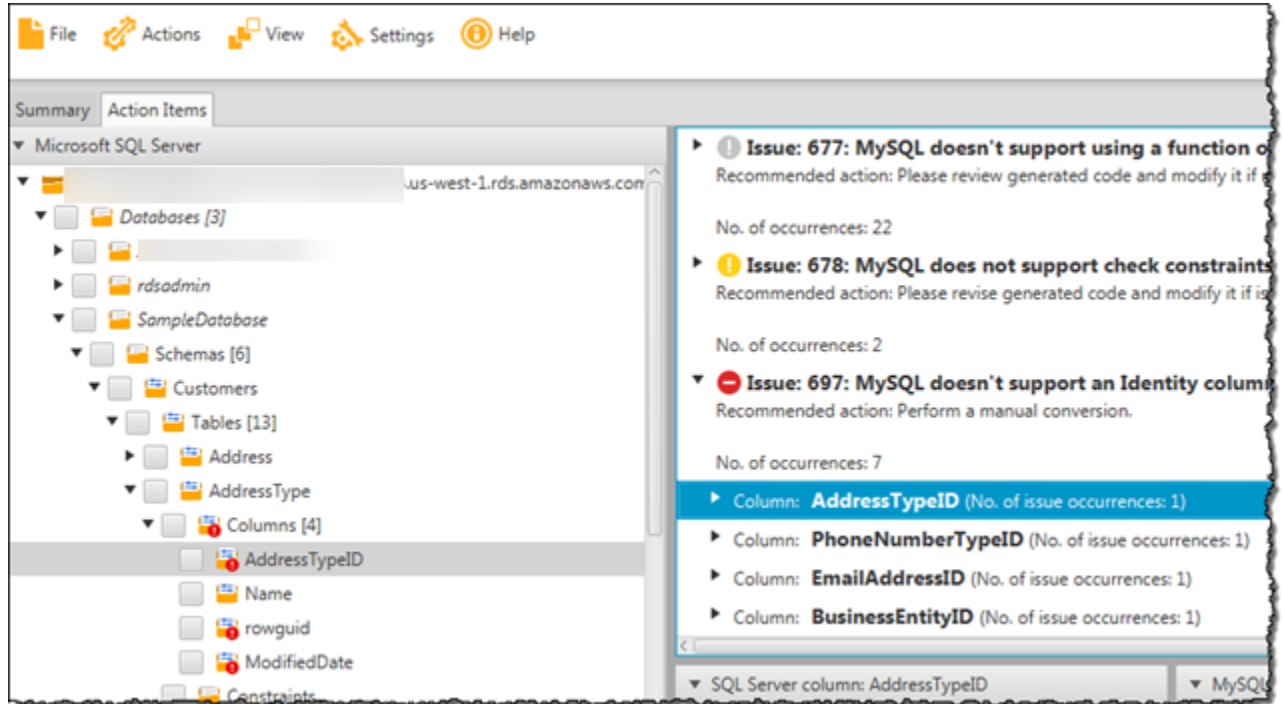
- **Simple** – Actions that can be completed in less than 1 hour.
- **Medium** – Actions that are more complex and can be completed in 1 to 4 hours.
- **Significant** – Actions that are very complex and take more than 4 hours to complete.

The section **License Evaluation and Cloud Support** contains information about moving your existing on-premises database schema to an Amazon RDS DB instance running the same engine. For example, if you want to change license types, this section of the report tells you which features from your current database should be removed.

Assessment Report Action Items

The assessment report view also includes an **Action Items** tab. This tab contains a list of items that can't be converted automatically to the database engine of your target Amazon RDS DB instance. If you select an action item from the list, AWS SCT highlights the item from your schema that the action item applies to.

The report also contains recommendations for how to manually convert the schema item. For more information about deciding how to handle manual conversions, see [Handling Manual Conversions in the AWS Schema Conversion Tool](#) (p. 80).



Saving the Assessment Report

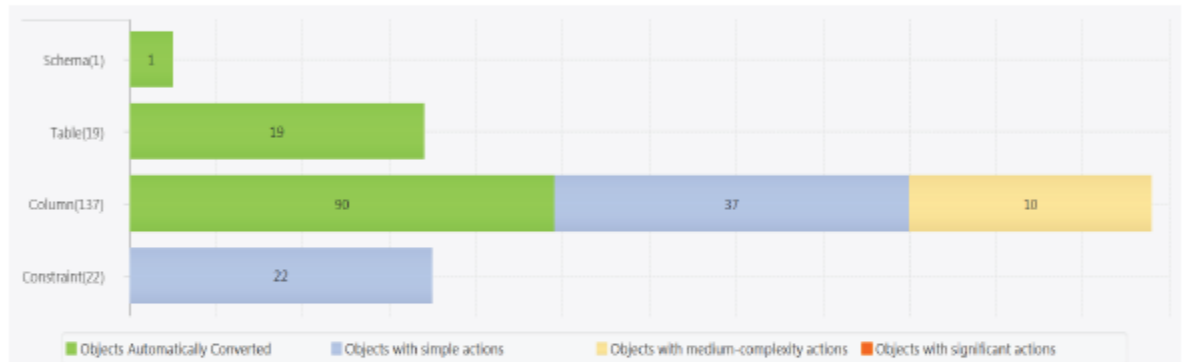
You can save a local copy of the database migration assessment report as either a PDF file or a comma-separated values (CSV) file. The CSV file contains only action item information. The PDF file contains both the summary and action item information, as shown in the following example.

Database Objects with Conversion Actions for MySQL

Of the total 179 database storage object(s) in the source database, we were able to identify 169 (94%) database storage object(s) that can be converted automatically or with minimal changes to MySQL.

10 (6%) database storage object(s) required 58 medium and 10 significant user action(s) to complete the conversion.

Figure: Conversion statistics for database storage objects



Detailed Recommendations for MySQL Migrations

If you choose to migrate your SQL Server database to MySQL, we recommend the following actions.

Storage Object Actions

Constraint Changes

Some changes are required to CONSTRAINTs that cannot be converted automatically. You'll need to address these issues manually.

Handling Manual Conversions in the AWS Schema Conversion Tool

The assessment report includes a list of items that can't be converted automatically to the database engine of your target Amazon RDS DB instance. For each item that can't be converted, there is an action item on the **Action Items** tab.

You can respond to the action items in the assessment report in the following ways:

- Modify your source database schema.
- Modify your target database schema.

Modifying Your Source Schema

For some items, it might be easier to modify the database schema in your source database to schema that can be converted automatically. First, verify that the new changes are compatible with your application architecture, then update the schema in your source database. Finally, refresh your project with the updated schema information. You can then convert your updated schema, and generate a new database migration assessment report. The action items no longer appear for the items that changed in the source schema.

The advantage of this process is that your updated schema is always available when you refresh from your source database.

Modifying Your Target Schema

For some items, it might be easier to apply the converted schema to your target database, and then add equivalent schema items manually to your target database for the items that couldn't be converted automatically. You can write all of the schema that can be converted automatically to your target DB instance by applying the schema. For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 82).

The schema that are written to your target DB instance don't contain the items that can't be converted automatically. After applying the schema to your target DB instance, you can then manually create schema in your target DB instance that are equivalent to those in the source database. The action items in the database migration assessment report contain suggestions for how to create the equivalent schema.

Warning

If you manually create schema in your target DB instance, save a copy of any manual work that you do. If you apply the converted schema from your project to your target DB instance again, it overwrites the manual work you have done.

In some cases, you can't create equivalent schema in your target DB instance. You might need to rearchitect a portion of your application and database to use the functionality that is available from the DB engine for your target DB instance. In other cases, you can simply ignore the schema that can't be converted automatically.

Related Topics

- [Converting Your Schema by Using the AWS Schema Conversion Tool](#) (p. 72)
- [Creating and Using the Assessment Report in the AWS Schema Conversion Tool](#) (p. 76)
- [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 81)

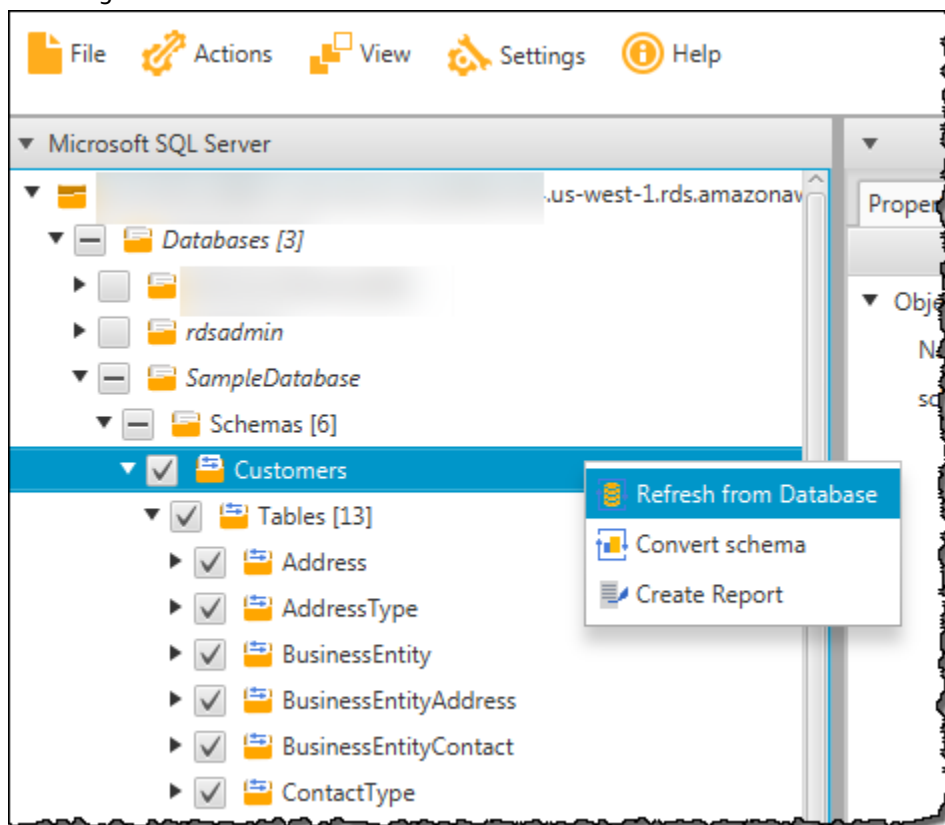
Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool

You can update both the source schema and the target schema in your AWS Schema Conversion Tool (AWS SCT) project.

- **Source** – If you update the schema for your source database, AWS SCT replaces the schema in your project with the latest schema from your source database. Using this functionality, you can update your project if changes have been made to the schema of your source database.
- **Target** – If you update the schema for your target Amazon RDS DB instance, AWS SCT replaces the schema in your project with the latest schema from your target DB instance. If you haven't applied any

schema to your target DB instance, AWS SCT clears the converted schema from your project. You can then convert the schema from your source database for a clean target DB instance.

You update the schema in your AWS SCT project by choosing **Refresh from Database**, as shown following.



Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool

When the AWS Schema Conversion Tool (AWS SCT) generates converted schema (as shown in [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 72\)](#)), it doesn't immediately apply the converted schema to the target DB instance. Instead, converted schema are stored locally in your project until you are ready to apply them to the target DB instance. Using this functionality, you can work with schema items that can't be converted automatically to your target DB engine. For more information on items that can't be converted automatically, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool \(p. 76\)](#).

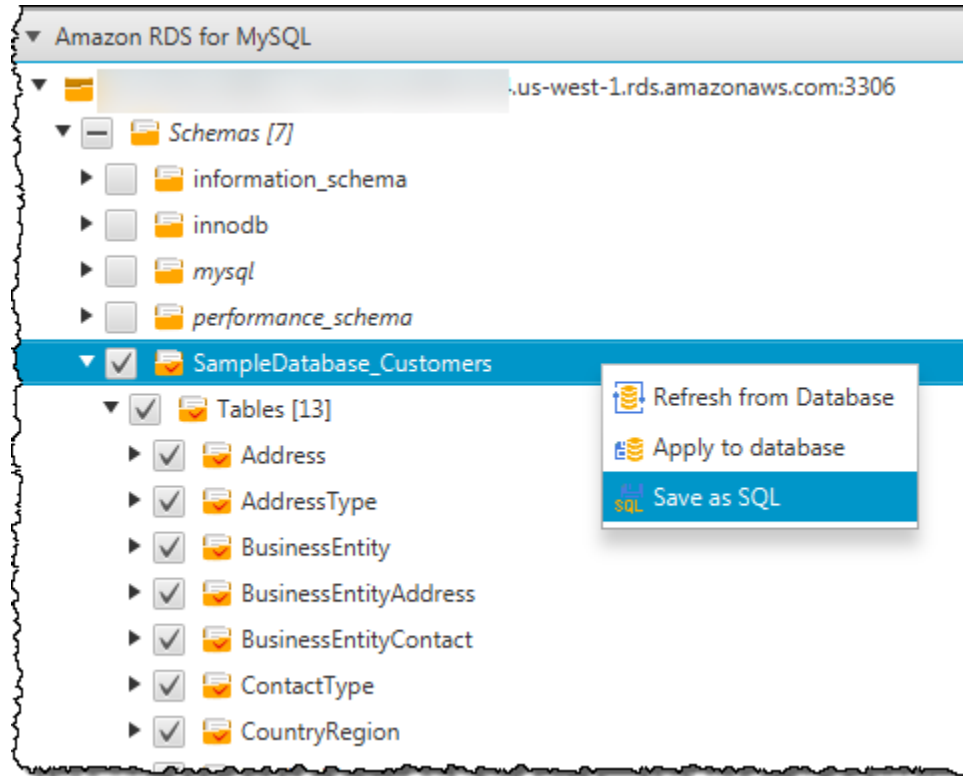
You can optionally have the tool save your converted schema to a file as a SQL script prior to applying the schema to your target DB instance. You can also have the tool apply the converted schema directly to your target DB instance.

Saving Your Converted Schema to a File

You can save your converted schema as SQL scripts in a text file. By using this approach, you can modify the generated SQL scripts from AWS SCT to address items that the tool can't convert automatically. You

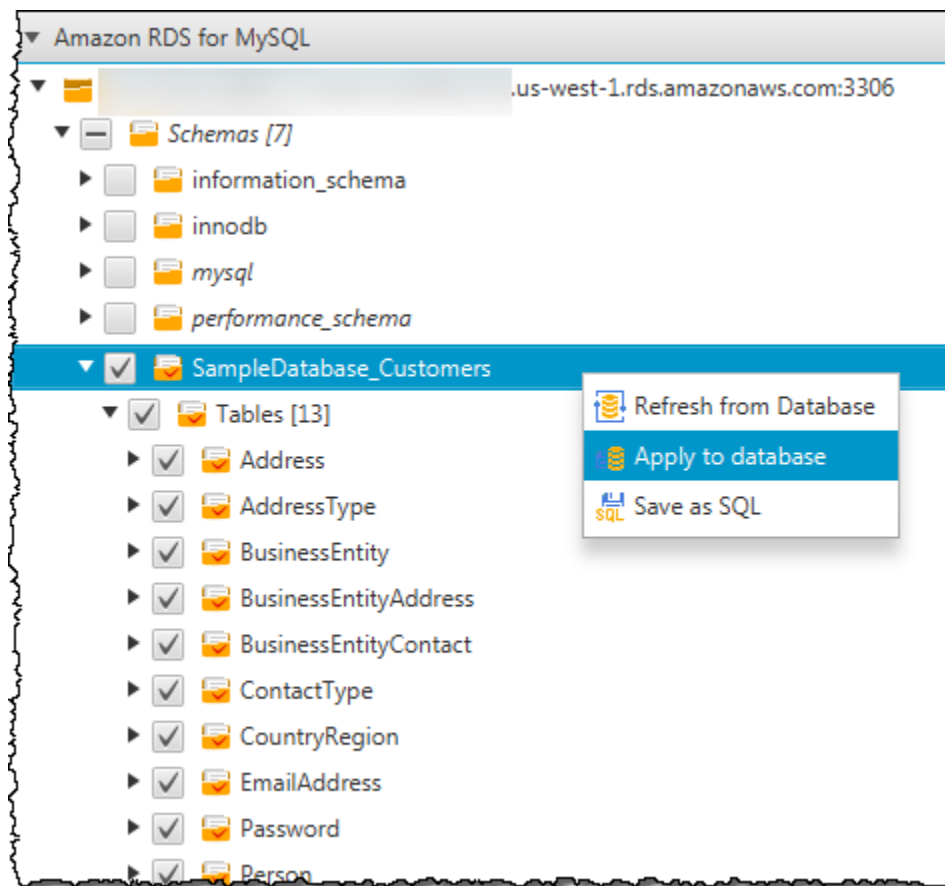
can then run your updated scripts on your target DB instance to apply your converted schema to your target database.

To save your converted schema as SQL scripts, open the context (right-click) menu for the schema element, and choose **Save as SQL**, as shown following.



Applying Your Converted Schema

When you are ready to apply your converted schema to your target Amazon RDS DB instance, choose the schema element from the right panel of your project. Open the context (right-click) menu for the schema element, and then choose **Apply to database**, as shown following.



The Extension Pack Schema

The first time that you apply your converted schema to your target DB instance, AWS SCT adds an additional schema to your target DB instance. This schema implements system functions of the source database that are required when writing your converted schema to your target DB instance. The schema is called the extension pack schema.

Don't modify the extension pack schema, or you might encounter unexpected results in the converted schema that is written to your target DB instance. When your schema is fully migrated to your target DB instance, and you no longer need AWS SCT, you can delete the extension pack schema.

The extension pack schema is named according to your source database as follows:

- Microsoft SQL Server: `AWS_SQLSERVER_EXT`
- MySQL: `AWS_MYSQL_EXT`
- Oracle: `AWS_ORACLE_EXT`
- PostgreSQL: `AWS_POSTGRESQL_EXT`

For more information, see [The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases](#) (p. 85).

The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases

When you convert your database schema, the AWS Schema Conversion Tool (AWS SCT) adds an additional schema to your target DB instance. This schema implements system functions of the source database that are required when writing your converted schema to your target DB instance. The schema is called the extension pack schema.

If you are converting a data warehouse to Amazon Redshift, instead see [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses \(p. 109\)](#).

The extension pack schema is named according to your source database as follows:

- Microsoft SQL Server: `AWS_SQLSERVER_EXT`
- MySQL: `AWS_MYSQL_EXT`
- Oracle: `AWS_ORACLE_EXT`
- PostgreSQL: `AWS_POSTGRESQL_EXT`

In two cases, you might want to install the extension pack manually:

- You accidentally delete the extension pack database schema from your target database.
- You want to use AWS services to emulate database functionality.

Using AWS Services to Emulate Database Functionality

In some cases, database features can't be converted to equivalent Amazon Relational Database Service (Amazon RDS) features. Some examples are Oracle send email calls that use `UTL_SMTP`, and Microsoft SQL Server jobs that use a job scheduler. If you host and self-manage a database on the Amazon Elastic Compute Cloud (Amazon EC2) platform, you can emulate these features by substituting AWS services for them.

The AWS SCT extension pack wizard helps you install, create, and configure AWS Lambda functions to emulate email, job scheduling, and other features.

Before You Begin

Almost all work you do with the AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

Before you install the extension pack, you need to convert your database schema. For more information, see [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool \(p. 69\)](#).

Applying the Extension Pack

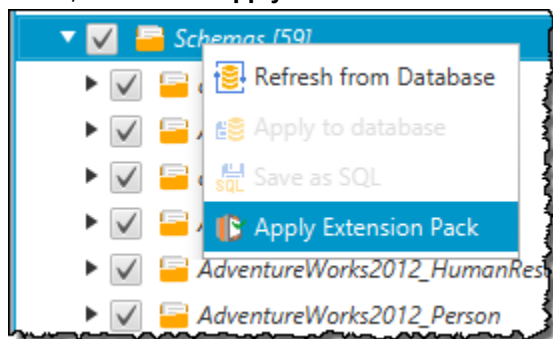
Use the following procedure to apply the extension pack.

Important

The AWS service emulation features are supported only for databases installed and self-managed on the Amazon EC2 platform. Don't install the service emulation features if your target database is on an Amazon RDS DB instance.

To apply the extension pack

1. In the AWS Schema Conversion Tool, in the target database tree, open the context (right-click) menu, and choose **Apply Extension Pack**.



The extension pack wizard appears.

2. Read the **Welcome** page, and then choose **Next**.
3. On the **AWS Services Settings** page, do the following:
 - If you are reinstalling the extension pack database schema only, choose **Skip this step for now**, and then choose **Next**.
 - If you are installing AWS services, provide the credentials to connect to your AWS account. You can use your AWS Command Line Interface (AWS CLI) credentials if you have the AWS CLI installed. You can also use credentials that you previously stored in a profile in the global application settings and associated with the project. If necessary, choose **Navigate to Project Settings** to associate a different profile with the project. If necessary, choose **Global Settings** to create a new profile. For more information, see [Storing AWS Profiles in the AWS Schema Conversion Tool \(p. 135\)](#).
4. On the **Email Sending Service** page, do the following:
 - If you are reinstalling the extension pack database schema only, choose **Skip this step for now**, and then choose **Next**.
 - If you are installing AWS services and you have an existing AWS Lambda function, you can provide it. Otherwise the wizard creates it for you. When you are done, choose **Next**.
5. On the **Job Emulation Service** page, do the following:
 - If you are reinstalling the extension pack database schema only, choose **Skip this step for now**, and then choose **Next**.
 - If you are installing AWS services and you have an existing AWS Lambda function, you can provide it. Otherwise the wizard creates it for you. When you are done, choose **Next**.
6. On the **Functions Emulation** page, choose **Create Extension Pack**. Messages appear with the status of the extension pack operations. When you are done, choose **Finish**.

Working with Data Warehouses

You can use the AWS Schema Conversion Tool (AWS SCT) to convert your existing data warehouse schema. Your converted schema is suitable for an Amazon Redshift cluster.

In some cases, source database features can't be converted to equivalent Amazon Redshift features. The AWS SCT extension pack contains a custom Python library that emulates some source database functionality on Amazon Redshift.

You can use data extraction agents to extract data from your data warehouse to prepare to migrate it to Amazon Redshift. To manage the data extraction agents, you can use AWS SCT. For more information, see [Using Data Extraction Agents \(p. 115\)](#).

You can use AWS SCT to optimize your existing Amazon Redshift database. AWS SCT recommends sort keys and distribution keys to optimize your database.

Following, you can find more information about these processes.

Topics

- [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#)
- [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses \(p. 109\)](#)
- [Optimizing Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 111\)](#)

Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool

The AWS Schema Conversion Tool (AWS SCT) automates much of the process of converting your data warehouse schema to a Amazon Redshift database schema. The source and target database engines contain many different features and capabilities, and AWS SCT attempts to create an equivalent schema in your target database wherever possible. If no direct conversion is possible, AWS SCT provides a list of possible actions for you to take.

AWS SCT supports the following data warehouse conversions.

Source Database	Target Database on Amazon Redshift
Greenplum Database (version 4.3 and later)	Amazon Redshift
Microsoft SQL Server (version 2008 and later)	Amazon Redshift
Netezza (version 7.0.3 and later)	Amazon Redshift
Oracle (version 10 and later)	Amazon Redshift
Teradata (version 13 and later)	Amazon Redshift
Vertica (version 7.2.2 and later)	Amazon Redshift

If you want to convert an online transaction processing (OLTP) database schema, see [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool \(p. 69\)](#).

Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

To convert your data warehouse schema to Amazon RDS, you take the following high-level steps:

- **Choose your optimization strategy** – To optimize how AWS SCT converts your data warehouse schema, you can choose the strategies and rules you want the tool to use.

For more information, see [Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool \(p. 90\)](#).

- **Collect statistics** – To optimize how AWS SCT converts your data warehouse schema, you can provide statistics from your source database that the tool can use. You can either collect statistics directly from the database, or upload an existing statistics file.

For more information, see [Collecting or Uploading Statistics for the AWS Schema Conversion Tool \(p. 91\)](#).

- **Create mapping rules** – Before you convert your schema with AWS SCT, you can set up rules that change the data type of columns, move objects from one schema to another, and change the names of objects.

For more information, see [Creating Mapping Rules in the AWS Schema Conversion Tool \(p. 92\)](#).

- **Convert your schema** – AWS SCT creates a local version of the converted schema for you to review, but it doesn't apply it to your target database until you are ready.

For more information, see [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 94\)](#).

- **Manage and customize keys** – After you convert your schema, you can manage and edit your keys. Key management is the heart of a data warehouse conversion.

For more information, see [Managing and Customizing Keys in the AWS Schema Conversion Tool \(p. 98\)](#).

- **Create a database migration assessment report** – AWS SCT creates a database migration assessment report that details the schema elements that can't be converted automatically. You can use this report to identify where you need to manually create a schema in your target database that is compatible with your source database.

For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool \(p. 99\)](#).

- **Decide how to handle manual conversions** – If you have schema elements that can't be converted automatically, you have two choices: update the source schema and then convert again, or create equivalent schema elements in your target database.

For more information, see [Handling Manual Conversions in the AWS Schema Conversion Tool \(p. 103\)](#).

- **Update and refresh the schema in your AWS SCT project** – You can update your AWS SCT project with the most recent schema from your source database.

For more information, see [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool \(p. 104\)](#).

- **Apply the converted schema to your target database** – When you are ready, have AWS SCT apply the converted schema in your local project to your target database.

For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool \(p. 105\)](#).

Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool

To optimize how the AWS Schema Conversion Tool (AWS SCT) converts your data warehouse schema, you can choose the strategies and rules you want the tool to use. After converting your schema, and reviewing the suggested keys, you can adjust your rules or change your strategy to get the results you want.

To choose your optimization strategies and rules

1. Choose **Settings**, and then choose **Project Settings**. The **Current project settings** dialog box appears.
2. In the left pane, choose **Optimization Strategies**. The optimization strategies appear in the right pane with the defaults selected.
3. For **Strategy Sector**, choose the optimization strategy you want to use. You can choose from the following:
 - **Use metadata, ignore statistical information** – In this strategy, only information from the metadata is used for optimization decisions. For example, if there is more than one index on a source table, the source database sort order is used, and the first index becomes a distribution key.
 - **Ignore metadata, use statistical information** – In this strategy, optimization decisions are derived from statistical information only. This strategy applies only to tables and columns for which statistics are provided. For more information, see [Collecting or Uploading Statistics for the AWS Schema Conversion Tool \(p. 91\)](#).
 - **Use metadata and use statistical information** – In this strategy, both metadata and statistics are used for optimization decisions.
4. After you choose your optimization strategy, you can choose the rules you want to use. You can choose from the following:
 - **Choose Distribution Key and Sort Keys using metadata**
 - **Choose fact table and appropriate dimension for collation**
 - **Analyze cardinality of indexes' columns**
 - **Find the most used tables and columns from QueryLog table**

For each rule, you can enter a weight for the sort key and a weight for the distribution key. AWS SCT uses the weights you choose when it converts your schema. Later, when you review the suggested keys, if you are not satisfied with the results, you can return here and change your settings. For more information, see [Managing and Customizing Keys in the AWS Schema Conversion Tool \(p. 98\)](#).

Related Topics

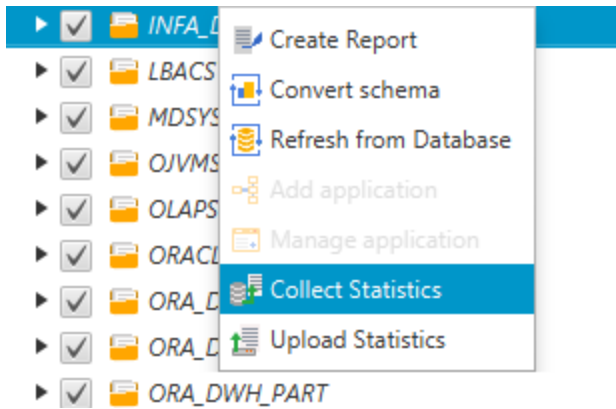
- [Choose the Best Sort Key](#)
- [Choose the Best Distribution Style](#)

Collecting or Uploading Statistics for the AWS Schema Conversion Tool

To optimize how the AWS Schema Conversion Tool (AWS SCT) converts your data warehouse schema, you can provide statistics from your source database that the tool can use. You can either collect statistics directly from the database, or upload an existing statistics file.

To provide and review statistics

1. Connect to your source database. For more information, see [Connecting to Your Source Database \(p. 14\)](#).
2. Choose a schema object from the left panel of your project, and open the context (right-click) menu for the object. Choose **Collect Statistics** or **Upload Statistics** as shown following.



3. Choose a schema object from the left panel of your project, and then choose the **Statistics** tab. You can review the statistics for the object.

The screenshot shows the 'Statistics' tab for table 'T_PROD_SPEC'. It displays the following information:

- Table: T_PROD_SPEC
- Stats Collection Date: 2016-06-14 15:41:23
- Stats Collection Mode: online
- Stats Reference Count: (partially visible)
- Stats Row Count: (partially visible)

Column Name	Stats Collection Date	Stats collection mode	Stats usage co
PART_ID	2016-06-14 15:41:23	online	
ADJUSTER_ID	2016-06-14 15:41:23	online	
SPEC_ID	2016-06-14 15:41:23	online	

Later, when you review the suggested keys, if you are not satisfied with the results, you can collect additional statistics and repeat this procedure. For more information, see [Managing and Customizing Keys in the AWS Schema Conversion Tool \(p. 98\)](#).

Related Topics

- [Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool \(p. 90\)](#)
- [Choose the Best Sort Key](#)
- [Choose the Best Distribution Style](#)

Creating Mapping Rules in the AWS Schema Conversion Tool

Before you convert your schema with the AWS Schema Conversion Tool (AWS SCT), you can set up rules that change the data type of columns, move objects from one schema to another, and change the names

of objects. For example, if you have a set of tables in your source schema named `test_TABLE_NAME`, you can set up a rule that changes the prefix `test_` to the prefix `demo_` in the target schema.

Note

You can only create mapping rules if your source database engine and target database engine are different.

You can create mapping rules that perform the following tasks:

- Change data type
- Move objects
- Rename objects
- Prefix - add prefix, remove prefix, replace prefix
- Suffix - add suffix, remove suffix, replace suffix

You can create mapping rules for the following objects:

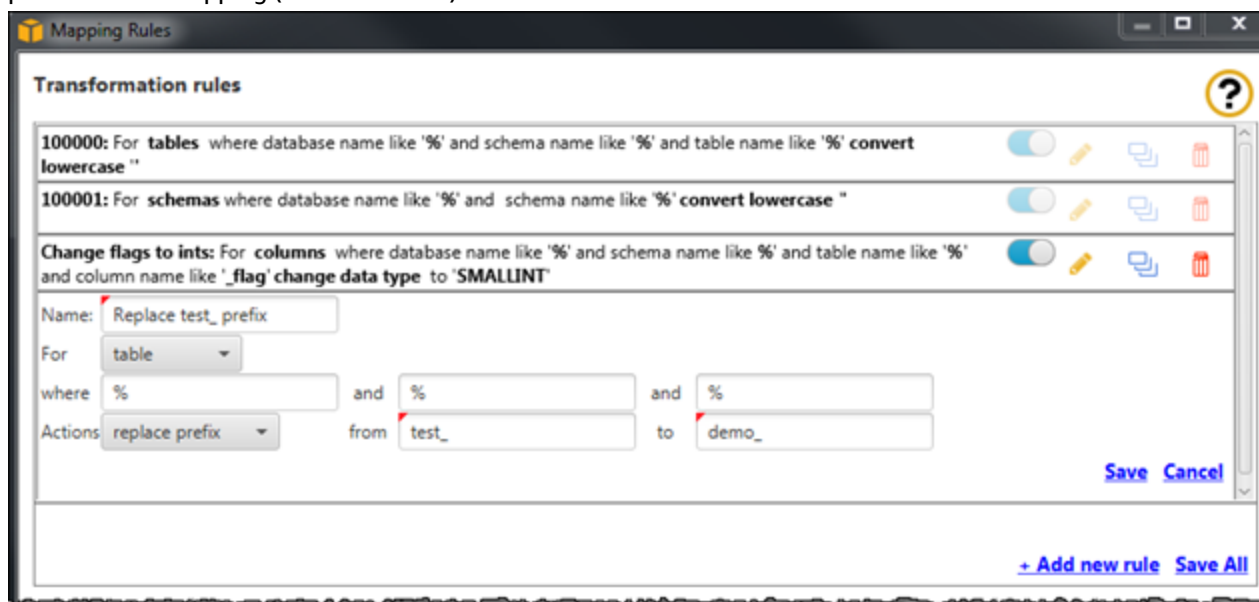
- Database
- Schema
- Table
- Column

Creating Mapping Rules

You can create mapping rules and save the rules as part of your project. With your project open, use the following procedure to create mapping rules.

To create mapping rules

1. Choose **Mapping Rules** from the **Settings** menu. The **Mapping Rules** dialog box appears. The top pane contains mapping (transformation) rules.



2. In the **Transformation Rules** pane, choose **Add new rule**.
3. Configure your transformation rule.

- a. For **Name**, type a name for your rule.
 - b. For **For**, choose the type of object that the rule applies to.
 - c. For **where**, type a filter to apply to objects before applying the mapping rule. The where clause is evaluated by using a like clause. You can enter an exact name to select one object, or you can enter a pattern to select multiple objects.

The fields available for the **where** clause are different depending on the type of the object. For example, if the object type is schema there is only one field available, for the schema name.
 - d. For **Actions**, choose the type of mapping rule you want to create.
 - e. Depending on the rule type, type one or two additional values. For example, to rename an object, type the new name of the object. To replace a prefix, type the old prefix and the new prefix.
4. After you have configured your mapping rule, choose **Save** to save your rule. You can also choose **Cancel** to cancel your changes.
 5. After you are done adding, editing, and deleting rules, choose **Save All** to save all your changes.
 6. Choose **Close** to close the **Mapping Rules** dialog box.

You can use the toggle icon to turn off a mapping rule without deleting it. You can use the copy icon to duplicate an existing mapping rule. You can use the delete icon to delete an existing mapping rule. To save any changes you make to your mapping rules, choose **Save All**.

Viewing Mapping Rules for Objects

After you set up your mapping rules, you can view the effect of the rules on specific objects in your schema before you convert your schema. In the source schema tree, choose the object you are interested in. In the main view, choose the **Mapping** tab. The **Mapping** tab opens and displays a list of all mapping rules that are applied to the object. You can see the name of the object in the source schema and the new name of the object in the target schema. If you have data type rules, you also see the data type of the column in the source schema and the new data type of the column in the target schema.

Exporting Mapping Rules

If you use AWS Database Migration Service (AWS DMS) to migrate your data from your source database to your target database, you can provide information about your mapping rules to AWS DMS. For more information about tasks, see [Working with AWS Database Migration Service Replication Tasks](#).

To export mapping rules

1. In the AWS Schema Conversion Tool, in the source schema tree, open the context (right-click) menu and choose **Export script for DMS**. The save dialog box opens.
2. Browse to the location where you want to save your script, and then choose **Save**. Your mapping rules are saved as a JSON script that can be consumed by AWS DMS.

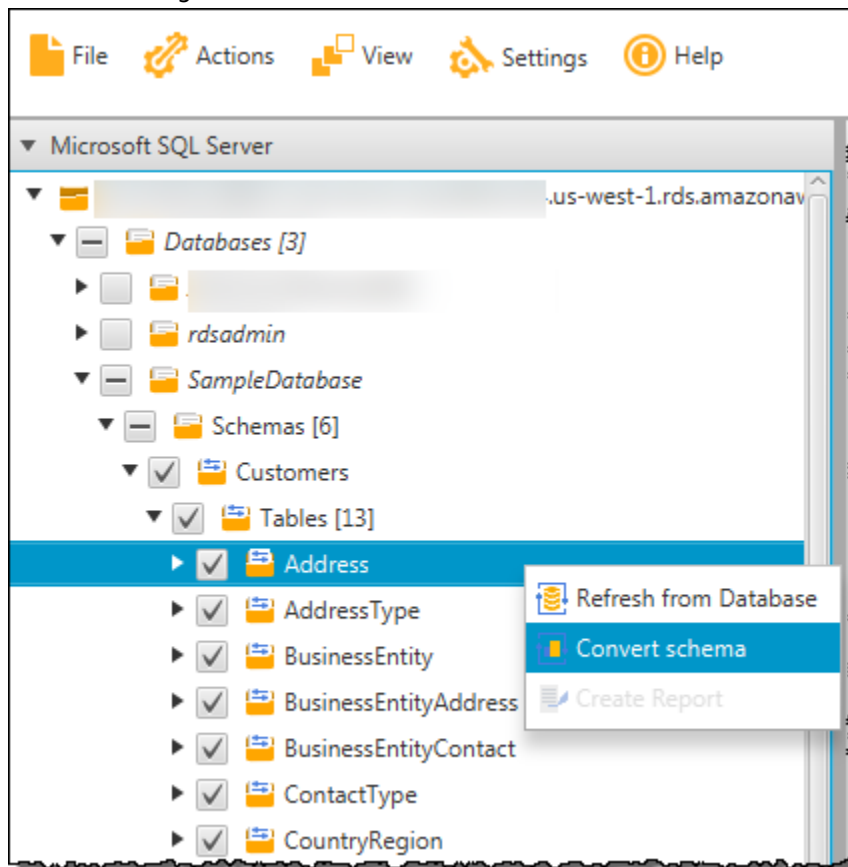
Converting Your Schema by Using the AWS Schema Conversion Tool

After you have connected your project to both your source database and your target database, your AWS Schema Conversion Tool (AWS SCT) project displays the schema from your source database in the left panel. The schema is presented in a tree-view format, and each node of the tree is lazy loaded. When you choose a node in the tree view, AWS SCT requests the schema information from your source database at that time.

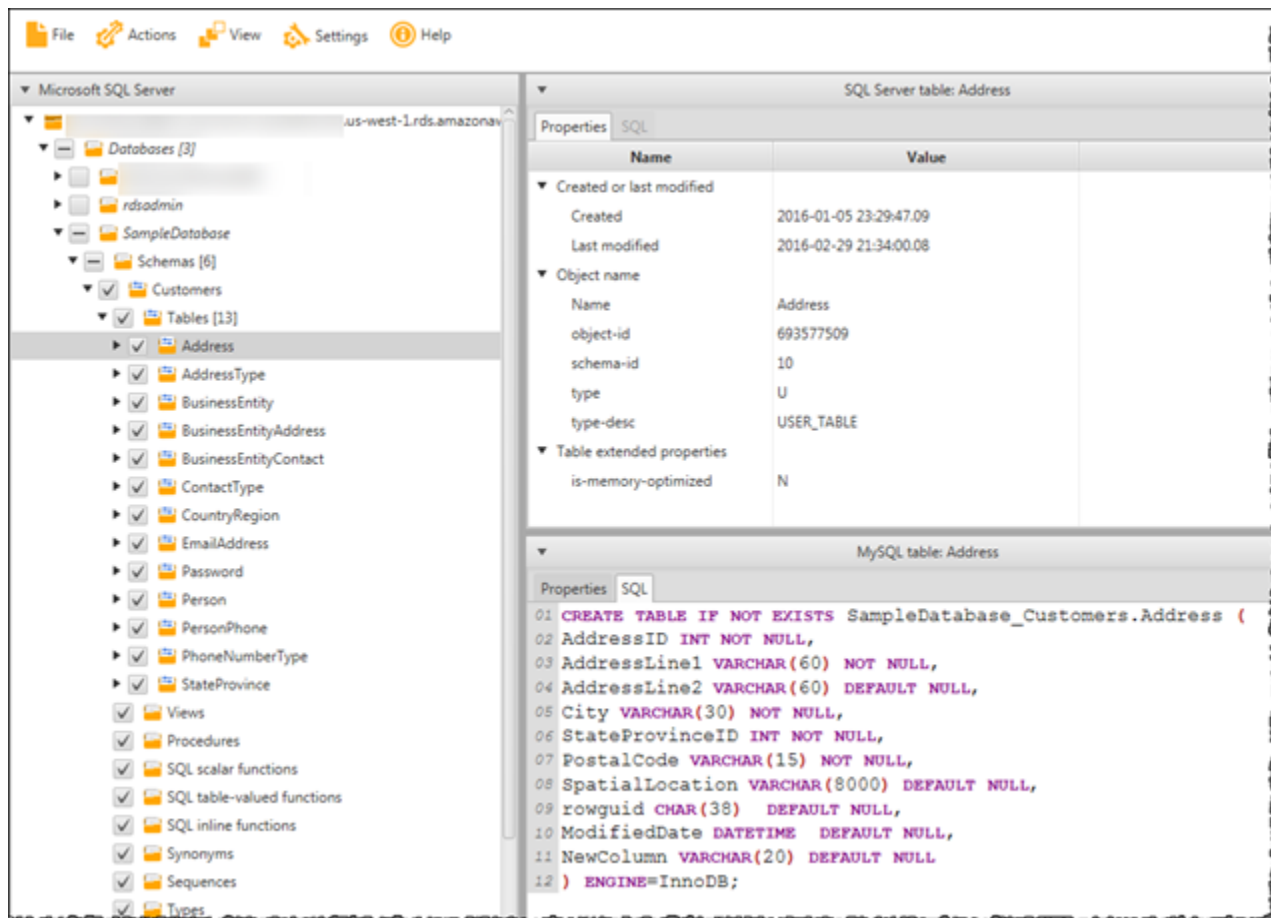
You can choose schema items from your source database and then convert the schema to equivalent schema for the database engine of your target database. You can choose any schema item from your source database to convert. If the schema item that you choose depends on a parent item, then AWS SCT also generates the schema for the parent item. For example, if you choose a column from a table to convert, then AWS SCT generates the schema for the column, the table that the column is in, and the database that the table is in.

Converting Schema

To convert schema from your source database, choose a schema object to convert from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Convert schema**, as shown following.



After you have converted the schema from your source database, you can choose schema items from the left panel of your project and view the converted schema in the center panels of your project. The lower-center panel displays the properties of and the SQL command to create the converted schema, as shown following.



After you have converted your schema, you can save your project. The schema information from your source database is saved with your project. This functionality means that you can work offline without being connected to your source database. AWS SCT connects to your source database to update the schema in your project if you choose **Refresh from Database** for your source database. For more information, see [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool](#) (p. 104).

You can create a database migration assessment report of the items that can't be converted automatically. The assessment report is useful for identifying and resolving schema items that can't be converted automatically. For more information, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool](#) (p. 99).

When AWS SCT generates a converted schema, it doesn't immediately apply it to the target database. Instead, the converted schema is stored locally until you are ready to apply it to the target database. For more information, see [Applying Your Converted Schema](#) (p. 106).

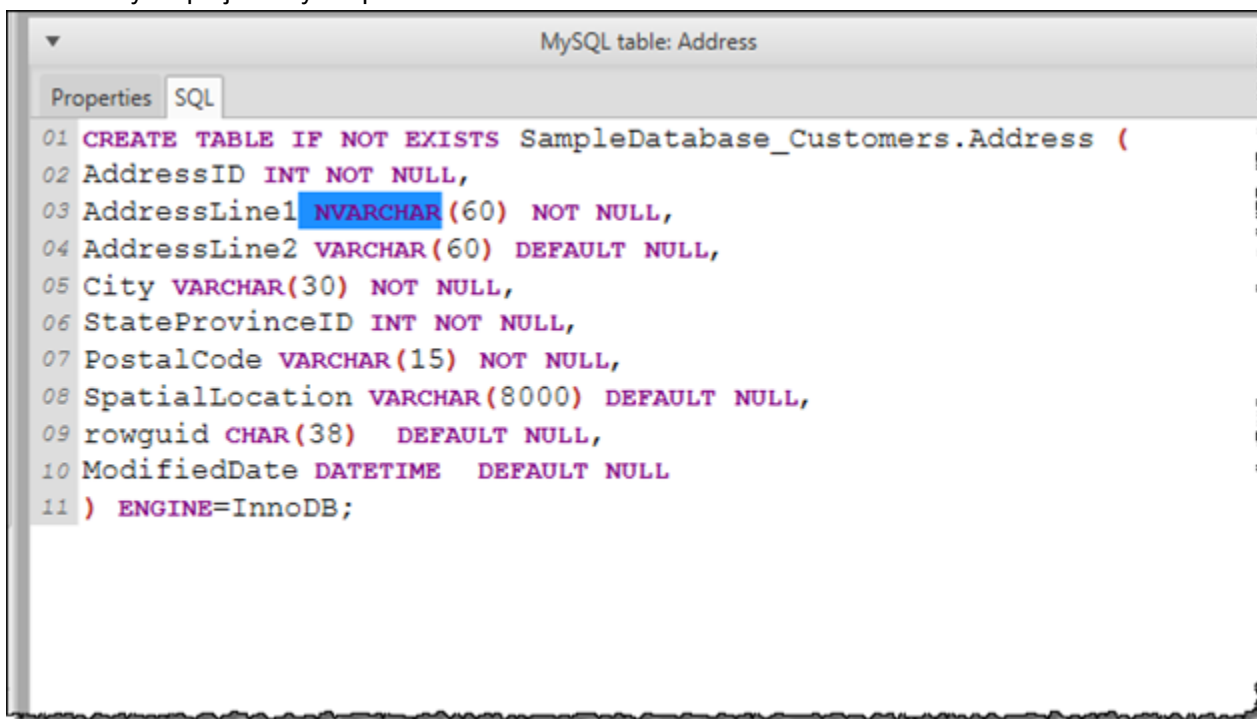
Editing Converted Schema

You can edit converted schema and save the changes as part of your project.

To edit converted schema

1. In the left panel that displays the schema from your source database, choose the schema item that you want to edit the converted schema for.

2. In the lower-center panel that displays the converted schema for the selected item, choose the **SQL** tab.
3. In the text displayed for the **SQL** tab, change the schema as needed. The schema is automatically saved with your project as you update it.



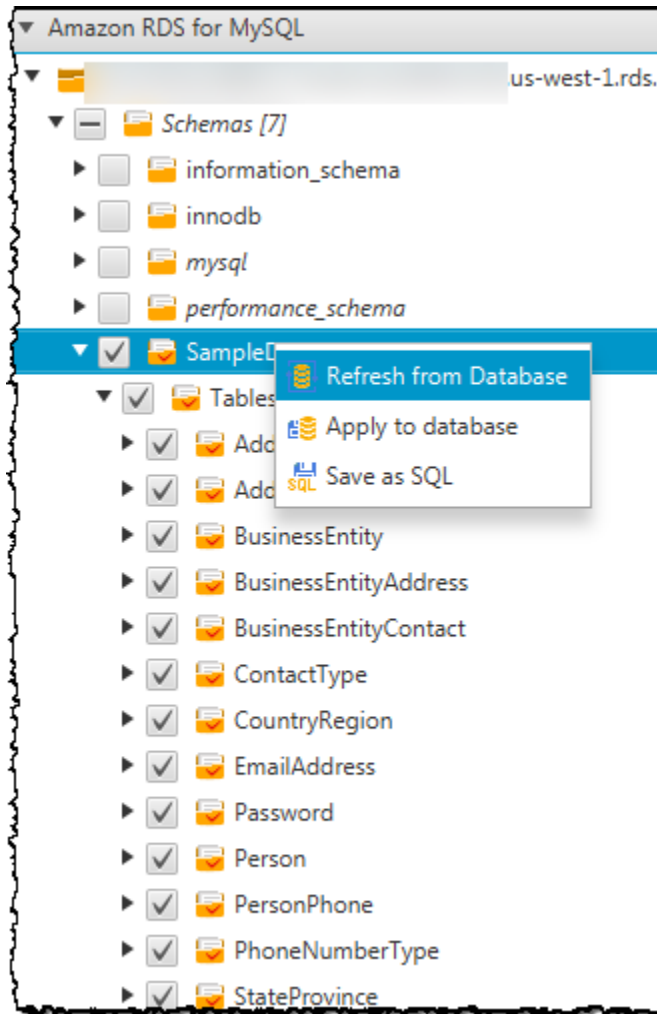
The screenshot shows a window titled "MySQL table: Address". It has two tabs: "Properties" and "SQL". The "SQL" tab is active, displaying the following SQL code:

```
01 CREATE TABLE IF NOT EXISTS SampleDatabase_Customers.Address (  
02 AddressID INT NOT NULL,  
03 AddressLine1 NVARCHAR(60) NOT NULL,  
04 AddressLine2 VARCHAR(60) DEFAULT NULL,  
05 City VARCHAR(30) NOT NULL,  
06 StateProvinceID INT NOT NULL,  
07 PostalCode VARCHAR(15) NOT NULL,  
08 SpatialLocation VARCHAR(8000) DEFAULT NULL,  
09 rowguid CHAR(38) DEFAULT NULL,  
10 ModifiedDate DATETIME DEFAULT NULL  
11 ) ENGINE=InnoDB;
```

The changes that you make to converted schema are stored with your project as you make updates. If you newly convert a schema item from your source database, and you have made updates to previously converted schema for that item, those existing updates are replaced by the newly converted schema item based on your source database.

Clearing a Converted Schema

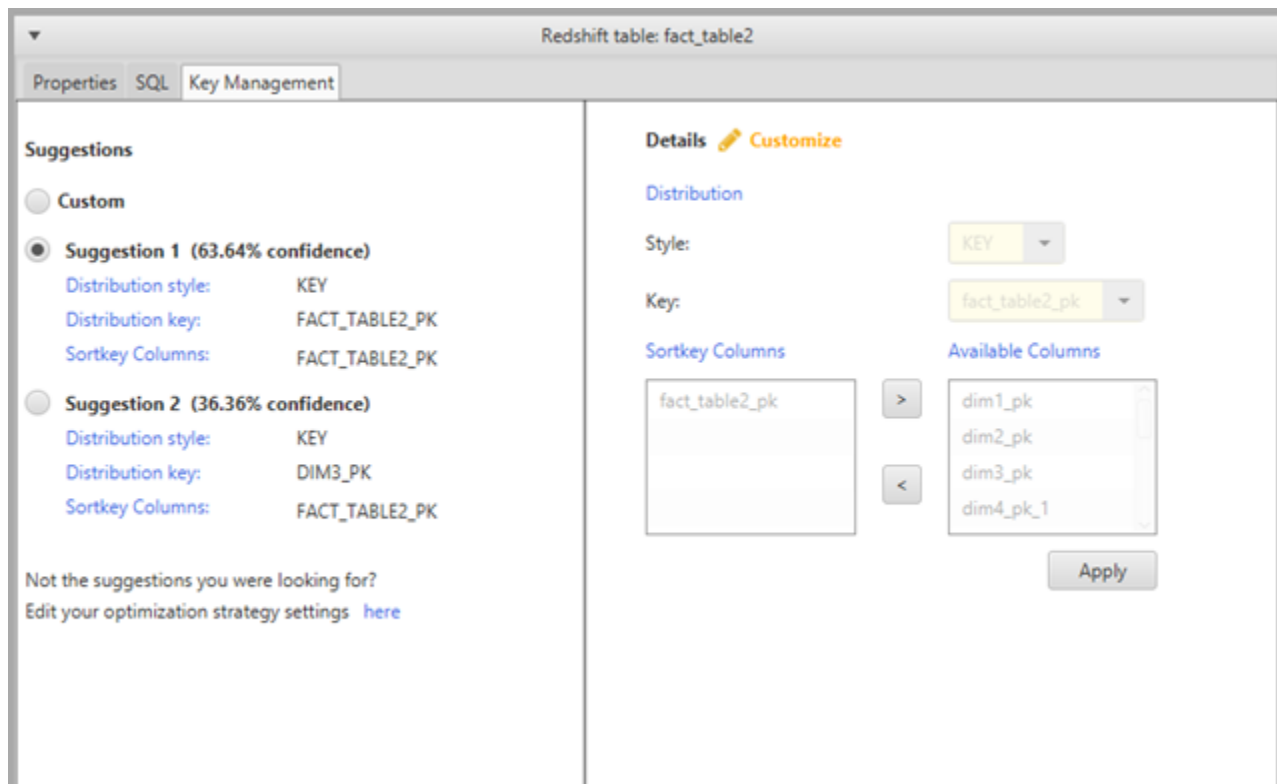
Until you apply the schema to your target database, AWS SCT only stores the converted schema locally in your project. You can clear the planned schema from your project by choosing the tree-view node for your target database, and then choosing **Refresh from Database**. Because no schema has been written to your target database, refreshing from the database removes the planned schema elements in your AWS SCT project to match what exists in your target database.



Managing and Customizing Keys in the AWS Schema Conversion Tool

After you convert your schema with the AWS Schema Conversion Tool (AWS SCT), you can manage and edit your keys. Key management is the heart of a data warehouse conversion.

To manage keys, select a table in your target database, and then choose the **Key Management** tab as shown following.



The left pane contains key suggestions, and includes the confidence rating for each suggestion. You can choose one of the suggestions, or you can customize the key by editing it in the right pane.

If the choices for the key don't look like what you expected, you can edit your edit your optimization strategies, and then retry the conversion. For more information, see [Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool](#) (p. 90).

Related Topics

- [Choose the Best Sort Key](#)
- [Choose the Best Distribution Style](#)

Creating and Using the Assessment Report in the AWS Schema Conversion Tool

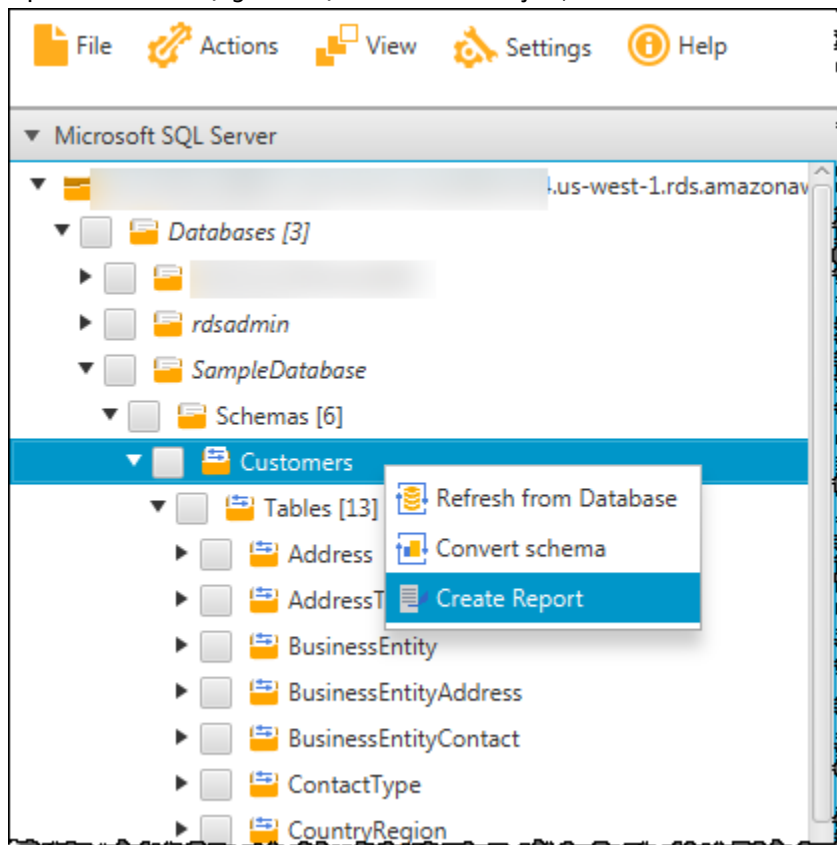
The AWS Schema Conversion Tool (AWS SCT) creates a *database migration assessment report* to help you convert your schema. The database migration assessment report provides important information about the conversion of the schema from your source database to your target database. The report summarizes all of the schema conversion tasks and details the action items for schema that can't be converted to the DB engine of your target database. The report also includes estimates of the amount of effort that it will take to write the equivalent code in your target database that can't be converted automatically.

Creating a Database Migration Assessment Report

Use the following procedure to create a database migration assessment report.

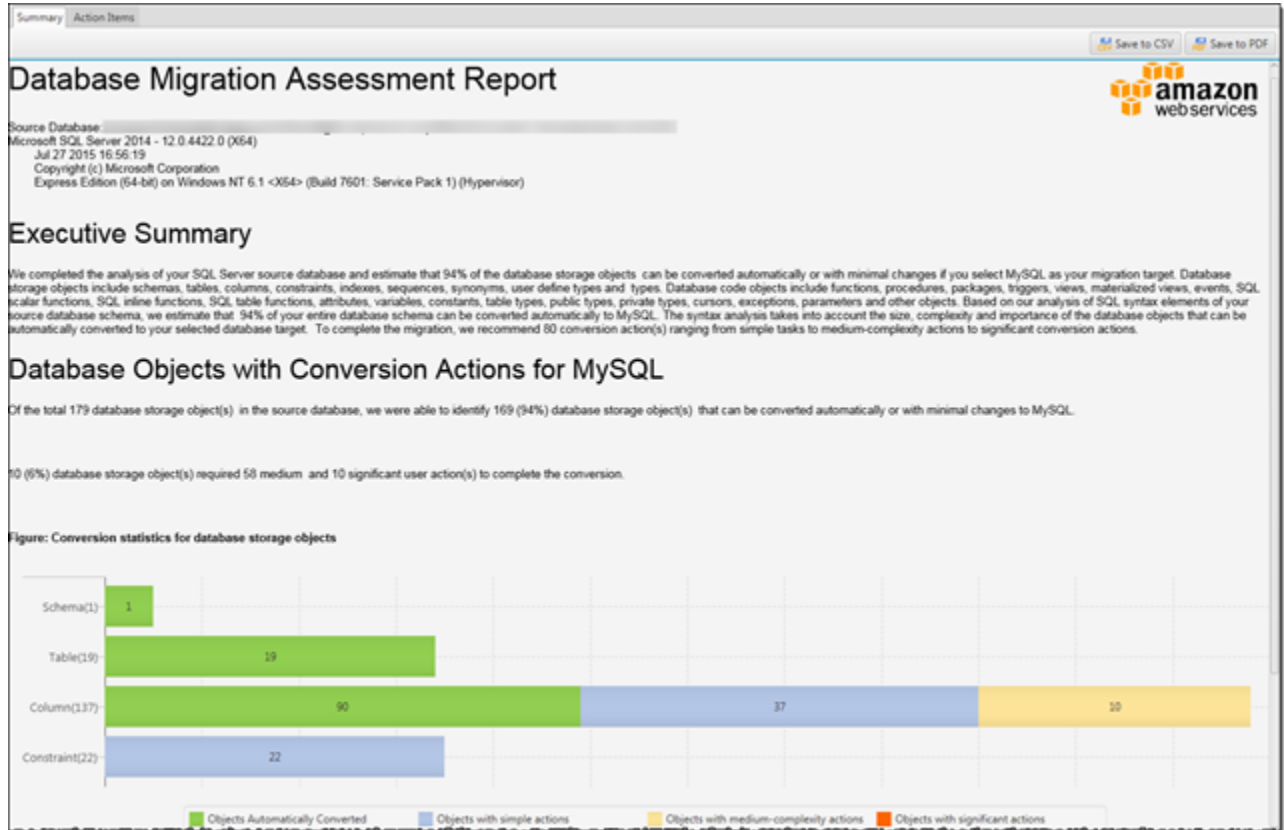
To create a database migration assessment report

1. In the left panel that displays the schema from your source database, choose a schema object to create an assessment report for.
2. Open the context (right-click) menu for the object, and then choose **Create Report**.



Assessment Report Summary

After you create an assessment report, the assessment report view opens, showing the **Summary** tab. The **Summary** tab displays the summary information from the database migration assessment report. It shows items that were converted automatically, and items that were not converted automatically.



For schema items that can't be converted automatically to the target database engine, the summary includes an estimate of the effort that it will take to create schema items in your target database that are equivalent to those in your source database.

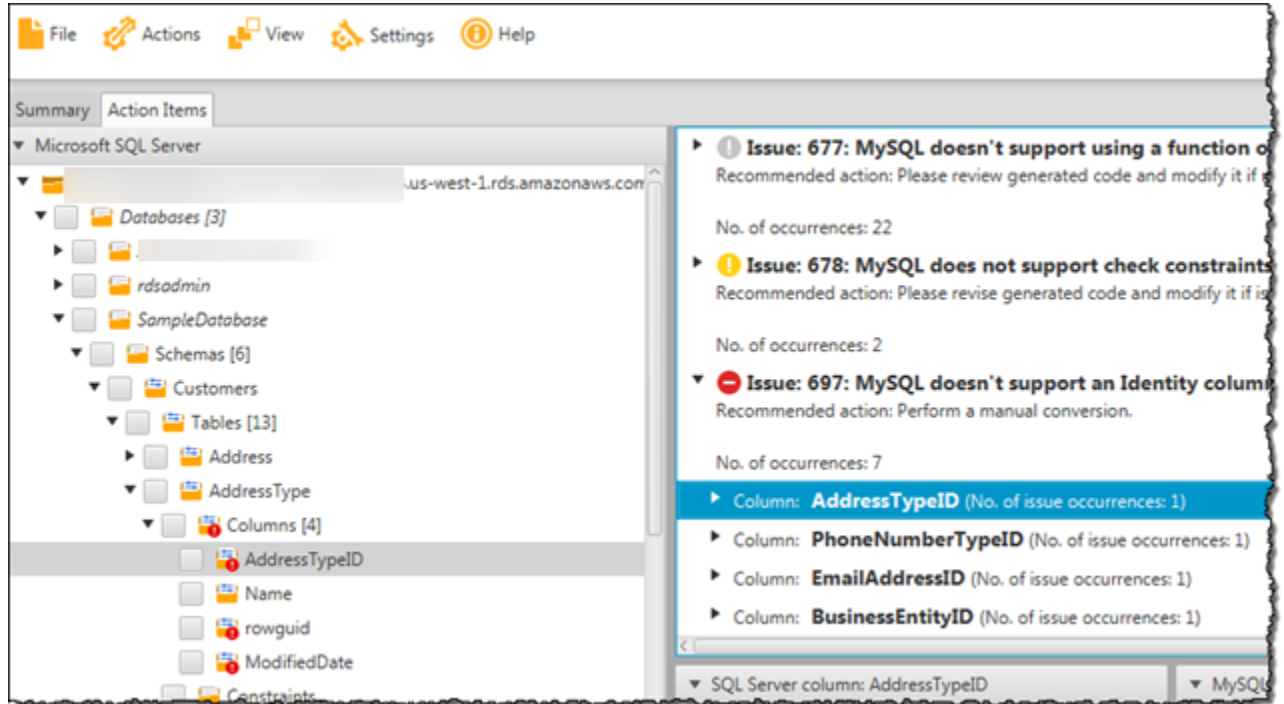
The report categorizes the estimated time to convert these schema items as follows:

- **Simple** – Actions that can be completed in less than 1 hour.
- **Medium** – Actions that are more complex and can be completed in 1 to 4 hours.
- **Significant** – Actions that are very complex and take more than 4 hours to complete.

Assessment Report Action Items

The assessment report view also includes an **Action Items** tab. This tab contains a list of items that can't be converted automatically to the database engine of your target database. If you select an action item from the list, AWS SCT highlights the item from your schema that the action item applies to.

The report also contains recommendations for how to manually convert the schema item. For more information about deciding how to handle manual conversions, see [Handling Manual Conversions in the AWS Schema Conversion Tool \(p. 103\)](#).



Saving the Assessment Report

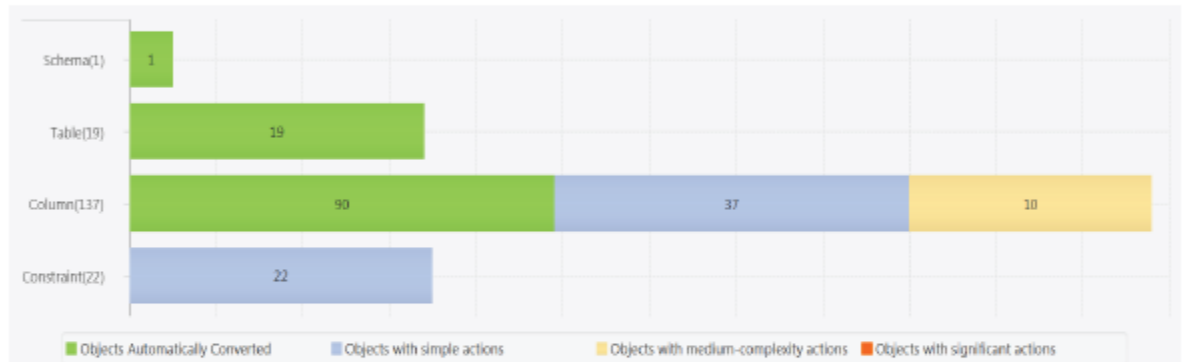
You can save a local copy of the database migration assessment report as either a PDF file or a comma-separated values (CSV) file. The CSV file contains only action item information. The PDF file contains both the summary and action item information, as shown in the following example.

Database Objects with Conversion Actions for MySQL

Of the total 179 database storage object(s) in the source database, we were able to identify 169 (94%) database storage object(s) that can be converted automatically or with minimal changes to MySQL.

10 (6%) database storage object(s) required 58 medium and 10 significant user action(s) to complete the conversion.

Figure: Conversion statistics for database storage objects



Detailed Recommendations for MySQL Migrations

If you choose to migrate your SQL Server database to MySQL, we recommend the following actions.

Storage Object Actions

Constraint Changes

Some changes are required to CONSTRAINTs that cannot be converted automatically. You'll need to address these issues manually.

Handling Manual Conversions in the AWS Schema Conversion Tool

The assessment report includes a list of items that can't be converted automatically to the database engine of your target database. For each item that can't be converted, there is an action item on the **Action Items** tab.

You can respond to the action items in the assessment report in the following ways:

- Modify your source database schema.
- Modify your target database schema.

Modifying Your Source Schema

For some items, it might be easier to modify the database schema in your source database to schema that can be converted automatically. First, verify that the new changes are compatible with your application architecture, then update the schema in your source database. Finally, refresh your project with the updated schema information. You can then convert your updated schema, and generate a new database migration assessment report. The action items no longer appear for the items that changed in the source schema.

The advantage of this process is that your updated schema is always available when you refresh from your source database.

Modifying Your Target Schema

For some items, it might be easier to apply the converted schema to your target database, and then add equivalent schema items manually to your target database for the items that couldn't be converted automatically. You can write all of the schema that can be converted automatically to your target database by applying the schema. For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool \(p. 105\)](#).

The schema that are written to your target database don't contain the items that can't be converted automatically. After applying the schema to your target database, you can then manually create schema in your target database that are equivalent to those in the source database. The action items in the database migration assessment report contain suggestions for how to create the equivalent schema.

Warning

If you manually create schema in your target database, save a copy of any manual work that you do. If you apply the converted schema from your project to your target database again, it overwrites the manual work you have done.

In some cases, you can't create equivalent schema in your target database. You might need to rearchitect a portion of your application and database to use the functionality that is available from the engine for your target database. In other cases, you can simply ignore the schema that can't be converted automatically.

Related Topics

- [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 94\)](#)
- [Creating and Using the Assessment Report in the AWS Schema Conversion Tool \(p. 99\)](#)
- [Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool \(p. 104\)](#)

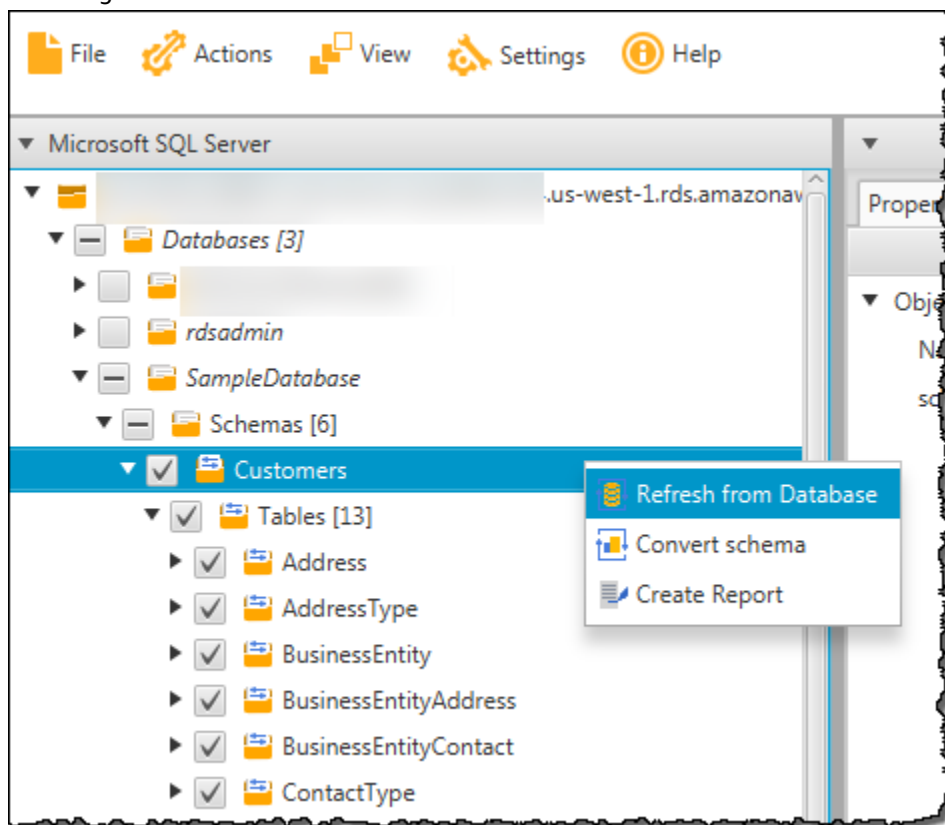
Updating and Refreshing Your Converted Schema in the AWS Schema Conversion Tool

You can update both the source schema and the target schema in your AWS Schema Conversion Tool (AWS SCT) project.

- **Source** – If you update the schema for your source database, AWS SCT replaces the schema in your project with the latest schema from your source database. Using this functionality, you can update your project if changes have been made to the schema of your source database.
- **Target** – If you update the schema for your target database, AWS SCT replaces the schema in your project with the latest schema from your target database. If you haven't applied any schema to your

target database, AWS SCT clears the converted schema from your project. You can then convert the schema from your source database for a clean target database.

You update the schema in your AWS SCT project by choosing **Refresh from Database**, as shown following.



Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool

When the AWS Schema Conversion Tool (AWS SCT) generates converted schema (as shown in [Converting Your Schema by Using the AWS Schema Conversion Tool \(p. 94\)](#)), it doesn't immediately apply the converted schema to the target database. Instead, converted schema are stored locally in your project until you are ready to apply them to the target database. Using this functionality, you can work with schema items that can't be converted automatically to your target database engine. For more information on items that can't be converted automatically, see [Creating and Using the Assessment Report in the AWS Schema Conversion Tool \(p. 99\)](#).

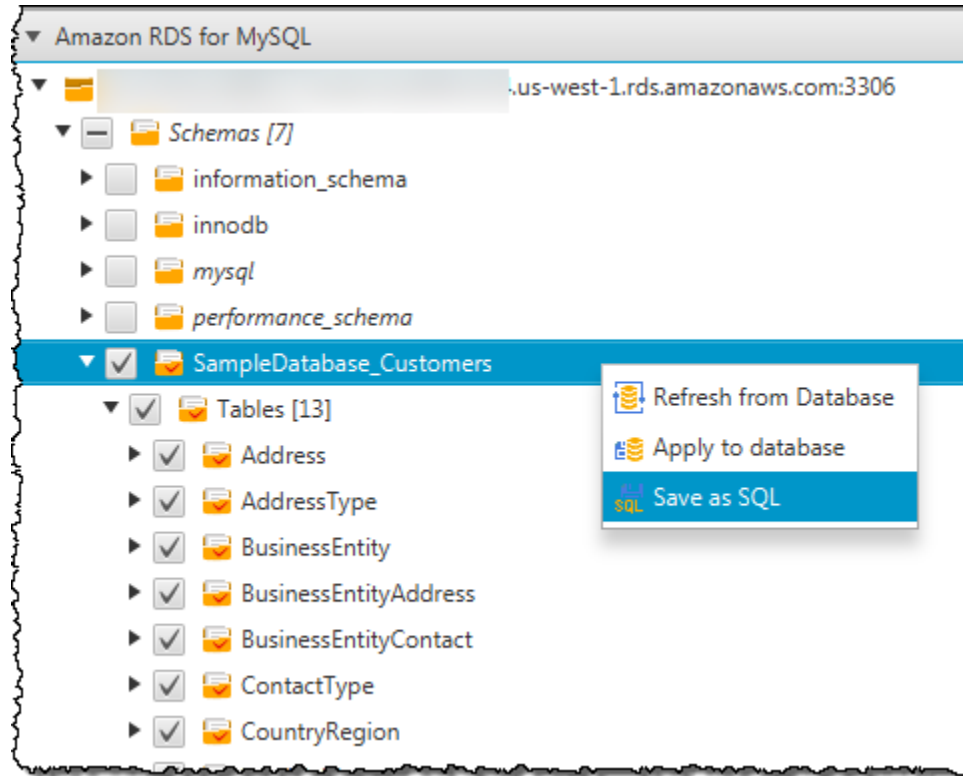
You can optionally have the tool save your converted schema to a file as a SQL script prior to applying the schema to your target database. You can also have the tool apply the converted schema directly to your target database.

Saving Your Converted Schema to a File

You can save your converted schema as SQL scripts in a text file. By using this approach, you can modify the generated SQL scripts from AWS SCT to address items that the tool can't convert automatically.

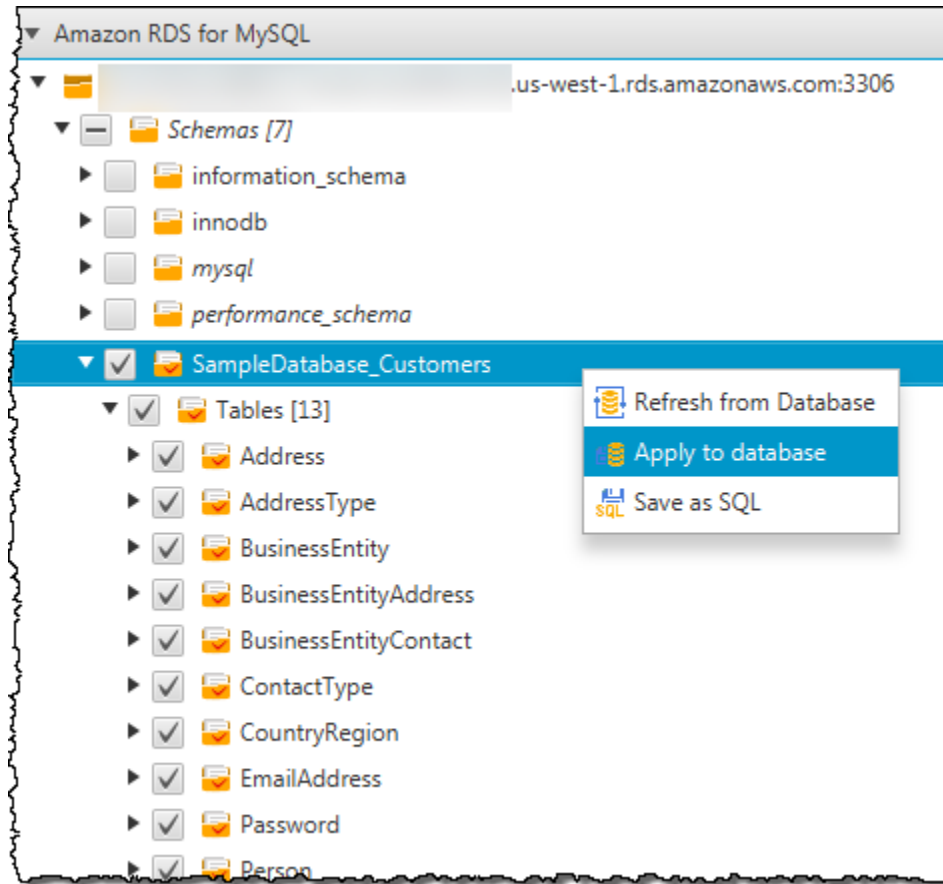
You can then run your updated scripts on your target database to apply your converted schema to your target database.

To save your converted schema as SQL scripts, open the context (right-click) menu for the schema element, and choose **Save as SQL**, as shown following.



Applying Your Converted Schema

When you are ready to apply your converted schema to your target database, choose the schema element from the right panel of your project. Open the context (right-click) menu for the schema element, and then choose **Apply to database**, as shown following.



The Extension Pack Schema

The first time that you apply your converted schema to your target DB instance, AWS SCT adds an additional schema to your target DB instance. This schema implements system functions of the source database that are required when writing your converted schema to your target DB instance. The schema is called the extension pack schema.

Don't modify the extension pack schema, or you might encounter unexpected results in the converted schema that is written to your target DB instance. When your schema is fully migrated to your target DB instance, and you no longer need AWS SCT, you can delete the extension pack schema.

The extension pack schema is named according to your source database as follows:

- Greenplum: `AWS_GREENPLUM_EXT`
- Microsoft SQL Server: `AWS_SQLSERVER_EXT`
- Netezza: `AWS_NETEZZA_EXT`
- Oracle: `AWS_ORACLE_EXT`
- Teradata: `AWS_TERADATA_EXT`
- Vertica: `AWS_VERTICA_EXT`

For more information, see [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses](#) (p. 109).

Python Libraries

To create custom functions in Amazon Redshift, you use the Python language. Use the AWS SCT extension pack to install python libraries for your Amazon Redshift database. For more information, see [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses \(p. 109\)](#).

The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses

When you convert your data warehouse schema, the AWS Schema Conversion Tool (AWS SCT) adds an additional schema to your target DB instance. This schema implements system functions of the source database that are required when writing your converted schema to your target DB instance. The schema is called the extension pack schema.

If you are converting a transactional database, instead see [The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases \(p. 85\)](#).

The extension pack schema is named according to your source database as follows:

- Greenplum: `AWS_GREENPLUM_EXT`
- Microsoft SQL Server: `AWS_SQLSERVER_EXT`
- Netezza: `AWS_NETEZZA_EXT`
- Oracle: `AWS_ORACLE_EXT`
- Teradata: `AWS_TERADATA_EXT`
- Vertica: `AWS_VERTICA_EXT`

In two cases, you might want to install the extension pack manually:

- You accidentally delete the extension pack database schema from your target database.
- You want to upload custom Python libraries to emulate database functionality.

Using AWS Services to Upload Custom Python Libraries

In some cases, source database features can't be converted to equivalent Amazon Redshift features. AWS SCT contains a custom Python library that emulates some source database functionality on Amazon Redshift.

The AWS SCT extension pack wizard helps you install the custom Python library.

Before You Begin

Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

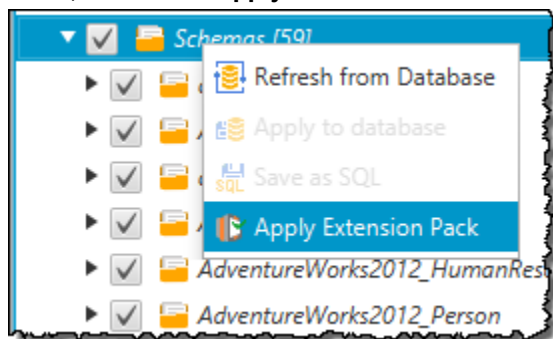
Before you install the extension pack, you need to convert your database schema. For more information, see [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#).

Applying the Extension Pack

Use the following procedure to apply the extension pack.

To apply the extension pack

1. In the AWS Schema Conversion Tool, in the target database tree, open the context (right-click) menu, and choose **Apply Extension Pack**.



The extension pack wizard appears.

2. Read the **Welcome** page, and then choose **Next**.
3. On the **AWS Services Settings** page, do the following:
 - If you are reinstalling the extension pack database schema only, choose **Skip this step for now**, and then choose **Next**.
 - If you are uploading the Python library, provide the credentials to connect to your AWS account. You can use your AWS Command Line Interface (AWS CLI) credentials if you have the AWS CLI installed. You can also use credentials that you previously stored in a profile in the global application settings and associated with the project. If necessary, choose **Navigate to Project Settings** to associate a different profile with the project. If necessary, choose **Global Settings** to create a new profile. For more information, see [Storing AWS Profiles in the AWS Schema Conversion Tool \(p. 135\)](#).
4. On the **Python Library Upload** page, do the following:
 - If you are reinstalling the extension pack database schema only, choose **Skip this step for now**, and then choose **Next**.
 - If you are uploading the Python library, provide the Amazon S3 path, and then choose **Upload Library to S3**. When you are done, choose **Next**.
5. On the **Functions Emulation** page, choose **Create Extension Pack**. Messages appear with the status of the extension pack operations. When you are done, choose **Finish**.

Related Topics

- [Python Language Support for UDFs](#)

Optimizing Amazon Redshift by Using the AWS Schema Conversion Tool

You can use the AWS Schema Conversion Tool (AWS SCT) to optimize your Amazon Redshift database. Using your Amazon Redshift database as a source, and a test Amazon Redshift database as the target, AWS SCT recommends sort keys and distribution keys to optimize your database.

Before You Begin

Almost all work you do with AWS SCT starts with the same three steps. Complete the following steps before you optimize your Amazon Redshift database:

1. Create an AWS SCT project. For more information, see [Creating an AWS Schema Conversion Tool Project \(p. 13\)](#).
2. Connect to your source database. For more information, see [Connecting to an Amazon Redshift Source Database \(p. 43\)](#).
3. Connect to your target database. For more information, see [Connecting to Your Target Database \(p. 14\)](#).

Important

Don't use the same cluster for both the source and target of your optimization.

Before you optimize your Amazon Redshift database, you should also choose your optimization strategies. For more information, see [Choosing Optimization Strategies and Rules for Use with the AWS Schema Conversion Tool \(p. 90\)](#).

Optimizing Your Amazon Redshift Database

Use the following procedure to optimize your Amazon Redshift database.

To optimize your Amazon Redshift database

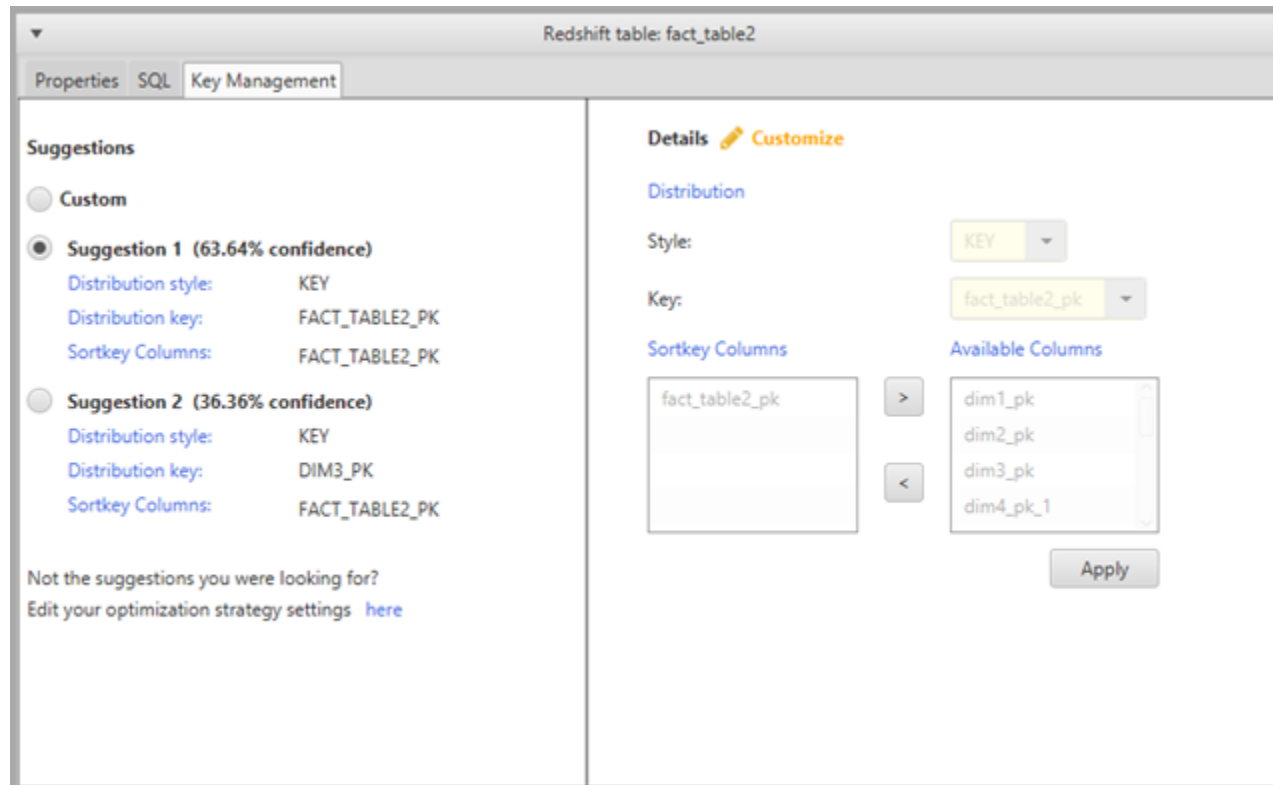
1. Take a manual snapshot of your Amazon Redshift cluster as a backup. You can delete the snapshot after you are done optimizing your Amazon Redshift cluster and testing any changes that you make. For more information, see [Amazon Redshift Snapshots](#).
2. Choose a schema object to convert from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Collect Statistics**.

AWS SCT uses the statistics to make suggestions for sort keys and distribution keys.

3. Choose a schema object to optimize from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Run Optimization**.

AWS SCT makes suggestions for sort keys and distribution keys.

4. To review the suggestions, expand the tables node under your schema in the left panel of your project, and then choose a table. Choose the **Key Management** tab as shown following.



The left pane contains key suggestions, and includes the confidence rating for each suggestion. You can choose one of the suggestions, or you can customize the key by editing it in the right pane.

5. You can create a report that contains the optimization suggestions. To create the report, do the following:
 - a. Choose a schema object that you optimized from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Create Report**.

The report opens in the main window, and the **Summary** tab appears. The number of objects with optimization suggestions appears in the report.
 - b. Choose the **Action Items** tab to see the key suggestions in a report format.
 - c. You can save a local copy of the optimization report as either a PDF file or a comma-separated values (CSV) file. The CSV file contains only action item information. The PDF file contains both the summary and action item information.
6. To apply suggested optimizations to your database, choose an object in the right panel of your project. Open the context (right-click) menu for the object, and then choose **Apply to database**.

Related Topics

- [Choose the Best Sort Key](#)
- [Choose the Best Distribution Style](#)
- [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Working with the AWS Database Migration Service Using the AWS Schema Conversion Tool

AWS Database Migration Service (AWS DMS) is a web service that you can use to migrate data to and from most widely used commercial and open-source databases. You can use the AWS Schema Conversion Tool (AWS SCT) to create AWS DMS endpoints and tasks. You can run and monitor the tasks from AWS SCT. For more information about AWS DMS, see [What Is AWS Database Migration Service?](#) in the *AWS DMS User Guide*.

Before You Begin

Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool](#) (p. 12).

Because AWS DMS interacts with the target schema, you need to convert your database schema before you can integrate with AWS DMS. To convert your database schema, see [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool](#) (p. 69).

Credentials for Working with AWS DMS

To create AWS DMS tasks, AWS SCT must connect to AWS DMS with your credentials. You can use credentials that you previously stored in a profile in the global application settings and associated with the project. For more information, see [Storing AWS Profiles in the AWS Schema Conversion Tool](#) (p. 135).

Creating an AWS DMS Task

After you use AWS SCT to convert your database schema, you can create AWS DMS tasks. Use the following procedure to create an AWS DMS task.

To create an AWS DMS task

1. In the AWS Schema Conversion Tool, choose a database or a schema object from the left panel of your project. Open the context (right-click) menu for the object, and then choose **Create DMS Task**.

The **Create DMS task** dialog box appears.

2. For **Task name**, type a name for your task.
3. For **Replication instance**, choose the replication instance that you want to use. For more information, see [Replication Instances for AWS Database Migration Service](#).
4. For **Source endpoint**, choose an existing endpoint. You can also choose **Create new** to create a new endpoint. When you choose **Create new**, a new endpoint is created for you by using the current connection information. You can verify the connection information and give the endpoint a name before it is created.
5. For **Target endpoint**, choose an existing endpoint. You can also choose **Create new** to create a new endpoint. When you choose **Create new**, a new endpoint is created for you by using the current connection information. You can verify the connection information and give the endpoint a name before it is created.
6. For **Include LOB columns in replication**, choose **Don't include LOB columns**, **Full LOB mode**, or **Limited LOB mode**. For more information, see [LOB Support for Source Databases](#).
7. For **LOB chunk size (kb)**, type the LOB chunk size in kilobytes.
8. Select **Enable logging** to turn on logging for the task.
9. Choose **Preview JSON** to see the JSON that is created for the task.
10. Choose **Save JSON** to save a copy of the JSON that is created for the task.
11. After you have configured your task, choose **Create** to save your task. You can also choose **Close** to cancel your changes.

Running and Monitoring an AWS DMS Task

After you create AWS DMS tasks, you can run and monitor them in the data migration view. Use the following procedure to run and monitor your AWS DMS tasks.

To run and monitor your AWS DMS tasks

1. Open the **View** menu, and then choose **Data Migration View**.
2. In the list of tasks, choose the task you want to run. Choose **Start** to run the task. The **Status** column shows the status of the task.
3. When you have a running task, you can choose the following actions:
 - Choose **Stop** to stop a task.
 - Choose **Resume** to resume a task.
4. Choose **Delete** to delete a task.
5. Choose **Refresh** to refresh the task list. The **Status** column shows the current status of all tasks.
6. Choose **Show Log** to see the task log. If you selected **Enable logging** when you created the task, the task log shows log information.

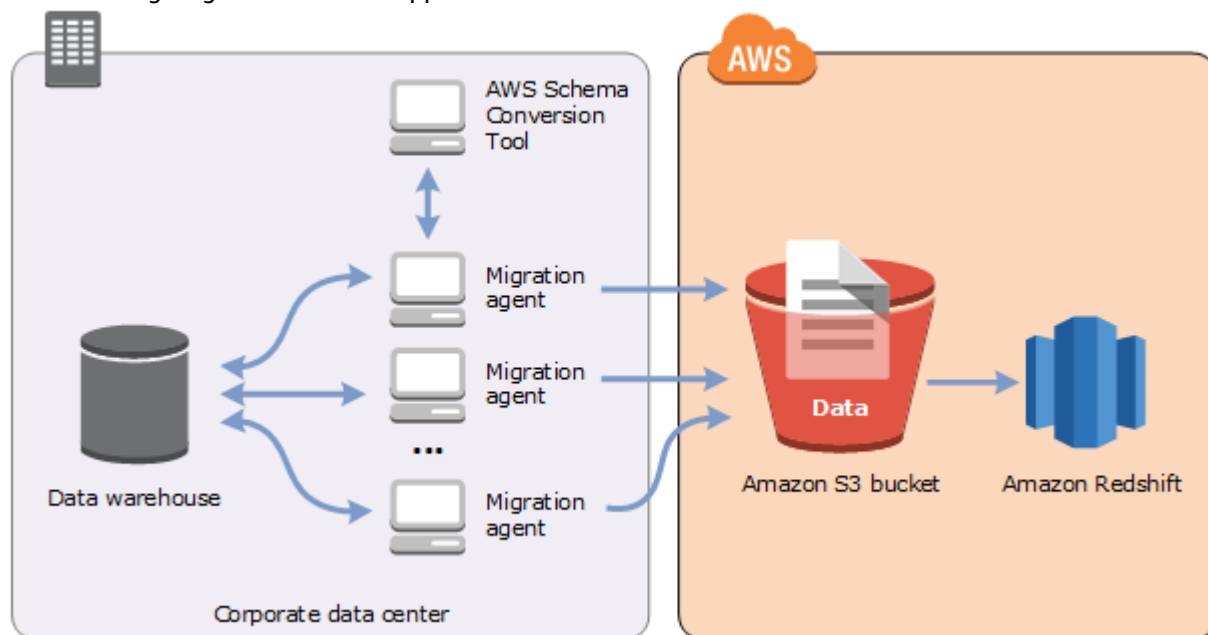
Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Working with AWS Database Migration Service Replication Tasks](#)
- [Monitoring AWS Database Migration Service Tasks](#)

Using Data Extraction Agents

You can use data extraction agents to extract data from your on-premises data warehouse and migrate it to Amazon Redshift. To manage the data extraction agents, you can use AWS SCT. Data extraction agents can work in the background while AWS SCT is closed. After your agents extract your data, they upload the data to Amazon S3 and then copy the data into Amazon Redshift.

The following diagram shows the supported scenario.



Data extraction agents are currently supported for the following source data warehouses:

- Greenplum Database (version 4.3 and later)
- Microsoft SQL Server (version 2008 and later)
- Netezza (version 7.0.3 and later)
- Oracle (version 10 and later)
- Teradata (version 13 and later)
- Vertica (version 7.2.2 and later)

You can connect to FIPS endpoints for Amazon Redshift if you need to comply with the Federal Information Processing Standard security requirements. FIPS endpoints are available in the following AWS Regions:

- US East (N. Virginia) Region (redshift-fips.us-east-1.amazonaws.com)
- US East (Ohio) Region (redshift-fips.us-east-2.amazonaws.com)
- US West (N. California) Region (redshift-fips.us-west-1.amazonaws.com)
- US West (Oregon) Region (redshift-fips.us-west-2.amazonaws.com)

Use the information in the following topics to learn how to work with data extraction agents.

Topics

- [Prerequisite Settings for Amazon S3 and Security for Data Extraction Agents \(p. 116\)](#)
- [Installing, Configuring, and Starting Data Extraction Agents \(p. 117\)](#)
- [Registering Extraction Agents with the AWS Schema Conversion Tool \(p. 121\)](#)
- [Creating Data Extraction Filters in the AWS Schema Conversion Tool \(p. 122\)](#)
- [Managing Data Extraction Tasks with the AWS Schema Conversion Tool \(p. 123\)](#)
- [Data Extraction Task Output \(p. 125\)](#)
- [Best Practices and Troubleshooting for Data Extraction Agents \(p. 126\)](#)

Prerequisite Settings for Amazon S3 and Security for Data Extraction Agents

Before you work with data extraction agents, store your Amazon S3 bucket information and set up your Secure Sockets Layer (SSL) trust and key store.

Amazon S3 Settings

After your agents extract your data, they upload it to your Amazon S3 bucket. Before you continue, you must provide the credentials to connect to your AWS account and your Amazon S3 bucket. You store your credentials and bucket information in a profile in the global application settings, and then associate the profile with your AWS SCT project. If necessary, choose **Global Settings** to create a new profile. For more information, see [Storing AWS Profiles in the AWS Schema Conversion Tool \(p. 135\)](#).

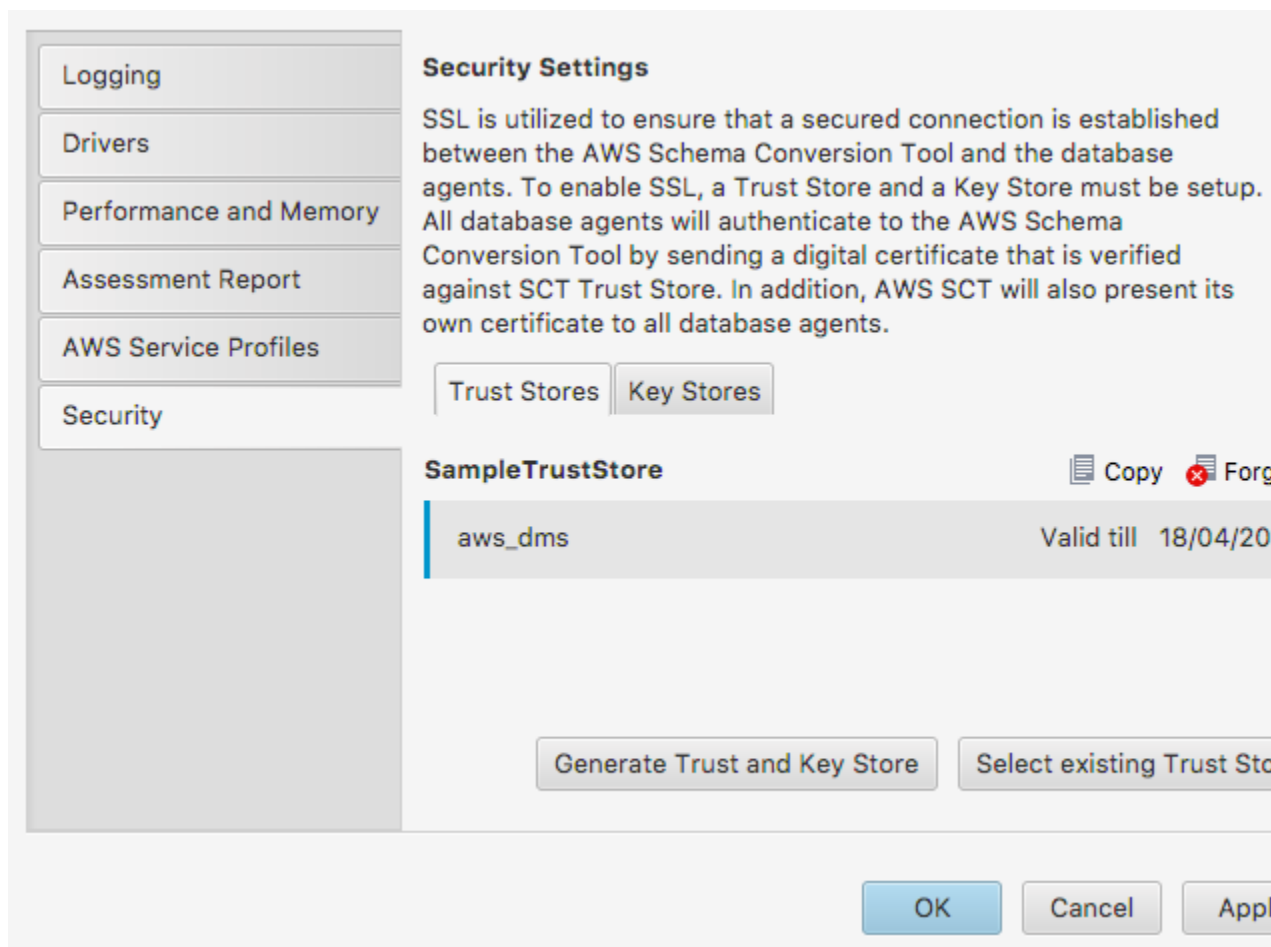
Security Settings

The AWS Schema Conversion Tool and the extraction agents can communicate through Secure Sockets Layer (SSL). To enable SSL, set up a trust store and key store.

To set up secure communication with your extraction agent

1. Start the AWS Schema Conversion Tool.
2. Open the **Settings** menu, and then choose **Global Settings**. The **Global settings** dialog box appears.

Choose the **Security** tab as shown following.



3. Choose **Generate Trust and Key Store**, or choose **Select existing Trust and Key Store**.

If you choose **Generate Trust and Key Store**, you then specify the name and password for the trust and key stores, and the path to the location for the generated files. You use these files in later steps.

If you choose **Select existing Trust and Key Store**, you then specify the password and file name for the trust and key stores. You use these files in later steps.

4. After you have specified the trust store and key store, choose **OK** to close the **Global Settings** dialog box.

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Installing, Configuring, and Starting Data Extraction Agents \(p. 117\)](#)

Installing, Configuring, and Starting Data Extraction Agents

Use the information in the following sections to install, configure, and start your data extraction agents.

Installing Extraction Agents

We recommend that you install multiple extraction agents on individual computers, separate from the computer that is running the AWS Schema Conversion Tool.

Extraction agents are currently supported on the following operating systems:

- macOS
- Microsoft Windows
- Red Hat Enterprise Linux (RHEL) 6.0
- Ubuntu Linux (version 14.04 and later)

Use the following procedure to install extraction agents. Repeat this procedure for each computer that you want to install an extraction agent on.

To install an extraction agent

1. If you have not already downloaded the AWS SCT installer file, follow the instructions at [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#) to download it. The .zip file that contains the AWS SCT installer file also contains the extraction agent installer file.
2. Locate the installer file for your extraction agent in a subfolder named agents. For each computer operating system, the correct file to install the extraction agent is shown following.

Operating System	File Name
macOS	aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .dmg
Microsoft Windows	aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .msi
RHEL	aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .x86_64.rpm
Ubuntu Linux	aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .deb

3. To install the extraction agent on a separate computer, copy the installer file to the new computer.
4. Run the installer file. Use the instructions for your operating system, shown following.

Operating System	Install Instructions
macOS	In Finder , open aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .dmg. Drag aws-schema-conversion-tool-extractor-1.0. <i>build-number</i> .dmg to the Applications folder.
Microsoft Windows	Double-click the file to run the installer.
RHEL	Run the following command in the folder that you downloaded or moved the file to: <pre>sudo rpm -ivh aws-schema-conversion-tool-extractor-1.0.<i>build-number</i>.x86_64.rpm</pre>

Operating System	Install Instructions
Ubuntu Linux	<p>Run the following command in the folder that you downloaded or moved the file to:</p> <pre>sudo dpkg -i aws-schema-conversion-tool-extractor-1.0.<i>build-number</i>.deb</pre>

5. Install the Java Database Connectivity (JDBC) drivers for your source database engine. For instructions and download links, see [Installing the Required Database Drivers \(p. 8\)](#). Follow the instructions for your source database engine only, not your target database engine.
6. Copy the SSL trust and key stores (.zip or individual files) that you generated in an earlier procedure. If you copy the .zip file to a new computer, extract the individual files from the .zip file on the new computer.

You can put the files anywhere you want. However, note the locations because in a later procedure you tell the agent where to find the files.

Continue installing your extraction agent by completing the procedure in the following section.

Configuring Extraction Agents

Use the following procedure to configure extraction agents. Repeat this procedure on each computer that has an extraction agent installed.

To configure your extraction agent

- From the location where you installed the agent, run the setup program. For RHEL and Ubuntu, the file is named `sct-extractor-setup.sh`. For macOS and Microsoft Windows, the file is named `AWS SCT Data Extractor Agent`, and you can double-click the file to run it.

The setup program prompts you for information. For each prompt, a default value appears. You can accept the default value, or type a new value. You specify the following information:

- The data warehouse engine.
- The port number the agent listens on.
- The location where you installed the JDBC drivers.
- The working folder. Your extracted data goes into a subfolder of this location. The working folder can be on a different computer from the agent, and a single working folder can be shared by multiple agents on different computers.
- The location of the key store file.
- The password for the key store.
- The location of the trust store file.
- The password for the trust store.

The setup program updates the settings file for the extraction agent. The settings file is named `Settings.properties`, and is located where you installed the extraction agent. The following is a sample settings file.

```
port=8888
vendor=ORACLE
driver.jars=<driver path>/Install/Drivers/ojdbc7.jar
location=<output path>/dmt/8888/out
extractor.log.folder=<log path>/dmt/8888/log
```

```
extractor.storage.folder=<storage path>/dmt/8888/storage
extractor.start.fetch.size=20000
extractor.out.file.size=10485760
ssl.option=OFF
#ssl.option=ON
#ssl.keystore.path=<key store path>/dmt/8888/vault/keystore
#ssl.truststore.path=<trust store path>/dmt/8888/vault/truststore
```

Starting Extraction Agents

Use the following procedure to start extraction agents. Repeat this procedure on each computer that has an extraction agent installed.

Extraction agents act as listeners. When you start an agent with this procedure, the agent starts listening for instructions. You send the agents instructions to extract data from your data warehouse in a later section.

To start your extraction agent

- On the computer that has the extraction agent installed, run the command listed following for your operating system.

Operating System	Start Command
macOS	Run the <code>StartAgent.command</code> file.
Microsoft Windows	Double-click the <code>StartAgent.bat</code> batch file.
RHEL	Run the following command in the path to the folder that you installed the agent: <code>sudo initctl start sct-extractor</code>
Ubuntu Linux	Run the following command in the path to the folder that you installed the agent. Use the command appropriate for your version of Ubuntu. Ubuntu 14.04: <code>sudo initctl start sct-extractor</code> Ubuntu 15.04 and later: <code>sudo systemctl start sct-extractor</code>

To check the status of the agent, run the same command but replace `start` with `status`.

To stop an agent, run the same command but replace `start` with `stop`.

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Prerequisite Settings for Amazon S3 and Security for Data Extraction Agents \(p. 116\)](#)
- [Registering Extraction Agents with the AWS Schema Conversion Tool \(p. 121\)](#)
- [Best Practices and Troubleshooting for Data Extraction Agents \(p. 126\)](#)

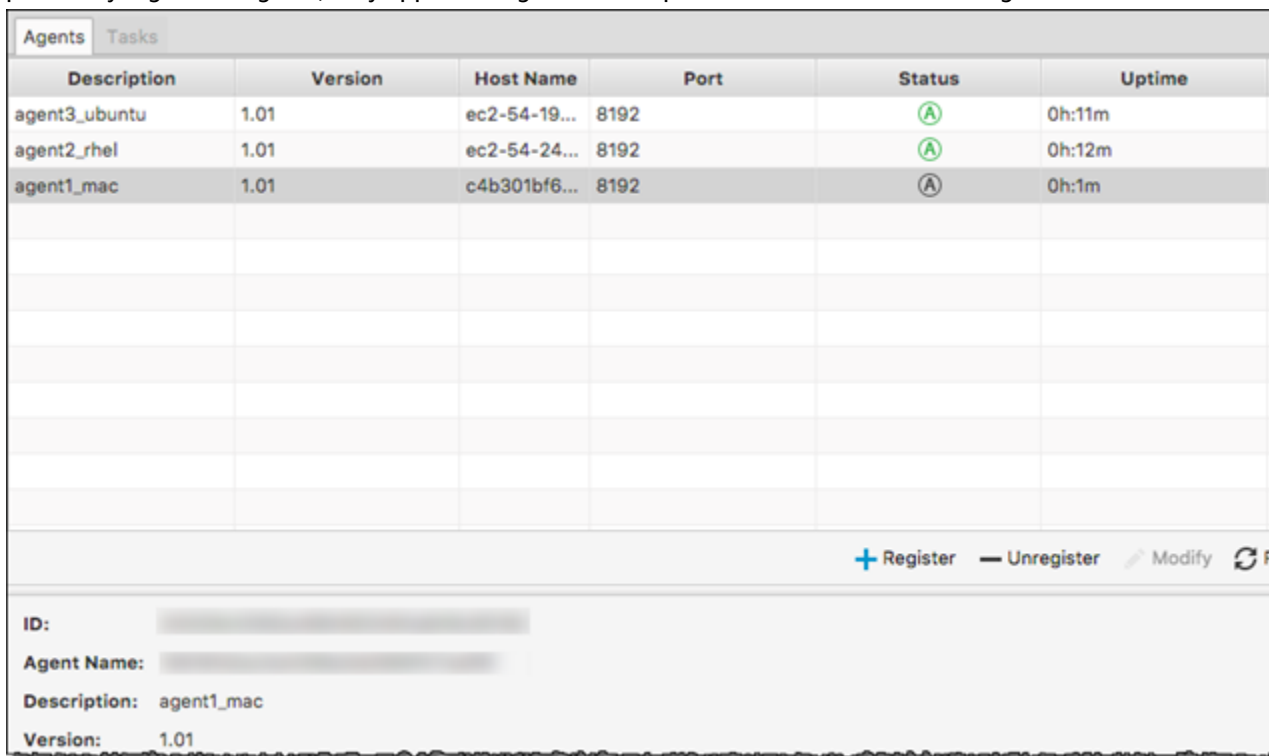
Registering Extraction Agents with the AWS Schema Conversion Tool

You manage your extraction agents by using AWS SCT. The extraction agents act as listeners. When they receive instructions from AWS SCT, they extract data from your data warehouse.

Use the following procedure to register extraction agents with your AWS SCT project.

To register an extraction agent

1. Start the AWS Schema Conversion Tool, and open a project.
 - a. Open an existing or create a new project. For more information, see [Creating an AWS Schema Conversion Tool Project \(p. 13\)](#).
 - b. Connect to your source database. For more information, see [Connecting to Your Source Database \(p. 14\)](#).
 - c. Connect to your target database. For more information, see [Connecting to Your Target Database \(p. 14\)](#).
 - d. Convert your schema. For more information, see [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#).
 - e. Apply your schema. For more information, see [Saving and Applying Your Converted Schema in the AWS Schema Conversion Tool \(p. 105\)](#).
2. Open the **View** menu, and then choose **Data Migration View**. The **Agents** tab appears. If you have previously registered agents, they appear in a grid at the top of the tab as shown following.



Description	Version	Host Name	Port	Status	Uptime
agent3_ubuntu	1.01	ec2-54-19...	8192	Ⓐ	0h:11m
agent2_rhel	1.01	ec2-54-24...	8192	Ⓐ	0h:12m
agent1_mac	1.01	c4b301bf6...	8192	Ⓐ	0h:1m

+ Register - Unregister / Modify ↻

ID:

Agent Name:

Description: agent1_mac

Version: 1.01

3. Choose **Register**. The **New Agent Registration** dialog box appears.

Note

After you register an agent with an AWS SCT project, you can't register the same agent with a different project. If you're no longer using an agent in an AWS SCT project, you can unregister it. You can then register it with a different project.

4. Enter your information in the **New Agent Registration** dialog box:
 - a. For **Description**, type a description of the agent.
 - b. For **Host Name**, type the host name or IP address of the computer of the agent.
 - c. For **Port**, type the port number that the agent is listening on.
 - d. Choose **Register** to register the agent with your AWS SCT project.
5. Repeat the previous steps to register multiple agents with your AWS SCT project.

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Installing, Configuring, and Starting Data Extraction Agents \(p. 117\)](#)
- [Creating Data Extraction Filters in the AWS Schema Conversion Tool \(p. 122\)](#)

Creating Data Extraction Filters in the AWS Schema Conversion Tool

Before you extract your data with the AWS Schema Conversion Tool (AWS SCT), you can set up filters that reduce the amount of data that you extract. You can create data extraction filters by using `where` clauses to reduce the data that you extract. For example, you can write a `where` clause that selects data from a single table.

You can create data extraction filters and save the filters as part of your project. With your project open, use the following procedure to create data extraction filters.

To create data extraction filters

1. On the **Settings** menu, choose **Mapping Rules**. The **Mapping Rules** dialog box appears. The top pane contains transformation rules, and the bottom pane contains filtering rules.
2. In the **Filtering Rules** pane, choose **Add new rule**.
3. Configure your filter.
 - a. For **Name**, type a name for your filter.
 - b. For **Where schema name like**, type a filter to apply to schemas. The `where` clause is evaluated by using a `like` clause. You can enter an exact name to select one schema, or you can enter a pattern to select multiple schemas.
 - c. For **table name like**, type a filter to apply to tables. The `where` clause is evaluated by using a `like` clause. You can enter an exact name to select one table, or you can enter a pattern to select multiple tables.
 - d. For **Where clause**, type a `where` clause to filter data.
4. After you have configured your filter, choose **Save** to save your filter. You can also choose **Cancel** to cancel your changes.
5. After you are done adding, editing, and deleting filters, choose **Save All** to save all your changes.
6. Choose **Close** to close the **Mapping Rules** dialog box.

You can use the toggle icon to turn off a filter without deleting it. You can use the copy icon to duplicate an existing filter. You can use the delete icon to delete an existing filter. To save any changes you make to your filters, choose **Save All**.

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Registering Extraction Agents with the AWS Schema Conversion Tool \(p. 121\)](#)
- [Managing Data Extraction Tasks with the AWS Schema Conversion Tool \(p. 123\)](#)

Managing Data Extraction Tasks with the AWS Schema Conversion Tool

Use the following procedures to create, run, and monitor data extraction tasks.

To assign tasks to agents and migrate data

1. In the AWS Schema Conversion Tool, after you have converted your schema, choose one or more tables from the left panel of your project. You can choose all tables, but we recommend against that for performance reasons. We recommend that you create multiple tasks for multiple tables based on the size of the tables in your data warehouse.
2. Open the context (right-click) menu for the tables, and then choose **Create Task**. The **Create Local Task** dialog box opens, as shown following.

The screenshot shows the 'Create Local Task' dialog box. It has a title bar with a yellow icon and the text 'Create Local Task'. Below the title bar are two tabs: 'General' and 'Advanced'. The 'General' tab is active. It contains the following fields and controls:

- Task Name:** A text input field containing 'NewTask'.
- Migration Mode:** A dropdown menu with 'Extract Only' selected.
- Extract LOBs:** A checked checkbox.
- Logging:** A section header.
- Enable Task Logging:** A checked checkbox.
- Logging Level:** A dropdown menu with 'TRACE' selected.

At the bottom of the dialog are three buttons: 'Test Task', 'Cancel', and 'Create'.

3. For **Task Name**, type a name for the task.
4. For **Migration Mode**, choose one of the following:
 - **Extract Only** – Extract your data, and save the data to your local working folders.
 - **Extract and Upload** – Extract your data, and upload your data to Amazon S3.
 - **Extract, Upload and Copy** – Extract your data, upload your data to Amazon S3, and copy it into your Amazon Redshift data warehouse.

5. Select **Extract LOBs** to extract large objects. If you don't need to extract large objects, you can clear the check box. This reduces the amount of data that you extract.
6. If you want to see detailed information about a task, select **Enable Task Logging**. You can use the task log to debug problems.

If you enable task logging, choose the level of detail that you want to see. The levels are the following, with each level including all messages from the previous level:

- **ERROR** – The smallest amount of detail.
 - **WARNING**
 - **INFO**
 - **DEBUG**
 - **TRACE** – The largest amount of detail.
7. Choose **Test Task** to verify that you can connect to your working folder, Amazon S3 bucket, and Amazon Redshift data warehouse. The verification depends on the migration mode you chose.
 8. Choose **Create** to create the task.
 9. Repeat the previous steps to create tasks for all the data that you want to migrate.

To run and monitor tasks

1. Open the **View** menu, and then choose **Data Migration View**. The **Agents** tab appears.
2. Choose the **Tasks** tab. Your tasks appear in the grid at the top as shown following.

The screenshot shows the 'Tasks' tab in the AWS Schema Conversion Tool. The top grid displays tasks with columns for Task Name, Status, Complete %, and a progress bar. Below the grid is a 'Task details' section with a 'Subtasks' tab. The subtasks table has columns for ID, Parent Agent, Source Name, Target Name, and Status.

Task Name	Status	Complete %	Time
Task1_PersonTables			
Task1_PersonTables(Local)	▶	60%	0h:00m:29s
Task1_PersonTables(S3)	▶	50%	0h:00m:29s
Task2_PlayerTables			
Task2_PlayerTables(Local)	✔	33%	0h:00m:13s
Task2_PlayerTables(S3)	▶	0%	0h:00m:13s

ID	Parent Agent	Source Name	Target Name	Status
7cae6776aef74db28bd67b...	agent3_ubuntu	DMS_SAMPLE.PERSON	dms_sample.person	RUNNING
38dce72c914149ff8764be...	agent2_rhel	DMS_SAMPLE.DATA	dms_sample.data	COMPLETED

3. Select a task in the top grid and expand it. Depending on the migration mode you chose, you see the task divided into **Extract**, **Upload**, and **Copy**.
4. Select a task in the top grid. You can see the status of the task in the top grid, and the status of its subtasks in the bottom grid.
5. To run a task, choose **Start**. You can monitor the status of your tasks while they work. The subtasks run in parallel. The extract, upload, and copy also run in parallel.

6. If you enabled logging when you set up the task, you can view the log.
 - a. Choose **Download Log**. A message appears with the name of the folder that contains the log file. Dismiss the message.
 - b. A link appears in the **Task details** tab. Choose the link to open the folder that contains the log file.

You can close AWS SCT, and your agents and tasks continue to run. You can reopen AWS SCT later to check the status of your tasks and view the task logs.

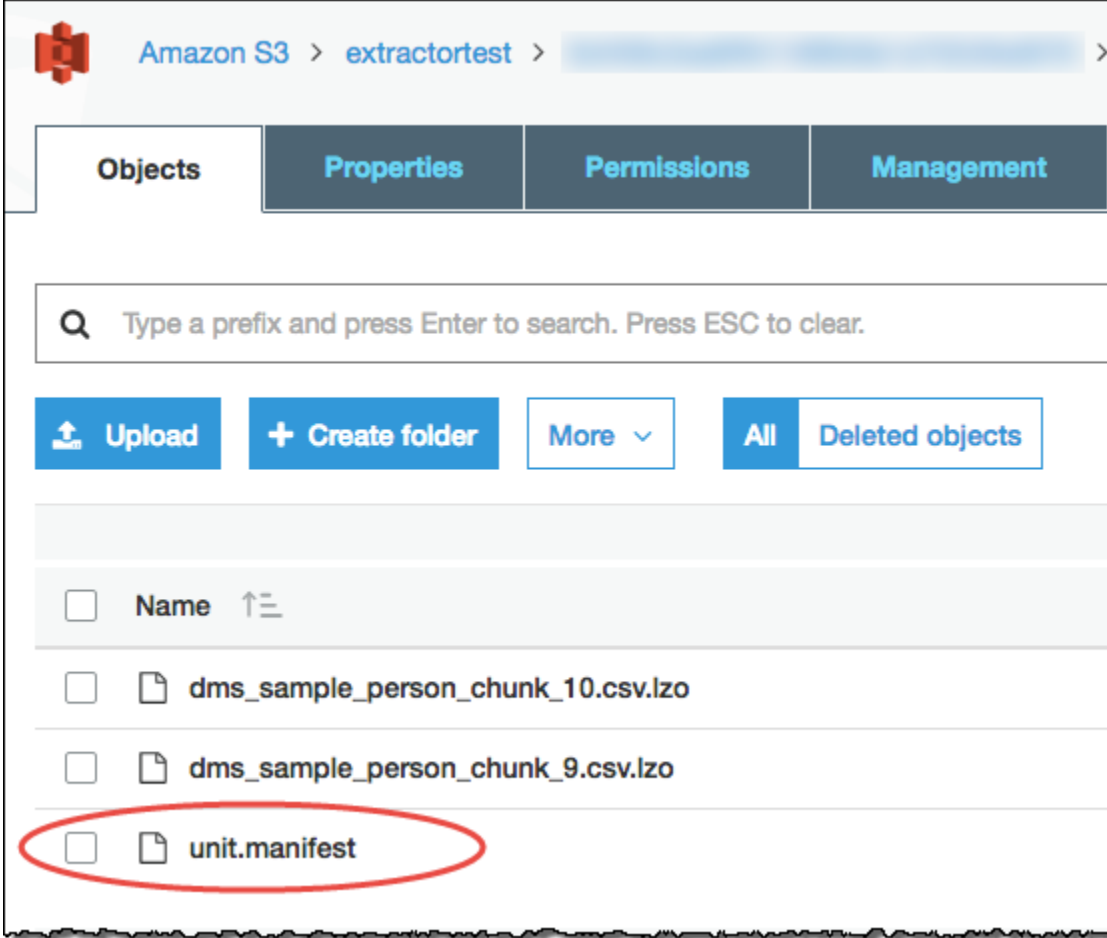
Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Creating Data Extraction Filters in the AWS Schema Conversion Tool \(p. 122\)](#)
- [Data Extraction Task Output \(p. 125\)](#)
- [Best Practices and Troubleshooting for Data Extraction Agents \(p. 126\)](#)

Data Extraction Task Output

After your migration tasks complete, your data is ready. Use the following information to determine how to proceed based on the migration mode you chose and the location of your data.

Migration Mode	Data Location
Extract, Upload and Copy	The data is already in your Amazon Redshift data warehouse. You can verify that the data is there, and start using it. For more information, see Connecting to Clusters From Client Tools and Code .
Extract and Upload	<p>The extraction agents saved your data as files in your Amazon S3 bucket. You can use the Amazon Redshift COPY command to load your data to Amazon Redshift. For more information, see Loading Data from Amazon S3 in the Amazon Redshift documentation.</p> <p>There are multiple folders in your Amazon S3 bucket, corresponding to the extraction tasks that you set up. When you load your data to Amazon Redshift, specify the name of the manifest file created by each task. The manifest file appears in the task folder in your S3 bucket as shown following.</p>

Migration Mode	Data Location
	
Extract Only	The extraction agents saved your data as files in your working folder. Manually copy your data to your Amazon S3 bucket, and then proceed with the instructions for Extract and Upload .

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Managing Data Extraction Tasks with the AWS Schema Conversion Tool \(p. 123\)](#)

Best Practices and Troubleshooting for Data Extraction Agents

The following are some best practices and troubleshooting suggestions for using extraction agents.

Issue	Troubleshooting Suggestions
Performance is slow	To improve performance, we recommend the following: <ul style="list-style-type: none">• Install multiple agents.• Install agents on computers close to your data warehouse.• Don't run all tables on a single agent task.
Contention delays	Avoid having too many agents accessing your data warehouse at the same time.
An agent goes down temporarily	If an agent is down, the status of each of its tasks appears as failed in AWS SCT. If you wait, in some cases the agent can recover. In this case, the status of its tasks updates in AWS SCT.
An agent goes down permanently	If the computer running an agent goes down permanently, and that agent is running a task, you can substitute a new agent to continue the task. You can substitute a new agent only if the working folder of the original agent was not on the same computer as the original agent. To substitute a new agent, do the following: <ul style="list-style-type: none">• Install an agent on a new computer.• Configure the new agent with the same settings, including port number and working folder, as the original agent.• Start the agent. After the agent starts, the task discovers the new available agent and continues running on the new agent.

Related Topics

- [Using Data Extraction Agents \(p. 115\)](#)
- [Installing, Configuring, and Starting Data Extraction Agents \(p. 117\)](#)
- [Managing Data Extraction Tasks with the AWS Schema Conversion Tool \(p. 123\)](#)

Converting Application SQL by Using the AWS Schema Conversion Tool

When you convert your database schema from one engine to another, you also need to update the SQL code in your applications to interact with the new database engine instead of the old one. You can use the AWS Schema Conversion Tool (AWS SCT) to convert the SQL code in your C++, C#, Java, or other application code. You can view, analyze, edit, and save the converted SQL code.

Before You Begin

Almost all work you do with AWS SCT starts with the following three steps:

1. Create an AWS SCT project.
2. Connect to your source database.
3. Connect to your target database.

If you have not created an AWS SCT project yet, see [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#).

Because the SQL code in your application interacts with your database schema, you need to convert your database schema before you can convert your application SQL. To convert your database schema, use the procedures in one of the following topics:

- [Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool \(p. 69\)](#)
- [Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool \(p. 88\)](#)

Overview of Converting Application SQL

To convert the SQL code in your application, you take the following high-level steps:

- **Create an application conversion project** – The application conversion project is a child of the database schema conversion project. Each database schema conversion project can have one or more child application conversion projects.

For more information, see [Creating Application Conversion Projects in the AWS Schema Conversion Tool \(p. 129\)](#).

- **Analyze and convert your SQL code** – AWS SCT analyzes your application, extracts the SQL code, and creates a local version of the converted SQL for you to review and edit. The tool doesn't change the code in your application until you are ready.

For more information, see [Analyzing and Converting Your SQL Code by Using the AWS Schema Conversion Tool \(p. 132\)](#).

- **Create an application assessment report** – The application assessment report provides important information about the conversion of the application SQL code from your source database schema to your target database schema.

For more information, see [Creating and Using the Assessment Report \(p. 133\)](#).

- **Edit, apply changes to, and save your converted SQL code** – The assessment report includes a list of SQL code items that can't be converted automatically. For these items, you can edit the SQL code manually to perform the conversion.

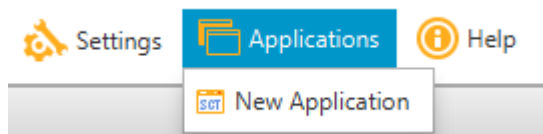
For more information, see [Editing and Saving Your Converted SQL Code with the AWS Schema Conversion Tool \(p. 134\)](#).

Creating Application Conversion Projects in the AWS Schema Conversion Tool

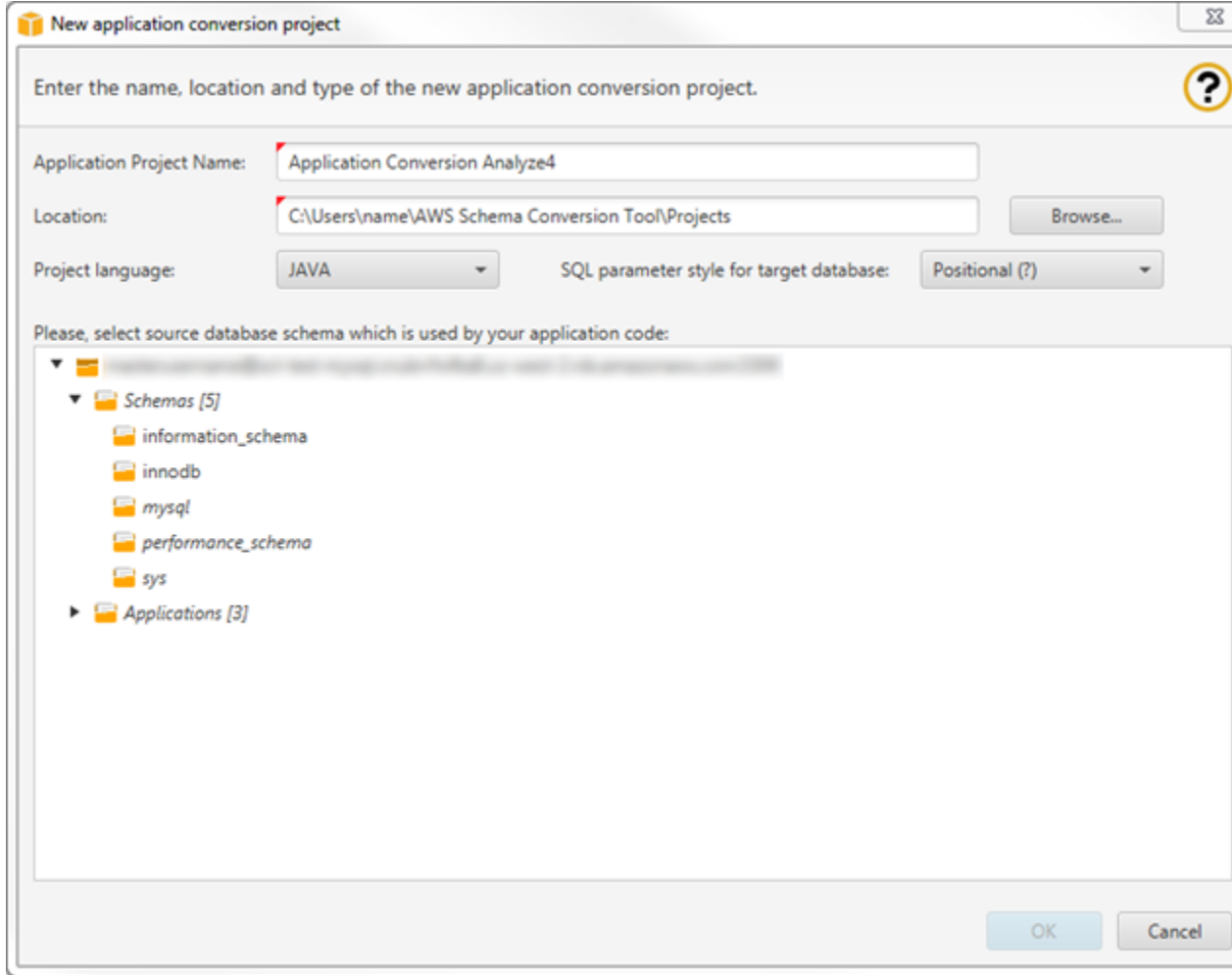
In the AWS Schema Conversion Tool (AWS SCT), the application conversion project is a child of the database schema conversion project. Each database schema conversion project can have one or more child application conversion projects. Use the following procedure to create an application conversion project.

To create an application conversion project

1. In the AWS Schema Conversion Tool, choose **New Application** from the **Applications** menu.



The **New application conversion project** dialog box appears.



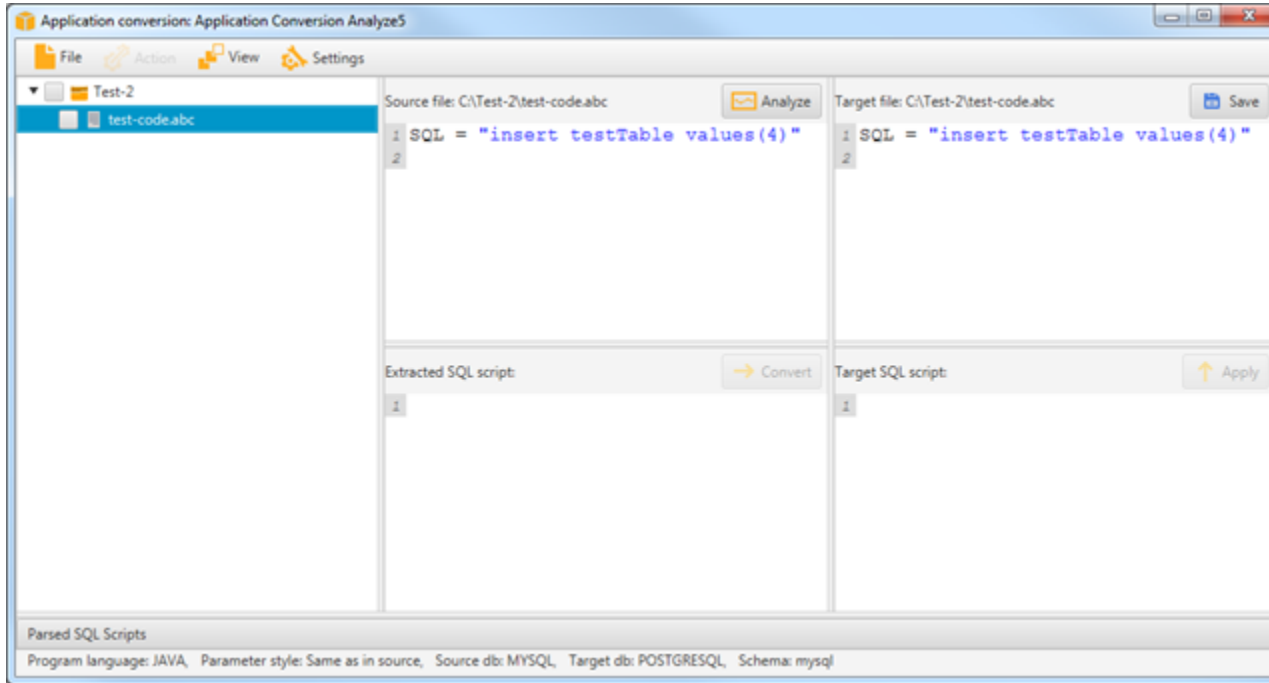
2. Add the following project information.

For This Parameter	Do This
Application Project Name	Type a name for your application conversion project. Each database schema conversion project can have one or more child application conversion projects, so choose a name that makes sense if you add more projects later.
Location	Type the location of the source code for your application.
Project language	Choose one of the following: <ul style="list-style-type: none"> • JAVA • C++ • C# • Any
SQL parameter style for target database	Choose one of the following: <ul style="list-style-type: none"> • Same as in source • Positional (?)

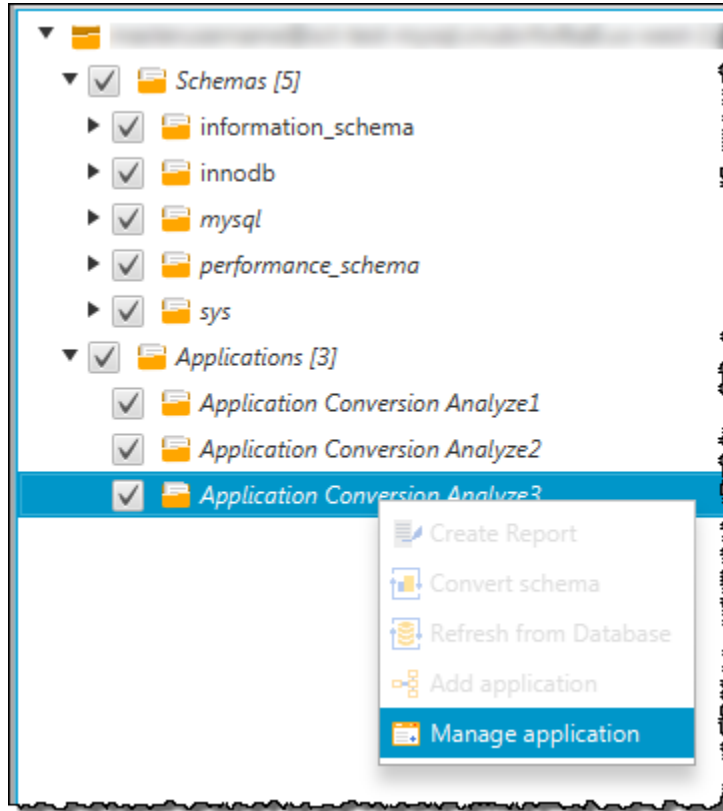
For This Parameter	Do This
	<ul style="list-style-type: none">• Indexed (:1)• Indexed (\$1)• Named (@name)• Named (:name)
Select source database schema	In the source tree, choose the schema used by your application code.

3. Choose **OK** to create your application conversion project.

The project window opens.



4. The first time you create an application conversion project, the project window opens automatically. To open an existing application conversion project, select the project node in the source tree, open the context (right-click) menu, and then choose **Manage application**.



5. You can add additional application conversion projects by choosing **New Application** from the **Applications** menu, or by selecting the **Applications** node in the source tree, opening the context (right-click) menu, and then choosing **Add application**.

Analyzing and Converting Your SQL Code by Using the AWS Schema Conversion Tool

Use the following procedure to analyze and convert your SQL code by using the AWS Schema Conversion Tool (AWS SCT).

To analyze and convert your SQL code

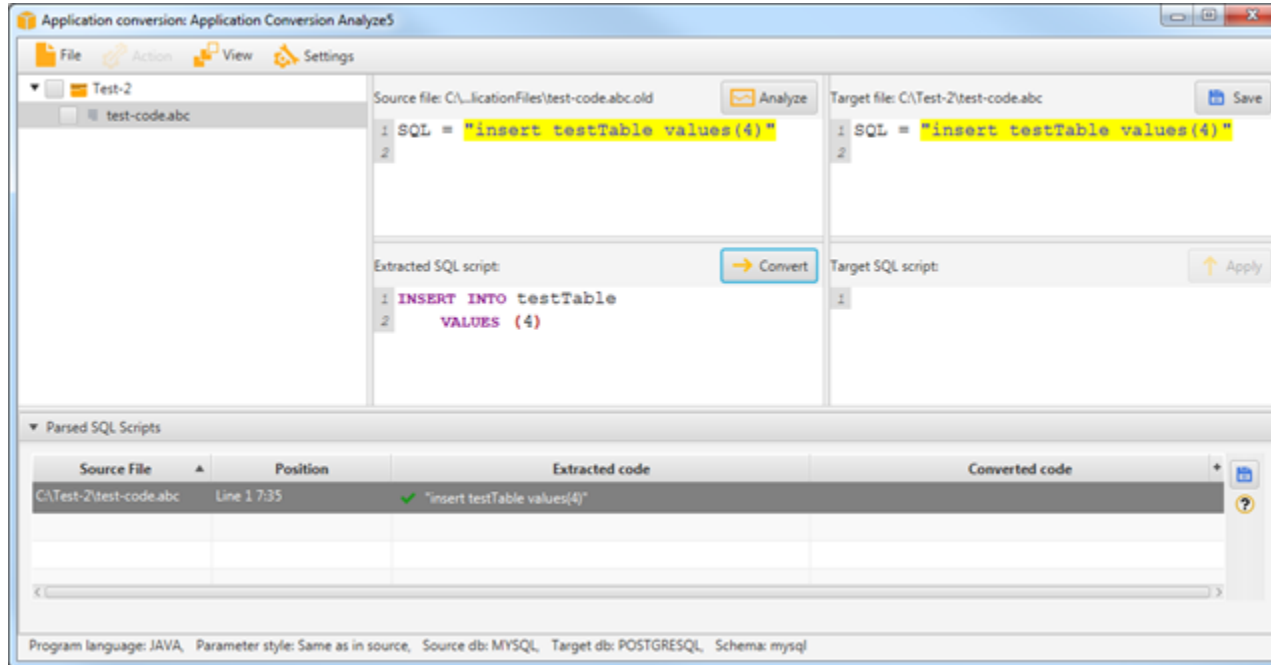
1. In the application conversion project, choose **Analyze**.

AWS SCT analyzes your application code and extracts the SQL code. A list of all extracted SQL code appears in the **Parsed SQL Scripts** pane at the bottom of the window. The selected item in the list also appears in the **Extracted SQL script** pane.

2. You can analyze each SQL code item in the list, and when you are ready, choose **Convert** to convert the SQL to SQL for your target database.

Note

You can edit the converted SQL code in a later procedure.



Creating and Using the Assessment Report

The *application assessment report* provides important information about converting the application SQL code from your source database schema to your target database schema. The report details all of the SQL extracted from the application, all of the SQL converted, and action items for SQL that can't be converted. The report also includes estimates of the amount of effort that it will take to manually convert the SQL code that can't be converted automatically.

Creating an Application Assessment Report

Use the following procedure to create an application assessment report.

To create an application assessment report

1. In the application conversion project window, choose **Create Report** from the **Actions** menu. The report is created and opens in the application conversion project window.
2. Review the **Summary** tab.

The **Summary** tab shown following, displays the summary information from the application assessment report. It shows the SQL code items that were converted automatically, and items that were not converted automatically.



3. Choose the **SQL Conversion Actions** tab and review the information.

The **SQL Conversion Actions** tab contains a list of SQL code items that can't be converted automatically. There are also recommendations for how to manually convert the SQL code. You edit your converted SQL code in a later step. For more information, see [Editing and Saving Your Converted SQL Code with the AWS Schema Conversion Tool \(p. 134\)](#).



4. You can save a local copy of the application assessment report as either a PDF file or a comma-separated values (CSV) file. The PDF file contains both the summary and action item information. The CSV file contains only action item information.

Editing and Saving Your Converted SQL Code with the AWS Schema Conversion Tool

The assessment report includes a list of SQL code items that can't be converted automatically. For each item that can't be converted, there is an action item on the **SQL Conversion Actions** tab. For these items, you can edit the SQL code manually to perform the conversion.

Use the following procedure to edit your converted SQL code, apply the changes, and then save them.

To edit, apply changes to, and save your converted SQL code

1. Edit your converted SQL code directly in the **Target SQL script** pane. If there is no converted code shown, you can click in the pane and start typing.
2. After you are finished editing your converted SQL code, choose **Apply**. At this point, the changes are saved in memory, but not yet written to your file.
3. Choose **Save** to save your changes to your file.

Important

When you choose **Save** you overwrite your original file. Make a copy of your original file before saving so you have a record of your original application code.

Storing AWS Profiles in the AWS Schema Conversion Tool

You can store your AWS credentials in the AWS Schema Conversion Tool (AWS SCT). AWS SCT uses your credentials when you use features that integrate with AWS services. For example, AWS SCT integrates with Amazon S3, AWS Lambda, and AWS Database Migration Service.

AWS SCT asks you for your AWS credentials when you access a feature that requires them. You can store your credentials in the global application settings. When AWS SCT asks for your credentials, you can select the stored credentials.

You can store different sets of AWS credentials in the global application settings. For example, you can store one set of credentials that you use in test scenarios, and a different set of credentials that you use in production scenarios. You can also store different credentials for different AWS regions.

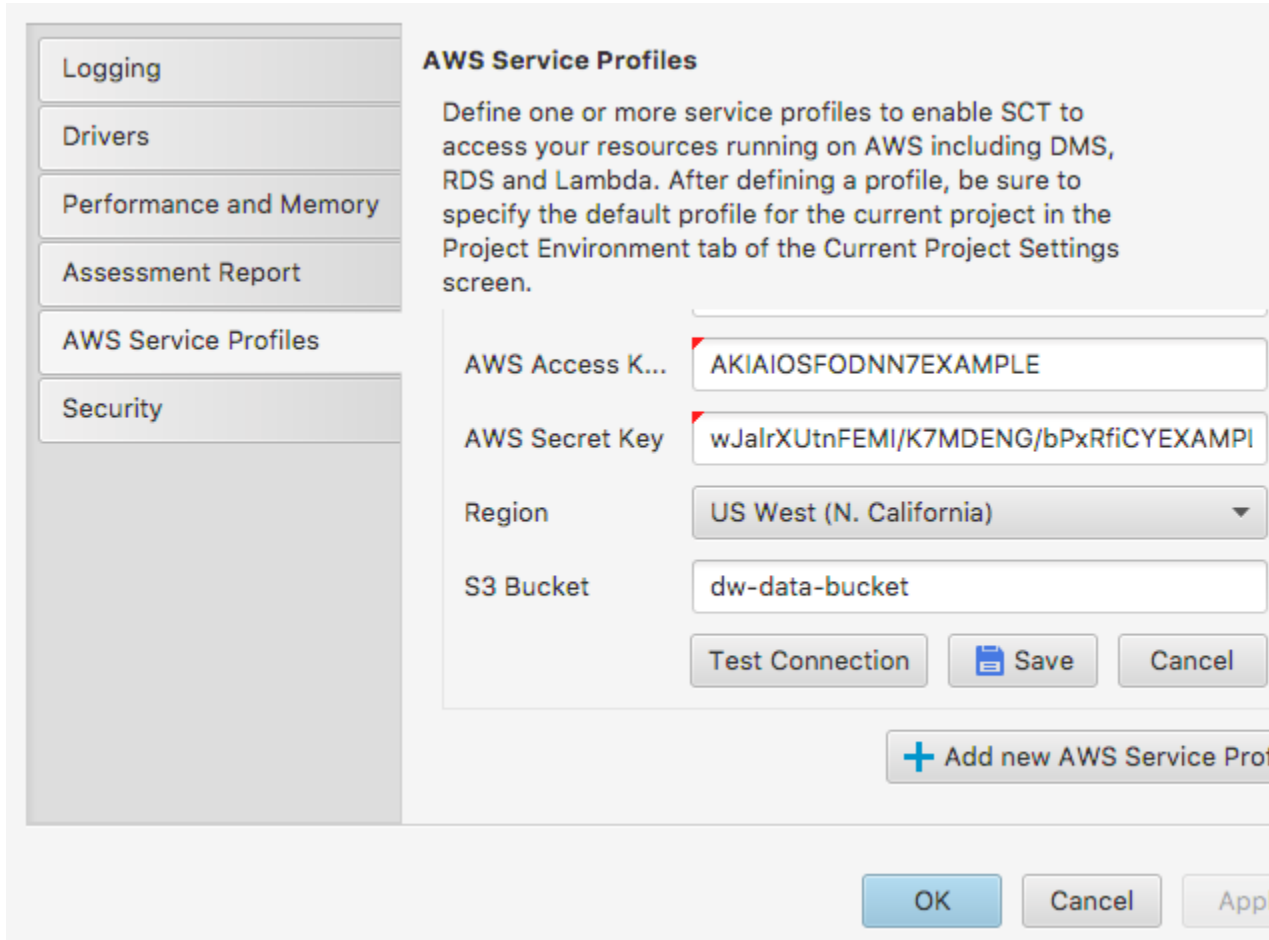
Storing AWS Credentials

Use the following procedure to store AWS credentials globally.

To store AWS credentials

1. Start the AWS Schema Conversion Tool.
2. Open the **Settings Menu**, and then choose **Global Settings**. The **Global settings** dialog box appears.

Choose the **AWS Service Profiles** tab, as shown following.



3. Choose **Add new AWS Service Profile**. A new row is added to the list of profiles.
4. Choose the edit icon to configure your profile.
 - a. For **Profile name**, type a name for your profile.
 - b. For **AWS Access Key**, type your AWS access key.
 - c. For **AWS Secret Key**, type your AWS secret key.
 - d. For **Region**, choose the region for your profile.
 - e. For **S3 Bucket**, choose the S3 bucket for your profile. You only need to specify a bucket if you are using a feature that connects to S3.
 - f. Select **Use FIPS endpoint for S3** if you need to comply with the Federal Information Processing Standard security requirements. FIPS endpoints are available in the following AWS Regions:
 - US East (N. Virginia) Region
 - US East (Ohio) Region
 - US West (N. California) Region
 - US West (Oregon) Region
5. Choose **Test Connection** to verify that your credentials are correct and active.

The **Test Connection** dialog box appears. You can see the status for each of the services connected to your profile. **Pass** indicates that the profile can successfully access the service.

6. After you have configured your profile, choose **Save** to save your profile. You can also choose **Cancel** to cancel your changes.

7. Choose **OK** to close the **Global Settings** dialog box.

Setting the Default Profile for a Project

You can set the default profile for an AWS SCT project. Doing this associates the AWS credentials stored in the profile with the project. With your project open, use the following procedure to set the default profile.

To set the default profile for a project

1. Start the AWS Schema Conversion Tool.
2. Open the **Settings Menu**, and then choose **Project Settings**. The **Current project settings** dialog box appears.
3. Choose the **Project Environment** tab.
4. For **AWS Service Profile**, choose the profile that you want to associate with the project.
5. Choose **OK** to close the **Current project settings** dialog box. You can also choose **Cancel** to cancel your changes.

Related Topics

- [The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases \(p. 85\)](#)
- [The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses \(p. 109\)](#)
- [Working with the AWS Database Migration Service Using the AWS Schema Conversion Tool \(p. 113\)](#)

Best Practices for the AWS Schema Conversion Tool

Following, you can find information on best practices and options for using the AWS Schema Conversion Tool (AWS SCT).

General Memory Management and Performance Options

You can configure the AWS Schema Conversion Tool with different memory performance settings. Increasing memory speeds up the performance of your conversion but uses more memory resources on your desktop.

To set your memory management option, choose **Global Settings** from the **Settings** menu, and choose the **Performance and Memory** tab. Choose one of the following options:

- **Fast conversion, but large memory consumption** – This option optimizes for speed of the conversion, but might require more memory for the object reference cache.
- **Low memory consumption, but slower conversion** – This option minimizes the amount of memory used, but results in a slower conversion. Use this option if your desktop has a limited amount of memory.
- **Balance speed with memory consumption** – This option optimizes provides a balance between memory use and conversion speed.

Configuring Additional Memory

For converting large database schemas, for example a database with 3,500 stored procedures, you can configure the amount of memory available to the AWS Schema Conversion Tool.

To modify the amount of memory AWS SCT consumes

1. Locate the folder where the configuration file is (C:\Program Files\AWS Schema Conversion Tool\App).
2. Open the configuration file `AWS Schema Conversion Tool.cfg` with Notepad or your favorite text editor.
3. Edit the `JVMUserOptions` section to set the minimum and maximum memory available. The following example sets the minimum to 4 GB and the maximum to 40 GB.

```
[JVMUserOptions]
-Xmx48960m
-Xms4096m
```

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Troubleshooting Issues with the AWS Schema Conversion Tool

Following, you can find information about troubleshooting issues with the AWS Schema Conversion Tool (AWS SCT).

Cannot load objects from an Oracle source database

When you attempt to load schema from an Oracle database, you might encounter one of the following errors.

```
Cannot load objects tree.
```

```
ORA-00942: table or view does not exist
```

These errors occur because the user whose ID you used to connect to the Oracle database doesn't have sufficient permissions to read the schema, as required by AWS SCT.

You can resolve the issue by granting the user `select_catalog_role` permission and also permission to any dictionary in the database. These permissions provide the read-only access to the views and system tables that is required by AWS SCT. The following example creates a user ID named `min_privs` and grants the user with this ID the minimum permissions required to convert schema from an Oracle source database.

```
create user min_privs identified by min_privs;  
grant connect to min_privs;  
grant select_catalog_role to min_privs;  
grant select any dictionary to min_privs;
```

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

AWS Schema Conversion Tool Reference

Following, you can find reference material for the AWS Schema Conversion Tool (AWS SCT):

- [MySQL to PostgreSQL Supported Schema Conversion \(p. 215\)](#)
- [Oracle to MySQL Supported Schema Conversion \(p. 245\)](#)
- [Oracle to PostgreSQL Supported Schema Conversion \(p. 274\)](#)
- [PostgreSQL to MySQL Supported Schema Conversion \(p. 215\)](#)
- [Microsoft SQL Server to MySQL Supported Schema Conversion \(p. 141\)](#)
- [Microsoft SQL Server to PostgreSQL Supported Schema Conversion \(p. 161\)](#)

Microsoft SQL Server to MySQL Supported Schema Conversion

The following sections list the schema elements from Microsoft SQL Server and whether they are supported for automatic conversion to MySQL using the AWS Schema Conversion Tool.

Topics

- [Statements \(p. 141\)](#)
- [Procedures \(p. 145\)](#)
- [Flow Control \(p. 145\)](#)
- [Functions \(p. 146\)](#)
- [Operators \(p. 153\)](#)
- [Transactions \(p. 156\)](#)
- [Data Types \(p. 157\)](#)
- [Data Definition Language \(DDL\) \(p. 159\)](#)
- [Cursors \(p. 160\)](#)
- [Related Topics \(p. 161\)](#)

Statements

Topics

- [SELECT \(p. 142\)](#)
- [INSERT \(p. 142\)](#)
- [UPDATE \(p. 143\)](#)
- [DELETE \(p. 144\)](#)

SELECT

Clause	Automatically Converted	Details
WITH	No	Try converting the WITH(query) expression to a subquery and use it in a SELECT query.
UNION	Yes	
EXCEPT INTERSECT	No	
ALL DISTINCT	Yes	
TOP	Partial	MySQL doesn't support TOP with the PERCENT or WITH TIES options. Try using the LIMIT option or perform a manual conversion.
<select_list>	Yes	
INTO	No	
FROM	Partial	MySQL doesn't support the CONTAINS and FREETEXT predicates in the FROM clause search condition.
WHERE	Yes	
GROUP BY	Partial	MySQL doesn't support ORDER BY with the ROLLUP, CUBE, and GROUPING SETS options. Try creating a stored procedure to replace the query.
HAVING	Yes	
ORDER BY	Partial	MySQL doesn't support ORDER BY with the OFFSET or FETCH options. Try creating a stored procedure to replace the query. MySQL doesn't support ORDER BY with the COLLATE option. You must use the collation settings that were assigned when the database was created.
FOR	No	
OPTION	No	

INSERT

Clause	Automatically Converted	Details
WITH	No	Try to convert the WITH query to a subquery, and then use the subquery in the INSERT query.
INSERT	Yes	

Clause	Automatically Converted	Details
TOP	No	Get a count of all rows, and then use the following expression, where count_of_all_rows is your row count: percent_expression * count_of_all_rows / 100
INTO	Yes	
table_name	Yes	
WITH (<Table_Hint_Limited>)	No	Use MySQL methods of performance tuning.
OUTPUT	No	Create a trigger for INSERT statements for the table, and then save the inserted rows in a temporary table. After the INSERT operation, you can make use of the rows saved in the temporary table. This logic can be placed in a stored procedure.
VALUES	Yes	
derived_table	Yes	
execute_statement	No	Create a temporary table, fill it with the data to insert, and use the temporary table in the query.
<dml_table_source>	No	
DEFAULT VALUES	No	

UPDATE

Clause	Automatically Converted	Details
WITH	No	Try to convert the WITH query to a subquery, and then use the subquery in the UPDATE query.
TOP	Partial	MySQL doesn't support UPDATE with the PERCENT option. Get a count of all rows, and then use the following expression, where count_of_all_rows is your row count: percent_expression * count_of_all_rows / 100
WITH (<Table_Hint_Limited>)	No	Use MySQL methods of performance tuning.
SET	No	
OUTPUT	No	Create a trigger for UPDATE statements for the table, and then save the changed rows in a temporary table. After the UPDATE operation, you can make use of the rows saved in the temporary table. This logic can be placed in a stored procedure.
FROM	Partial	MySQL doesn't support a FROM clause. AWS Schema Conversion Tool moves <table-source> to the UPDATE clause.

Clause	Automatically Converted	Details
WHERE	Yes	
CURRENT OF	No	Replace the CURRENT OF clause with an expression that consists of conditions that use unique key fields of the table.
GLOBAL	No	
OPTION	No	

Note

MySQL doesn't support FILESTREAM DATA. Perform a manual conversion to update the data in the file system file.

DELETE

Clause	Automatically Converted	Details
WITH	No	Try to convert the WITH query to a subquery, and then use the subquery in the DELETE query.
TOP	Partial	MySQL doesn't support the PERCENT option. A manual conversion is required. Get a count of all rows, and then use the following expression, where count_of_all_rows is your row count: percent_expression * count_of_all_rows / 100
FROM	Yes	
table_or_view_name	Yes	
rowset_function_limited	No	
WITH	No	MySQL doesn't support hints in DELETE statements, so the AWS Schema Conversion Tool skips options in the format WITH(Table_Hint_Limited). Use MySQL methods of performance tuning.
OUTPUT	No	Create a trigger for DELETE statements for the table, and then save the deleted rows in a temporary table. After the DELETE operation, you can make use of the rows saved in the temporary table. This logic can be placed in a stored procedure.
WHERE	Yes	
CURRENT OF	No	Replace the CURRENT OF clause with an expression that consists of conditions that use unique key fields of the table.
GLOBAL	No	
OPTION	No	

Procedures

Topics

- [CREATE PROCEDURE \(p. 145\)](#)

CREATE PROCEDURE

Clause	Automatically Converted	Details
@parameter	Yes	MySQL doesn't support procedure arguments of the CURSOR data type. Change the business logic to eliminate the need to send cursors through arguments to a stored procedure.
VARYING	Partial	
OUT		
OUTPUT	Partial	
READONLY	Partial	
WITH	No	
FOR REPLICATION	No	
AS BEGIN ... END	No	
ENCRYPTION	Yes	
RECOMPILE	No	
EXECUTE AS	No	

Flow Control

Clause	Automatically Converted	Details
TRY...CATCH THROW	No	Try using the DECLARE HANDLER for the CATCH emulation block and the SIGNAL SQLSTATE operator to emulate the THROW operator.
WAITFOR	No	Perform a manual conversion.
DECLARE	No	Perform a manual conversion.
GOTO	No	Revise your code to eliminate GOTO operators, using BEGIN...END blocks in combination with LEAVE, REPEAT, UNTIL, and WHILE operators.
BREAK	Yes	
CONTINUE	Yes	

Clause	Automatically Converted	Details
IF...ELSE	Yes	
RETURN	No	MySQL does not support returning a value from a procedure using the RETURN statement. To return a value, use the OUT parameter or a result set.
WHILE	Yes	
CASE	Yes	
COALESCE	Yes	
NULLIF	Yes	

Functions

Topics

- [Aggregate Functions \(p. 146\)](#)
- [Date and Time Functions \(p. 147\)](#)
- [Mathematical Functions \(p. 148\)](#)
- [String Functions \(p. 148\)](#)
- [Configuration Functions \(p. 149\)](#)
- [Conversion Functions \(p. 150\)](#)
- [Security Functions \(p. 150\)](#)
- [System Functions \(p. 151\)](#)
- [Logical Functions \(p. 152\)](#)
- [CREATE FUNCTION \(p. 153\)](#)
- [EXECUTE \(p. 153\)](#)

In this section, you can find a list of the Microsoft SQL Server built-in functions that indicates whether the AWS Schema Conversion Tool performs an automatic conversion. Where MySQL doesn't support a function, consider creating a user-defined function.

Aggregate Functions

Function	Automatically Converted	Details
AVG	Yes	
COUNT	Yes	
COUNT_BIG	Partial	
MIN	Yes	
MAX	Yes	
SUM	Yes	

Function	Automatically Converted	Details
CHECKSUM_AGG	No	
STDEV	Partial	MySQL doesn't support the STDEV function with the DISTINCT clause.
STDEVP	Partial	MySQL doesn't support the STDEVP function with the DISTINCT clause.
GROUPING	No	
GROUPING_ID	No	
VAR	Partial	MySQL doesn't support the VAR function with the DISTINCT clause.
VARP	Partial	MySQL doesn't support the VARP function with the DISTINCT clause.

Date and Time Functions

Function	Automatically Converted	Details
DATEADD	Partial	MySQL doesn't support the DATEADD function with the nanosecond datepart.
DATEDIFF	Partial	MySQL doesn't support the DATEDIFF function with the week, millisecond, or nanosecond datepart.
DATENAME	Partial	MySQL doesn't support the DATENAME function with the millisecond, nanosecond, or TZoffset datepart.
DATEPART	Partial	MySQL doesn't support the DATEPART function with the millisecond, nanosecond, or TZoffset datepart.
DAY	Partial	
GETDATE	Partial	
GETDATE + 1	Partial	
GETUTCDATE	Partial	
MONTH	Partial	
YEAR	Partial	
DATETIMEOFFSETFROMPARTS	No	
ISDATE	No	
SWITCHOFFSET	No	
SYSDATETIMEOFFSET	No	
TODATETIMEOFFSET	No	

Mathematical Functions

Function	Automatically Converted	Details
ABS	Yes	
ACOS	Yes	
ASIN	Yes	
ATN2	Partial	
ATAN	Yes	
CEILING	Yes	
COS	Yes	
COT	Yes	
DEGREES	Yes	
EXP	Yes	
FLOOR	Yes	
LOG10	Yes	
LOG	Yes	
PI	Yes	
POWER	Yes	
RADIANS	Yes	
RAND	Yes	
ROUND	Partial	MySQL doesn't support the ROUND function with the function argument.
SIGN	Yes	
SIN	Yes	
SQRT	Yes	
SQUARE	No	
TAN	Yes	

String Functions

Function	Automatically Converted	Details
ASCII	Yes	

Function	Automatically Converted	Details
CHAR	Yes	
CHARINDEX	No	
CONCAT	Yes	
DIFFERENCE	No	
FORMAT	No	
LEFT	Yes	
LEN	Partial	
LOWER	Yes	
LTRIM	Yes	
NCHAR	No	
PATINDEX	No	
QUOTENAME	Partial	
REPLACE	Yes	
REPLICATE	Partial	
REVERSE	Yes	
RIGHT	Yes	
RTRIM	Yes	
SOUNDEX	No	
SPACE	Yes	
STR	No	
STUFF	No	
SUBSTRING	Yes	
UNICODE	No	
UPPER	Yes	

Configuration Functions

Function	Automatically Converted	Details
@@DATEFIRST	No	
@@DBTS	Yes	

Function	Automatically Converted	Details
@@LANGID	Yes	
@@LANGUAGE	No	
@@LOCK_TIMEOUT	Yes	
@@MAX_CONNECTIONS	No	
@@MAX_PRECISION	Yes	
@@NESTLEVEL	Yes	
@@OPTIONS	Yes	
@@REMSERVER	Yes	
@@SERVERNAME	No	
@@SERVICENAME	Yes	
@@SPID	Yes	
@@TEXTSIZE	Yes	
@@VERSION	No	

Conversion Functions

Function	Automatically Converted	Details
CAST and CONVERT	No	
PARSE	No	
TRY_CAST	No	
TRY_CONVERT	No	
TRY_PARSE	No	

Security Functions

Function	Automatically Converted	Details
CERTENCODED	Yes	
CERTPRIVATEKEY	Yes	
CURRENT_USER	No	
DATABASE_PRINCIPAL_ID	No	

Function	Automatically Converted	Details
HAS_PERMS_BY_NAME	Yes	
IS_MEMBER	Yes	
IS_ROLEMEMBER	Yes	
IS_SRVROLEMEMBER	Yes	
ORIGINAL_LOGIN	No	
PERMISSIONS	Yes	
PWDCOMPARE	Yes	
PWDENCRYPT	Yes	
SCHEMA_ID	Yes	
SCHEMA_NAME	No	
SESSION_USER	No	
SUSER_ID	Yes	
SUSER_NAME	Yes	
SUSER_SID	Yes	
SUSER_SNAME	Yes	
sys.fn_built_in_permissions	Yes	
sys.fn_get_audit_file	Yes	
sys.fn_my_permissions	Yes	
SYSTEM_USER	Yes	
USER_ID	Yes	
USER_NAME	Yes	

System Functions

Function	Automatically Converted	Details
\$PARTITION	Yes	
@@ERROR	Yes	
@@IDENTITY	Yes	
@@PACK_RECEIVED	Yes	
@@ROWCOUNT	Yes	

Function	Automatically Converted	Details
@@TRANCOUNT	Yes	
BINARY_CHECKSUM	Yes	
CHECKSUM	No	
CONNECTIONPROPERTY	Yes	
CONTEXT_INFO	Yes	
CURRENT_REQUEST_ID	Yes	
ERROR_LINE	No	
ERROR_MESSAGE	No	
ERROR_NUMBER	No	
ERROR_PROCEDURE	No	
ERROR_SEVERITY	No	
ERROR_STATE	No	
FORMATMESSAGE	No	
GET_FILESTREAM_TRANSACTION_CONTEXT	Yes	
GETANSINULL	Yes	
HOST_ID	No	
HOST_NAME	No	
ISNULL	No	
MIN_ACTIVE_ROWVERSION	Yes	
NEWID	Yes	
NEWSEQUENTIALID	Yes	
PARSENAME	Yes	
ROWCOUNT_BIG	Yes	
XACT_STATE	Yes	

Logical Functions

Function	Automatically Converted	Details
CHOOSE	No	
IIF	No	

CREATE FUNCTION

Clause	Automatically Converted	Details
@parameter_name	Yes	
type_schema_name	Yes	
parameter_data_type	No	
= default	Yes	
READONLY	No	
RETURNS return_data_type	No	
WITH function_option	Yes	
BEGIN ... END	No	
RETURN scalar_expression	Yes	MySQL supports only functions that return a scalar value.

EXECUTE

Clause	Automatically Converted	Details
@parameter_name	Yes	
type_schema_name	Yes	
parameter_data_type	No	
= default	Yes	
READONLY	No	
RETURNS return_data_type	No	
WITH function_option	Yes	
BEGIN ... END	No	
RETURN scalar_expression	Yes	MySQL supports only functions that return a scalar value.

Operators

Topics

- [Arithmetic Operators \(p. 154\)](#)
- [Assignment Operator \(p. 154\)](#)
- [Bitwise Operators \(p. 154\)](#)
- [Comparison Operators \(p. 155\)](#)
- [Logical Operators \(p. 155\)](#)

- [Set Operators \(p. 155\)](#)
- [String Concatenation Operator \(p. 156\)](#)
- [Unary Operators \(p. 156\)](#)

Arithmetic Operators

Clause	Automatically Converted	Details
+	Yes	
+=	Yes	
-	Yes	
-=	Yes	
*	Yes	
*=	Yes	
/	Yes	
/=	Yes	
%	Yes	

Assignment Operator

Clause	Automatically Converted	Details
=	Yes	

Bitwise Operators

Clause	Automatically Converted	Details
&	Yes	
	Yes	
^	Yes	
~	Yes	

Comparison Operators

Clause	Automatically Converted	Details
=	Yes	
>	Yes	
<	Yes	
>=	Yes	
<=	Yes	
<>	Yes	
!=	Yes	
!<	Yes	
!>	Yes	

Logical Operators

Clause	Automatically Converted	Details
ALL	Yes	
AND	Yes	
ANY	Yes	
BETWEEN	Yes	
EXISTS	Yes	
IN	Yes	
LIKE	Yes	
NOT	Yes	
OR	Yes	
SOME	Partial	The AWS Schema Conversion Tool converts SOME to ANY.

Set Operators

Clause	Automatically Converted	Details
UNION	Yes	
UNION ALL	Yes	

Clause	Automatically Converted	Details
EXCEPT	No	Perform a manual conversion.
INTERSECT	No	Perform a manual conversion.

String Concatenation Operator

Clause	Automatically Converted	Details
+	Yes	

Unary Operators

Clause	Automatically Converted	Details
+	Yes	
-	Yes	

Transactions

Topics

- [BEGIN TRANSACTION \(p. 156\)](#)

BEGIN TRANSACTION

Clause	Automatically Converted	Details
DISTRIBUTED	No	MySQL doesn't support distributed transactions. Revise your architecture so that it doesn't use distributed transactions.
transaction_name	No	MySQL doesn't support named transactions. Revise your code to eliminate marked transactions.
WITH MARK	No	MySQL doesn't support the WITH MARK option. Revise your code to eliminate marked transactions.
SAVE	Partial	
ROLLBACK	Partial	
transaction_name	No	MySQL doesn't support named transactions. Revise your code to eliminate named transactions.

Clause	Automatically Converted	Details
savepoint_name	Partial	
COMMIT	Partial	
DELAYED_DURABILITY	No	

Data Types

Topics

- [Numerics \(p. 157\)](#)
- [Date and Time \(p. 157\)](#)
- [Character Strings \(p. 158\)](#)
- [Binary Strings \(p. 158\)](#)
- [Special Types \(p. 159\)](#)

Numerics

Data type	Automatically Converted	Default Conversion	Details
bigint	Yes	bigint	
int	Yes	int	
smallint	Yes	smallint	
tinyint	Yes	tinyint unsigned	
bit	Yes	bit(1)	
money	Yes	numeric(19,4)	
smallmoney	Yes	numeric(10,4)	
numeric	Yes	numeric	
decimal	Yes	decimal	
float	Yes	double	
real	Yes	float	

Date and Time

Data type	Automatically Converted	Default Conversion	Details
date	Partial	date	0001-01-01 through 9999-12-31

Data type	Automatically Converted	Default Conversion	Details
datetime2(7)	Partial	datetime	0001-01-01 through 9999-12-31 00:00:00 through 23:59:59.9999999
datetime	Partial	datetime	1753-01-01, through 9999-12-31 00:00:00 through 23:59:59.997
datetimeoffset(7)	No		
smalldatetime	Yes	datetime	
time(7)	Yes	time	

Character Strings

Data type	Automatically Converted	Default Conversion	Details
char(len)	Partial	char	len= 1 - 255 (2 ⁸ - 1) symbols (loss range)
varchar(len)	Yes	varchar	
varchar(max)	Yes	longtext	
text	Yes	longtext	
nchar(len)	len = 1 - 4000	char	len= 1 - 255 (2 ⁸ - 1) symbols (loss range)
nvarchar(len)	Yes	varchar	
nvarchar(max)	Yes	longtext	
ntext	Yes	longtext	

Binary Strings

Data type	Automatically Converted	Default Conversion	Details
binary(len)	Yes	blob	
varbinary(len)	Yes	blob	
varbinary(max)	Yes	longblob	
hierarchyid	No		
sql_variant	No		
table	No		

Data type	Automatically Converted	Default Conversion	Details
uniqueidentifier	No		
xml	No		
geography	No		
geometry	No		

Special Types

Data type	Automatically Converted	Default Conversion	Details
UNIQUEIDENTIFIER	Partial	CHAR(38) OR BINARY(16)	
DOUBLE PRECISION	Partial	FLOAT(53)	
IDENTITY	Partial	AUTO_INCREMENT	
SYSNAME	Partial	NVARCHAR(128) NOT NULL	

Data Definition Language (DDL)

Topics

- [CREATE TABLE \(p. 159\)](#)
- [CREATE INDEX \(p. 159\)](#)
- [CREATE TRIGGER \(p. 159\)](#)
- [CREATE VIEW \(p. 160\)](#)

CREATE TABLE

CREATE TABLE statements are converted automatically except for the following clause.

Clause	Automatically Converted	Details
SET DEFAULT	No	MySQL doesn't support the SET DEFAULT option for FOREIGN KEY.

CREATE INDEX

The AWS Schema Conversion Tool does not support automatic migration of data definition language (DDL) code to create an index.

CREATE TRIGGER

CREATE TRIGGER statements are converted automatically except for the following clause.

Clause	Automatically Converted	Details
FOR	No	MySQL doesn't support a FOR clause in a CREATE TRIGGER statement.

CREATE VIEW

CREATE VIEW statements are converted automatically except for the following clauses.

Clause	Automatically Converted	Details
SCHEMABINDING	No	AWS Schema Conversion Tool ignores this clause.
ENCRYPTION	No	AWS Schema Conversion Tool ignores this clause.
VIEW_METADATA	No	AWS Schema Conversion Tool ignores this clause.

Cursors

Topics

- [DECLARE CURSOR \(p. 160\)](#)

DECLARE CURSOR

Clause	Automatically Converted	Details
LOCAL	Yes	Change the global cursor to a local cursor, or revise your code so it doesn't require global cursors.
GLOBAL	No	Setting this option corresponds to the typical behavior of cursors in MySQL, so the AWS Schema Conversion Tool skips this option during conversion.
FORWARD_ONLY	Partial	Revise your code to eliminate cursors with the SCROLL option.
SCROLL	No	
STATIC	Yes	The membership and order of rows never changes for cursors in MySQL, so the AWS Schema Conversion Tool skips this option during conversion. Verify that the converted schema behavior is acceptable.
KEYSET	Partial	Revise your code to eliminate dynamic cursors.
DYNAMIC	No	Setting this option corresponds to the typical behavior of cursors in MySQL, so the AWS Schema Conversion Tool skips this option during conversion.

Clause	Automatically Converted	Details
FAST_FORWARD	Yes	All MySQL cursors are read-only, so the AWS Schema Conversion Tool skips this option during conversion.
READ_ONLY	Yes	MySQL doesn't support the option SCROLL_LOCKS, so the AWS Schema Conversion Tool ignores this option during conversion. Verify that the converted schema behavior is acceptable.
SCROLL_LOCKS	Partial	MySQL doesn't support the option OPTIMISTIC, so the AWS Schema Conversion Tool ignores this option during conversion. Verify that the converted schema behavior is acceptable.
OPTIMISTIC	Partial	MySQL doesn't support the option TYPE_WARNING, so the AWS Schema Conversion Tool ignores this option during conversion. Verify that the converted schema behavior is acceptable.
TYPE_WARNING	Partial	Change the global cursor to a local cursor, or revise your code so it doesn't require global cursors.

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Microsoft SQL Server to PostgreSQL Supported Schema Conversion

The following sections list the schema elements from a Microsoft SQL Server database and whether they are supported for automatic conversion to PostgreSQL using the AWS Schema Conversion Tool.

Data Definition Language (DDL)

Naming Objects

Clause	Automatically Converted	Details
Schemas	Yes	
Tables, Columns	Yes	

Statements

Clause	Automatically Converted	Details
ALTER INDEX	No	
ALTER PROCEDURE	No	
ALTER TRIGGER	No	
ALTER VIEW	No	
CREATE FUNCTION	Partial	Automatic migration of inline functions is not supported. Default parameters specified for any function are skipped.
CREATE PROCEDURE	Yes	
CREATE SEQUENCE	Yes	
IDENTITY (x,y)	Yes	
TRUNCATE TABLE	Yes	
UPDATE STATISTICS	No	

ALTER TABLE

Clause	Automatically Converted	Details
ADD	Yes	
ALTER COLUMN	No	
DROP COLUMN	No	
DROP CONSTRAINT	No	
ENABLE/DISABLE TRIGGER	Yes	

CREATE INDEX

Clause	Automatically Converted	Details
CLUSTER option	Partial	
COLUMNSTORE index	Partial	
INCLUDE clause	Partial	
Other INDEX options	Partial	

Clause	Automatically Converted	Details
SPATIAL index	Partial	
UNIQUE option	Yes	
XML index	Partial	

CREATE TABLE

Clause	Automatically Converted	Details
Partitioned tables	Yes	
Regular tables	Yes	
Temporary tables	Partial	
Wide tables	Yes	

CREATE TRIGGER

Clause	Automatically Converted	Details
FOR EACH STATEMENT Triggers	Partial	
INSTEAD OF Triggers	Partial	
Trigger event (INSERT/UPDATE/DELETE)	Partial	
Trigger time (BEFORE/AFTER)	Partial	

CREATE VIEW

Clause	Automatically Converted	Details
CREATE VIEW	Yes	
Updatable Views	Yes	
VIEW	Yes	

DROP Statements

Clause	Automatically Converted	Details
DROP TABLE	Yes	
DROP VIEW	Yes	
Other types of objects	Yes	

Data Manipulation Language (DML)

Clause for Statements

Clause	Automatically Converted	Details
Join Hints	No	
Query Hints	No	
Table Hint	No	
WHERE	Yes	

Joins

Cross Joins

Clause	Automatically Converted	Details
Cross join	Yes	
Cross join with condition	Yes	

Inner Joins

Clause	Automatically Converted	Details
Inner Joins	Yes	

Outer Joins

Clause	Automatically Converted	Details
Full join	Yes	
Full outer join	Yes	
Left join	Yes	
Left outer join	Yes	
Right join	Yes	
Right outer join	Yes	

Predicates for Statements

Clause	Automatically Converted	Details
Boolean conditions	Yes	
Exists conditions	Yes	
In conditions	Yes	
Null conditions	Yes	
Pattern matching conditions	Yes	
Range conditions	Yes	

Comparison Conditions

Clause	Automatically Converted	Details
=, <>, <, <=, >, >=	Yes	
ANY, SOME, ALL	Partial	

Statements

DELETE

Clause	Automatically Converted	Details
CURRENT OF	Yes	

Clause	Automatically Converted	Details
FROM	Partial	
OUTPUT	Partial	
TOP	No	
VALUES	Yes	
WITH	Yes	

INSERT

Clause	Automatically Converted	Details
DEFAULT	Yes	
DEFAULT VALUES	Yes	
FROM	Yes	
INTO	Yes	
OUTPUT	No	
TOP	No	
VALUES	Yes	
WITH	Yes	

SELECT

Clause	Automatically Converted	Details
GROUP BY clause with an expression	Yes	
GROUP BY clause with multiple tables	Yes	
GROUP BY CUBE	No	
Group By Grouping Sets	No	
GROUP BY ROLLUP	No	
Limiting the number of rows returned	Yes	
Select all columns (using the * and aliases)	Yes	

Clause	Automatically Converted	Details
Select all columns (using the *)	Yes	
Select subset of the columns	Yes	
Select with calculations	Yes	
Select with column heading	Yes	
Select with constants	Yes	
Specifying a ascending order	Yes	
Specifying a collation	Yes	
Specifying a conditional order	Yes	
Specifying a descending order	Yes	
Specifying a percentage	Yes	
Specifying a percentage (argument 100 percent)	Yes	
Specifying an alias as the sort column	Yes	
Specifying an expression as the sort column	Yes	
Specifying both ascending and descending order	Yes	
TOP with a variable	Yes	
Using ORDER BY and OFFSET command	Yes	
Using ORDER BY in a ranking function	Yes	
Using ORDER BY with UNION	Yes	
Using WITH TIES	No	
WHERE with BETWEEN	Yes	
WHERE with combination of predicates	Yes	
WHERE with CONTAINS	No	

Clause	Automatically Converted	Details
WHERE with EXISTS subquery	Yes	
WHERE with FREETEXT	No	
WHERE with IN custom list	Yes	
WHERE with IN subquery	Yes	
WHERE with LIKE	Partial	
WHERE with negates a Boolean input	Yes	
WHERE with NULL	Yes	

DISTINCT

Clause	Automatically Converted	Details
DISTINCT	Yes	

FROM

Clause	Automatically Converted	Details
FROM	Yes	

GROUP BY

Clause	Automatically Converted	Details
GROUP BY	Yes	

HAVING

Clause	Automatically Converted	Details
HAVING	Yes	

ORDER BY

Clause	Automatically Converted	Details
ORDER BY	Yes	

TOP

Clause	Automatically Converted	Details
TOP	Yes	

WHERE

Clause	Automatically Converted	Details
WHERE	Yes	

UPDATE

Clause	Automatically Converted	Details
CURRENT OF	Yes	
DEFAULT	Yes	
FILESTREAM Data	No	
FROM	Yes	
OUTPUT	Partial	
TOP	No	
VALUES	Yes	
WITH	Yes	

Data Types

Clause	Automatically Converted	Details
BIGINT	Yes	
BINARY	Yes	

Clause	Automatically Converted	Details
BIT	Yes	
CHAR, CHARACTER	Yes	
DATETIME	Yes	
DATETIMEOFFSET	No	
DEC, DECIMAL	Yes	
DOUBLE PRECISION	Yes	
FLOAT	Yes	
GEOGRAPHY	No	
GEOMETRY	No	
HIERARCHYID	No	
IDENTITY	No	
INT, INTEGER	Yes	
MONEY	Yes	
NCHAR	Yes	
NTEXT	Yes	
NUMERIC	Yes	
NVARCHAR	Yes	
REAL	Yes	
ROWVERSION	No	
SMALL MONEY	Yes	
SMALLDATETIME	Yes	
SMALLINT	Yes	
SQL_VARIANT	No	
SYSNAME	Yes	
TABLE	No	
TEXT	Yes	
TIMESTAMP	Yes	
TINYINT	Yes	
UNIQUEIDENTIFIER	Yes	Converted to a universally unique identifier (UUID) if the PostgreSQL instance has the uuid-oss module installed. The uuid-oss module provides functions to generate UUIDs using one of several standard algorithms.

Clause	Automatically Converted	Details
VARBINARY	Yes	
VARCHAR	Yes	
XML	No	

Database Mail

Clause	Automatically Converted	Details
Accounts and Profiles	Yes	
Database Mail Configuration Objects	Yes	
Database Mail Messaging Objects	Yes	
Database Mail Settings	Yes	
Security	Yes	
sp_send_dbmail	No	
sysmail_add_account_sp	No	
sysmail_add_principalprofile_sp	No	
sysmail_add_profile_sp	No	
sysmail_add_profileaccount_sp	No	
sysmail_allitems	No	
sysmail_configure_sp	No	
sysmail_delete_account_sp	No	
sysmail_delete_log_sp	No	
sysmail_delete_mailitems_sp	No	
sysmail_delete_principalprofile_sp	No	
sysmail_delete_profile_sp	No	
sysmail_delete_profileaccount_sp	No	
sysmail_event_log	No	
sysmail_faileditems	No	
sysmail_help_account_sp	No	
sysmail_help_configure_sp	No	

Clause	Automatically Converted	Details
sysmail_help_principalprofile_sp	No	
sysmail_help_profile_sp	No	
sysmail_help_profileaccount_sp	No	
sysmail_help_queue_sp	No	
sysmail_help_status_sp	No	
sysmail_help_status_sp	No	
sysmail_mailattachments	No	
sysmail_sentitems	No	
sysmail_start_sp	No	
sysmail_start_sp	No	
sysmail_stop_sp	No	
sysmail_stop_sp	No	
sysmail_unsentitems	No	
sysmail_update_account_sp	No	
sysmail_update_principalprofile_sp	No	
sysmail_update_principalprofile_sp	No	
sysmail_update_profile_sp	No	
sysmail_update_profileaccount_sp	No	
System State	Yes	

Functions

@@functions

Clause	Automatically Converted	Details
@@CONNECTIONS	No	
@@CPU_BUSY	No	
@@CURSOR_ROWS	No	
@@DATEFIRST	No	
@@DBTS	No	
@@ERROR	No	

Clause	Automatically Converted	Details
@@FETCH_STATUS	Yes	
@@IDENTITY	Yes	
@@IDLE	No	
@@IO_BUSY	No	
@@LANGID	No	
@@LANGUAGE	No	
@@LOCK_TIMEOUT	No	
@@MAX_CONNECTIONS	No	
@@MAX_PRECISION	No	
@@NESTLEVEL	No	
@@OPTIONS	No	
@@PACK_RECEIVED	No	
@@PACK_SENT	No	
@@PACKET_ERRORS	No	
@@PROCID	No	
@@REMSERVER	No	
@@ROWCOUNT	No	
@@SERVERNAME	No	
@@SERVICENAME	No	
@@SPID	No	
@@TEXTSIZE	No	
@@TIMETICKS	No	
@@TOTAL_ERRORS	No	
@@TOTAL_READ	No	
@@TOTAL_WRITE	No	
@@TRANCOUNT	No	
@@VERSION	No	

Aggregate

Clause	Automatically Converted	Details
AVG	Partial	
CHECKSUM_AGG	No	
COUNT	Yes	
COUNT_BIG	Yes	
GROUPING	No	
GROUPING_ID	No	
MAX	Yes	
MIN	Yes	
STDEV	Yes	
STDEVP	Yes	
SUM	Yes	
VAR	Yes	
VARP	Yes	

Built-in Functions

Clause	Automatically Converted	Details
Aggregate Functions	Yes	
Cursor Functions	No	
Metadata Functions	No	
Security Functions	No	
System Functions	No	
System Statistical Functions	No	
Text and Image Functions	No	

Conversion

Clause	Automatically Converted	Details
CAST	Partial	
CONVERT	Partial	
PARSE	No	
TRY_CAST	No	
TRY_CONVERT	No	
TRY_PARSE	No	

Date and Time

Clause	Automatically Converted	Details
CURRENT_TIMESTAMP	Partial	
DATEADD	Partial	
DATEDIFF	Partial	
DATEFROMPARTS	Yes	
DATENAME	Partial	
DATEPART	Partial	
DATETIME2FROMPARTS	Partial	
DATETIMEFROMPARTS	Partial	
DATETIMEOFFSETFROMPARTS	Partial	
DAY	Yes	
EOMONTH	Yes	
GETDATE	Yes	
GETUTCDATE	Yes	
ISDATE	No	
MONTH	Yes	
SMALLDATETIMEFROMPARTS	Yes	
SWITCHOFFSET	No	
SYSDATETIME	Yes	
SYSDATETIMEOFFSET	No	

Clause	Automatically Converted	Details
SYSUTCDATETIME	Yes	
TIMEFROMPARTS	Yes	
TODATETIMEOFFSET	No	
YEAR	Yes	

Mathematical Functions

Clause	Automatically Converted	Details
ABS()	Yes	
ACOS()	Yes	
ASIN()	Yes	
ATAN()	Yes	
ATN2 ()	Yes	
CEILING()	Yes	
COS()	Yes	
COT()	Yes	
DEGREES()	Yes	
EXP()	Yes	
FLOOR()	Yes	
LOG()	Yes	
LOG10()	Yes	
PI()	Yes	
POWER()	Yes	
RADIANS()	Yes	
RAND()	Yes	
RAND(seed)	Partial	
ROUND()	Partial	
ROUND()	Partial	
SIGN()	Yes	
SIN()	Yes	

Clause	Automatically Converted	Details
SQRT()	Yes	
SQUARE()	Yes	
TAN()	Yes	

Ranking Functions

Clause	Automatically Converted	Details
DENSE_RANK	Yes	
NTILE	Yes	
RANK	Yes	
ROW_NUMBER	Yes	

Rowset Functions

Clause	Automatically Converted	Details
CONTAINSTABLE	No	
FREETEXTTABLE	No	
OPENDATASOURCE	No	
OPENQUERY	No	
OPENROWSET	No	
OPENXML	No	

Scalar Functions

Clause	Automatically Converted	Details
ABS()	Yes	
ACOS()	Yes	
ASCII	Yes	
ASIN()	Yes	
ATAN()	Yes	

Clause	Automatically Converted	Details
ATN2 ()	Yes	
AVG	Yes	
CEILING()	Yes	
CHAR	Yes	
CHARINDEX	Yes	
CHECKSUM	No	
CHECKSUM_AGG	No	
CHOOSE	No	
CONCAT	Yes	
COS()	Yes	
COT()	Yes	
COUNT	Yes	
COUNT_BIG	Yes	
CURRENT_TIMESTAMP	Yes	
CURRENT_USER	No	
DATABASE_PRINCIPAL_ID	No	
DATEADD	Partial	
DATEDIFF	Partial	
DATEFROMPARTS	Yes	
DATENAME	Partial	
DATEPART	Partial	
DATETIME2FROMPARTS	Yes	
DATETIMEFROMPARTS	Yes	
DATETIMEOFFSETFROMPARTS	No	
DAY	Yes	
DEGREES()	Yes	
DIFFERENCE	No	
EOMONTH	Yes	
ERROR_LINE	No	
ERROR_MESSAGE	No	

Clause	Automatically Converted	Details
ERROR_NUMBER	No	
ERROR_PROCEDURE	No	
ERROR_SEVERITY	No	
ERROR_STATE	No	
EXP()	Yes	
FLOOR()	Yes	
FORMAT	No	
FORMATMESSAGE	No	
GETDATE	Yes	
GETUTCDATE	Yes	
GROUPING	No	
GROUPING_ID	No	
HOST_ID	No	
HOST_NAME	No	
IIF	No	
ISDATE	No	
ISNULL	Yes	
ISNUMERIC	Implemented in Extension Library	
LEFT	Yes	
LEN	Yes	
LOG()	Yes	
LOG10()	Yes	
LOWER	Yes	
LTRIM	Yes	
MAX	Yes	
MIN	Yes	
MONTH	Yes	
NCHAR	No	
ORIGINAL_LOGIN	No	

Clause	Automatically Converted	Details
PARSENAME	No	
PATINDEX	No	
PI()	Yes	
POWER()	Yes	
PRINT	Yes	
QUOTENAME	Partial	
RADIANS()	Yes	
RAND()	Yes	
REPLACE	Yes	
REPLICATE	Yes	
REVERSE	Yes	
RIGHT	Yes	
ROUND()	Yes	
RTRIM	Yes	
SCHEMA_NAME	No	
SESSION_USER	No	
SIGN()	Yes	
SIN()	Yes	
SMALLDATETIMEFROMPARTS	Yes	
SOUNDEX	Yes	
SPACE	Yes	
SQRT()	Yes	
SQUARE()	Yes	
STDEV	Yes	
STDEVP	Yes	
STR	No	
STUFF	Yes	
SUBSTRING	Yes	
SUM	Yes	
SWITCHOFFSET	No	

Clause	Automatically Converted	Details
SYSDATETIME	Yes	
SYSDATETIMEOFFSET	No	
SYSUTCDATETIME	Yes	
TAN()	Yes	
TIMEFROMPARTS	Yes	
TODATETIMEOFFSET	No	
UNICODE	No	
UPPER	Yes	
VAR	Yes	
VARP	Yes	
YEAR	Yes	

String

Clause	Automatically Converted	Details
ASCII	Yes	
CHAR	Yes	
CHARINDEX	Yes	
CONCAT	Yes	
DIFFERENCE	No	
FORMAT	No	
LEFT	Yes	
LEN	Yes	
LOWER	Yes	
LTRIM	Yes	
NCHAR	No	
PATINDEX	No	
QUOTENAME	Partial	
REPLACE	Yes	
REPLICATE	Yes	

Clause	Automatically Converted	Details
REVERSE	Yes	
RIGHT	Yes	
RTRIM	Yes	
SOUNDEX	No	
SPACE	Yes	
STR	No	
STUFF	Yes	
SUBSTRING	Yes	
UNICODE	No	
UPPER	Yes	

Operators

Arithmetic Operators

Clause	Automatically Converted	Details
- (Subtract)	Partial	
% (Modulo)	Partial	
* (Multiply)	Partial	
/ (Divide)	Partial	
+ (Add)	Partial	

Assignment Operator

Clause	Automatically Converted	Details
=	Yes	

Bitwise Operators

Clause	Automatically Converted	Details
& (Bitwise AND)	Yes	

Clause	Automatically Converted	Details
^ (Bitwise Exclusive OR)	No	
(Bitwise OR)	Yes	

Comparison Operators

Clause	Automatically Converted	Details
Comparison Operators	Partial	

Logical Operators

Clause	Automatically Converted	Details
Logical Operators	Yes	

Set Operators

Clause	Automatically Converted	Details
EXCEPT	Yes	
INTERSECT	Yes	
UNION	Yes	
UNION ALL	Yes	

String Concatenation Operator

Clause	Automatically Converted	Details
String Concatenation Operator	Yes	

Unary Operators

Clause	Automatically Converted	Details
Unary Operators	Yes	

Other

Clause	Automatically Converted	Details
CHECKSUM	No	
CHOOSE	No	
CURRENT_USER	No	
DATABASE_PRINCIPAL_ID	No	
ERROR_LINE	No	
ERROR_MESSAGE	No	
ERROR_NUMBER	No	
ERROR_PROCEDURE	No	
ERROR_SEVERITY	No	
ERROR_STATE	No	
FORMATMESSAGE	No	
HOST_ID	No	
HOST_NAME	No	
IIF	No	
ISNULL	Yes	
ISNUMERIC	No	
Logical Functions	Yes	
NEWID	Yes	
ORIGINAL_LOGIN	No	
Other	Yes	
PRINT	Yes	
SCHEMA_NAME	No	
Security Functions	Yes	
SESSION_USER	No	
System Functions	Yes	

Service Broker

Clause	Automatically Converted	Details
ALTER QUEUE	No	
ALTER SERVICE	No	
BEGIN CONVERSATION TIMER	No	
BEGIN DIALOG CONVERSATION	No	
CREATE QUEUE	No	
CREATE SERVICE	No	
DROP QUEUE	No	
DROP SERVICE	No	
END CONVERSATION	No	
GET CONVERSATION GROUP	No	
GET TRANSMISSION_STATUS	No	
MOVE CONVERSATION	No	
Queue management	Yes	
RECEIVE	No	
SEND	No	
Service Broker Catalog Views	Yes	
Service Broker Related Dynamic Management Views	Yes	
Service Broker Statements	Yes	
Service management	Yes	
sys.conversation_endpoints	No	
sys.conversation_groups	No	
sys.conversation_priorities	No	
sys.dm_broker_activated_tasks	No	
sys.dm_broker_connections	No	
sys.dm_broker_forwarded_messages	No	

Clause	Automatically Converted	Details
sys.dm_broker_queue_monitors	No	
sys.message_type_xml_schemas	No	collection_usages
sys.remote_service_bindings	No	
sys.routes	No	
sys.service_contract_message_usages	No	
sys.service_contract_usages	No	
sys.service_contracts	No	
sys.service_message_types	No	
sys.service_queue_usages	No	
sys.service_queues	No	
sys.services	No	
sys.transmission_queue	No	

SQL Server Agent

Clause	Automatically Converted	Details
sp_add_alert	No	
sp_add_category	No	
sp_add_job	No	
sp_add_jobschedule	No	
sp_add_jobserver	No	
sp_add_jobstep	No	
sp_add_notification	No	
sp_add_operator	No	
sp_add_proxy	No	
sp_add_schedule	No	
sp_add_targetservergroup	No	
sp_add_targetsvrgrp_member	No	
sp_apply_job_to_targets	No	
sp_attach_schedule	No	

Clause	Automatically Converted	Details
sp_cycle_agent_errorlog	No	
sp_cycle_errorlog	No	
sp_delete_alert	No	
sp_delete_category	No	
sp_delete_job	No	
sp_delete_jobschedule	No	
sp_delete_jobserver	No	
sp_delete_jobstep	No	
sp_delete_jobsteplog	No	
sp_delete_notification	No	
sp_delete_operator	No	
sp_delete_proxy	No	
sp_delete_schedule	No	
sp_delete_targetserver	No	
sp_delete_targetservergroup	No	
sp_delete_targetsvrgrp_member	No	
sp_detach_schedule	No	
sp_enum_login_for_proxy	No	
sp_enum_proxy_for_subsystem	No	
sp_enum_sqlagent_subsystem	No	
sp_grant_login_to_proxy	No	
sp_grant_proxy_to_subsystem	No	
sp_help_alert	No	
sp_help_category	No	
sp_help_downloadlist	No	
sp_help_job	No	
sp_help_jobactivity	No	
sp_help_jobcount	No	
sp_help_jobhistory	No	
sp_help_jobs_in_schedule	No	

Clause	Automatically Converted	Details
sp_help_jobschedule	No	
sp_help_jobserver	No	
sp_help_jobstep	No	
sp_help_jobsteplog	No	
sp_help_notification	No	
sp_help_operator	No	
sp_help_proxy	No	
sp_help_schedule	No	
sp_help_targetserver	No	
sp_help_targetservergroup	No	
sp_manage_jobs_by_login	No	
sp_msx_defect	No	
sp_msx_enlist	No	
sp_msx_get_account	No	
sp_msx_set_account	No	
sp_notify_operator	No	
sp_post_msx_operation	No	
sp_purge_jobhistory	No	
sp_remove_job_from_targetss	No	
sp_resync_targetserver	No	
sp_revoke_login_from_proxy	No	
sp_revoke_proxy_from_subsystem	No	
sp_start_job	No	
sp_stop_job	No	
sp_update_alert	No	
sp_update_category	No	
sp_update_job	No	
sp_update_jobschedule	No	
sp_update_jobstep	No	
sp_update_notification	No	

Clause	Automatically Converted	Details
sp_update_operator	No	
sp_update_proxy	No	
sp_update_schedule	No	
sp_update_targetservergroup	No	

SQL Server Backup

Clause	Automatically Converted	Details
BACKUP	No	
BACKUP CERTIFICATE	No	
BACKUP MASTER KEY	No	
BACKUP SERVICE MASTER KEY	No	
RESTORE	No	
RESTORE FILELISTONLY	No	
RESTORE HEADERONLY	No	
RESTORE LABELONLY	No	
RESTORE MASTER KEY	No	
RESTORE REWINDONLY	No	
RESTORE SERVICE MASTER KEY	No	
RESTORE VERIFYONLY	No	

T-SQL

BACKUP and RESTORE

Clause	Automatically Converted	Details
BACKUP and RESTORE Statements	No	

Collation

Clause	Automatically Converted	Details
Collation	No	

Control-of-Flow Language

Clause	Automatically Converted	Details
BEGIN...END	Yes	
BREAK	Yes	
CONTINUE	Yes	
GOTO	No	
IF...ELSE	Yes	
RETURN	Yes	
THROW	Yes	
TRY...CATCH	Yes	
WAITFOR	Partial	
WHILE	Yes	

Cursors

DECLARE CURSOR

Clause	Automatically Converted	Details
DECLARE CURSOR	Yes	
CURSOR with option GLOBAL	No	
CURSOR with options FORWARD_ONLY or SCROLL	No	

FETCH

Clause	Automatically Converted	Details
FETCH	Yes	

OPEN

Clause	Automatically Converted	Details
OPEN	Yes	

DUMP and LOAD Statements

Clause	Automatically Converted	Details
DUMP and LOAD Statements	No	

Microsoft SQL Server to PostgreSQL Conversion Issue Reference

Use the following sections to get information on the types of issues you might encounter during a Microsoft SQL Server to PostgreSQL conversion. Choose the link in the **Issue** column to get more detailed resolution information about the issue if it is available.

Arithmetic Operators

Item	Issue	Resolution
Arithmetic operators	Issue 7775: Check the data type conversion for possible loss of accuracy (p. 200)	Cast values as the intended type.
Arithmetic operators and date types	Issue 7773: Unable to perform an automated migration of arithmetic operations with dates (p. 200)	Cast values as the intended type.
Arithmetic operators and mixed types	Issue 7774: Unable to perform an automated migration of the arithmetic operations with mixed types of operands (p. 200)	Cast values as the intended type.

Built-In Services

Item	Issue	Resolution
Database Mail	Issue 7900: PostgreSQL does not have functionality similar to SQL Server Database Mail (p. 201)	Try using the Amazon Simple Notification Service (Amazon SNS).
Service Broker	Issue 7901: PostgreSQL does not have functionality similar to SQL Server Service Broker (p. 201)	Try using the Amazon Simple Queue Service (Amazon SQS).
SQL Server Agent	Issue 7902: PostgreSQL does not have functionality similar to SQL Server Agent (p. 201)	Try using the AWS Lambda service with Scheduled Events.
SQL Server Backup	Issue 7903: PostgreSQL does not have functionality similar to SQL Server Backup (p. 201)	Try using the Amazon Glacier storage service.

Built-In SQL Functions

Item	Issue	Resolution
SOUNDEX	Issue 7811: PostgreSQL doesn't support the SOUNDEX function (p. 201)	Create a user-defined function.

CONTROL FLOW

Item	Issue	Resolution
Cursors	Issue 7801: The table can be locked open cursor (p. 201)	Review your transformed code and modify it if necessary.
DECLARE / DEFAULT VALUE	Issue 7826: Check the default value for a DateTime variable (p. 202)	Check the default value for a DateTime variable.
GOTO	Issue 7628: PostgreSQL doesn't support the GOTO option. Automatic conversion can't be performed. (p. 202)	Revise your code to eliminate GOTO operators, using BEGIN...END blocks in combination with EXIT, REPEAT, UNTIL, and WHILE operators.
Procedures	Issue 7802: A table that is created within the procedure, must be deleted before the end of the procedure (p. 202)	Review your transformed code and modify it if necessary.
Result sets	Issue 7800: PostgreSQL doesn't support result sets in the style of MSSQL (p. 201)	Review your transformed code and modify it if necessary.

Item	Issue	Resolution
WAITFOR DELAY	Issue 7821: Automatic conversion operator WAITFOR with a variable is not supported (p. 202)	Perform a manual conversion.
WAITFOR TIME	Issue 7691: PostgreSQL doesn't support WAITFOR TIME feature (p. 202)	Perform a manual conversion.

CREATE

Item	Issue	Resolution
	Issue 7696: Unable convert object due to %s not created (p. 202)	Review the %s object.
Encrypted objects	Issue 7813: Encrypted objects (p. 202)	Decrypt the object before conversion.

CURSORS

Item	Issue	Resolution
CURSOR with option GLOBAL	Issue 7637: PostgreSQL doesn't support GLOBAL CURSORS. Requires manual conversion. (p. 203)	Change the global cursor to a local cursor, or revise your code so it doesn't require global cursors.
CURSOR with options STATIC or KEYSET or DYNAMIC or FAST_FORWARD	Issue 7639: PostgreSQL doesn't support DYNAMIC cursors (p. 203)	Revise your code to eliminate dynamic cursors.
FAST_FORWARD option	Issue 7701: Setting this option corresponds to the typical behavior of cursors in PostgreSQL, so this option is skipped (p. 203)	Use cursors without this option.
FOR UPDATE	Issue 7803: PostgreSQL doesn't support the option FOR UPDATE, so this option is skipped (p. 203)	Review your transformed code and modify it if necessary.
KEYSET option	Issue 7700: The membership and order of rows never changes for cursors in PostgreSQL, so this option is skipped (p. 203)	Use cursors without this option.
OPTIMISTIC option	Issue 7704: PostgreSQL doesn't support the option OPTIMISTIC, so this option is skipped (p. 203)	Use cursors without this option.
READ_ONLY option	Issue 7702: All PostgreSQL cursors are read-only, so this option is skipped (p. 203)	Use cursors without this option.
TYPE_WARNING option	Issue 7705: PostgreSQL doesn't support the option TYPE_WARNING, so this option is skipped (p. 203)	Use cursors without this option.

DATA TYPES

Item	Issue	Resolution
BYTEA	Issue 7818: PostgreSQL doesn't support arithmetic operations with binary data types (p. 203)	Perform a manual conversion.
geography	Issue 7662: PostgreSQL doesn't support this type. A manual conversion is required. (p. 204)	To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.
geometry	Issue 7664: PostgreSQL doesn't support this type. A manual conversion is required. (p. 204)	To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.
hierarchyid	Issue 7657: PostgreSQL doesn't support this type. A manual conversion is required. (p. 204)	To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.
rowversion	Issue 7706: PostgreSQL doesn't support this type (p. 204)	Perform a manual conversion.
sql_variant	Issue 7658: PostgreSQL doesn't support this type. A manual conversion is required. (p. 204)	To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.
table	Issue 7659: The scope table-variables and temporary tables is different. You must apply manual conversion, if you are using recursion. (p. 204)	Perform a manual conversion.
UDT	Issue 7690: PostgreSQL doesn't support table types (p. 204)	Perform a manual conversion.
XML	Issue 7816: PostgreSQL doesn't support any methods for data type XML (p. 205)	Perform a manual conversion.
XML	Issue 7817: PostgreSQL doesn't support option [for xml path] in the SQL queries (p. 205)	Perform a manual conversion.

DDL

Item	Issue	Resolution
CREATE INDEX	Issue 7675: PostgreSQL doesn't support sorting options (ASC DESC) for constraints (p. 205)	Use indexes without this option.
CREATE INDEX	Issue 7681: PostgreSQL doesn't support clustered indexes (p. 205)	Use nonclustered indexes.

AWS Schema Conversion Tool User Guide
Microsoft SQL Server to PostgreSQL
Conversion Issue Reference

Item	Issue	Resolution
CREATE INDEX	Issue 7682: PostgreSQL doesn't support the INCLUDE option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7781: PostgreSQL doesn't support the PAD_INDEX option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7782: PostgreSQL doesn't support the SORT_IN_TEMPDB option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7783: PostgreSQL doesn't support the IGNORE_DUP_KEY option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7784: PostgreSQL doesn't support the STATISTICS_NORECOMPUTE option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7785: PostgreSQL doesn't support the STATISTICS_INCREMENTAL option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7786: PostgreSQL doesn't support the DROP_EXISTING option in indexes (p. 205)	Use index without this option.
CREATE INDEX	Issue 7787: PostgreSQL doesn't support the ONLINE option in indexes (p. 206)	Use index without this option.
CREATE INDEX	Issue 7788: PostgreSQL doesn't support the ALLOW_ROW_LOCKS option in indexes (p. 206)	Use index without this option.
CREATE INDEX	Issue 7789: PostgreSQL doesn't support the ALLOW_PAGE_LOCKS option in indexes (p. 206)	Use index without this option.
CREATE INDEX	Issue 7790: PostgreSQL doesn't support the MAXDOP option in indexes (p. 206)	Use index without this option.
CREATE INDEX	Issue 7791: PostgreSQL doesn't support the DATA_COMPRESSION option in indexes (p. 206)	Use index without this option.
CREATE TABLE	Issue 7679: PostgreSQL doesn't support computed columns (p. 206)	Try using a trigger.
CREATE TABLE	Issue 7680: PostgreSQL doesn't support global temporary tables (p. 206)	Use local temporary or regular tables.
CREATE TABLE	Issue 7812: Temporary table must be removed before the end of the function (p. 206)	Review your transformed code and modify it if necessary.
CREATE TABLE	Issue 7825: The default value for a DateTime column removed (p. 206)	Review generated code and modify it if necessary.
Inline Function	Issue 7776: Automatic migration of inline functions not supported (p. 206)	Perform a manual conversion.

DELETE

Item	Issue	Resolution
DELETE with Table Hint Limited	Issue 7623: PostgreSQL doesn't support hints in DELETE statements. The conversion skips options in the format WITH(Table_Hint_Limited). (p. 206)	Use PostgreSQL methods for performance tuning.
TOP	Issue 7798: PostgreSQL doesn't support TOP option in the operator DELETE (p. 207)	Perform a manual conversion.

DML, FROM

Item	Issue	Resolution
Table Hints	Issue 7823: PostgreSQL doesn't support table hints in DML statements (p. 207)	Use PostgreSQL methods of performance tuning.

EXECUTE

Item	Issue	Resolution
	Issue 7695: PostgreSQL doesn't support the execution of a procedure as a variable (p. 207)	Perform a manual conversion using dynamic SQL.
EXECUTE a character string	Issue 7672: Automatic conversion of this command is not supported (p. 207)	Perform a manual conversion.
Execute a pass-through command against a linked server	Issue 7645: PostgreSQL doesn't support executing a pass-through command on a linked server (p. 208)	Use other methods to execute statements on a remote server.
EXECUTE with execute options [RECOMPILE]	Issue 7640: The EXECUTE with RECOMPILE option is ignored (p. 208)	Use EXECUTE command without this option.
EXECUTE with execute options [RESULT SETS (<result_sets_definition>)]	Issue 7643: The EXECUTE with RESULT SETS <result set definition> option is ignored (p. 208)	Use EXECUTE command without this option.
EXECUTE with execute options [RESULT SETS NONE]	Issue 7642: The EXECUTE with RESULT SETS NONE option is ignored (p. 208)	Use EXECUTE command without this option.
EXECUTE with execute options [RESULT SETS UNDEFINED]	Issue 7641: The EXECUTE with RESULT SETS UNDEFINED option is ignored (p. 208)	Use EXECUTE command without this option.

INSERT

Item	Issue	Resolution
EXECUTE	Issue 7819: PostgreSQL doesn't support INSERT...EXECUTE statements (p. 208)	Perform a manual conversion.
TOP	Issue 7799: PostgreSQL doesn't support TOP option in the operator INSERT (p. 208)	Perform a manual conversion.

Operators

Item	Issue	Resolution
Bitwise Operators	Issue 7804: PostgreSQL doesn't support operator [Bitwise exclusive OR] (p. 209)	Perform a manual conversion.
Comparison Operators	Issue 7805: PostgreSQL doesn't support operator [Not less than] (p. 209)	Perform a manual conversion.
Comparison Operators	Issue 7806: PostgreSQL doesn't support operator [Not greater than] (p. 209)	Perform a manual conversion.

Parser Error

Item	Issue	Resolution
Parser Error	Issue 7663: Unable to resolve the object (p. 209)	Verify if object <object name> is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

SELECT

Item	Issue	Resolution
GROUP BY CUBE	Issue 7653: PostgreSQL doesn't support the option GROUP BY ROLLUP. Automatic conversion can't be performed. (p. 209)	Try creating a stored procedure to replace the query.
GROUP BY CUBE	Issue 7654: PostgreSQL doesn't support the option GROUP BY CUBE. Automatic conversion can't be performed. (p. 210)	Try creating a stored procedure to replace the query.
Group By Grouping Sets	Issue 7655: PostgreSQL doesn't support the option GROUP BY GROUPING SETS. Automatic conversion can't be performed. (p. 210)	Try creating a stored procedure to replace the query.

Item	Issue	Resolution
ORDER BY Specifying a collation	Issue 7646: PostgreSQL doesn't support the COLLATE option in the ORDER BY clause. Automatic conversion can't be performed. (p. 210)	You must use the COLLATION settings that were assigned when the database was created.
search condition	Issue 7687: PostgreSQL doesn't support the CONTAINS predicate (p. 210)	Perform a manual conversion.
search condition	Issue 7688: PostgreSQL doesn't support the FREETEXT predicate (p. 210)	Perform a manual conversion.
search condition	Issue 7795: PostgreSQL is case sensitive. Check the string comparison. (p. 210)	Check the string comparison.
TOP WITH TIES	Issue 7605: PostgreSQL doesn't support the WITH TIES option (p. 210)	Perform a manual conversion.

SEQUENCE

Item	Issue	Resolution
Specific data type	Issue 7793: PostgreSQL doesn't support changing data type in sequences (p. 210)	Review your transformed code and modify it if necessary.

SYNONYMS

Item	Issue	Resolution
CREATE	Issue 7792: PostgreSQL doesn't support synonyms (p. 211)	Replace a synonym with original database object. If original object is table or view you can try to create a view in one database that relies on a table or view in another database.

SYS objects

Item	Issue	Resolution
SYS objects	Issue 7904: Unable to convert system object <object name> (p. 211)	Perform a manual conversion.

TRANSACTION

Item	Issue	Resolution
Transaction	Issue 7807: Transactions in functions PostgreSQL doesn't support (p. 211)	Perform a manual conversion.

TRIGGERS

Item	Issue	Resolution
Triggers	Issue 7809: In PostgreSQL there are no analogues of the tables 'inserted' and 'deleted' (p. 211)	Perform a manual conversion.

Unknown

Item	Issue	Resolution
	Issue 7627: This syntactic element conversion is not supported yet (p. 212)	Perform a manual conversion.
Unknown Clause	Issue 7674: Automatic conversion of <clause name> clause of <statement name> statement is not supported (p. 213)	Perform a manual conversion.
Unparsed SQL	Issue 7808: Unparsed SQL (p. 213)	Perform a manual conversion.

UPDATE

Item	Issue	Resolution
TOP	Issue 7796: PostgreSQL doesn't support TOP option in the operator UPDATE (p. 213)	Perform a manual conversion.
UPDATE	Issue 7797: PostgreSQL doesn't have analog for option "OUTPUT" with using the implicit table "deleted" in the operator for UPDATE (p. 213)	This case requires manual conversion.
Variable assignment	Issue 7829: Unable to convert variable assignment by UPDATE statement (p. 214)	Perform a manual conversion.

User Types

Item	Issue	Resolution
User-Defined Data Types	Issue 7794: PostgreSQL doesn't support user-defined data types (p. 215)	Perform a manual conversion.

Conversion Issues with Arithmetic Operators

Issue 7775: Check the data type conversion for possible loss of accuracy

To avoid loss of accuracy for types in arithmetic operations, cast values as the intended type, as in the example following.

```
BEGIN
  SELECT
    5.1E+1::REAL
    INTO lBinaryFloat;
  SELECT
    - 5.2E-1::DOUBLE PRECISION
    INTO lBinaryDouble;

  var1 := (var_varchar1||var_varchar2)::INTEGER;
  var2 := (var_varchar1||var_varchar3)::NUMERIC;

  var3 := var_varchar1::NUMERIC + var_numeric2;
  var4 := var_timestamp2 + (var_numeric1::NUMERIC || 'day')::INTERVAL;
END;
```

Issue 7773: Unable to perform an automated migration of arithmetic operations with dates

The Schema Conversion Tool cannot automatically migrate arithmetic operations that use date types. Cast values as the intended type, as in the example following.

```
BEGIN
  var3 := var_varchar1::NUMERIC + var_numeric2;
  var4 := var_timestamp2 + (var_numeric1::NUMERIC || 'day')::INTERVAL;
END;
```

Issue 7774: Unable to perform an automated migration of the arithmetic operations with mixed types of operands

The Schema Conversion Tool cannot automatically migrate arithmetic operations that mix types. For example, multiplying a variable of type FLOAT with a variable of type INT. Cast values as the intended type before performing any arithmetic operations, as in the example following.

```
BEGIN
  var1 := (var_varchar1||var_varchar2)::INTEGER;
  var2 := (var_varchar1||var_varchar3)::NUMERIC;
END;
```

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with BUILT-IN SERVICES

Issue 7900: PostgreSQL does not have functionality similar to SQL Server Database Mail

Try using the Amazon Simple Queue Service (Amazon SQS). For more information, see [What is Amazon Simple Queue Service?](#)

Issue 7901: PostgreSQL does not have functionality similar to SQL Server Service Broker

Try using the Amazon Simple Queue Service (SQS). For more information, see [What is Amazon Simple Queue Service?](#) in the Amazon SQS documentation.

Issue 7902: PostgreSQL does not have functionality similar to SQL Server Agent

Try using the AWS Lambda service with Scheduled Events. For more information, see [What Is AWS Lambda?](#) in the *AWS Lambda Developer Guide*.

Issue 7903: PostgreSQL does not have functionality similar to SQL Server Backup

Try using the Amazon Glacier storage service. For more information, see [What Is Amazon Glacier?](#) in the *Amazon Glacier Developer Guide*.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with BUILT-IN SQL FUNCTIONS

Issue 7811: PostgreSQL doesn't support the SOUNDEX function

Create a user-defined function.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CONTROL FLOW

Issue 7800: PostgreSQL doesn't support result sets in the style of MSSQL

Review your transformed code and modify it if necessary.

Issue 7801: The table can be locked open cursor

In Microsoft SQL Server, a local temporary table created in a stored procedure is dropped automatically when the stored procedure completes. Temporary tables in PostgreSQL are automatically dropped at the

end of a session. The conversion uses the DROP TABLE IF EXISTS statement to emulate this behavior and remove the temporary table before the end of the function that created it. In some cases though, this statement might fail because the temporary table is locked due to an open cursor. In this case, the DROP TABLE statement must be commented out. Review your transformed code and modify it if necessary.

Issue 7802: A table that is created within the procedure, must be deleted before the end of the procedure

Review your transformed code and modify it if necessary.

Issue 7826: Check the default value for a DateTime variable

Check the default value for a DateTime variable.

Issue 7628: PostgreSQL doesn't support the GOTO option. Automatic conversion can't be performed.

Revise your code to eliminate GOTO operators, using BEGIN...END blocks in combination with EXIT, REPEAT, UNTIL, and WHILE operators.

Issue 7821: Automatic conversion operator WAITFOR with a variable is not supported

You must perform a manual conversion in cases where you have a WAITFOR DELAY statement that uses a variable, for example `WAITFOR DELAY @time;`, instead of a literal value, for example `WAITFOR DELAY '00:00:05';`.

Issue 7691: PostgreSQL doesn't support WAITFOR TIME feature

Perform a manual conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE

Issue 7696: Unable convert object due to %s not created

Review the <object name> object.

Issue 7813: Encrypted objects

Decrypt the object before conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CURSORS

Issue 7637: PostgreSQL doesn't support GLOBAL CURSORS. Requires manual conversion.

Change the global cursor to a local cursor, or revise your code so it doesn't require global cursors.

Issue 7639: PostgreSQL doesn't support DYNAMIC cursors

Revise your code to remove the DYNAMIC option from any cursors.

Issue 7701: Setting this option corresponds to the typical behavior of cursors in PostgreSQL, so this option is skipped

The FAST_FORWARD option is skipped during conversion, because PostgreSQL cursors provide the same behavior as the FAST_FORWARD option by default. Review your transformed code and modify it if necessary.

Issue 7803: PostgreSQL doesn't support the option FOR UPDATE, so this option is skipped

The FOR_UPDATE option is skipped during conversion because PostgreSQL doesn't support it. Review your transformed code and modify it if necessary.

Issue 7700: The membership and order of rows never changes for cursors in PostgreSQL, so this option is skipped

Use cursors without this option.

Issue 7704: PostgreSQL doesn't support the option OPTIMISTIC, so this option is skipped

Use cursors without this option.

Issue 7702: All PostgreSQL cursors are read-only, so this option is skipped

Use cursors without this option.

Issue 7705: PostgreSQL doesn't support the option TYPE_WARNING, so this option is skipped

Use cursors without this option.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DATA TYPES

Issue 7818: PostgreSQL doesn't support arithmetic operations with binary data types

Perform a manual conversion.

Issue 7662: PostgreSQL doesn't support this type. A manual conversion is required.

PostgreSQL doesn't support the geography data type. To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

Issue 7664: PostgreSQL doesn't support this type. A manual conversion is required.

PostgreSQL doesn't support the geometry data type. To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

Issue 7657: PostgreSQL doesn't support this type. A manual conversion is required.

PostgreSQL doesn't support the hierarchyid data type. To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

Issue 7706: PostgreSQL doesn't support this type

PostgreSQL doesn't support the rowversion data type. To store data of this type in PostgreSQL, use a PostgreSQL-compatible type.

Issue 7658: PostgreSQL doesn't support this type. A manual conversion is required.

PostgreSQL doesn't support the sql_variant data type. To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

Issue 7659: The scope table-variables and temporary tables is different. You must apply manual conversion, if you are using recursion.

Perform a manual conversion.

Issue 7690: PostgreSQL doesn't support table types

PostgreSQL doesn't support table parameters in CREATE PROCEDURE statements, as in the example following.

```
CREATE TYPE Employee2Client AS TABLE(  
    EmployeeID int,  
    EmployeeName varchar(160),  
    ClientID int,  
    ClientName varchar(160)  
)  
go  
  
CREATE PROCEDURE [dbo].[PROC_CREATE_PROC_005]  
    @p_E2C Employee2Client READONLY  
AS  
BEGIN  
    select * from @p_E2C where upper(ClientName) like concat('%', upper('bank'), '%') ;
```

Convert to temporary tables or row variables (for example, name table_name%ROWTYPE;).

Issue 7816: PostgreSQL doesn't support any methods for data type XML

Microsoft SQL Server provides methods for the xml data type that you can use to read and manipulate XML data, but PostgreSQL's xml data type does not offer parallel functionality. Perform a manual conversion to use PostgreSQL XML handling functionality instead.

Issue 7817: PostgreSQL doesn't support option [for xml path] in the SQL queries

PostgreSQL doesn't support using the FOR XML clause in PATH mode for DML statements. Perform a manual conversion to use PostgreSQL XML handling functionality instead.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DDL

Issue 7675: PostgreSQL doesn't support sorting options (ASC | DESC) for constraints

Remove these options from indexes and constraints.

Issue 7681: PostgreSQL doesn't support clustered indexes

Use nonclustered indexes.

Issue 7682: PostgreSQL doesn't support the INCLUDE option in indexes

Use the index without this option.

Issue 7781: PostgreSQL doesn't support the PAD_INDEX option in indexes

Use the index without this option.

Issue 7782: PostgreSQL doesn't support the SORT_IN_TEMPDB option in indexes

Use the index without this option.

Issue 7783: PostgreSQL doesn't support the IGNORE_DUP_KEY option in indexes

Use the index without this option.

Issue 7784: PostgreSQL doesn't support the STATISTICS_NORECOMPUTE option in indexes

Use the index without this option.

Issue 7785: PostgreSQL doesn't support the STATISTICS_INCREMENTAL option in indexes

Use the index without this option.

Issue 7786: PostgreSQL doesn't support the DROP_EXISTING option in indexes

Use the index without this option.

Issue 7787: PostgreSQL doesn't support the ONLINE option in indexes

Use the index without this option.

Issue 7788: PostgreSQL doesn't support the ALLOW_ROW_LOCKS option in indexes

Use index without this option.

Issue 7789: PostgreSQL doesn't support the ALLOW_PAGE_LOCKS option in indexes

Use index without this option.

Issue 7790: PostgreSQL doesn't support the MAXDOP option in indexes

Use index without this option.

Issue 7791: PostgreSQL doesn't support the DATA_COMPRESSION option in indexes

Use index without this option.

Issue 7679: PostgreSQL doesn't support computed columns

Try using a trigger instead.

Issue 7680: PostgreSQL doesn't support global temporary tables

Use local temporary or regular tables.

Issue 7812: Temporary table must be removed before the end of the function

Review your transformed code and modify it if necessary.

Issue 7825: The default value for a DateTime column removed

Review generated code and modify it if necessary.

Issue 7776: Automatic migration of inline functions not supported

Perform a manual conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DELETE

Issue 7623: PostgreSQL doesn't support hints in DELETE statements. The conversion skips options in the format WITH(Table_Hint_Limited).

PostgreSQL doesn't support using hints (for example, WITH FORCESCAN) to override the default behavior of the query optimizer when executing Data Manipulation Language (DML) statements. Use PostgreSQL methods for performance tuning instead.

Issue 7798: PostgreSQL doesn't support TOP option in the operator DELETE

Perform a manual conversion for any DELETE statements that contain a TOP argument, as in the example following.

```
delete top (<expression >)[PERCENT] from <table-name>;
```

In PostgreSQL, you can use a subquery in the WHERE clause to identify the rows you want to delete.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DML, FROM

Issue 7823: PostgreSQL doesn't support table hints in DML statements

PostgreSQL doesn't support using hints (for example, WITH FORCESCAN) to override the default behavior of the query optimizer when executing Data Manipulation Language (DML) statements. Use PostgreSQL methods of performance tuning instead.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with EXECUTE

Issue 7695: PostgreSQL doesn't support the execution of a procedure as a variable

Perform a manual conversion (one possibility is to use dynamic SQL). This issue occurs when you pass in the name of a procedure to an EXECUTE statement by using a variable, as in the example following.

```
ALTER PROCEDURE [dbo].[PROC_EXECUTE_009]
AS
  SET NOCOUNT ON
BEGIN
  declare @v_ID int;
  declare @v_ProcName varchar(20) = '[dbo].[PROC_DML_015]';
  exec @v_ProcName
  @v_ID output
```

Issue 7672: Automatic conversion of this command is not supported

This issue is often caused by an EXECUTE statement that executes a string or variable, for example EXEC('select * from Customer'); or EXEC(@variable);. Convert each of these statements to a pair of PREPARE and EXECUTE statements in PostgreSQL.

Issue 7645: PostgreSQL doesn't support executing a pass-through command on a linked server

Use other methods to execute statements on a remote server.

Issue 7640: The EXECUTE with RECOMPILE option is ignored

Use the EXECUTE command without this option.

Issue 7643: The EXECUTE with RESULT SETS <result set definition> option is ignored

Use the EXECUTE command without this option.

Issue 7642: The EXECUTE with RESULT SETS NONE option is ignored

Use the EXECUTE command without this option.

Issue 7641: The EXECUTE with RESULT SETS UNDEFINED option is ignored

Use the EXECUTE command without this option.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with INSERT

Issue 7819: PostgreSQL doesn't support INSERT...EXECUTE statements

Perform a manual conversion. PostgreSQL doesn't support using EXECUTE within an INSERT statement to insert the results of a stored procedure into a table.

Issue 7799: PostgreSQL doesn't support TOP option in the operator INSERT

Perform a manual conversion for any INSERT statements that contain a TOP argument, as in the example following.

```
insert TOP(2) percent into Customer(  
  id  
  ,sname  
  ,name  
  ,typeid  
  ,taxcode  
  ,stateid  
  ,opendate  
  ,closedate  
)select  
  id+1000  
  ,name  
  ,name  
  ,4  
  ,61+id  
  ,1  
  ,getdate()  
  ,null  
from
```

```
employees e
where
name not in(select name from Customer)
and exists(select 1 from Employees where ManagerId = e.id)
```

In PostgreSQL, you can use the `row_number()` [window function](#) in the FROM clause or a subquery in the WHERE clause of the SELECT statement you use to select the data to be inserted to identify the rows you want to insert.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Operators

Issue 7804: PostgreSQL doesn't support operator [Bitwise exclusive OR]

Perform a manual conversion.

Issue 7805: PostgreSQL doesn't support operator [Not less than]

Perform a manual conversion.

Issue 7806: PostgreSQL doesn't support operator [Not greater than]

Perform a manual conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Parser Error

Issue 7663: Unable to resolve the object

Verify if object <object name> is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT

Issue 7653: PostgreSQL doesn't support the option GROUP BY ROLLUP. Automatic conversion can't be performed.

Try creating a stored procedure to replace the query.

Issue 7654: PostgreSQL doesn't support the option GROUP BY CUBE. Automatic conversion can't be performed.

Try creating a stored procedure to replace the query.

Issue 7655: PostgreSQL doesn't support the option GROUP BY GROUPING SETS. Automatic conversion can't be performed.

Try creating a stored procedure to replace the query.

Issue 7646: PostgreSQL doesn't support the COLLATE option in the ORDER BY clause. Automatic conversion can't be performed.

You must use the collation settings that were assigned when the database was created.

Issue 7687: PostgreSQL doesn't support the CONTAINS predicate

Perform a manual conversion by re-writing the SELECT statement to use PostgreSQL full-text search functionality instead. For more information about full-text search in PostgreSQL, see [Tables and Indexes](#) in the PostgreSQL documentation.

Issue 7688: PostgreSQL doesn't support the FREETEXT predicate

Perform a manual conversion by re-writing the SELECT statement to use PostgreSQL full-text search functionality instead. For more information about full-text search in PostgreSQL, see [Tables and Indexes](#) in the PostgreSQL documentation.

Issue 7795: PostgreSQL is case sensitive. Check the string comparison.

Check string comparisons in the WHERE clauses of statements, for example `WHERE [Name] LIKE '%BLOCKED%'`; . All strings used in comparisons must be lowercase.

Issue 7605: PostgreSQL doesn't support the WITH TIES option

Perform a manual conversion of any TOP statements that use the WITH TIES clause. You can use the `rank()` [window function](#) in PostgreSQL to provide similar results.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SEQUENCE

Issue 7793: PostgreSQL doesn't support changing data type in sequences

PostgreSQL doesn't support setting a sequence's data type by using the AS clause of the CREATE SEQUENCE statement. Review your transformed code and modify it if necessary.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SYNONYMS

Issue 7792: PostgreSQL doesn't support synonyms

PostgreSQL doesn't support the CREATE SYNONYM statement. To convert, replace the synonym in any statement with the original database object. If the original object is a table or view, you can try to create a view in one database that relies on a table or view in another database.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SYS objects

Issue 7904: Unable to convert system object <object name>

Perform a manual conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with TRANSACTION

Issue 7807: Transactions in functions PostgreSQL doesn't support

Perform a manual conversion to correct this issue.

PostgreSQL supports local transactions within a given client session. You must manually convert Microsoft SQL Server transactions to use the PostgreSQL transaction management statements instead. PostgreSQL doesn't support named or distributed transactions, marking transactions, or transaction management within functions. For more information about using transactions in PostgreSQL, see [TRANSACTIONS](#) in the PostgreSQL documentation.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with TRIGGERS

Issue 7809: In PostgreSQL there are no analogues of the tables 'inserted' and 'deleted'

PostgreSQL doesn't support the inserted or deleted virtual tables. To manually convert a trigger that uses these tables from Microsoft SQL Server to PostgreSQL, perform the following steps:

1. Create a trigger function. The trigger function must be declared as a function taking no arguments and returning type trigger. The returned value must be NEW, OLD or NULL, depending on the type of a trigger. The function's body must be PostgreSQL-supported SQL to replace the body of the Microsoft SQL Server trigger without using the virtual tables. The name of the function should be formed as `schema_name.fn_<%trigger_name%>()`.
2. Create a trigger with CREATE TRIGGER that executes the trigger function.

For example, consider the following Microsoft SQL Server trigger.

```
CREATE TRIGGER [dbo].[tr_customerStateExt_aiu]
  ON [dbo].[CustomerStateExt]
  AFTER INSERT, UPDATE
  AS
  BEGIN
    IF EXISTS (SELECT *
              FROM inserted
              WHERE LEN(Name) < 5)
    BEGIN
      RAISERROR ('The name must be longer than five characters.', 16, 1);
      ROLLBACK TRANSACTION;
      RETURN
    END;
  END
GO
```

Using the approach described, this trigger is converted into the following PostgreSQL trigger.

```
CREATE OR REPLACE FUNCTION mydb_dbo.fn_tr_customerStateExt_aiu()
  RETURNS trigger
  AS
  $BODY$
  BEGIN
  <SQL to write around use of the inserted and deleted virtual tables>
  END;
  $BODY$
  LANGUAGE plpgsql;

CREATE TRIGGER tr_customerStateExt_aiu
  AFTER INSERT OR UPDATE
  ON mydb_dbo.CustomerStateExt
  FOR EACH STATEMENT
  EXECUTE PROCEDURE mydb_dbo.fn_tr_customerStateExt_aiu();
```

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Unknown

Issue 7627: This syntactic element conversion is not supported yet

Perform a manual conversion.

Issue 7674: Automatic conversion of <clause name> clause of <statement name> statement is not supported

Perform a manual conversion to correct this issue.

A common cause of this issue is SET statements in Microsoft SQL Server that affect current session handling (a complete list of these SET statements is available at [SET Statements \(Transact-SQL\)](#)). These statements are not supported in PostgreSQL, so you need to perform a manual conversion to a PostgreSQL SET statement. For more information about using a SET statement in PostgreSQL, see [SET](#) in the PostgreSQL documentation.

Issue 7808: Unparsed SQL

Perform a manual conversion.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with UPDATE

Issue 7796: PostgreSQL doesn't support TOP option in the operator UPDATE

Perform a manual conversion for any UPDATE statements that contain a TOP argument, as in the example following.

```
update top(10)
  Account
set
  AccountBalance = AccountBalance - 1
where
  AccountBalance > 15723
;
```

or:

```
update top(5) percent
  Account
set
  AccountBalance = AccountBalance - 1
where
  AccountBalance > 15723
;
```

In PostgreSQL, you can use the `row_number()` [window function](#) in the FROM clause or a subquery in the WHERE clause of the UPDATE statement to identify the rows you want to modify.

Issue 7797: PostgreSQL doesn't have analog for option "OUTPUT" with using the implicit table "deleted" in the operator for UPDATE

Perform a manual conversion for any UPDATE statements that uses the OUTPUT clause and references the deleted virtual table, as in the example following.

```
update Customer set
```

```
name = 'Mister ' + name
output
deleted.id,
deleted.name,
inserted.id,
inserted.name
where
TypeID = 4
```

In PostgreSQL, use a subquery in the RETURNING clause to return the information about the deleted table. This approach requires the table to have a primary key field.

```
update mydb_dbo.Customer c set
name = 'Mister ' || name
where
TypeID = 4
returning
(select id from mydb_dbo.Customer where id = c.id)
,(select name from mydb_dbo.Customer where id = c.id)
,id
,name
```

If the primary key is composed of multiple fields, you must reference all of those fields in the subquery. For example, take the following statement in SQL Server where id1 and id2 are the primary key fields for the TestMultiPK table.

```
UPDATE [TestMultiPK]
SET [Description] = 'Test'
output
deleted.id1,
deleted.id2,
deleted.Description,
inserted.id1,
inserted.id2,
inserted.Description
where ID = 0
```

The PostgreSQL conversion is as follows.

```
UPDATE [TestMultiPK] t
SET [Description] = 'Test'
WHERE ID = 0
returning
select (id1 from TestMultiPK where id1 = t.id1 and id2 = t.id2)
, select (id2 from TestMultiPK where id1 = t.id1 and id2 = t.id2)
, select (Description from TestMultiPK where id1 = t.id1 and id2 = t.id2)
, id1
, id2
, Description
```

Issue 7829: Unable to convert variable assignment by UPDATE statement

Perform a manual conversion. This issue occurs when you use an UPDATE statement to change the value of a variable, as in the example following.

```
CREATE PROCEDURE [dbo].[PROC_DML_UPDATE_FROM_3]
AS
BEGIN
DECLARE @AccountBalance FLOAT = 400;
```

```
UPDATE bank
  SET @AccountBalance = 500
  FROM bank
  WHERE id = 50005;
END
```

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with User Types

Issue 7794: PostgreSQL doesn't support user-defined data types

To correct this issue, perform a manual conversion of each `sp_addtype` or `CREATE TYPE` statement in Microsoft SQL Server to a `CREATE TYPE` statement in PostgreSQL. For more information about creating a user-defined data type in PostgreSQL, see [CREATE TYPE](#) in the PostgreSQL documentation.

Related Topics

- [Microsoft SQL Server to PostgreSQL Conversion Issue Reference \(p. 191\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Related Topics

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

MySQL to PostgreSQL Supported Schema Conversion

The following sections list the schema elements from a MySQL database and whether they are supported for automatic conversion to PostgreSQL using the AWS Schema Conversion Tool.

DDL

Alter Statements

ALTER TABLE

Clause	Automatically Converted	Details
ADD CONSTRAINT	Yes	

Clause	Automatically Converted	Details
ADD INDEX	Yes	

Create Statements

CREATE INDEX

Clause	Automatically Converted	Details
INDEX	Yes	
UNIQUE index	Yes	

CREATE PROCEDURE

Clause	Automatically Converted	Details
PARAMETERS: IN, OUT, INOUT	Yes	
With any output parameters	Yes	
Without output parameters	Yes	

CREATE TABLE

Clause	Automatically Converted	Details
Regular tables	Yes	
Temporary tables	Yes	

CREATE TRIGGER

Clause	Automatically Converted	Details
FOR EACH ROW Triggers	Yes	
Trigger event (Insert / Update / Delete)	Yes	
Trigger time (Before / After)	Yes	

CREATE VIEW

Clause	Automatically Converted	Details
With ALGORITHM OPTION	Partial	Issue 8824: ALGORITHM option is not supported (p. 245)
With CHECK OPTION	Yes	

Naming Objects

Clause	Automatically Converted	Details
Schemas	Yes	
Tables, Columns	Yes	

DML

Clauses

FROM

Clause	Automatically Converted	Details
CROSS JOIN	Yes	
INNER JOIN	Yes	
join_condition: ...ON conditional_expr	Yes	
join_condition: ... USING (column_list)	Yes	
LEFT JOIN	Yes	
LEFT OUTER JOIN	Yes	
NATURAL LEFT JOIN	Yes	
NATURAL LEFT OUTER JOIN	Yes	
NATURAL RIGHT JOIN	Yes	
NATURAL RIGHT OUTER JOIN	Yes	
RIGHT JOIN	Yes	

Clause	Automatically Converted	Details
RIGHT OUTER JOIN	Yes	
STRAIGHT_JOIN	Yes	
Subqueries in the FROM Clause	Yes	
tables_list	Yes	

Statements

DELETE

Clause	Automatically Converted	Details
IGNORE	Partial	Issue 8831: PostgreSQL doesn't have an option similar to IGNORE for DML statements (p. 241)
LIMIT	Yes	
LOW_PRIORITY	Partial	Issue 8830: PostgreSQL doesn't have an option similar to LOW_PRIORITY for DML statements (p. 241)
Multiple-Table Syntax	Partial	Issue 8834: PostgreSQL can't delete from several tables at the same time (p. 242)
Multiple-Table Syntax with USING	Partial	Issue 8834: PostgreSQL can't delete from several tables at the same time (p. 242)
ORDER BY	Yes	
QUICK	Partial	Issue 8832: PostgreSQL doesn't have an option similar to QUICK for DML statements (p. 241)

INSERT

Clause	Automatically Converted	Details
IGNORE	Partial	Issue 8831: PostgreSQL doesn't have an option similar to IGNORE for DML statements (p. 241)
LOW_PRIORITY DELAYED HIGH_PRIORITY	Partial	Issue 8830: PostgreSQL doesn't have an option similar to LOW_PRIORITY for DML statements (p. 241)
ON DUPLICATE KEY UPDATE	No	Issue 8829: PostgreSQL doesn't have an analog of clause ON DUPLICATE KEY UPDATE (p. 241)
SELECT-statement for insert	Yes	
SET col1=expr1,..	Yes	

Clause	Automatically Converted	Details
VALUES(...) VALUE(...)	Yes	

SELECT

Clause	Automatically Converted	Details
DISTINCT	Yes	
GROUP BY ASC, DESC	Yes	
GROUP BY col_name,...	Yes	
GROUP BY expression	Yes	
GROUP BY mixed options	Yes	
GROUP BY position,...	Yes	
GROUP BY WITH ROLLUP	No	Issue 8653: PostgreSQL doesn't support the option GROUP BY ROLLUP (p. 244)
HAVING	Yes	
HIGH_PRIORITY	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
LIMIT offset, row_count	Yes	
LIMIT row_count OFFSET offset	Yes	
MAX_STATEMENT_TIME = N	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
ORDER BY col_name,...	Yes	
ORDER BY expression	Yes	
ORDER BY mixed options	Yes	
ORDER BY position,...	Yes	
SQL_BIG_RESULT	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
SQL_BUFFER_RESULT	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
SQL_CACHE	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
SQL_CALC_FOUND_ROWS	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)

Clause	Automatically Converted	Details
SQL_NO_CACHE	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
SQL_SMALL_RESULT	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
STRAIGHT_JOIN	Partial	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)
Select all columns (using * and aliases)	Yes	
Select all columns (using *)	Yes	
Select subset of the columns	Yes	
Select with calculations	Yes	
Select with column heading	Yes	
Select with constants	Yes	
Subquery as scalar operand	Yes	

UPDATE

Clause	Automatically Converted	Details
DEFAULT VALUES	Yes	
IGNORE	Partial	Issue 8831: PostgreSQL doesn't have an option similar to IGNORE for DML statements (p. 241)
LIMIT	Yes	
LOW_PRIORITY	Partial	Issue 8830: PostgreSQL doesn't have an option similar to LOW_PRIORITY for DML statements (p. 241)
Multiple-table syntax	Yes	
ORDER BY	Yes	

Functions

Aggregate

Clause	Automatically Converted	Details
AVG()	Yes	

Clause	Automatically Converted	Details
BIT_AND()	Yes	
BIT_OR()	Yes	
BIT_XOR()	No	
COUNT()	Yes	
COUNT(DISTINCT)	No	
GROUP_CONCAT()	Yes	
MAX()	Yes	
MIN()	Yes	
STD()	Yes	
STDDEV()	Yes	
STDDEV_POP()	Yes	
STDDEV_SAMP()	Yes	
SUM()	Yes	
VAR_POP()	Yes	
VAR_SAMP()	Yes	
VARIANCE()	Yes	

Bit

Clause	Automatically Converted	Details
BIT_COUNT()	Yes	

Control Flow

Clause	Automatically Converted	Details
CASE	Yes	
IF()	Yes	
IFNULL()	Yes	
NULLIF()	Yes	

Conversion

Clause	Automatically Converted	Details
BINARY	Yes	
CAST()	Yes	
CONVERT()	Yes	

Date and Time

Clause	Automatically Converted	Details
ADDDATE()	Yes	
ADDTIME()	Yes	
CONVERT_TZ()	Yes	
CURDATE()	Yes	
CURRENT_DATE(), CURRENT_DATE	Yes	
CURRENT_TIME(), CURRENT_TIME	Yes	
CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP	Yes	
CURTIME()	Yes	
DATE()	Yes	
DATE_ADD()	Yes	
DATE_FORMAT()	Yes	
DATE_SUB()	Yes	
DATEDIFF()	Yes	
DAY()	Yes	
DAYNAME()	Yes	
DAYOFMONTH()	Yes	
DAYOFWEEK()	Yes	
DAYOFYEAR()	Yes	
EXTRACT()	Yes	
FROM_DAYS()	Yes	
FROM_UNIXTIME()	Yes	

Clause	Automatically Converted	Details
GET_FORMAT()	Yes	
HOUR()	Yes	
LAST_DAY	Yes	
LOCALTIME(), LOCALTIME	Yes	
LOCALTIMESTAMP, LOCALTIMESTAMP()	Yes	
MAKEDATE()	Yes	
MAKETIME()	Yes	
MICROSECOND()	Yes	
MINUTE()	Yes	
MONTH()	Yes	
MONTHNAME()	Yes	
NOW()	Yes	
PERIOD_ADD()	Yes	
PERIOD_DIFF()	Yes	
QUARTER()	Yes	
SEC_TO_TIME()	Yes	
SECOND()	Yes	
STR_TO_DATE()	Yes	
SUBDATE()	Yes	
SUBTIME()	Yes	
SYSDATE()	No	
TIME()	Yes	
TIME_FORMAT()	Yes	
TIME_TO_SEC()	Yes	
TIMEDIFF()	Yes	
TIMESTAMP()	Yes	
TIMESTAMPADD()	Yes	
TIMESTAMPDIFF()	Yes	
TO_DAYS()	Yes	
TO_SECONDS()	Yes	

Clause	Automatically Converted	Details
UNIX_TIMESTAMP()	Yes	
UTC_DATE()	Yes	
UTC_TIME()	Yes	
UTC_TIMESTAMP()	Yes	
WEEK()	No	
WEEKDAY()	Yes	
WEEKOFYEAR()	Yes	
YEAR()	Yes	
YEARWEEK()	No	

Encryption and Compression

Clause	Automatically Converted	Details
AES_DECRYPT()	No	
AES_ENCRYPT()	No	
COMPRESS()	No	
DECODE()	No	
DES_DECRYPT() (deprecated 5.7.6)	No	
DES_ENCRYPT() (deprecated 5.7.6)	No	
ENCODE()	No	
ENCRYPT() (deprecated 5.7.6)	No	
MD5()	No	
OLD_PASSWORD()	No	
PASSWORD() (deprecated 5.7.6)	No	
RANDOM_BYTES()	No	
SHA1(), SHA()	No	
SHA2()	No	
UNCOMPRESS()	No	

Clause	Automatically Converted	Details
UNCOMPRESSED_LENGTH()	No	
VALIDATE_PASSWORD_STRENGTH()	No	

Information

Clause	Automatically Converted	Details
BENCHMARK()	No	
CHARSET()	No	
COERCIBILITY()	No	
COLLATION()	No	
CONNECTION_ID()	No	
CURRENT_USER(), CURRENT_USER()	No	
DATABASE()	No	
FOUND_ROWS()	No	
LAST_INSERT_ID()	No	
ROW_COUNT()	No	
SCHEMA()	No	
SESSION_USER()	No	
SYSTEM_USER()	No	
USER()	No	
VERSION()	No	

Mathematical

Clause	Automatically Converted	Details
ABS()	Yes	
ACOS()	Yes	
ASIN()	Yes	
ATAN()	Yes	
ATAN2()	Yes	

Clause	Automatically Converted	Details
CEIL()	Yes	
CEILING()	Yes	
CONV()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
COS()	Yes	
COT()	Yes	
CRC32()	Yes	
DEGREES()	Yes	
EXP()	Yes	
FLOOR()	Yes	
LN()	Yes	
LOG()	Yes	
LOG10()	Yes	
LOG2()	Yes	
MOD()	Yes	
PI()	Yes	
POW()	Yes	
POWER()	Yes	
RADIANS()	Yes	
RAND()	Yes	
ROUND(X)	Yes	
SIGN()	Yes	
SIN()	Yes	
SQRT()	Yes	
TAN()	Yes	
TRUNCATE()	Yes	

String

Clause	Automatically Converted	Details
ASCII()	Yes	
BIN()	Yes	
BIT_LENGTH()	Yes	
CHAR()	Yes	
CHAR_LENGTH()	Yes	
CHARACTER_LENGTH()	Yes	
CONCAT()	No	
CONCAT_WS()	No	
ELT()	Yes	
EXPORT_SET()	Yes	
FIELD()	Yes	
FIND_IN_SET()	Yes	
FORMAT()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
FROM_BASE64()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
HEX()	Yes	
INSERT()	Yes	
INSTR()	Yes	
LCASE()	Yes	
LEFT()	Yes	
LENGTH()	Yes	
LIKE	Yes	
LOAD_FILE()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
LOCATE()	Yes	
LOWER()	Yes	
LPAD()	Yes	
LTRIM()	Yes	
MAKE_SET()	Yes	

Clause	Automatically Converted	Details
MATCH	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
MID()	Yes	
NOT REGEXP	Yes	
NOT RLIKE	Yes	
OCT()	Yes	
OCTET_LENGTH()	Yes	
ORD()	Yes	
POSITION()	Yes	
QUOTE()	Yes	
REGEXP	Yes	
REPEAT()	Yes	
REPLACE()	Yes	
REVERSE()	Yes	
RIGHT()	Yes	
RLIKE	Yes	
RPAD()	Yes	
RTRIM()	No	
SOUNDEX()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
SOUNDS LIKE	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
SPACE()	Yes	
STRCMP()	Yes	
SUBSTR()	Yes	
SUBSTRING()	Yes	
SUBSTRING_INDEX()	Yes	
TO_BASE64()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
TRIM()	Yes	
UCASE()	Yes	
UNHEX()	Yes	

Clause	Automatically Converted	Details
UPPER()	Yes	
WEIGHT_STRING()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)

XML

Clause	Automatically Converted	Details
ExtractValue()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)
UpdateXML()	No	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)

Data Types

Date and Time

Data type	Automatically Converted to	Details
DATE	date	
DATETIME	timestamp without time zone	
TIME	time	
TIMESTAMP	timestamp without time zone	
YEAR	smallint	
YEAR (M)	smallint	

JSON

Data type	Automatically Converted to	Details
JSON	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)

Numeric

Integer Types

Data type	Automatically Converted to	Details
BIGINT [SIGNED]	bigint	
BIGINT UNSIGNED	numeric	
BOOL	boolean	Issue 8848: In MySQL the BOOLEAN type is a synonym for TINYINT (p. 240)
BOOLEAN	boolean	Issue 8848: In MySQL the BOOLEAN type is a synonym for TINYINT (p. 240)
INT [INTEGER] [SIGNED]	integer	
INT [INTEGER] UNSIGNED	bigint	
MEDIUMINT [SIGNED]	integer	
MEDIUMINT UNSIGNED	integer	
SMALLINT [SIGNED]	smallint	
SMALLINT UNSIGNED	integer	
TINYINT [SIGNED]	smallint	
TINYINT UNSIGNED	smallint	

Bit

Data type	Automatically Converted to	Details
BIT	bit	

Fixed-Point

Data type	Automatically Converted to	Details
DEC	numeric	
DEC (p)	numeric	
DEC (p,s)	numeric	
DECIMAL	numeric	

Data type	Automatically Converted to	Details
DECIMAL (p)	numeric	
DECIMAL (p,s)	numeric	
NUMERIC	numeric	
NUMERIC (p)	numeric	
NUMERIC (p,s)	numeric	

Floating-Point

Data type	Automatically Converted to	Details
DOUBLE	double precision	
DOUBLE PRECISION	double precision	
DOUBLE PRECISION (p)	double precision	
DOUBLE PRECISION (p,s)	double precision	
DOUBLE(p)	double precision	
DOUBLE(p,s)	double precision	
FLOAT	real	
FLOAT(p)	precision from 1-24: real precision > 24: double precision	
FLOAT(p,s)	double precision	
REAL	double precision	
REAL(p)	double precision	

Data type	Automatically Converted to	Details
REAL(p,s)	double precision	

Spatial Data Types

Data type	Automatically Converted to	Details
GEOMETRY	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
GEOMETRYCOLLECTION	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
LINestring	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
MULTILINESTRING	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
MULTIPOINT	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
MULTIPOLYGON	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
POINT	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
POLYGON	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)

String

Data type	Automatically Converted to	Details
BINARY	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
BLOB	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
CHAR	char	
CHAR(len)	char	

Data type	Automatically Converted to	Details
ENUM	CREATE ENUM user type	
LONGBLOB	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
LONGTEXT	text	
MEDIUMBLOB	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
MEDIUMTEXT	text	
SET	VARCHAR(8000)	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
TEXT	text	
TINYBLOB	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
TINYTEXT	text	
VARBINARY(len)	bytea	Issue 8706: PostgreSQL doesn't support %s type (p. 240)
VARCHAR(len)	varchar	

MySQL to PostgreSQL Conversion Reference

BUILT-IN SQL FUNCTIONS

Item	Issue	Resolution
Function	Issue 8811: PostgreSQL doesn't support the %s function (p. 238)	Create a user-defined function.
Function	Issue 8849: PostgreSQL doesn't support the %s function with parameter values %s (p. 238)	Create a user-defined function.
Function	Issue 8850: Some parameter values for the function %s aren't supported (p. 238)	You need to check the result of the conversion
Function	Issue 8851: PostgreSQL doesn't support using aggregate functions in the SELECT list without a GROUP BY clause (p. 238)	Perform a manual conversion.
Function	Issue 8854: PostgreSQL doesn't support the %s function with parameters (p. 238)	Perform a manual conversion.

CONTROL FLOW

Item	Issue	Resolution
CURSORS	Issue 8845: Rows count functionality is not supported behind the "open cursor" statement (p. 238)	Perform a manual conversion.
DECLARE / DEFAULT VALUE	Issue 8826: Check the default value for a Date or DateTime variable (p. 238)	Review generated code and modify it if necessary.
DECLARE / DEFAULT VALUE	Issue 8828: User-defined variables are not supported in PostgreSQL (p. 239)	Perform a manual conversion.
error handling	Issue 8844: Error codes are not the same (p. 239)	You need to check the result of the conversion
error handling	Issue 8846: PostgreSQL doesn't support the statement option %s (p. 239)	Perform a manual conversion.
error handling	Issue 8847: PostgreSQL isn't able to return to a call point after error handling (p. 239)	Perform a manual conversion.
LABEL	Issue 8827: Statement Label Syntax for BEGIN...END blocks is not supported in PostgreSQL (p. 239)	Perform a manual conversion.
PROCESS_LIST	Issue 8856: The information of the server processes is different on different servers (p. 240)	Check your conversion result.
REPLICATION	Issue 8857: Automatic conversion for replication commands is not supported (p. 240)	Perform a manual conversion.
SLEEP	Issue 8853: Perform a manual conversion if using SLEEP() with other expressions (p. 240)	Perform a manual conversion.

CREATE

Item	Issue	Resolution
	Issue 8654: Unable to convert object due to %s not created (p. 240)	Review the %s object.

DATATYPES

Item	Issue	Resolution
BOOLEAN	Issue 8848: In MySQL the BOOLEAN type is a synonym for TINYINT (p. 240)	Check it when you insert data from a BOOLEAN variables into the database.

Item	Issue	Resolution
DATATYPES	Issue 8706: PostgreSQL doesn't support %s type (p. 240)	To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

DDL

Item	Issue	Resolution
CREATE TABLE	Issue 8825: Check the default value for a Date or DateTime column (p. 240)	Review generated code and modify it if necessary.
DROP TABLE	Issue 8801: The table can be locked open cursor (p. 241)	Review your transformed code and modify it if necessary.

DML

Item	Issue	Resolution
DML	Issue 8829: PostgreSQL doesn't have an analog of clause ON DUPLICATE KEY UPDATE (p. 241)	Perform a manual conversion.
DML	Issue 8830: PostgreSQL doesn't have an option similar to LOW_PRIORITY for DML statements (p. 241)	Use DML statement without this option.
DML	Issue 8831: PostgreSQL doesn't have an option similar to IGNORE for DML statements (p. 241)	Use DML statement without this option.
DML	Issue 8832: PostgreSQL doesn't have an option similar to QUICK for DML statements (p. 241)	Use DML statement without this option.
DML	Issue 8833: PostgreSQL can't make updates to several tables at the same time (p. 242)	Perform a manual conversion.
DML	Issue 8834: PostgreSQL can't delete from several tables at the same time (p. 242)	Perform a manual conversion.
SELECT	Issue 8835: PostgreSQL doesn't have an option similar to DELAYED for DML statements (p. 242)	Use DML statement without this option.
SELECT	Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements (p. 242)	Use DML statement without this option.
SELECT	Issue 8837: PostgreSQL doesn't support clause STRAIGHT_JOIN (p. 242)	Use DML statement without this option

Item	Issue	Resolution
SELECT	Issue 8838: PostgreSQL doesn't support clause SQL_SMALL_RESULT (p. 242)	Use DML statement without this option
SELECT	Issue 8839: PostgreSQL doesn't support clause SQL_BIG_RESULT (p. 242)	Use DML statement without this option
SELECT	Issue 8840: PostgreSQL doesn't support clause SQL_BUFFER_RESULT (p. 242)	Use DML statement without this option
SELECT	Issue 8841: PostgreSQL doesn't support clause SQL_CACHE (p. 242)	Use DML statement without this option
SELECT	Issue 8842: PostgreSQL doesn't support clause SQL_NO_CACHE (p. 242)	Use DML statement without this option
SELECT	Issue 8843: PostgreSQL doesn't support clause SQL_CALC_FOUND_ROWS (p. 242)	Use DML statement without this option
WHERE	Issue 8795: PostgreSQL is case sensitive. Check the string comparison. (p. 242)	Check the string comparison.

EVENTS

Item	Issue	Resolution
EVENTS	Issue 8855: Automatic conversion of EVENT syntax is not supported (p. 243)	Perform a manual conversion.

EXECUTE

Item	Issue	Resolution
EXECUTE a character string	Issue 8672: Automatic conversion of this command is not supported (p. 243)	Perform a manual conversion.

Operators

Item	Issue	Resolution
Arithmetic operators	Issue 8773: The tool can not currently perform automated migration of arithmetic operations with dates (p. 243)	Perform a manual conversion.
Arithmetic operators	Issue 8774: The tool can not currently perform automated migration of arithmetic operations with mixed types of operands (p. 243)	Perform a manual conversion.

Parser Error

Item	Issue	Resolution
Parser Error	Issue 8663: Unable to resolve the object (p. 243)	Verify if object %s is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

SELECT

Item	Issue	Resolution
GROUP BY	Issue 8653: PostgreSQL doesn't support the option GROUP BY ROLLUP (p. 244)	Perform a manual conversion.

TRANSACTION

Item	Issue	Resolution
TRANSACTION	Issue 8807: PostgreSQL doesn't support transactions in functions (p. 244)	Perform a manual conversion.

Unknown

Item	Issue	Resolution
	Issue 8655: This syntactic element conversion is not supported yet (p. 244)	Perform a manual conversion.
Unknown Clause	Issue 8674: Automatic conversion of %s clause of %s statement is not supported (p. 244)	Perform a manual conversion.

VIEW

Item	Issue	Resolution
CREATE VIEW	Issue 8824: ALGORITHM option is not supported (p. 245)	Perform a manual conversion.

Conversion Issues with BUILT-IN SQL FUNCTIONS

Issue 8811: PostgreSQL doesn't support the %s function

Create a user-defined function.

Issue 8849: PostgreSQL doesn't support the %s function with parameter values %s

Create a user-defined function.

Issue 8850: Some parameter values for the function %s aren't supported

Review the transformed code, and correct it if necessary.

Issue 8851: PostgreSQL doesn't support using aggregate functions in the SELECT list without a GROUP BY clause

In MySQL, it is possible to use an aggregate function on a field in the SELECT list without adding the other fields in the SELECT list to a GROUP BY clause. In PostgreSQL, adding these fields to a GROUP BY clause is required. Perform a manual conversion in this case.

Issue 8854: PostgreSQL doesn't support the %s function with parameters

Perform a manual conversion.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CONTROL FLOW

Issue 8845: Rows count functionality is not supported behind the "open cursor" statement

In PostgreSQL, the FOUND_ROWS() function can't be called within a cursor. Perform a manual conversion instead.

Issue 8826: Check the default value for a Date or DateTime variable

When converting default values for Date, Time, or DateTime variables, they are explicitly cast to the data type of the target variable. For example, the following MySQL code:

```
DECLARE ValueDate1 DATE DEFAULT '2000-01-01';
```

Is converted to the following PostgreSQL code:

```
ValueDate1 DATE DEFAULT '2000-01-01'::DATE;
```

If the default value is of the form '0000-00-00' or similar ('0000-00-00', '0000-00-00 00:00:00', '0000-00-00 00:00:00.000000'), it is converted to the literal 'epoch'. For example, the following MySQL code:

```
DECLARE ValueDate2 DATE DEFAULT '0000-00-00';
```

Is converted to the following PostgreSQL code:

```
ValueDate2 DATE DEFAULT 'epoch'::DATE;
```

In these cases, the default value must be manually updated. Review the generated code and modify it if necessary.

Issue 8828: User-defined variables are not supported in PostgreSQL

Perform a manual conversion.

Issue 8844: Error codes are not the same

Some exit handler conditions require manual conversion. MySQL code with any of the following exit handler statements must be converted to use PostgreSQL error handling with OTHERS or SQLSTATE conditions instead:

```
DECLARE EXIT HANDLER FOR SQLWARNING, SQLEXCEPTION, NOT FOUND  
DECLARE EXIT HANDLER FOR SQLWARNING  
DECLARE EXIT HANDLER FOR SQLEXCEPTION  
DECLARE EXIT HANDLER FOR NOT FOUND  
DECLARE EXIT HANDLER FOR SQLWARNING, NOT FOUND  
DECLARE EXIT HANDLER FOR <mysql_error_code>  
DECLARE EXIT HANDLER FOR SQLSTATE <sqlstate_value>
```

Review the transformed code, and correct it if necessary.

Issue 8846: PostgreSQL doesn't support the statement option %s

Some of the condition information items that are used with the DECLARE CONDITION and GET DIAGNOSTICS CONDITION statements in MySQL are not supported by PostgreSQL. These include:

- CLASS_ORIGIN
- SUBCLASS_ORIGIN
- MYSQL_ERRNO
- CONSTRAINT_CATALOG
- CONSTRAINT_SCHEMA
- CATALOG_NAME
- CURSOR_NAME

Manually convert the code wherever you use these condition information items.

Issue 8847: PostgreSQL isn't able to return to a call point after error handling

If an error occurs in a BEGIN...END block, PostgreSQL exits the block without executing any further statements in that block. Check your error handling logic and perform a manual conversion instead.

Issue 8827: Statement Label Syntax for BEGIN...END blocks is not supported in PostgreSQL

Perform a manual conversion.

Issue 8856: The information of the server processes is different on different servers

Check your conversion result.

Issue 8857: Automatic conversion for replication commands is not supported

Perform a manual conversion for all replication-related code.

Issue 8853: Perform a manual conversion if using SLEEP() with other expressions

If you are using SLEEP() in a SELECT statement along with other expressions, for example `SELECT NOW(), SLEEP(2), AccountNo FROM Account`, you must manually convert that code.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE

Issue 8654: Unable to convert object due to %s not created

Review the %s object.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DATATYPES

Issue 8848: In MySQL the BOOLEAN type is a synonym for TINYINT

Check your code wherever you insert data from a BOOLEAN variable into the database.

Issue 8706: PostgreSQL doesn't support %s type

To store data of this type in PostgreSQL, use a PostgreSQL-compatible type or use a composite type.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DDL

Issue 8825: Check the default value for a Date or DateTime column

When converting default values for Date, Time, or DateTime variables, they are explicitly cast to the data type of the target variable. For example, the following MySQL code:

```
DECLARE ValueDate1 DATE DEFAULT '2000-01-01';
```

Is converted to the following PostgreSQL code:

```
ValueDate1 DATE DEFAULT '2000-01-01'::DATE;
```

If the default value is of the form '0000-00-00' or similar ('0000-00-00', '0000-00-00 00:00:00', '0000-00-00 00:00:00.000000'), it is converted to the literal 'epoch'. For example, the following MySQL code:

```
DECLARE ValueDate2 DATE DEFAULT '0000-00-00';
```

Is converted to the following PostgreSQL code:

```
ValueDate2 DATE DEFAULT 'epoch'::DATE;
```

In these cases, the default value must be manually updated. Review the generated code and modify it if necessary.

Issue 8801: The table can be locked open cursor

The conversion uses the DROP TABLE IF EXISTS statement to remove a temporary table before the end of the function in which it was created. In some cases though, this statement might fail because the temporary table is locked due to an open cursor. In this case, the DROP TABLE statement must be commented out. Review your transformed code and modify it if necessary.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DML

Issue 8829: PostgreSQL doesn't have an analog of clause ON DUPLICATE KEY UPDATE

Perform a manual conversion.

Issue 8830: PostgreSQL doesn't have an option similar to LOW_PRIORITY for DML statements

Use the DML statement without this option.

Issue 8831: PostgreSQL doesn't have an option similar to IGNORE for DML statements

Use the DML statement without this option.

Issue 8832: PostgreSQL doesn't have an option similar to QUICK for DML statements

Use the DML statement without this option.

Issue 8833: PostgreSQL can't make updates to several tables at the same time

PostgreSQL doesn't support using an UPDATE statement to update several tables at once, for example `update Account, Customer set Account.AccountBalance = Account.AccountBalance + 100, Customer.SName = 'Sofan Hubert'`. Perform a manual conversion in this case.

Issue 8834: PostgreSQL can't delete from several tables at the same time

PostgreSQL doesn't support using a DELETE statement to delete several tables at once, for example `delete table1, table3`. Perform a manual conversion in this case.

Issue 8835: PostgreSQL doesn't have an option similar to DELAYED for DML statements

Use the DML statement without this option.

Issue 8836: PostgreSQL doesn't have an option similar to %s for DML statements

Use the DML statement without this option.

Issue 8837: PostgreSQL doesn't support clause STRAIGHT_JOIN

Use the DML statement without this option

Issue 8838: PostgreSQL doesn't support clause SQL_SMALL_RESULT

Use the DML statement without this option

Issue 8839: PostgreSQL doesn't support clause SQL_BIG_RESULT

Use the DML statement without this option

Issue 8840: PostgreSQL doesn't support clause SQL_BUFFER_RESULT

Use the DML statement without this option

Issue 8841: PostgreSQL doesn't support clause SQL_CACHE

Use the DML statement without this option

Issue 8842: PostgreSQL doesn't support clause SQL_NO_CACHE

Use the DML statement without this option

Issue 8843: PostgreSQL doesn't support clause SQL_CALC_FOUND_ROWS

Use the DML statement without this option

Issue 8795: PostgreSQL is case sensitive. Check the string comparison.

Check string comparisons in the WHERE clauses of statements, for example `WHERE [Name] LIKE '%BLOCKED%'`; . All strings used in comparisons must be lowercase.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with EVENTS

Issue 8855: Automatic conversion of EVENT syntax is not supported

Event-related statements like CREATE EVENT or DROP EVENT are not supported in PostgreSQL. Perform a manual conversion of this code.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with EXECUTE

Issue 8672: Automatic conversion of this command is not supported

Code that uses the PREPARE...EXECUTE...DEALLOCATE PREPARE statements can't be automatically converted. Perform a manual conversion instead.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Operators

Issue 8773: The tool can not currently perform automated migration of arithmetic operations with dates

Perform a manual conversion.

Issue 8774: The tool can not currently perform automated migration of arithmetic operations with mixed types of operands

Code that uses an arithmetic operator and has one operand that has a DATE, TIME or TIMESTAMP data type can't be automatically converted. For example, set `d = now() + 1;`. Perform a manual conversion instead.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Parser Error

Issue 8663: Unable to resolve the object

Verify if object %s is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT

Issue 8653: PostgreSQL doesn't support the option GROUP BY ROLLUP

PostgreSQL doesn't support the GROUP BY <field_name> WITH ROLLUP clause for the SELECT statement. Perform a manual conversion of this code.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with TRANSACTION

Issue 8807: PostgreSQL doesn't support transactions in functions

Code that uses transaction management statements such as START TRANSACTION or COMMIT within a function or stored procedure can't be automatically converted. Perform a manual conversion instead.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Unknown

Issue 8655: This syntactic element conversion is not supported yet

Perform a manual conversion.

Issue 8674: Automatic conversion of %s clause of %s statement is not supported

This error is typically raised by code that sets transaction options or replication-related variables that aren't supported in PostgreSQL. Such code must be manually converted. Statements that raise this error include the following:

- SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
- SET TRANSACTION READ ONLY;
- SET SESSION TRANSACTION READ WRITE;
- SET SESSION TRANSACTION ISOLATION LEVEL REPEATABLE READ;
- SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
- SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

- SET GLOBAL TRANSACTION READ WRITE;
- SET GLOBAL tx_isolation=<isolation_level>;
- SET SQL_LOG_BIN=<value>;
- SET GLOBAL sql_slave_skip_counter=<value>;

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with VIEW

Issue 8824: ALGORITHM option is not supported

PostgreSQL doesn't support the ALGORITHM clause for the CREATE VIEW or ALTER VIEW statements. Perform a manual conversion in this case.

Related Topics

- [MySQL to PostgreSQL Conversion Reference \(p. 233\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Related Topics

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Oracle to MySQL Supported Schema Conversion

The following sections list the schema elements from an Oracle database and whether they are supported for automatic conversion to MySQL using the AWS Schema Conversion Tool.

Topics

- [Statements \(p. 246\)](#)
- [Procedures \(p. 250\)](#)
- [Flow Control \(p. 251\)](#)
- [Packages \(p. 251\)](#)
- [Functions \(p. 251\)](#)
- [Operators \(p. 261\)](#)
- [Data Types \(p. 264\)](#)
- [Data Definition Language \(DDL\) \(p. 266\)](#)
- [Cursors \(p. 270\)](#)
- [Hints \(p. 270\)](#)
- [Exceptions \(p. 273\)](#)
- [Related Topics \(p. 274\)](#)

Statements

Topics

- [SELECT](#) (p. 246)
- [INSERT](#) (p. 247)
- [UPDATE](#) (p. 248)
- [DELETE](#) (p. 249)
- [MERGE](#) (p. 250)
- [TRUNCATE](#) (p. 250)
- [LOCK TABLE](#) (p. 250)

SELECT

Clause	Automatically Converted	Details
WITH	No	Use a stored procedure to prepare data, or rewrite your query to avoid the WITH clause.
AS	No	
SELECT	Yes	
DISTINCT UNIQUE ALL	Yes	
select_list	Yes	
BULK COLLECT INTO	No	You can try to include all of the fields from your table in an INTO clause.
INTO	Yes	
record_name	No	
FROM	Yes	
@dblink	No	
materialized view	No	
TABLE (collection_expression)	No	
MODEL	No	MySQL doesn't support the MODEL statement.
START WITH	No	MySQL doesn't support hierarchical queries. Use a stored procedure to prepare data.
CONNECT BY	No	MySQL doesn't support hierarchical queries. Use a stored procedure to prepare data.
PIVOT	No	
XML	No	
UNPIVOT	No	

Clause	Automatically Converted	Details
WHERE	Yes	
GROUP BY	Yes	
CUBE	No	Use a stored procedure to prepare data.
GROUPING SETS	No	Use a stored procedure to prepare data.
HAVING	Yes	
ORDER BY	Yes	
SIBLINGS	No	
NULLS FIRST NULLS LAST	No	MySQL doesn't support NULLS FIRST and NULLS LAST. In MySQL NULL sorting, NULL values go first for an ascending order and last for a descending order. Try rewriting the ORDER BY clause with CASE.
FOR UPDATE	Yes	
OF	No	Try using FOR UPDATE instead of FOR UPDATE OF.
NOWAIT WAIT	No	MySQL doesn't support WAIT and NOWAIT clauses. Try using FOR UPDATE without NOWAIT.
SKIP LOCKED	No	Try using FOR UPDATE without SKIP LOCKED.
UNION	Yes	
INTERSECT MINUS]	Yes	

INSERT

Clause	Automatically Converted	Details
INTO table	Yes	
PARTITION	Yes	
PARTITION FOR	No	
SUBPARTITION	No	Either insert data into the overlying partition, or perform a manual conversion using the INSERT statement.
VIEW	No	Target a table with the INSERT statement instead of a VIEW. If the target view has an INSTEAD OF trigger, parse and execute the INSTEAD OF trigger code.
MATERIALIZED VIEW	No	Perform a manual conversion.
subquery	No	Perform this operation on the underlying tables instead.
WITH table_collection_expression	No	

Clause	Automatically Converted	Details
column ...	Yes	
VALUES	Yes	
subquery	Yes	
RETURNING ... INTO	No	To perform this operation, divide the INSERT statement with the RETURNING clause into an INSERT statement with following SELECT statements and use the same key conditions in each SELECT. You can also use the last value that was generated by AUTO_INCREMENT column in the key condition.
LOG ERRORS	No	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

UPDATE

Clause	Automatically Converted	Details
UPDATE [hint]	Yes	
table	Yes	
PARTITION	Yes	
PARTITION FOR	No	
SUBPARTITION	No	Either insert data into the overlying partition, or perform a manual transformation using the UPDATE statement.
VIEW	No	Perform an update on the underlying tables instead.
MATERIALIZED VIEW	No	
subquery	No	Perform this operation on the underlying tables instead.
WITH	No	
table_collection_expression	Yes	
SET	Yes	
VALUE		
WHERE condition	Yes	
RETURNING ... INTO	No	To perform this operation, divide the UPDATE statement with the RETURNING clause into an UPDATE statement with following INSERT statements that have the specified key conditions in the SELECT part.

Clause	Automatically Converted	Details
LOG ERRORS	No	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

Note

MySQL doesn't support FILESTREAM DATA. Perform a manual conversion to update the data in the file system file.

DELETE

Clause	Automatically Converted	Details
DELETE	Yes	
FROM	Yes	
PARTITION	Yes	
PARTITION FOR	No	Either insert data into the overlying partition, or perform a manual transformation using the DELETE statement.
SUBPARTITION	No	
VIEW	No	Perform a manual conversion.
MATERIALIZED VIEW	No	Perform a manual conversion.
subquery	No	Perform this operation on the underlying tables instead.
WITH	No	
table_collection_expression	Yes	
WHERE condition	Yes	
RETURNING ... INTO	No	To perform this operation, divide the DELETE statement with the RETURNING clause into a DELETE statement with following INSERT statements and use the same key conditions in each SELECT.
LOG ERRORS	No	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

MERGE

Statement	Automatically Converted	Details
MERGE	No	To achieve the effect of a MERGE statement, use separate INSERT, DELETE, and UPDATE statements.

TRUNCATE

Clause	Automatically Converted	Details
TRUNCATE TABLE	Yes	
PRESERVE MATERIALIZED VIEW LOG	No	
PURGE MATERIALIZED VIEW LOG	No	
DROP STORAGE	No	
REUSE STORAGE	No	

LOCK TABLE

Clause	Automatically Converted	Details
PARTITION	No	
SUBPARTITION	No	
NOWAIT	No	

Procedures

Item	Automatically Converted	Details
LOCK TABLE	No	MySQL doesn't support the LOCK TABLE statement inside a stored procedure.
dbms_output.put_line	No	Try using INSERT in the log table. To do this, you must add code into AWS_ORACLE_EXT.PUT_LINE.
dbms_output.put	No	Try using INSERT in the log table. To do this, you must add code into AWS_ORACLE_EXT.PUT.

Flow Control

Clause	Automatically Converted	Details
GOTO	No	
FORALL	No	Try using a WHILE DO statement.
EXECUTE IMMEDIATE	No	
BULK COLLECT	No	
RETURNING BULK COLLECT INTO	No	
LABEL	No	Try rewriting variables without using labels.

Packages

Item	Automatically Converted	Details
Initialization block, BEGIN ... END	Partial	MySQL doesn't support EXCEPTION BLOCK in initialization blocks in packages. Try using CONTINUE HANDLER.
User type	No	
Global cursor	No	
Global user exception	No	

Functions

Topics

- [Aggregate Functions \(p. 252\)](#)
- [Date and Time Functions \(p. 253\)](#)
- [Mathematical Functions \(p. 254\)](#)
- [Character \(String\) Functions \(p. 256\)](#)
- [Conversion Functions \(p. 257\)](#)
- [General Comparison Functions \(p. 258\)](#)
- [Encoding and Decoding Functions \(p. 259\)](#)
- [Email \(p. 259\)](#)
- [HTTP \(p. 259\)](#)
- [SMS \(p. 259\)](#)
- [NULL Functions \(p. 259\)](#)
- [User-Defined Functions \(p. 259\)](#)
- [Large Object Functions \(p. 261\)](#)
- [Hierarchical Functions \(p. 261\)](#)

- [Analytic Functions \(p. 261\)](#)

In this section, you can find a list of the Microsoft SQL Server built-in functions that indicates whether the AWS Schema Conversion Tool performs an automatic conversion. Where MySQL doesn't support a function, consider creating a user-defined function.

Aggregate Functions

Function	Automatically Converted	Details
AVG	Yes	
COLLECT	No	
CORR	No	
CORR_*	No	
COUNT	Yes	
COVAR_POP	No	
COVAR_SAMP	No	
CUME_DIST	No	
DENSE_RANK	No	
FIRST	No	
GROUP_ID	No	
GROUPING	No	
GROUPING_ID	No	
LAST	No	
MAX	Yes	
MEDIAN	No	
MIN	Yes	
PERCENTILE_CONT	No	
PERCENTILE_DISC	No	
PERCENT_RANK	No	
RANK	No	
REGR_ (Linear Regression) Functions	No	
STATS_BINOMIAL_TEST	No	
STATS_CROSSTAB	No	
STATS_F_TEST	No	

Function	Automatically Converted	Details
STATS_KS_TEST	No	
STATS_MODE	No	
STATS_MW_TEST	No	
STATS_ONE_WAY_ANOVA	No	
STATS_T_TEST_*	No	
STATS_WSR_TEST	No	
STDDEV	Yes	
STDDEV_POP	Yes	
STDDEV_SAMP	Yes	
SUM	Yes	
TRUNC	Yes	Converts to: TRUNCATE()
VAR_POP	Yes	
VAR_SAMP	Yes	
VARIANCE	Yes	

Date and Time Functions

Function	Automatically Converted	Details
ADD_MONTHS(date, num)	Partial	Converts to: TIMESTAMPADD(MONTH, num, date)
CURRENT_DATE	Partial	Converts to: NOW()
CURRENT_TIMESTAMP	Partial	Converts to: NOW()
DBTIMEZONE	No	
EXTRACT(YEAR FROM date)	Partial	Converts to: YEAR(date)
EXTRACT(MONTH FROM date)	Partial	Converts to: MONTH(date)
EXTRACT(DAY FROM date)	Partial	Converts to: DAY(date)
EXTRACT(HOUR FROM time)	Partial	Converts to: HOUR(time)
EXTRACT(MINUTE FROM time)	Partial	Converts to: MINUTE(time)
EXTRACT(SECOND FROM time)	Partial	Converts to: SECOND(time)

Function	Automatically Converted	Details
FROM_TZ	No	
LAST_DAY(date)	Yes	
LOCALTIMESTAMP	Yes	
LOCALTIMESTAMP(prec)	Partial	The maximum precision for the MySQL LOCALTIMESTAMP function is 6. If you need greater precision, create a user-defined function.
MONTHS_BETWEEN(date1, date2)	No	
NEW_TIME	No	
NEXT_DAY	No	
NUMTODSINTERVAL	No	
NUMTOYMINTERVAL	No	
ROUND (date)	No	
SESSIONTIMEZONE	No	
SYS_EXTRACT_UTC	No	
SYSDATE	Partial	Converts to: SYSDATE()
SYSTIMESTAMP	Partial	Converts to: CURRENT_TIMESTAMP
TO_CHAR (datetime, format)	Partial	Converts to: DATE_FORMAT() Note that the TO_CHAR and DATE_FORMAT format strings are different.
TO_TIMESTAMP(exp)	No	
TO_TIMESTAMP_TZ	No	
TO_DSINTERVAL	No	
TO_YMINTERVAL	No	
TRUNC (datetime)	Partial	Converts to: DATE(datetime)
TZ_OFFSET	No	

Mathematical Functions

Function	Automatically Converted	Details
ABS(num)	Yes	
ACOS(num)	Yes	

Function	Automatically Converted	Details
ASIN(num)	Yes	
ATAN(num)	Yes	
ATAN2(x,y)	Yes	
BITAND(exp1, exp2)	Partial	Converts to: (exp1 & exp2)
CEIL(num)	Yes	
COS(num)	Yes	
COSH(num)	Partial	Converts to: (EXP(num) + EXP(-num)) / 2
EXP(n)	Yes	
FLOOR(num)	Yes	
LN(num)	Yes	
LOG(num1, num2)	Yes	
MOD(dividend, divisor)	Yes	
NANVL(n2, n1)	No	
POWER(value, n)	Yes	
REMAINDER(n1, n2)	Partial	Converts to: (n1 - n2*ROUND(n1/n2))
ROUND (num, integer)	Yes	
SIGN(exp)	Yes	
SIN(num)	Yes	
SINH(num)	Yes	
SQRT(num)	Yes	
TAN(num)	Yes	
TANH(num)	Yes	
TRUNC (number)	Partial	Converts to: TRUNCATE(num, 0)
TRUNC(num, num2)	Yes	
VALUE(variable)	No	
WIDTH_BUCKET	No	

Character (String) Functions

Function	Automatically Converted	Details
ASCII(str)	Yes	
CHR(num)	Partial	Converts to: CHAR(num USING ASCII)
CONCAT(char1, char2)	Yes	
INITCAP(string)	No	
INSTR(str, substr)	Yes	
INSTR(str, substr, pos)	Partial	Converts to: LOCATE (substr, str, pos) This function is the same as the two-argument form of INSTR(), except that the order of the arguments is reversed.
INSTR(str, substr, pos, num)	No	
LENGTH(string)	Partial	Converts to: LENGTH (string)
LOWER(string)	Yes	
LPAD(string, len)	Partial	Converts to: LPAD(string, len, ' ')
LPAD(string, len, pad)	Yes	
LTRIM(string)	Yes	
LTRIM(string, set)	Partial	Converts to: TRIM(LEADING set FROM string)
NLS_INITCAP	No	
NLS_LOWER	No	
NLS_UPPER	No	
NLSSORT	No	
REGEXP_INSTR	No	
REGEXP_REPLACE	No	
REGEXP_SUBSTR	No	
REPLACE(str, search)	Partial	Converts to: REPLACE(str, search, "")
REPLACE(str, search, replace)	Yes	
RPAD(string, len)	Partial	Converts to: RPAD(string, len, ' ')
RPAD(string, len, pad)	Yes	
RTRIM(string)	Yes	
RTRIM(string, set)	Partial	Converts to: TRIM(TRAILING set FROM string)

Function	Automatically Converted	Details
SOUNDEX(string)	Yes	
SUBSTR(string, pos, len)	Yes	
TRANSLATE(string, from, to)	No	
TREAT		
TRIM([type trim FROM] string)	Yes	
UPPER(string)	Yes	

Conversion Functions

Function	Automatically Converted	Details
ASCIISTR(string)	No	
BIN_TO_NUM(bit1, bit2, ...)	No	
CAST	Yes	
CHARTOROWID	No	
COMPOSE	No	
CONVERT(string, charset)	Partial	Converts to: CONVERT(string USING charset)
DECOMPOSE	No	
HEXTORAW	No	
NUMTODSINTERVAL	No	
NUMTOYMINTERVAL	No	
RAWTOHEX	No	
RAWTONHEX	No	
ROWIDTOCHAR	No	
ROWIDTONCHAR	No	
SCN_TO_TIMESTAMP	No	
TIMESTAMP_TO_SCN	No	
TO_BINARY_DOUBLE	No	
TO_BINARY_FLOAT	No	
TO_CHAR (character)	No	

Function	Automatically Converted	Details
TO_CHAR (datetime, format)	Partial	Converts to: DATE_FORMAT(datetime, format) Note that the TO_CHAR and DATE_FORMAT format strings are different.
TO_CHAR (number, format)	Partial	Converts to: FORMAT(number, decimal_digits) In MySQL, you can use FORMAT function as well as other string functions and expressions.
TO_CLOB	Partial	
TO_DATE	Partial	Converts to: STR_TO_DATE(string, format) Note that the TO_DATE and STR_TO_DATE format strings are different.
TO_DSINTERVAL	No	
TO_LOB	No	
TO_MULTI_BYTE	No	
TO_NCHAR (character)	No	
TO_NCHAR (datetime)	No	
TO_NCHAR (number)	No	
TO_NCLOB	No	
TO_NUMBER	No	
TO_DSINTERVAL	No	
TO_SINGLE_BYTE	No	
TO_TIMESTAMP	No	
TO_TIMESTAMP_TZ	No	
TO_YMINTERVAL	No	
TRANSLATE ... USING	No	
UNISTR	Partial	Converts to: CHAR(string USING UCS2)

General Comparison Functions

Function	Automatically Converted	Details
GREATEST(exp, exp2, ...)	Yes	
LEAST(exp, exp2, ...)	Yes	

Encoding and Decoding Functions

Function	Automatically Converted	Details
DECODE(exp, when, then, ...)	Partial	Converts to: CASE expression
DUMP	No	
ORA_HASH	No	
VSIZE	No	

Email

MySQL doesn't support sending e-mail. To send email, use Amazon Simple Email Service (Amazon SES).

HTTP

MySQL doesn't support sending messages to HTTP endpoints. To send messages to HTTP and HTTPS endpoints, you can use the Amazon Simple Notification Service (Amazon SNS).

SMS

MySQL doesn't support sending notifications by using SMS. To send and receive SMS notifications, you can use the Amazon Simple Notification Service (Amazon SNS).

NULL Functions

Function	Automatically Converted	Details
COALESCE(exp1, exp2, ...)	Yes	
LNNVL	No	
NULLIF(exp1, exp2)	Partial	Converts to: NULLIF(exp1, exp2)
NVL(exp, replacement)	Partial	Converts to: IFNULL(exp, replacement)
NVL2(exp1, exp2, exp3)	Partial	Converts to: CASE expression

User-Defined Functions

Return

In MySQL, you can use statements that return a result set within a stored procedure but not within a stored function.

Statement	Automatically Converted	Details
RETURN resultset	No	Try changing the function to a stored procedure and use a table to store the results.
RETURN record-type	No	Try changing the function to a stored procedure and change record-type items to separate parameters.
TYPE .. IS TABLE OF .. INDEX BY	No	Try changing the function to a stored procedure and use a table to store the results.
TYPE .. IS TABLE OF	No	Try changing the function to a stored procedure and use a table to store the results.
TYPE .. IS VARRAY(..) OF	No	Try changing the function to a stored procedure and use a table to store the results.
RETURN .. PIPELINED	No	Try changing the function to a stored procedure and use a table to store the results.
TYPE ... IS REF CURSOR	No	Try changing the function to a stored procedure and use a table to store the results.
object-type	No	

Collections

Item	Automatically Converted	Details
TYPE .. IS TABLE OF...	No	MySQL doesn't support table type variables. Try using a table.
collection_name.First, collection_name.Last, collection_name.Count, collection_name.Next	No	MySQL doesn't support table type collection methods. Try using a table.
:= collection_type(...)	No	MySQL doesn't support the constructor for the collection type. Try using a table.

Arguments

Item	Automatically Converted	Details
UDT record-type	No	
UDT collection-type	No	
UDT object-type	No	

Large Object Functions

Function	Automatically Converted	Details
BFILENAME	No	
EMPTY_BLOB	Partial	Converts to an empty string.
EMPTY_CLOB	Partial	Converts to an empty string.

Hierarchical Functions

Function	Automatically Converted	Details
SYS_CONNECT_BY_PATH	No	Try creating a user-defined function.

Analytic Functions

MySQL doesn't support analytic functions. Try creating a user-defined function.

Operators

Topics

- [Arithmetic Operators \(p. 261\)](#)
- [Assignment Operator \(p. 262\)](#)
- [Comparison Operators \(p. 262\)](#)
- [Logical Operators \(p. 262\)](#)
- [String Concatenation Operator \(p. 263\)](#)
- [Date and time format specifiers \(p. 263\)](#)

Arithmetic Operators

Clause	Automatically Converted	Details
+	Yes	
-	Yes	
+ -	Partial	MySQL handles date and time arithmetic differently. Verify that the converted schema is accurate.
* /	Yes	

Assignment Operator

Clause	Automatically Converted	Details
=	Yes	

Comparison Operators

Clause	Automatically Converted	Details
>	Yes	
<	Yes	
>=	Yes	
<=	Yes	
<>	Yes	
!=	Yes	
!<	Yes	
!>	Yes	

Logical Operators

Clause	Automatically Converted	Details
IN	Yes	
NOT IN	Yes	
ANY	Yes	
SOME	Yes	
ALL	Yes	
BETWEEN x	Yes	
EXISTS	Yes	
LIKE	Yes	
IS NULL	Yes	
NOT	Yes	
AND	Yes	
OR	Yes	

String Concatenation Operator

Clause	Automatically Converted	Details
	Yes	

Date and time format specifiers

Clause	Automatically Converted	Default Conversion
YEAR	No	
YYYY	Partial	%Y
YYY	No	
YY	Partial	%y
Y	No	
IYY	No	
IY	Partial	%y
I	No	
IYYY	Partial	%Y
RRRR	No	
Q	No	
MM	Partial	%m
MON	Partial	%b
MONTH	Partial	%M
RM	No	
WW	Partial	%V
W	No	
IW	Partial	%V
D	Partial	%w
Day of the week (1=Sunday, 7=Saturday)	Partial	Day of the week (0=Sunday, 6=Saturday)
DAY	Partial	%W
DD	Partial	%e
DDD	Partial	%j

Clause	Automatically Converted	Default Conversion
DY	Partial	%a
J	No	
HH	Partial	%h
HH12	Partial	%h
HH24	Partial	%H
MI	Partial	%i
SS	Partial	%S
SSSSS	No	
FF	No	
AM, PM	Partial	%p
A.M., P.M.	No	
AD or A.D	No	
BC or B.C.	No	
TZD	No	
TZH	No	
TZM	No	
TZR	No	

Data Types

Data type	Automatically Converted	Default Conversion	Details
BFILE	Partial	VARCHAR(255)	Contains the file path to the BFILE. MySQL does not support the BFILE. Because BFILE data is made up of the path to a file, you can either store a filename and create a routine that gets the file from the file system, or store the file contents as a LONGBLOB.
BINARY_FLOAT	Yes	FLOAT	
BINARY_DOUBLE	Yes	DOUBLE	
BLOB	Yes	LONGBLOB	

Data type	Automatically Converted	Default Conversion	Details
CHAR	Yes	TEXT	
CHAR(n), CHARACTER(n)	Yes	CHAR(n), CHARACTER(n)	
CHAR(n), CHARACTER(n)	Yes	VARCHAR(n)	
CLOB	Yes	LONGTEXT	
DATE	Yes	DATETIME	
DECIMAL, DEC	Yes	DOUBLE	
DECIMAL(p,s), DEC(p,s)	Yes	DECIMAL(p,s), DEC(p,s)	
DOUBLE PRECISION	Yes	DOUBLE PRECISION	
FLOAT	Yes	DOUBLE	
FLOAT(p)	Yes	DOUBLE	
INTEGER, INT	Yes	DECIMAL(38)	
INTERVAL YEAR TO MONTH	Yes	DOUBLE	
INTERVAL YEAR(p) TO MONTH	Yes	VARCHAR(30)	
INTERVAL DAY TO SECOND	Yes	VARCHAR(30)	
INTERVAL DAY(p) TO SECOND(s)	Yes	VARCHAR(30)	
LONG	Yes	LONGTEXT	
LONG RAW	Yes	LOB	
NCHAR	Yes	TEXT	
NCHAR(n)	Yes	NCHAR(n)	
NCHAR(n)	Yes	NVARCHAR(n)	
NCHAR VARYING	Yes	TEXT	
NCHAR VARYING(n)	Yes	NCHAR VARYING(n)	
NCLOB	Yes	LONGTEXT	
NUMBER(p,0), NUMBER(p)	Yes	DECIMAL(p,0), DECIMAL(p)	
NUMBER(p,s)	Yes	DECIMAL(p,s)	
NUMBER, NUMBER(*), NUMERIC	Yes	DOUBLE	
NUMERIC(p,s)	Yes	NUMERIC(p,s)	
NVARCHAR2	Yes	TEXT	
NVARCHAR2(n)	Yes	NVARCHAR(n)	

Data type	Automatically Converted	Default Conversion	Details
RAW	Yes	VARBINARY(2000)	
RAW(n)	Yes	BINARY(n)	
RAW(n)	Yes	VARBINARY(n)	
REAL	Yes	DOUBLE	
ROWID	Yes	CHAR(10)	
SMALLINT	Yes	DECIMAL(38)	
TIMESTAMP	Yes	DATETIME	
TIMESTAMP(p)	Yes	DATETIME(p)	
TIMESTAMP(p)	Partial		MySQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) precision.
TIMESTAMP(p) WITH TIME ZONE	Partial	DATETIME	There is not a data type in MySQL that can store time zone information. DATETIME data type stores timestamps in the MySQL server time zone.
UROWID	Yes	TEXT	
UROWID(n)	Yes	VARCHAR(n)	
VARCHAR	Yes	TEXT	
VARCHAR(n)	Yes	VARCHAR(n)	
VARCHAR2	Yes	TEXT	
VARCHAR2(n)	Yes	VARCHAR(n)	
XMLTYPE	Yes	LONGTEXT	

Data Definition Language (DDL)

Topics

- [CREATE TABLE \(p. 267\)](#)
- [CREATE INDEX \(p. 267\)](#)
- [CREATE TRIGGER \(p. 267\)](#)
- [CREATE VIEW \(p. 269\)](#)
- [CREATE CONSTRAINT \(p. 269\)](#)
- [CREATE SEQUENCE \(p. 270\)](#)

CREATE TABLE

CREATE TABLE statements are converted automatically except for the following.

Clause	Automatically converted	Details
OBJECT TABLE	No	MySQL doesn't support OBJECT TABLE. Revise your code to avoid OBJECT TABLE.
CLUSTERED TABLE	No	MySQL doesn't support CLUSTERED TABLE. Try using a table with triggers.
EXTERNAL TABLE	No	MySQL doesn't support EXTERNAL TABLE. Try using a table.
GLOBAL TEMPORARY TABLE	No	MySQL doesn't support GLOBAL TEMPORARY TABLE. Try using a temporary table.
Partitioned tables	No	MySQL doesn't support all partition types. Perform a manual conversion for the partition types that aren't supported.
Virtual columns	No	MySQL doesn't support virtual columns. Represent virtual columns using a view.
Functions as default values	No	MySQL doesn't support functions as default values. Try using a trigger.
NESTED TABLE columns	No	MySQL doesn't support tables with NESTED TABLE columns. Revise any tables with NESTED TABLE columns to avoid these columns.
COLUMN(TABLE)	No	MySQL doesn't support the objects column. Create a user-defined function.

CREATE INDEX

CREATE INDEX statements are converted automatically except that MySQL does not support bitmap, function-based, or domain indexes.

CREATE TRIGGER

CREATE TRIGGER statements are converted automatically except for the following.

Clause	Automatically converted	Details
COMPOUND	No	MySQL doesn't support compound triggers. Create a single trigger for each part of the compound trigger.
INSTEAD OF	No	MySQL doesn't support INSTEAD OF. Try using a BEFORE trigger.
INSERTING UPDATING DELETING	No	MySQL doesn't support conditional predicates.

Clause	Automatically converted	Details
without FOR EACH ROW	No	MySQL doesn't support statement triggers. Try including FOR EACH ROW.
FOLLOWS PRECEDES	No	MySQL doesn't support the FOLLOWS PRECEDES clause. Try using a FOR EACH ROW trigger.
UPDATE OF	No	MySQL doesn't support the UPDATE OF clause. Try using a FOR EACH ROW trigger.
REFERENCING NEW AS OLD AS	No	MySQL doesn't support the REFERENCING clauses. Modify references to pseudorows to use OLD and NEW instead.
WHEN(condition)	No	MySQL doesn't support WHEN(condition) triggers. Apply the condition for WHEN in the trigger body.
ON NESTED TABLE	No	
trigger-status=DISABLED	No	MySQL doesn't support the DISABLED clause. Drop the trigger instead.
create-date	No	MySQL doesn't support create-date for triggers. Try using a FOR EACH ROW trigger.
modify-date	No	MySQL doesn't support modify-date for triggers. Try using a FOR EACH ROW trigger.
status	No	MySQL doesn't support trigger status values. Try using a FOR EACH ROW trigger.
description	No	<p>MySQL doesn't support descriptions for triggers.</p> <p>In MySQL, you can get a trigger description using INFORMATION_SCHEMA.TRIGGERS. MySQL doesn't support the REFERENCING clause. The following are the corresponding values between Oracle(USER_TRIGGERS) and MySQL(INFORMATION_SCHEMA.TRIGGERS):</p> <ul style="list-style-type: none"> • TABLE_OWNER corresponds to TRIGGER_SCHEMA • TRIGGER_NAME corresponds to TRIGGER_NAME • TRIGGER_TYPE corresponds to ACTION_TIMING • TRIGGERING_EVENT corresponds to EVENT_MANIPULATION • TABLE_NAME corresponds to EVENT_OBJECT_TABLE • REFERENCING_NAMES corresponds to ACTION_REFERENCE_NEW_ROW(always is NEW), ACTION_REFERENCE_OLD_ROW(always is OLD)
triggering-event=ddl_events	No	
triggering-event=database_events	No	
base-object-type=VIEW	No	
base-object-type=SCHEMA	No	

Clause	Automatically converted	Details
base-object-type=DATABASE	No	
action-type=CALL or PL/SQL	No	
crossedition=FORWARD/REVERSE	No	
fire-once=YES	No	
apply-server-only=YES	No	
referencing-names=PARENT	No	

CREATE VIEW

CREATE VIEW statements are converted automatically except for the following.

Clause	Automatically converted	Details
FORCE	No	
NOFORCE	No	NOFORCE is the default behavior for MySQL. You don't need to specify this option.
WITH READ ONLY	No	
create-date	No	
modify-date	No	
status	No	
oid-text	No	
view-type-owner	No	
view-type	No	
superview-name	No	
editioning-view	No	
encrypted-view	No	
CAST	No	MySQL doesn't support views with nested table columns.

CREATE CONSTRAINT

CREATE CONSTRAINT statements are converted automatically except for the following.

Clause	Automatically converted	Details
DISABLE, status=DISABLED	No	My SQL doesn't support constraints with the status DISABLE. Drop the constraint.
Foreign keys of different types	No	MySQL doesn't support foreign keys with different types of columns and referenced columns. Modify the column types using an equivalent of the column data types in the original database.
constraint-type	No	MySQL doesn't support the constraint check option on a view. Try using a trigger.

CREATE SEQUENCE

MySQL doesn't support sequences. Try developing a system for sequences in your application.

Cursors

Cursors are converted automatically except for the following.

Clause	Automatically converted	Details
SQLAttribute	Partial	MySQL doesn't support the cursor attribute <code>SQL%ISOPEN</code> MySQL doesn't support the cursor attribute <code>SQL%BULK_ROWCOUNT</code> . To calculate processed rows, use a variable.
REF CURSOR	No	MySQL doesn't support the REF CURSOR object. Perform a manual conversion.
SYS_REFCURSOR	No	MySQL doesn't support a variable of SYS_REFCURSOR type. Perform a manual conversion.
CURSOR	Partial	The AWS Schema Conversion Tool cannot automatically transform the SELECT statement for an implicit cursor. Try rewriting the SELECT statement.
REF_CURSOR variable	Partial	The SELECT statement for the REF_CURSOR variable cannot be transformed. Try rewriting the SELECT statement.

Hints

MySQL doesn't support hints. Try using MySQL performance tuning methods or perform a manual conversion.

Clause	Automatically converted	Details
<code>/*+ ALL_ROWS */</code>	No	

Clause	Automatically converted	Details
<code>/*+ AND_EQUAL(table index...) */</code>	No	
<code>/*+ APPEND */</code>	No	
<code>/*+ CACHE(table) */</code>	No	
<code>/*+ CHOOSE */</code>	No	
<code>/*+ CLUSTER(table) */</code>	No	
<code>/*+ CURSOR_SHARING_EXACT */</code>	No	
<code>/*+ DRIVING_SITE(table) */</code>	No	
<code>/*+ DYNAMIC_SAMPLING */</code>	No	
<code>/*+ EXPAND_GSET_TO_UNION */</code>	No	
<code>/*+ FACT(table) */</code>	No	
<code>/*+ FIRST_ROWS(n) */</code>	No	
<code>/*+ FULL(table) */</code>	No	
<code>/*+ HASH */</code>	No	
<code>/*+ HASH_SJ */</code>	No	
<code>/*+ INDEX(table index) */</code>	No	
<code>/*+ INDEX_ASC(table index) */</code>	No	
<code>/*+ INDEX_COMBINE(table index) */</code>	No	
<code>/*+ INDEX_DESC(table index) */</code>	No	
<code>/*+ INDEX_FFS(table index) */</code>	No	
<code>/*+ LEADING(table) */</code>	No	
<code>/*+ MERGE(table) */</code>	No	
<code>/*+ MERGE_SJ */</code>	No	
<code>/*+ NL_SJ */</code>	No	
<code>/*+ NO_EXPAND */</code>	No	
<code>/*+ NO_INDEX(table index) */</code>	No	

Clause	Automatically converted	Details
<code>/*+ NO_MERGE(table) */</code>	No	
<code>/*+ NO_PARALLEL(table) */</code>	No	
<code>/*+ NO_PARALLEL_INDEX(table index) */</code>	No	
<code>/*+ NO_PUSH_PRED(subquery) */</code>	No	
<code>/*+ NO_PUSH_SUBQ(subquery) */</code>	No	
<code>/*+ NO_REWRITE */</code>	No	
<code>/*+ NO_UNNEST */</code>	No	
<code>/*+ NOAPPEND */</code>	No	
<code>/*+ NOCACHE(table) */</code>	No	
<code>/*+ NOFACT(table) */</code>	No	
<code>/*+ ORDERED */</code>	No	
<code>/*+ ORDERED_PREDICATES */</code>	No	
<code>/*+ PARALLEL(table server_num) */</code>	No	
<code>/*+ PARALLEL_INDEX(table index server_num) */</code>	No	
<code>/*+ PQ_DISTRIBUTE(table out_distr in_distr) */</code>	No	
<code>/*+ PUSH_PRED(subquery) */</code>	No	
<code>/*+ PUSH_SUBQ(subquery) */</code>	No	
<code>/*+ REWRITE */</code>	No	
<code>/*+ ROWID(table) */</code>	No	
<code>/*+ RULE */</code>	No	
<code>/*+ STAR */</code>	No	
<code>/*+ STAR_TRANSFORMATION */</code>	No	

Clause	Automatically converted	Details
<code>/*+ UNNEST */</code>	No	
<code>/*+ USE_CONCAT */</code>	No	
<code>/*+ USE_HASH(table1 table2) */</code>	No	
<code>/*+ USE_MERGE(table1 table2) */</code>	No	

Exceptions

Item	Automatically converted	Details
RAISE	No	MySQL doesn't support the RAISE command. Review the exception, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.
%s	No	MySQL doesn't support the %s exception. Review the exception, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.
EXCEPTION	No	MySQL doesn't support the EXCEPTION declaration. Use the DECLARE ... CONDITION statement.
PRAGMA EXCEPTION_INIT	No	MySQL doesn't support the PRAGMA EXCEPTION_INIT declaration. Use the DECLARE ... CONDITION statement.
PROC_RAISE_APPLICATION_ERROR	No	MySQL doesn't support the PROC_RAISE_APPLICATION_ERROR statement. Use the DECLARE ... CONDITION statement. Try using the SIGNAL or RESIGNAL statement.

Built-In Exceptions

Exception	Automatically converted	Details
INVALID_NUMBER	No	Try creating a user exception.
TIMEOUT_ON_RESOURCE	No	Try creating a user exception.
TRANSACTION_BACKED_OUT	No	Try creating a user exception.
INVALID_CURSOR	No	You can use one of several handlers for invalid cursors. Choose the correct handler based on the issue.
NOT_LOGGED_ON	No	Try creating a user exception.
LOGIN_DENIED	No	Try creating a user exception.

Exception	Automatically converted	Details
STORAGE_ERROR	No	You can use one of several handlers for invalid cursors. Choose the correct handler based on the issue.
PROGRAM_ERROR	No	Try using the handler "1815 - Internal Error."

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Oracle to PostgreSQL Supported Schema Conversion

The following sections list the schema elements from an Oracle database and whether they are supported for automatic conversion to PostgreSQL using the AWS Schema Conversion Tool.

DDL

Statements

Clause	Automatically Converted	Details
ALTER CLUSTER	No	
ALTER DATABASE	No	
ALTER DIMENSION	No	
ALTER DISKGROUP	No	
ALTER FLASHBACK ARCHIVE	No	
ALTER FUNCTION	No	
ALTER TABLESPACE	No	
CREATE [OR REPLACE] FUNCTION	Yes	
CREATE [TEMPORARY] TABLESPACE	No	
CREATE CLUSTER	No	
CREATE CONTEXT	No	

Clause	Automatically Converted	Details
CREATE CONTROLFILE	No	
CREATE DATABASE	Yes	
CREATE DATABASE LINK	No	
CREATE DIMENSION	No	
CREATE DIRECTORY	No	
CREATE DISKGROUP	No	
CREATE FLASHBACK ARCHIVE	No	
DROP CLUSTER	No	
DROP DATABASE	No	
DROP DATABASE LINK	No	
DROP DIMENSION	No	
DROP DIRECTORY	No	
DROP DISKGROUP	No	
DROP FLASHBACK ARCHIVE	No	
DROP FUNCTION	No	
DROP TABLE	No	
DROP TABLESPACE	No	
DROP VIEW	Yes	

ALTER TABLE

Clause	Automatically Converted	Details
ADD CONSTRAINT	Yes	
ADD column	No	
DEALLOCATE UNUSED	No	
DISCARD/IMPORT TABLESPACE	No	
DROP CONSTRAINT	No	
ENABLE/DISABLE constraint	No	

Clause	Automatically Converted	Details
ENABLE/DISABLE triggers	No	
MODIFY	No	
MODIFY COLUMN	No	
PARTITION operations	No	
REMOVE PARTITIONING	No	
RENAME	No	
RENAME COLUMN	No	

CREATE [OR REPLACE] VIEW Statement

Clause	Automatically Converted	Details
FORCE	Yes	
NOFORCE	Yes	
READ ONLY	Yes	
Updatable Views	Yes	
WITH OBJECT IDENTIFIER	Yes	

CREATE INDEX Statement

Clause	Automatically Converted	Details
BITMAP	No	Issue 5206: PostgreSQL doesn't support bitmap indexes (p. 318)
DOMAIN	No	Issue 5208: PostgreSQL doesn't support domain indexes (p. 318)
FUNCTION-BASED BITMAP	No	Issue 5206: PostgreSQL doesn't support bitmap indexes (p. 318)
FUNCTION-BASED NORMAL	Yes	Issue 5555: PostgreSQL doesn't support functional indexes that aren't single-column (p. 318)
NONUNIQUE	Yes	
NORMAL	Yes	
UNIQUE	Yes	

CREATE TABLE

Clause	Automatically Converted	Details
Clustered tables	No	Issue 5199: PostgreSQL doesn't support CLUSTERED TABLE (p. 318)
External table	No	Issue 5200: PostgreSQL doesn't support EXTERNAL TABLES (p. 319)
Function as default value for column	Yes	
Object table	No	Issue 5196: PostgreSQL doesn't support OBJECT TABLE (p. 319)
Partitioned table	No	Issue 5201: PostgreSQL doesn't support all partition types (p. 318)
Regular tables	Yes	
Temporary tables	Partial	Issue 5198: PostgreSQL doesn't support GLOBAL TEMPORARY TABLE (p. 319)

Sequences

Clause	Automatically Converted	Details
CREATE	Yes	
Using	Yes	

DML

Built-in SQL Functions

Aggregate Functions

Clause	Automatically Converted	Details
AVG	Yes	
COLLECT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CORR	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CORR_*	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Clause	Automatically Converted	Details
COUNT	Yes	
COVAR_POP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
COVAR_SAMP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CUME_DIST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
DENSE_RANK	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
FIRST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
GROUP_ID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
GROUPING	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
GROUPING_ID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
LAST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
MAX	Yes	
MEDIAN	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
MIN	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PERCENT_RANK	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PERCENTILE_CONT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PERCENTILE_DISC	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
RANK	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REGR_ (Linear Regression) Functions	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_BINOMIAL_TEST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_CROSSTAB	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Clause	Automatically Converted	Details
STATS_F_TEST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_KS_TEST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_MODE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_MW_TEST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_ONE_WAY_ANOVA	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_T_TEST_*	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STATS_WSR_TEST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STDDEV	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STDDEV_POP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
STDDEV_SAMP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SUM	Yes	
VAR_POP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VAR_SAMP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VARIANCE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Analytic Functions

Clause	Automatically Converted	Details
AVG *	Yes	
CORR *	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
COUNT *	Yes	
COVAR_POP *	Yes	
COVAR_SAMP *	Yes	

Clause	Automatically Converted	Details
CUME_DIST	Yes	
DENSE_RANK	Yes	
FIRST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
FIRST_VALUE *	Yes	
LAG	Yes	
LAST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
LAST_VALUE *	Yes	
LEAD	Yes	
MAX *	Yes	
MIN *	Yes	
NTILE	Yes	
PERCENT_RANK	Yes	
PERCENTILE_CONT	Yes	
PERCENTILE_DISC	Yes	
RANK	Yes	
RATIO_TO_REPORT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REGR_ (Linear Regression) Functions *	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
ROW_NUMBER	Yes	
STDDEV *	Yes	
STDDEV_POP *	Yes	
STDDEV_SAMP *	Yes	
SUM *	Yes	
VAR_POP *	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VAR_SAMP *	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VARIANCE *	Yes	

Character Functions Returning Character Values

Clause	Automatically Converted	Details
CHR(num)	Yes	
CONCAT(char1, char2)	Yes	
INITCAP(string)	Yes	
LOWER(string)	Yes	
LPAD(string, len)	Yes	
LPAD(string, len, pad)	Yes	
LTRIM(string)	Yes	
NLS_INITCAP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NLS_LOWER	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NLS_UPPER	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NLSSORT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REGEXP_REPLACE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REGEXP_SUBSTR	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REPLACE(str, search)	Yes	
REPLACE(str, search, replace)	Yes	
RPAD(string, len)	Yes	
RTRIM(string)	Yes	
RTRIM(string, set)	Yes	
SOUNDEX(string)	Yes	
SUBSTR(string, pos, len)	Yes	
TRANSLATE(string, from, to)	Yes	
TREAT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TRIM([type trim FROM] string)	Yes	

Clause	Automatically Converted	Details
UPPER(string)	Yes	

Character Functions Returning Number Values

Clause	Automatically Converted	Details
ASCII(str)	Yes	
INSTR(str, substr)	Yes	
INSTR(str, substr, pos)	Implemented in Extension Library	
INSTR(str, substr, pos, num)	Implemented in Extension Library	
LENGTH(string)	Yes	
REGEXP_INSTR	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Collection Functions

Clause	Automatically Converted	Details
CARDINALITY	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
COLLECT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SET	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Conversion Functions

Clause	Automatically Converted	Details
ASCIISTR(string)	Implemented in Extension Library	
BIN_TO_NUM(bit1, bit2, ...)	Yes	
CAST	Yes	

Clause	Automatically Converted	Details
CHARTOROWID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
COMPOSE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CONVERT(string, charset)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
DECOMPOSE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
HEXTORAW	Yes	
NUMTODSINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NUMTOYMINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
RAWTOHEX	Yes	
RAWTONHEX	Yes	
ROWIDTOCHAR	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
ROWIDTONCHAR	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SCN_TO_TIMESTAMP	Yes	
TIMESTAMP_TO_SCN	Yes	
TO_BINARY_DOUBLE	Yes	
TO_BINARY_FLOAT	Yes	
TO_CHAR (character)	Yes	
TO_CHAR (datetime, format)	Yes	
TO_CHAR (number, format)	Implemented in Extension Library	
TO_CLOB	Yes	
TO_DATE	Implemented in Extension Library	
TO_DSINTERVAL	Yes	
TO_LOB	Yes	
TO_MULTI_BYTE	Yes	

Clause	Automatically Converted	Details
TO_NCHAR (character)	Yes	
TO_NCHAR (datetime)	Implemented in Extension Library	
TO_NCHAR (number)	Implemented in Extension Library	
TO_NCLOB	Implemented in Extension Library	
TO_NUMBER	Implemented in Extension Library	
TO_SINGLE_BYTE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TRANSLATE ... USING	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
UNISTR	Implemented in Extension Library	

Data Mining Functions

Clause	Automatically Converted	Details
CLUSTER_ID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CLUSTER_PROBABILITY	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
CLUSTER_SET	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
FEATURE_ID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
FEATURE_SET	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
FEATURE_VALUE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PREDICTION	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Clause	Automatically Converted	Details
PREDICTION_COST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PREDICTION_DETAILS	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PREDICTION_PROBABILITY	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PREDICTION_SET	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Datetime Functions

Clause	Automatically Converted	Details
ADD_MONTHS(date, num)	Implemented in Extension Library	
CURRENT_DATE	Yes	
CURRENT_TIMESTAMP	Yes	
DBTIMEZONE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
EXTRACT(DAY FROM date)	Yes	
EXTRACT(HOUR FROM time)	Yes	
EXTRACT(MINUTE FROM time)	Yes	
EXTRACT(MONTH FROM date)	Yes	
EXTRACT(SECOND FROM time)	Yes	
EXTRACT(YEAR FROM date)	Yes	
FROM_TZ	Implemented in Extension Library	
LAST_DAY(date)	Implemented in Extension Library	
LOCALTIMESTAMP	Yes	
LOCALTIMESTAMP([prec])	Yes	

Clause	Automatically Converted	Details
MONTHS_BETWEEN(date1, date2)	Yes	
NEW_TIME	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NEXT_DAY	Implemented in Extension Library	
NUMTODSINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NUMTOYMINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
ROUND (date)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SESSIONTIMEZONE	Yes	
SYS_EXTRACT_UTC	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYSDATE	Yes	
SYSTIMESTAMP	Yes	
TO_CHAR (datetime, format)	Yes	
TO_DSINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TO_TIMESTAMP(exp)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TO_TIMESTAMP_TZ	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TO_YMINTERVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TRUNC (datetime)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
TZ_OFFSET	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

dbms_output

Clause	Automatically Converted	Details
dbms_output.put()	Yes	

Clause	Automatically Converted	Details
dbms_output.put_line()	Yes	

Encoding and Decoding Functions

Clause	Automatically Converted	Details
DECODE(exp, when, then, ...)	Yes	
DUMP	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
ORA_HASH	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VSIZE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Environment and Identifier Functions

Clause	Automatically Converted	Details
SYS_CONTEXT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYS_GUID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYS_TYPEID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
UID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
USER	Yes	
USERENV('parameter')	Yes	

General Comparison Functions

Clause	Automatically Converted	Details
GREATEST(exp, exp2, ...)	Yes	
LEAST(exp, exp2, ...)	Yes	

Hierarchical Functions

Clause	Automatically Converted	Details
SYS_CONNECT_BY_PATH	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Large Object Functions

Clause	Automatically Converted	Details
BFILENAME	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
EMPTY_BLOB, EMPTY_CLOB	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Model Functions

Clause	Automatically Converted	Details
CV	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
ITERATION_NUMBER	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PRESENTNNV	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PRESENTV	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PREVIOUS	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

NLS Character Functions

Clause	Automatically Converted	Details
NLS_CHARSET_DECL_LEN	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NLS_CHARSET_ID	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NLS_CHARSET_NAME	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

NULL-Related Functions

Clause	Automatically Converted	Details
COALESCE(exp1, exp2, ...)	Yes	
LNNVL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
NULLIF(exp1, exp2)	Yes	
NVL(exp, replacement)	Yes	
NVL2(exp1, exp2, exp3)	Yes	

Numeric Functions

Clause	Automatically Converted	Details
ABS(num)	Yes	
ACOS(num)	Yes	
ASIN(num)	Yes	
ATAN(num)	Yes	
ATAN2(x,y)	Yes	
BITAND(exp1, exp2)	Yes	
CEIL(num)	Yes	
COS(num)	Yes	
COSH(num)	Yes	
EXP(n)	Yes	
FLOOR(num)	Yes	
LN(num)	Yes	
LOG(num1, num2)	Yes	
MOD(dividend, divisor)	Yes	
NANVL(n2, n1)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
POWER(value, n)	Yes	
REMAINDER(n1, n2)	Yes	
ROUND (num, integer)	Yes	
SIGN(exp)	Yes	

Clause	Automatically Converted	Details
SIN(num)	Yes	
SINH(num)	Yes	
SQRT(num)	Yes	
TAN(num)	Yes	
TANH(num)	Yes	
TRUNC (number)	Yes	
WIDTH_BUCKET	Yes	

Object Reference Functions

Clause	Automatically Converted	Details
DEREF	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
MAKE_REF	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REF	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
REFTOHEX	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
VALUE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

XML Functions

Clause	Automatically Converted	Details
APPENDCHILDXML	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
DELETEXML	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
DEPTH	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
EXISTSNODE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
EXTRACT (XML)	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Clause	Automatically Converted	Details
EXTRACTVALUE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
INSERTCHILDXML	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
INSERTXMLBEFORE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
PATH	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYS_DBURIGEN	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYS_XMLAGG	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
SYS_XMLGEN	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
UPDATEXML	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLAGG	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLCDATA	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLCOLATTVAL	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLCOMMENT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLCONCAT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLFOREST	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLPARSE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLPI	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLQUERY	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLROOT	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLSEQUENCE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Clause	Automatically Converted	Details
XMLSERIALIZE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLTABLE	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)
XMLTRANSFORM	No	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)

Hints

Clause	Automatically Converted	Details
Index and Cluster	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Join Order	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Joins	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Optimizer Mode	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Parallel Execution	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Query Transformation	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)
Other	No	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)

Joins

Clause	Automatically Converted	Details
Cross Joins	Yes	
Inner Joins	Yes	
Outer Joins	Yes	

Operators and Date Functions

Clause	Automatically Converted	Details
Arithmetic operators	Yes	
Concatenation operators	Yes	

Predicates

Clause	Automatically Converted	Details
Boolean conditions	Yes	
Comparison conditions: '=', '<>', '<', '<=', '>', '>=', ANY, SOME, ALL	Yes	
Exists conditions	Yes	
In conditions	Yes	
Null conditions	Yes	
Pattern matching conditions	Yes	
Range conditions	Yes	

Set operators

Concatenation operators

Clause	Automatically Converted	Details
INTERSECT, EXCEPT	Yes	
MINUS	Yes	
UNION, UNION ALL	Yes	

Statements

DELETE

Clause	Automatically Converted	Details
DELETE FROM MATERIALIZED VIEW	No	Issue 5095: PostgreSQL doesn't support using a materialized VIEW (p. 324)
DELETE FROM SUBQUERY	No	Issue 5068: PostgreSQL doesn't support the DELETE statement for a subquery (p. 322)
DELETE FROM TABLE	Yes	
DELETE FROM VIEW	Yes	
DELETE with Error Logging clause	No	Issue 5067: PostgreSQL doesn't support the DELETE statement with the ERROR LOG option (p. 321)
DELETE with RETURNING clause	Yes	

HIERARCHICAL QUERY AND HIERARCHICAL QUERY PSEUDOCOLUMNS

Clause	Automatically Converted	Details
CONNECT BY CLAUSE	Yes	
CONNECT BY NOCYCLE	No	Issue 5586: Automatic conversion for queries with NOCYCLE clause not supported (p. 326)
CONNECT_BY_ISLEAF pseudocolumn	Yes	
LEVEL PSEUDOCOLUMN	Yes	
ORDER SIBLINGS BY	Yes	
START WITH CLAUSE	Yes	
SYS_CONNECT_BY_PATH	Yes	

Hierarchical query operators

Clause	Automatically Converted	Details
CONNECT_BY_ROOT	Yes	
PRIOR	Yes	

INSERT

Clause	Automatically Converted	Details
ALL	Yes	
DEFAULT	Yes	
ELSE	Yes	
FIRST	Yes	
INSERT from clause "SUBQUERY"	Yes	
INSERT from clause "VALUES(...)"	Yes	
INSERT INTO SUBQUERY	No	Issue 5071: PostgreSQL doesn't support the INSERT statement for a subquery (p. 324)
INSERT INTO TABLE	Yes	
INSERT INTO VIEW	Yes	
INSERT with partition_extension_clause	No	Issue 5024: PostgreSQL doesn't support the INSERT statement with partition_extension_clause (p. 324)
Inserting Into a Table with Error Logging clause	No	Issue 5070: PostgreSQL doesn't support the INSERT statement with the ERROR LOG option (p. 324)
RETURNING INTO	Yes	
WHEN	Yes	

LOCK TABLE

Clause	Automatically Converted	Details
Locking table in exclusive mode	Yes	
Locking table in share mode	Yes	
Locking without waiting	Yes	

SELECT

Clause	Automatically Converted	Details
Add or subtract operators	Yes	

Clause	Automatically Converted	Details
BULK COLLECT INTO	No	Issue 5140: PostgreSQL doesn't support BULK COLLECT INTO (p. 326)
CUBE	No	Issue 5557: PostgreSQL doesn't support GROUPING SETS, CUBE, and ROLLUP functions (p. 326)
FOR UPDATE	No	Issue 5139: PostgreSQL doesn't support FOR UPDATE SKIP LOCKED (p. 326)
GROUP BY	Yes	
GROUPING SETS	No	Issue 5557: PostgreSQL doesn't support GROUPING SETS, CUBE, and ROLLUP functions (p. 326)
HAVING	Yes	
INTERSECT	Yes	
INTO	Yes	
MINUS	Yes	
MODEL	No	Issue 5126: PostgreSQL doesn't support the MODEL statement (p. 326)
ORDER BY	Yes	
PIVOT clause	No	Issue 5077: PostgreSQL doesn't support the PIVOT clause for the SELECT statement (p. 332)
ROLLUP	No	Issue 5557: PostgreSQL doesn't support GROUPING SETS, CUBE, and ROLLUP functions (p. 326)
ROWNUM pseudocolumn	Yes	
SELECT ALL	Yes	
SELECT DISTINCT	Yes	
SELECT LIST	Yes	
SELECT UNIQUE	Yes	
UNION	Yes	
UNION ALL	Yes	
UNPIVOT clause	Yes	
WHERE(field_name1, ...,fieldnameN) IN (subquery_with_multiple_fields)	Yes	
WHERE comparison_condition (=;<>!=;<=>>=>=<=>ANY;SOME;ALL)	Yes	

Clause	Automatically Converted	Details
WHERE compound_condition_with_or_and_not	Yes	
WHERE field_name BETWEEN arg1 AND arg2	Yes	
WHERE field_name IN (const1,...,constN)	Yes	
WHERE field_name IN (subquery)	Yes	
WHERE like_condition (operator LIKE)	Yes	
WITH	Yes	

UPDATE

Clause	Automatically Converted	Details
UPDATE (subquery)	No	Issue 5065: PostgreSQL doesn't support the UPDATE statement for a subquery (p. 331)
UPDATE MATERIALIZED VIEW	No	Issue 5095: PostgreSQL doesn't support using a materialized VIEW (p. 324)
UPDATE TABLE	Yes	
UPDATE VIEW	Yes	
UPDATE with Error Logging clause	No	Issue 5064: PostgreSQL doesn't support the UPDATE statement with the ERROR LOG option (p. 331)
UPDATE with partition extension clause	No	Issue 5558: PostgreSQL doesn't support the UPDATE statement for a PARTITION (p. 331)
UPDATE with RETURNING... INTO clause	Yes	
UPDATE with subqueries into SET-clause	Yes	
UPDATE-operator which based on data from others tables	Yes	

MERGE

Clause	Automatically Converted	Details
MERGE	No	Issue 5102: PostgreSQL doesn't support the MERGE statement (p. 325)
TRUNCATE TABLE	Yes	

PLSQL

Arguments of functions and procedures

Clause	Automatically Converted	Details
Built-in datatype scalar arguments	Yes	
Cursor-type arguments	Yes	
Default clause in declaration of arguments	Yes	
IN, OUT, IN OUT type of arguments	Yes	
Ref cursor-type arguments	Yes	
UDT collection-type arguments	Yes	
UDT object-type arguments	Yes	
UDT record-type	Yes	

Functions

Clause	Automatically Converted	Details
Functions with DML	Yes	
Functions without DML	Yes	
Return pipelined table	Yes	
Return ref cursor	Yes	
Return scalar	Yes	
Return table	Yes	

Clause	Automatically Converted	Details
Return UDT collection-type	Yes	
Return UDT object-type	Yes	
Return UDT record-type	Yes	

Packages

Clause	Automatically Converted	Details
Global cursor declaration	No	Issue 5569: PostgreSQL only supports session variables using SQL standard date and time types (p. 325)
Global user exception declaration	No	Issue 5569: PostgreSQL only supports session variables using SQL standard date and time types (p. 325)
Initialization block BEGIN ... END in packages	Yes	
Package functions	Yes	
Package procedures	Yes	
Package variables	Yes	
User type declaration	Yes	

PL SQL constructions

Clause	Automatically Converted	Details
block EXCEPTION when exception_name then...	Yes	
CASE_NOT_FOUND	Yes	
case when conditions then...else... end case;	Yes	
close cursor_variable	Yes	
COMMIT	Yes	
CURSOR_ALREADY_OPEN	Yes	
DUP_VAL_ON_INDEX	Yes	
execute immediate SQL_string_variable	No	Issue 5334: PostgreSQL doesn't support the dynamic SQL statement (p. 322)

Clause	Automatically Converted	Details
into variables_list use variables_list		
fetch cursor_variable into variables_list	Yes	
FORALL -cycle	No	Issue 5121: PostgreSQL doesn't support the FORALL statement (p. 330)
for I in(begN..endN) loop... end loop	Yes	
for I in(cursor_name) loop... end loop	Yes	
for I in(select_statement) loop... end loop	Yes	
FOR i IN REVERSE begN..endN LOOP... END LOOP;	Yes	
goto label;	No	Issue 5335: PostgreSQL doesn't support the GOTO operator (p. 327)
if conditions then...else if conditions then... end if;	Yes	
if conditions then... end if;	Yes	
INVALID_CURSOR	Yes	
INVALID_NUMBER	Yes	
LOGIN_DENIED	Yes	
LOOP...exit when conditions; END LOOP;	Yes	
NO_DATA_FOUND	Yes	
NOT_LOGGED_ON	Yes	
NULL Statement	Yes	
open cursor_variable	Yes	
open cursor_variable for select_statemet_string_variable	Yes	
others	Yes	
PRAGMA AUTONOMOUS_TRANSACTION	No	Issue 5350: The function can't use statements that explicitly or implicitly begin or end a transaction, such as START TRANSACTION, COMMIT, or ROLLBACK (p. 323)
PROGRAM_ERROR	Yes	

Clause	Automatically Converted	Details
raise_application_error(except_message, message)	Yes	
raise exception_name	Yes	
ROLLBACK	Yes	
select...bulk collect into table_variable from table_name	No	Issue 5140: PostgreSQL doesn't support BULK COLLECT INTO (p. 326)
SQL%BULK_ROWCOUNT	No	
SQL%FOUND	Yes	
SQL%ISOPEN	No	Issue 5084: PostgreSQL doesn't support the cursor attribute %ISOPEN (p. 321)
SQL%NOTFOUND	Yes	
SQL%ROWCOUNT	Yes	
STORAGE_ERROR	Yes	
TIMEOUT_ON_RESOURCE	Yes	
TOO_MANY_ROWS	Yes	
TRANSACTION_BACKED_OUT	Yes	
VALUE_ERROR	Yes	
while conditions loop... end loop;	Yes	
ZERO_DIVIDE	Yes	

PL SQL declaration block

Clause	Automatically Converted	Details
Declare built-in scalar variable	Yes	
Declare ref-cursor variable	Yes	
Declare static cursor	Yes	
Declare UDT collection variable	Yes	
Declare UDT object variable	Yes	
Declare UDT record variable	Yes	

Clause	Automatically Converted	Details
Declare UDT array types	Yes	
Declare user-exception	Yes	
Declare with %rowtype option	Yes	
Declare with %type option	Yes	

Procedures

Clause	Automatically Converted	Details
CREATE PROCEDURE statement	Yes	

Triggers

Clause	Automatically Converted	Details
BEFORE, AFTER - options	Yes	
FOR EACH ROW -option	Yes	
INSTEAD OF - options	Yes	
NESTED TABLE - clause	No	Issue 5240: PostgreSQL doesn't support triggers on nested table columns in views (p. 329)
REFERENCING - clause	No	Issue 5238: PostgreSQL doesn't support the REFERENCING clauses (p. 329)
Statement-trigger	No	
trigger on TABLE	Yes	
Trigger on VIEW	Yes	
WHEN(condition)- option	No	

Data Types

Data type	Automatically Converted to	Details
BFILE	character varying	Issue 5212: PostgreSQL doesn't have a data type BFILE (p. 318)
BINARY_FLOAT	real	
BINARY_DOUBLE	double precision	
BLOB	bytea	
CHAR(n)	character	
CHARACTER(n)	character	
CLOB	text	
DATE	timestamp(0) without time zone	
DECIMAL(p,s)	numeric	
DEC(p,s)	numeric	
FLOAT	double precision	
INTEGER	numeric	
INT	numeric	
INTERVAL YEAR(p) TO MONTH	interval year to month	
INTERVAL DAY(p) TO SECOND(s)	interval day to second(s)	
LONG	text	
LONG RAW	bytea	
NCHAR(n)	character	
NCHAR VARYING(n)	character varying	
NCLOB	text	
NUMBER(p)	numeric	
NUMBER(p,0)	numeric	
NUMBER(*,0)	numeric	
NUMBER(p,s)	numeric	

Data type	Automatically Converted to	Details
NUMBER(*)	double precision	
NUMBER	double precision	
NVARCHAR2(n)	character varying	
RAW(n)	bytea	
ROWID	character	Issue 5550: PostgreSQL doesn't have a data type ROWID (p. 319)
TIMESTAMP(p)	timestamp(p) without time zone	
TIMESTAMP(p)	timestamp(6) without time zone	Issue 5213: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)
TIMESTAMP(p) WITH TIME ZONE	timestamp(p) with time zone	
TIMESTAMP(p) WITH TIME ZONE	timestamp(6) with time zone	Issue 5552: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)
TIMESTAMP(p) WITH LOCAL TIME ZONE	timestamp(p) without time zone	
TIMESTAMP(p) WITH LOCAL TIME ZONE	timestamp(6) without time zone	Issue 5553: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)
UROWID	character varying	Issue 5551: PostgreSQL doesn't have a data type UROWID (p. 320)
VARCHAR	character varying	
VARCHAR2	character varying	
XMLTYPE	xml	

Oracle to PostgreSQL Conversion Reference

Use the following sections to get information on the types of issues you might encounter during an Oracle to PostgreSQL conversion. Choose the link in the Issue column to get more detailed resolution information about the issue if it is available.

Build_in_services

Item	Issue	Resolution
UTL_SMTP DBMS_JOB	Issue 5590: The function was converted as procedure (p. 316)	Can't return a reply because this functionality is asynchronous.

Built-In SQL Functions

Item	Issue	Resolution
DateTime function	Issue 5584: The function %s depends on the time zone setting (p. 317)	Review the transformed code, and set time zone manually if necessary.
FUNCTION	Issue 5340: PostgreSQL doesn't support the %s function (p. 317)	Use suitable function or create user defined function.
GREATEST(exp, exp2, ...)	Issue 5271: The Oracle GREATEST function and PostgreSQL GREATEST function might give different results (p. 317)	Review your transformed code and modify it if necessary. Change an argument's type if necessary.
LEAST(exp, exp2, ...)	Issue 5272: The Oracle LEAST function and PostgreSQL LEAST function might give different results (p. 317)	Review your transformed code and modify it if necessary. Change an argument's type if necessary.
TO_CHAR, TO_DATE, TO_NUMBER	Issue 5579: The second parameter of %s function can be processed incorrectly (p. 317)	Check converted code and make manual corrections if it's necessary.

CREATE

Item	Issue	Resolution
	Issue 5099: Unable convert object due to %s not created (p. 317)	Review the %s object.
	Issue 5332: The object has references on the object in %s schema (p. 317)	Convert the object in %s schema also.
Encrypted Objects	Issue 5582: Encrypted Objects (p. 317)	Decrypt the object before conversion.

CREATE CONSTRAINT

Item	Issue	Resolution
status=DISABLED	Issue 5326: PostgreSQL doesn't support constraints with the status DISABLED (p. 318)	Drop the constraint.

CREATE INDEX

Item	Issue	Resolution
Creating a Domain Index	Issue 5208: PostgreSQL doesn't support domain indexes (p. 318)	Revise your code and try to use simple index.
Creating a Function-Based Index	Issue 5555: PostgreSQL doesn't support functional indexes that aren't single-column (p. 318)	Revise your code and try to use simple index.
Creating an BITMAP Index	Issue 5206: PostgreSQL doesn't support bitmap indexes (p. 318)	Revise your code and try to use simple index.

CREATE TABLE

Item	Issue	Resolution
BFILE Data Type	Issue 5212: PostgreSQL doesn't have a data type BFILE (p. 318)	Either store a named file with the data and create a routine that gets that file from the file system, or store the data blob inside your database.
CREATING A PARTITIONED TABLE	Issue 5201: PostgreSQL doesn't support all partition types (p. 318)	Perform a manual conversion for the partition types that aren't supported.
CREATING CLUSTERED TABLE	Issue 5199: PostgreSQL doesn't support CLUSTERED TABLE (p. 318)	Try using a table with triggers.
CREATING EXTERNAL TABLES	Issue 5200: PostgreSQL doesn't support EXTERNAL TABLES (p. 319)	Try using a table.
CREATING GLOBAL TEMPORARY TABLE	Issue 5198: PostgreSQL doesn't support GLOBAL TEMPORARY TABLE (p. 319)	Try using a local temporary table.
CREATING OBJECT TABLES	Issue 5196: PostgreSQL doesn't support OBJECT TABLE (p. 319)	Revise your code to avoid OBJECT TABLE.
Index-organized table	Issue 5581: PostgreSQL doesn't support index-organized table (p. 319)	Revise your architecture with a custom solution for the table type.
NESTED TABLE	Issue 5348: PostgreSQL doesn't support NESTED TABLES (p. 319)	Revise your code to avoid NESTED TABLE.

Item	Issue	Resolution
ROWID data type	Issue 5550: PostgreSQL doesn't have a data type ROWID (p. 319)	Review your transformed code and modify it if necessary.
TABLE WITH VIRTUAL COLUMNS	Issue 5554: PostgreSQL doesn't support virtual columns (p. 319)	They were emulated by a trigger.
TIMESTAMP(n>6) Data Type	Issue 5213: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)	Review your transformed code and modify it if necessary to avoid a loss of accuracy.
TIMESTAMP(n>6) WITH TIME ZONE data type	Issue 5552: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)	Review your transformed code and modify it if necessary to avoid a loss of accuracy.
TIMESTAMP(n>6) WITH LOCAL TIME ZONE data type	Issue 5553: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds (6 digits) of precision (p. 320)	Review your transformed code and modify it if necessary to avoid a loss of accuracy.
UROWID data type	Issue 5551: PostgreSQL doesn't have a data type UROWID (p. 320)	Review your transformed code and modify it if necessary.

CREATE TYPE

Item	Issue	Resolution
OBJECT TYPE	Issue 5572: PostgreSQL doesn't support object type methods (p. 320)	Revise your architecture with a custom solution for the user type.

CURSOR

Item	Issue	Resolution
Cursor attributes	Issue 5084: PostgreSQL doesn't support the cursor attribute %ISOPEN (p. 321)	Revise how you are using this attribute in your code.
TYPE .. IS REF CURSOR	Issue 5226: PostgreSQL doesn't support TYPE ... IS REF CURSOR (p. 321)	Change the function to a stored procedure, and try using a table to store results.

Datetime Expressions

Item	Issue	Resolution
at local at time zone	Issue 5594: PostgreSQL doesn't support %s datetime expression. The loss of precision and/or accuracy of data is possible (p. 321)	Check, if the data violates these restrictions. If so, perform a manual migration.

DELETE

Item	Issue	Resolution
ERROR LOG	Issue 5067: PostgreSQL doesn't support the DELETE statement with the ERROR LOG option (p. 321)	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.
PARTITION	Issue 5098: PostgreSQL doesn't support the DELETE statement for a PARTITION (p. 321)	Either insert data into the overlying partition, or perform a manual conversion using the DELETE statement.
QUERY	Issue 5068: PostgreSQL doesn't support the DELETE statement for a subquery (p. 322)	Perform this operation on the underlying tables instead.

DYNAMIC SQL

Item	Issue	Resolution
BULK COLLECT	Issue 5088: PostgreSQL doesn't support the EXECUTE IMMEDIATE statement with BULK COLLECT (p. 322)	Perform a manual conversion.
EXECUTE IMMEDIATE	Issue 5334: PostgreSQL doesn't support the dynamic SQL statement (p. 322)	Review and modify the execution string.
RETURNING BULK COLLECT INTO	Issue 5087: PostgreSQL doesn't support the RETURNING BULK COLLECT INTO clause (p. 322)	Perform a manual conversion.

EXCEPTION

Item	Issue	Resolution
ACCESS_INTO_NULL	Issue 5561: PostgreSQL doesn't support the ACCESS_INTO_NULL exception (p. 322)	Review the exception used, and if possible convert it to another condition.

Item	Issue	Resolution
COLLECTION_IS_NULL	Issue 5562: PostgreSQL doesn't support the COLLECTION_IS_NULL exception (p. 322)	Review the exception used, and if possible convert it to another condition.
EMPTY_BLOCK	Issue 5580: The exception block was emptied after conversion (p. 322)	Check converted code and make manual corrections if it is necessary.
EXCEPTION	Issue 5570: PostgreSQL doesn't support exception name declaration (p. 322)	Review the exception used, and if possible convert it to another condition.
NO_DATA_NEEDED	Issue 5563: PostgreSQL doesn't support the NO_DATA_NEEDED exception (p. 322)	Review the exception used, and if possible convert it to another condition.
PROGRAM_ERROR	Issue 5560: PostgreSQL doesn't support the PROGRAM_ERROR exception (p. 323)	Review the exception used, and if possible convert it to another condition.
ROWTYPE_MISMATCH	Issue 5564: PostgreSQL doesn't support the ROWTYPE_MISMATCH exception (p. 323)	Review the exception used, and if possible convert it to another condition.
SELF_IS_NULL	Issue 5565: PostgreSQL doesn't support the SELF_IS_NULL exception (p. 323)	Review the exception used, and if possible convert it to another condition.
SUBSCRIPT_BEYOND_COUNT	Issue 5566: PostgreSQL doesn't support the SUBSCRIPT_BEYOND_COUNT exception (p. 323)	Review the exception used, and if possible convert it to another condition.
SUBSCRIPT_OUTSIDE_LIMIT	Issue 5567: PostgreSQL doesn't support the SUBSCRIPT_OUTSIDE_LIMIT exception (p. 323)	Review the exception used, and if possible convert it to another condition.
SYS_INVALID_ROWID	Issue 5568: PostgreSQL doesn't support the SYS_INVALID_ROWID exception (p. 323)	Review the exception used, and if possible convert it to another condition.

EXPLICIT CURSORS

Item	Issue	Resolution
RETURN	Issue 5559: PostgreSQL doesn't support the RETURN TYPE for the cursor (p. 323)	Use cursor declaration without this statement.

FUNCTION

Item	Issue	Resolution
FUNCTIONS WITH DML (START TRANSACTION, COMMIT, or ROLLBACK.)	Issue 5350: The function can't use statements that explicitly or implicitly begin or end a transaction, such as START TRANSACTION, COMMIT, or ROLLBACK (p. 323)	Revise your code to implement a transaction control at application side.

INSERT

Item	Issue	Resolution
ERROR LOG	Issue 5070: PostgreSQL doesn't support the INSERT statement with the ERROR LOG option (p. 324)	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.
INSERT with partition_extension_clause	Issue 5024: PostgreSQL doesn't support the INSERT statement with partition_extension_clause (p. 324)	Perform a manual conversion for the partition types that aren't supported.
QUERY	Issue 5071: PostgreSQL doesn't support the INSERT statement for a subquery (p. 324)	Perform this operation on the underlying tables instead.
SUBPARTITION	Issue 5090: PostgreSQL doesn't support the INSERT statement for a SUBPARTITION (p. 324)	Either insert data into the overlying partition, or perform a manual conversion using the INSERT statement.

MATERIALIZED VIEW

Item	Issue	Resolution
CREATE	Issue 5094: PostgreSQL doesn't support a materialized VIEW (p. 324)	Revise your code to replace materialized VIEW with another solution.
USING MATERIALIZED VIEW	Issue 5095: PostgreSQL doesn't support using a materialized VIEW (p. 324)	Revise your code to replace materialized VIEW with another solution.

MERGE

Item	Issue	Resolution
MERGE	Issue 5102: PostgreSQL doesn't support the MERGE statement (p. 325)	To achieve the effect of a MERGE statement, use separate INSERT, DELETE, and UPDATE statements.

Optimizer Hints

Item	Issue	Resolution
HINT	Issue 5103: PostgreSQL doesn't support the %s hint (p. 325)	Use PostgreSQL methods for performance tuning.

PACKAGES

Item	Issue	Resolution
Package global cursor	Issue 5330: PostgreSQL doesn't support global cursors (p. 325)	Use another method for this functionality.
Package global user EXCEPTION	Issue 5331: PostgreSQL doesn't support a global user exception (p. 325)	Use another method for this functionality.
Package user type declaration	Issue 5569: PostgreSQL only supports session variables using SQL standard date and time types (p. 325)	Revise your architecture with a custom solution for the user type.

Parser Error

Item	Issue	Resolution
Parser Error	Issue 5293: Unable to resolve the object (p. 325)	Verify if object %s is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

PRAGMA

Item	Issue	Resolution
PRAGMA	Issue 5571: PostgreSQL doesn't support PRAGMA options (p. 326)	Review the exception used, and if possible convert it to another condition.

SELECT

Item	Issue	Resolution
BULK COLLECT INTO	Issue 5140: PostgreSQL doesn't support BULK COLLECT INTO (p. 326)	You can try to include all of the fields from your table in an INTO clause.
FOR UPDATE SKIP LOCKED	Issue 5139: PostgreSQL doesn't support FOR UPDATE SKIP LOCKED (p. 326)	Try using FOR UPDATE without SKIP LOCKED.
FOR UPDATE WAIT	Issue 5144: PostgreSQL doesn't support FOR UPDATE WAIT clauses (p. 326)	Try using FOR UPDATE without WAIT.
Hierarchical queries	Issue 5586: Automatic conversion for queries with NOCYCLE clause not supported (p. 326)	Perform a manual conversion.
MODEL	Issue 5126: PostgreSQL doesn't support the MODEL statement (p. 326)	Revise your code and try using user procedures and temporary tables.
OUTER JOIN	Issue 5585: Automatic conversion of an outer join into a correlation query is not supported (p. 326)	Perform a manual conversion.
ROLLUP	Issue 5557: PostgreSQL doesn't support GROUPING SETS, CUBE, and ROLLUP functions (p. 326)	It can be emulated using CTE expressions. GROUPING SETS, CUBE, and ROLLUP are available in PostgreSQL database v.9.5 or higher.
Unsupported statement	Issue 5578: Unable to automatically transform the SELECT statement (p. 327)	Try rewriting the statement.

SEQUENCE

Item	Issue	Resolution
STATUS=INVALID	Issue 5574: PostgreSQL doesn't support the sequence's status (p. 327)	Check the original sequence for errors, and perform the conversion again.

Statements

Item	Issue	Resolution
GOTO	Issue 5335: PostgreSQL doesn't support the GOTO operator (p. 327)	Perform a manual conversion.

Synonym

Item	Issue	Resolution
private synonym	Issue 5333: This object uses a private synonym (p. 327)	Check that a referenced object created after synonym conversion exists.
public synonym	Issue 5592: This object uses a public synonym which references a table, a view, or a function (p. 327)	Check that a referenced object created after synonym conversion exists.
public synonym	Issue 5593: This object uses a public synonym which doesn't reference any other objects (p. 328)	Check that a referenced object created after synonym conversion exists.

System Package

Item	Issue	Resolution
APEX_UTIL package	Issue 5503: MySQL does not have functionality similar to %s module (p. 328)	Try using the Amazon Simple Workflow (Amazon SWF).
DBMS_DATAPUMP	Issue 5502: MySQL does not have functionality similar to %s module (p. 328)	Try using the AWS Import/Export Snowball.
DBMS_JOB module	Issue 5501: MySQL does not have functionality similar to %s module (p. 328)	Try using the AWS Lambda Service with Scheduled Events.
UTL_MAIL module	Issue 5500: MySQL does not have functionality similar to %s module (p. 328)	Try using the Amazon Simple Notification Service (Amazon SNS).

TRIGGERS

Item	Issue	Resolution
action-type=CALL or PL/SQL	Issue 5313: PostgreSQL doesn't support the action-type clause (p. 328)	Use another method for this functionality.
base-object-type=DATABASE	Issue 5415: PostgreSQL doesn't support system triggers (p. 328)	Revise your code to replace system triggers with another solution.
base-object-type=SCHEMA	Issue 5311: PostgreSQL doesn't support system triggers (p. 328)	Revise your code to replace system triggers with another solution.
COMPOUND TRIGGER	Issue 5242: PostgreSQL doesn't support COMPOUND TRIGGER (p. 329)	Create a single trigger for each part of the compound trigger.

Item	Issue	Resolution
FOLLOWS PRECEDES	Issue 5241: PostgreSQL doesn't support the FOLLOWS PRECEDES clause (p. 329)	Try using the FOR EACH ROW trigger.
ON NESTED TABLE	Issue 5240: PostgreSQL doesn't support triggers on nested table columns in views (p. 329)	Revise your code to replace nested table with another solution.
REFERENCING NEW AS .. OLD AS ..	Issue 5238: PostgreSQL doesn't support the REFERENCING clauses (p. 329)	Modify references to pseudo-rows to use OLD and NEW instead.
referencing-names=PARENT	Issue 5317: PostgreSQL doesn't support the PARENT referencing clause (p. 329)	Use another method for this functionality.
status=invalid	Issue 5306: Transformation from invalid trigger (p. 329)	Check the original trigger for errors, and perform the conversion again.
UPDATING(column_names)	Issue 5556: PostgreSQL doesn't support conditional predicates (p. 329)	Revise your code and try using simple triggers.

TRUNCATE TABLE

Item	Issue	Resolution
DROP STORAGE	Issue 5298: PostgreSQL doesn't support the DROP STORAGE clause in the TRUNCATE statement (p. 329)	Use TRUNCATE without this option.
PRESERVE MATERIALIZED VIEW LOG	Issue 5300: PostgreSQL doesn't support the PRESERVE clause in the TRUNCATE statement (p. 329)	Use TRUNCATE without this option.
PURGE MATERIALIZED VIEW LOG	Issue 5301: PostgreSQL doesn't support the PURGE clause in the TRUNCATE statement (p. 329)	Use TRUNCATE without this option.
REUSE STORAGE	Issue 5299: PostgreSQL doesn't support the REUSE STORAGE clause in the TRUNCATE statement (p. 330)	Use TRUNCATE without this option.

UDT

Item	Issue	Resolution
STATUS=INVALID	Issue 5577: PostgreSQL doesn't support a UDT with body (p. 330)	Perform a manual conversion.

UDT collection-type arguments

Item	Issue	Resolution
<code>:= collection_type(...)</code>	Issue 5120: PostgreSQL doesn't support constructors of the "collection" type (p. 330)	Use the "array" type.
<code>EXTEND(n[,i])</code>	Issue 5587: PostgreSQL doesn't support EXTEND method with parameters (p. 330)	Perform a manual conversion.
<code>FORALL</code>	Issue 5121: PostgreSQL doesn't support the FORALL statement (p. 330)	Use a FOR LOOP statement.
<code>TYPE .. IS TABLE OF...</code>	Issue 5118: PostgreSQL doesn't support table type variables (p. 330)	Use the "array" type.

Universal

Item	Issue	Resolution
Universal action item	Issue 5339: The %s object is invalid (p. 330)	To correct this, update the original source code to make the specified object valid.
Universal action item	Issue 5591: The %s synonym is system (p. 331)	Perform a manual conversion.

Unknown

Item	Issue	Resolution
	Issue 5025: This syntactic element conversion is not supported yet (p. 331)	Perform a manual conversion.
Unknown object	Issue 5351: Automatic conversion of %s object is not supported (p. 331)	Perform a manual conversion.
Unparsed SQL	Issue 5576: Unparsed SQL (p. 331)	Perform a manual conversion.

UPDATE

Item	Issue	Resolution
ERROR LOG	Issue 5064: PostgreSQL doesn't support the UPDATE statement with the ERROR LOG option (p. 331)	You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

Item	Issue	Resolution
PARTITION	Issue 5558: PostgreSQL doesn't support the UPDATE statement for a PARTITION (p. 331)	Either insert data into the overlying partition, or perform a manual transformation using the UPDATE statement.
QUERY	Issue 5065: PostgreSQL doesn't support the UPDATE statement for a subquery (p. 331)	Perform this operation on the underlying tables instead.

VIEW

Item	Issue	Resolution
CAST (MULTISET(constraint	Issue 5245: PostgreSQL doesn't support views with nested table columns (p. 332) Issue 5583: PostgreSQL doesn't support constraints for view (p. 332)	Revise your code to replace nested table with another solution. Perform a manual conversion.
OID_TEXT IS NOT NULL	Issue 5321: PostgreSQL doesn't support the object view (p. 332)	Perform a manual conversion.
PIVOT clause	Issue 5077: PostgreSQL doesn't support the PIVOT clause for the SELECT statement (p. 332)	You can use a stored procedure to prepare data, and then return the data with a VIEW.
STATUS=INVALID	Issue 5320: PostgreSQL doesn't support the view with invalid status (p. 332)	Perform a manual conversion.
VIEW_TYPE_OWNER IS NOT NULL	Issue 5322: PostgreSQL doesn't support the typed view (p. 332)	Perform a manual conversion.
WITH READ ONLY	Issue 5075: PostgreSQL doesn't support VIEW with the READ ONLY option (p. 332)	Use VIEW without this option instead.

Conversion Issues with Build_in_services

Issue 5590: The function was converted as procedure

Can't return a reply because this functionality is asynchronous.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with BUILT-IN SQL FUNCTIONS

Issue 5584: The function %s depends on the time zone setting

Review the transformed code, and set the time zone manually if necessary.

Issue 5340: PostgreSQL doesn't support the %s function

You have SQL code that uses an Oracle function that PostgreSQL doesn't support. Re-write the code to either use an alternative function or create a user-defined function instead. Some of the functions that are not supported are as follows:

- NANVL
- NLS_LOWER
- NLS_UPPER
- SYS_CONTEXT

Issue 5271: The Oracle GREATEST function and PostgreSQL GREATEST function might give different results

Review your transformed code and modify it to change the argument's type if necessary.

Issue 5272: The Oracle LEAST function and PostgreSQL LEAST function might give different results

Review your transformed code and modify it to change the argument's type if necessary.

Issue 5579: The second parameter of %s function can be processed incorrectly

Review your transformed code and modify it if necessary.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE

Issue 5099: Unable convert object due to %s not created

Review the %s object.

Issue 5332: The object has references on the object in %s schema

Convert the object in %s schema also.

Issue 5582: Encrypted Objects

Decrypt the object before conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE CONSTRAINT

Issue 5326: PostgreSQL doesn't support constraints with the status DISABLED

PostgreSQL doesn't support the DISABLE clause on constraints. Drop the constraint prior to conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE INDEX

Issue 5208: PostgreSQL doesn't support domain indexes

Revise your code to use a simple index instead.

Issue 5555: PostgreSQL doesn't support functional indexes that aren't single-column

Revise your code to use a simple index instead.

Issue 5206: PostgreSQL doesn't support bitmap indexes

Revise your code to use a simple index instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE TABLE

Issue 5212: PostgreSQL doesn't have a data type BFILE

Either store a named file with the data and create a routine that gets that file from the file system, or store the data blob inside your database.

Issue 5201: PostgreSQL doesn't support all partition types

Perform a manual conversion for the partition types that aren't supported.

Issue 5199: PostgreSQL doesn't support CLUSTERED TABLE

Try using a table with triggers instead.

Issue 5200: PostgreSQL doesn't support EXTERNAL TABLES

Try moving the data from the external data source to a regular database table.

Issue 5198: PostgreSQL doesn't support GLOBAL TEMPORARY TABLE

Use of the GLOBAL and LOCAL keywords is deprecated in PostgreSQL. Modify your code to create a temporary table without using these keywords.

Issue 5196: PostgreSQL doesn't support OBJECT TABLE

Revise your code to remove references to tables that are based on an OBJECT type, as shown in the following example:

```
CREATE TYPE department_typ AS OBJECT
  ( d_name      VARCHAR2(100),
    d_address   VARCHAR2(200) );
/
CREATE TABLE departments_obj_t OF department_typ;
```

Issue 5581: PostgreSQL doesn't support index-organized table

Revise your architecture with a custom solution for the table type. A typical approach is using a simple table instead in PostgreSQL.

Issue 5348: PostgreSQL doesn't support NESTED TABLES

Revise your code to avoid NESTED TABLE.

Issue 5550: PostgreSQL doesn't have a data type ROWID

Review your transformed code and modify it if necessary to use a different data type.

Issue 5554: PostgreSQL doesn't support virtual columns

Emulate the virtual column by using a trigger instead. For example, you could modify the following Oracle table:

```
CREATE TABLE employees2 (
  id          NUMBER,
  first_name  VARCHAR2(10),
  last_name   VARCHAR2(10),
  salary      NUMBER(9,2),
  comm1       NUMBER(3),
  comm2       NUMBER(3),
  salary1     AS (ROUND(salary*(1+comm1/100),2)),
  salary2     NUMBER GENERATED ALWAYS AS (ROUND(salary*(1+comm2/100),2)) VIRTUAL,
  CONSTRAINT employees_pk PRIMARY KEY (id)
);
```

To the following PostgreSQL statements:

```
CREATE TABLE IF NOT EXISTS TEST_ORA_PG.employees2 (
  id          DOUBLE PRECISION NOT NULL,
  first_name  character varying(10),
  last_name   character varying(10),
  salary      NUMERIC(9,2),
  comm1       NUMERIC(3,0),
```

```
comm2      NUMERIC(3,0),
salary1    DOUBLE PRECISION,
salary2    DOUBLE PRECISION
);

ALTER TABLE TEST_ORA_PG.EMPLOYEES2
ADD CONSTRAINT EMPLOYEES_PK PRIMARY KEY (ID);

CREATE OR REPLACE FUNCTION TEST_ORA_PG.f_trigger_vc$employees2 ()
RETURNS trigger
AS
$BODY$
BEGIN
    new.salary1 := ROUND(new.salary*(1+new.comm1/100),2);
    new.salary2 := ROUND(new.salary*(1+new.comm2/100),2);
return NEW;
END;
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER trigger_vc$employees2
before insert or update on TEST_ORA_PG.employees2
for each row
EXECUTE PROCEDURE TEST_ORA_PG.f_trigger_vc$employees2();
```

[Issue 5213: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds \(6 digits\) of precision](#)

Review your transformed code and modify it if necessary to avoid a loss of accuracy.

[Issue 5552: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds \(6 digits\) of precision](#)

Review your transformed code and modify it if necessary to avoid a loss of accuracy.

[Issue 5553: PostgreSQL expands fractional seconds support for TIME, DATETIME, and TIMESTAMP values, with up to microseconds \(6 digits\) of precision](#)

Review your transformed code and modify it if necessary to avoid a loss of accuracy.

[Issue 5551: PostgreSQL doesn't have a data type UROWID](#)

Review your transformed code and modify it if necessary to use a different data type.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE TYPE

[Issue 5572: PostgreSQL doesn't support object type methods](#)

Revise your architecture with a custom solution for the user type.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CURSOR

Issue 5084: PostgreSQL doesn't support the cursor attribute %ISOPEN

Revise how you are using this attribute in your code.

Issue 5226: PostgreSQL doesn't support TYPE ... IS REF CURSOR

PostgreSQL doesn't support the TYPE <type_name> IS REF CURSOR statement. Change the function to a stored procedure and try using a table to store results instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Datetime Expressions

Issue 5594: PostgreSQL doesn't support %s datetime expression. The loss of precision and/or accuracy of data is possible

PostgreSQL doesn't support the AT LOCAL or AT TIME ZONE <time_zone> datetime expressions. Review your code and modify it if necessary.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DELETE

Issue 5067: PostgreSQL doesn't support the DELETE statement with the ERROR LOG option

You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

Issue 5098: PostgreSQL doesn't support the DELETE statement for a PARTITION

Either insert data into the overlying partition, or perform a manual conversion using the DELETE statement.

Issue 5068: PostgreSQL doesn't support the DELETE statement for a subquery

Perform this operation on the underlying tables instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DYNAMIC SQL

Issue 5088: PostgreSQL doesn't support the EXECUTE IMMEDIATE statement with BULK COLLECT

Perform a manual conversion.

Issue 5334: PostgreSQL doesn't support the dynamic SQL statement

Review and modify the execution string.

Issue 5087: PostgreSQL doesn't support the RETURNING BULK COLLECT INTO clause

Perform a manual conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with EXCEPTION

Issue 5561: PostgreSQL doesn't support the ACCESS_INTO_NULL exception

Review the exception used, and if possible convert it to another condition.

Issue 5562: PostgreSQL doesn't support the COLLECTION_IS_NULL exception

Review the exception used, and if possible convert it to another condition.

Issue 5580: The exception block was emptied after conversion

Check converted code and make manual corrections if necessary.

Issue 5570: PostgreSQL doesn't support exception name declaration

Review the exception used, and if possible convert it to another condition.

Issue 5563: PostgreSQL doesn't support the NO_DATA_NEEDED exception

Review the exception used, and if possible convert it to another condition.

Issue 5560: PostgreSQL doesn't support the PROGRAM_ERROR exception

Review the exception used, and if possible convert it to another condition.

Issue 5564: PostgreSQL doesn't support the ROWTYPE_MISMATCH exception

Review the exception used, and if possible convert it to another condition.

Issue 5565: PostgreSQL doesn't support the SELF_IS_NULL exception

Review the exception used, and if possible convert it to another condition.

Issue 5566: PostgreSQL doesn't support the SUBSCRIPT_BEYOND_COUNT exception

Review the exception used, and if possible convert it to another condition.

Issue 5567: PostgreSQL doesn't support the SUBSCRIPT_OUTSIDE_LIMIT exception

Review the exception used, and if possible convert it to another condition.

Issue 5568: PostgreSQL doesn't support the SYS_INVALID_ROWID exception

Review the exception used, and if possible convert it to another condition.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with EXPLICIT CURSORS

Issue 5559: PostgreSQL doesn't support the RETURN TYPE for the cursor

PostgreSQL doesn't support specifying a return type for a cursor. Declare the cursor without this statement.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with FUNCTION

Issue 5350: The function can't use statements that explicitly or implicitly begin or end a transaction, such as START TRANSACTION, COMMIT, or ROLLBACK

Revise your code to implement transaction control on the application side. Some cases that use SAVEPOINT <name>... ROLLBACK TO <name> can be converted to BEGIN ... EXCEPTION ... END.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with INSERT

Issue 5070: PostgreSQL doesn't support the INSERT statement with the ERROR LOG option

You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

Issue 5024: PostgreSQL doesn't support the INSERT statement with partition_extension_clause

Perform a manual conversion for the partition types that aren't supported.

Issue 5071: PostgreSQL doesn't support the INSERT statement for a subquery

Perform this operation on the underlying tables instead.

Issue 5090: PostgreSQL doesn't support the INSERT statement for a SUBPARTITION

Either insert data into the overlying partition, or perform a manual conversion using the INSERT statement.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with MATERIALIZED VIEW

Issue 5094: PostgreSQL doesn't support a materialized VIEW

Revise your code to replace the CREATE MATERIALIZED VIEW statement with another solution.

Issue 5095: PostgreSQL doesn't support using a materialized VIEW

Revise your code to replace the materialized view with another solution.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with MERGE

Issue 5102: PostgreSQL doesn't support the MERGE statement

To achieve the effect of a MERGE statement, use separate INSERT, DELETE, and UPDATE statements.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Optimizer Hints

Issue 5103: PostgreSQL doesn't support the %s hint

Use PostgreSQL methods for performance tuning.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with PACKAGES

Issue 5330: PostgreSQL doesn't support global cursors

Use another method for this functionality.

Issue 5331: PostgreSQL doesn't support a global user exception

Use another method for this functionality.

Issue 5569: PostgreSQL only supports session variables using SQL standard date and time types

If your code uses variables of other types, such as associative arrays, cursors, ref cursors, or subtypes, revise your architecture with a custom solution for replacing that type.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Parser Error

Issue 5293: Unable to resolve the object

Verify if object <object_name> is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with PRAGMA

Issue 5571: PostgreSQL doesn't support PRAGMA options

Review the exception used, and if possible convert it to another condition.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT

Issue 5140: PostgreSQL doesn't support BULK COLLECT INTO

Include all of the fields from your table in an INTO clause instead.

Issue 5139: PostgreSQL doesn't support FOR UPDATE SKIP LOCKED

Try using FOR UPDATE without SKIP LOCKED.

Issue 5144: PostgreSQL doesn't support FOR UPDATE WAIT clauses

Try using FOR UPDATE without WAIT.

Issue 5586: Automatic conversion for queries with NOCYCLE clause not supported

Perform a manual conversion.

Issue 5126: PostgreSQL doesn't support the MODEL statement

Revise your code. Try using user procedures and temporary tables instead.

Issue 5585: Automatic conversion of an outer join into a correlation query is not supported

Perform a manual conversion.

Issue 5557: PostgreSQL doesn't support GROUPING SETS, CUBE, and ROLLUP functions

These functions can be emulated using CTE expressions instead. GROUPING SETS, CUBE, and ROLLUP functionality is available in PostgreSQL v.9.5 or higher.

Issue 5578: Unable to automatically transform the SELECT statement

The indicated SELECT statement contains language that can't be converted, for example an unsupported function. Try rewriting the statement.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SEQUENCE

Issue 5574: PostgreSQL doesn't support the sequence's status

Check the original sequence for errors, and perform the conversion again.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Statements

Issue 5335: PostgreSQL doesn't support the GOTO operator

PostgreSQL doesn't support the GOTO operator in stored procedures. Try one of the following alternatives instead:

- Set a flag and test for it.
- Return from the procedure if updates should be made, and add the updates after the return.
- Put the updates in another stored procedure, and call it in your IF statements.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Synonym

Issue 5333: This object uses a private synonym

Check that a referenced object created after synonym conversion exists.

Issue 5592: This object uses a public synonym which references a table, a view, or a function

Check that a referenced object created after synonym conversion exists.

Issue 5593: This object uses a public synonym which doesn't reference any other objects

Check that a referenced object created after synonym conversion exists.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with System Package

Issue 5503: MySQL does not have functionality similar to %s module

Try using Amazon Simple Workflow (Amazon SWF). For more information, see [What Is Amazon Simple Workflow Service?](#) in the *Amazon Simple Workflow Service Developer Guide*.

Issue 5502: MySQL does not have functionality similar to %s module

Try using AWS Snowball. For more information, see [What Is AWS Snowball?](#)

Issue 5501: MySQL does not have functionality similar to %s module

Try using the AWS Lambda service with Scheduled Events. For more information, see [What Is AWS Lambda?](#)

Issue 5500: MySQL does not have functionality similar to %s module

Try using the Amazon Simple Queue Service (Amazon SQS). For more information, see [What is Amazon Simple Queue Service?](#)

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with TRIGGERS

Issue 5313: PostgreSQL doesn't support the action-type clause

Use another method for this functionality.

Issue 5415: PostgreSQL doesn't support system triggers

Revise your code to replace system triggers with another solution.

Issue 5311: PostgreSQL doesn't support system triggers

Revise your code to replace system triggers with another solution.

Issue 5242: PostgreSQL doesn't support COMPOUND TRIGGER

Create a single trigger for each part of the compound trigger.

Issue 5241: PostgreSQL doesn't support the FOLLOWS | PRECEDES clause

Try using the FOR EACH ROW trigger.

Issue 5240: PostgreSQL doesn't support triggers on nested table columns in views

Revise your code to replace the nested table with another solution.

Issue 5238: PostgreSQL doesn't support the REFERENCING clauses

Modify references to pseudo-rows to use OLD and NEW instead.

Issue 5317: PostgreSQL doesn't support the PARENT referencing clause

Use another method for this functionality.

Issue 5306: Transformation from invalid trigger

Check the original trigger for errors, and perform the conversion again.

Issue 5556: PostgreSQL doesn't support conditional predicates

Revise your code and try using simple triggers instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with TRUNCATE TABLE

Issue 5298: PostgreSQL doesn't support the DROP STORAGE clause in the TRUNCATE statement

Use the TRUNCATE TABLE statement without this option.

Issue 5300: PostgreSQL doesn't support the PRESERVE clause in the TRUNCATE statement

Use the TRUNCATE TABLE statement without this option.

Issue 5301: PostgreSQL doesn't support the PURGE clause in the TRUNCATE statement

Use the TRUNCATE TABLE statement without this option.

Issue 5299: PostgreSQL doesn't support the REUSE STORAGE clause in the TRUNCATE statement

Use the TRUNCATE TABLE statement without this option.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with UDT

Issue 5577: PostgreSQL doesn't support a UDT with body

Perform a manual conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with UDT collection-type arguments

Issue 5120: PostgreSQL doesn't support constructors of the "collection" type

Use the "array" type instead.

Issue 5587: PostgreSQL doesn't support EXTEND method with parameters

Perform a manual conversion.

Issue 5121: PostgreSQL doesn't support the FORALL statement

Use a FOR LOOP statement instead.

Issue 5118: PostgreSQL doesn't support table type variables

Use the "array" type instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Universal

Issue 5339: The %s object is invalid

Revise the original source code to make the specified object valid.

Issue 5591: The %s synonym is system

Perform a manual conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Unknown

Issue 5025: This syntactic element conversion is not supported yet

Perform a manual conversion.

Issue 5351: Automatic conversion of %s object is not supported

Perform a manual conversion.

Issue 5576: Unparsed SQL

Perform a manual conversion.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with UPDATE

Issue 5064: PostgreSQL doesn't support the UPDATE statement with the ERROR LOG option

You can add error records by inserting them into the log in the exception block. Iterate through the errors in the exception block, add them to the log, and use the EXIT command when finished.

Issue 5558: PostgreSQL doesn't support the UPDATE statement for a PARTITION

Either insert data into the overlying partition, or perform a manual transformation using the UPDATE statement.

Issue 5065: PostgreSQL doesn't support the UPDATE statement for a subquery

Perform this operation on the underlying tables instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with VIEW

Issue 5245: PostgreSQL doesn't support views with nested table columns

Revise your code to replace the nested table with another solution.

Issue 5583: PostgreSQL doesn't support constraints for view

Perform a manual conversion.

Issue 5321: PostgreSQL doesn't support the object view

Perform a manual conversion.

Issue 5077: PostgreSQL doesn't support the PIVOT clause for the SELECT statement

You can use a stored procedure to prepare data, and then return the data with a view.

Issue 5320: PostgreSQL doesn't support the view with invalid status

Perform a manual conversion.

Issue 5322: PostgreSQL doesn't support the typed view

Perform a manual conversion.

Issue 5075: PostgreSQL doesn't support VIEW with the READ ONLY option

Create the view without this option instead.

Related Topics

- [Oracle to PostgreSQL Conversion Reference \(p. 304\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Related Topics

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

PostgreSQL to MySQL Supported Schema Conversion

The following sections list the schema elements from a PostgreSQL database and whether they are supported for automatic conversion to MySQL using the AWS Schema Conversion Tool.

DDL

Statements

Tables

Clause	Automatically Converted	Details
ALTER TABLE	Yes	
Auto-incremented columns	Yes	
Columns with default values	Partial	Issue 6115: MySQL doesn't support expressions in DEFAULT constraints, so they are converted to a BEFORE INSERT trigger (p. 377)
CREATE TABLE	Yes	
Deferrable constraints	Partial	Issue 6111: MySQL doesn't support deferrable constraints (p. 377)
DROP TABLE	Yes	
Expression as default value for column	Partial	Issue 6115: MySQL doesn't support expressions in DEFAULT constraints, so they are converted to a BEFORE INSERT trigger (p. 377) Issue 6117: Unable to keep the NOT NULL constraint for %s column while converting its DEFAULT constraint based on an expression (p. 378)
Foreign keys	Partial	Issue 6114: MySQL doesn't support SET DEFAULT as a referential action in foreign keys (p. 377)
Not-null columns	Partial	Issue 6117: Unable to keep the NOT NULL constraint for %s column while converting its DEFAULT constraint based on an expression (p. 378)
Primary keys	Yes	
Regular tables	Yes	
Table inheritance	No	Issue 6108: MySQL doesn't support inherited tables (p. 378)
Tables with check constraints	Partial	Issue 6113: MySQL doesn't support check constraints; the source constraint is converted to a conditional error raising in triggers (p. 377)
Tables with exclusion constraints	Partial	Issue 6112: MySQL doesn't support exclusion constraints (p. 377)
Tables without columns	No	Issue 6109: MySQL doesn't support typed tables (p. 379)
Trigger disablement	No	Issue 6087: MySQL doesn't support disabling triggers (p. 379)

Clause	Automatically Converted	Details
Typed tables	No	Issue 6110: MySQL doesn't support tables without columns (p. 378) Issue 6120: MySQL doesn't support user-defined data types (p. 381)
Unique keys	Yes	
Unlogged tables	No	Issue 6107: MySQL doesn't support unlogged tables (p. 378)

DML

Built-In SQL Functions

Aggregate

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

Character

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

Conversion

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

DateTime

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

Numeric

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

Other

Clause	Automatically Converted	Details
All	Partial	Issue 6667: MySQL doesn't support the %s function (p. 364)

Window

Clause	Automatically Converted	Details
All	No	Issue 6667: MySQL doesn't support the %s function (p. 364)

XML

Clause	Automatically Converted	Details
All	No	Issue 6667: MySQL doesn't support the %s function (p. 364)

Joins

TRUNCATE TABLE

Clause	Automatically Converted	Details
[NATURAL] [INNER] JOIN	Yes	
[NATURAL] {LEFT RIGHT FULL } [OUTER] JOIN	Yes	
CROSS JOIN	Yes	
USING (join_column [, ...])	Yes	

Operators and Date Functions

Bit String Operators

Clause	Automatically Converted	Details
,&, ,#,~,<<,>>,	Yes	

Date/Time Operators

Clause	Automatically Converted	Details
+, -, *, /	Partial	<p>Issue 6660: Unable to convert the %s operation with the interval value as one or more arguments (p. 372)</p> <p>Issue 6661: The operation will return the number of days instead of the interval value (p. 372)</p>

Mathematical Operators

Clause	Automatically Converted	Details
+, -, *, /, %, ^, , , !, !!, @, &, , #, ~, <<, >>,	Yes	

String Operators

Clause	Automatically Converted	Details
	Yes	

Predicates

SIMILAR Predicate

Clause	Automatically Converted	Details
SIMILAR	No	Issue 6607: Could not convert pattern to MySQL version; check pattern manually (p. 374)

Boolean Conditions

Clause	Automatically Converted	Details
All	Yes	

Comparison Condition

Clause	Automatically Converted	Details
'=, <>, <, <=, >, >='	Yes	
ANY, SOME, ALL	Yes	

Exists Conditions

Clause	Automatically Converted	Details
EXISTS	Yes	

In Conditions

Clause	Automatically Converted	Details
IN	Yes	

Null Conditions

Clause	Automatically Converted	Details
NULL	Yes	

Pattern-Matching Conditions

Clause	Automatically Converted	Details
All	Yes	

Range Conditions

Clause	Automatically Converted	Details
All	Yes	

Set Operators

Text Search Functions and Operators

Clause	Automatically Converted	Details
EXCEPT	Yes	
INTERSECT	Yes	
UNION	Yes	

Statements

INSERT

Clause	Automatically Converted	Details
INSERT INTO table DEFAULT VALUES;	Yes	
INSERT INTO table query;	Yes	
INSERT INTO table VALUES() RETURNING *;	No	Issue 6172: MySQL doesn't support the INSERT statement with the RETURNING option (p. 370)
INSERT INTO table VALUES() RETURNING {output_expression [AS output_name] [, ...]};	No	Issue 6172: MySQL doesn't support the INSERT statement with the RETURNING option (p. 370)
INSERT INTO table VALUES();	Yes	
INSERT INTO table VALUES({ expression DEFAULT } [, ...]),VALUES({ expression DEFAULT } [, ...]) [, ...];	Yes	
INSERT INTO table VALUES({ expression DEFAULT } [, ...]);	Yes	

Clause	Automatically Converted	Details
INSERT INTO table(array) VALUES({' ' }[...]);	Yes	
INSERT INTO table[(column [, ...])] VALUES();	Yes	
WITH RECURSIVE with_query [, ...] UPDATE table_name SET { column_name = { expression DEFAULT } [, ...]	No	Issue 6127: MySQL doesn't support the WITH RECURSIVE clause (p. 375)
WITH with_query [, ...] UPDATE table_name SET { column_name = { expression DEFAULT } [, ...]	Yes	

SELECT

Clause	Automatically Converted	Details
Add or subtract operators	Yes	
EXCEPT [ALL]	Partial	Issue 6612: Query with INTERSECT/EXCEPT ALL operator was transformed (p. 370)
FROM [ONLY] tablename [*]	No	
FROM { with_query_name function_name() }	No	Issue 6603: Can't use function in from clause (p. 368)
FROM {(subquery) [AS] alias (VALUES({ expression DEFAULT } [, ...])) [AS] alias }	Yes	
FROM tablename [[AS] alias]	Yes	
GROUP BY	Yes	
HAVING	Yes	
INTERSECT [ALL]	Partial	Issue 6612: Query with INTERSECT/EXCEPT ALL operator was transformed (p. 370)
LIMIT { count ALL } OFFSET start { ROW ROWS }	Partial	Issue 6611: LIMIT/OFFSET option was omitted (p. 371)

Clause	Automatically Converted	Details
OFFSET start { ROW ROWS } FETCH { FIRST NEXT } [count] { ROW ROWS } ONLY	Partial	Issue 6611: LIMIT/OFFSET option was omitted (p. 371)
ORDER BY [ASC DESC USING operator] [NULLS { FIRST LAST }] [, ...]	Yes	
SELECT { * expression [[AS] output_name] [, ...] } FROM table	Yes	
SELECT { expression [[AS] output_name] [, ...] }	Yes	
SELECT ALL	Yes	
SELECT DISTINCT	Yes	
SELECT DISTINCT ON	No	
UNION [ALL]	Yes	
WHERE (field_name1, ...,fieldnameN) IN (subquery_with_multiple_fields)	Yes	
WHERE comparison_condition (=;<>!=;>=>=<=>ANY;SOME;ALL)	Yes	
WHERE compound_condition_with_or_and_not	Yes	
WHERE field_name BETWEEN arg1 AND arg2	Yes	
WHERE field_name IN (const1,...,constN)	Yes	
WHERE field_name IN (subquery)	Yes	
WHERE like_condition (operator LIKE)	Yes	
WITH RECURSIVE with_query [, ...]	No	Issue 6127: MySQL doesn't support the WITH RECURSIVE clause (p. 375)
WITH with_query [, ...]	Yes	

WINDOW DEFINITION

DELETE

Clause	Automatically Converted	Details
DELETE FROM table_name WHERE CURRENT OF cursor_name	No	Issue 6672: MySQL does not support delete by cursor (p. 367)
DELETE FROM table_name FROM from_list WHERE condition	Yes	
DELETE FROM table_name RETURNING {output_expression [AS output_name] [, ...]}	No	Issue 6069: MySQL doesn't support the DELETE statement with the RETURNING option (p. 367)
DELETE FROM table_name RETURNING *	No	Issue 6069: MySQL doesn't support the DELETE statement with the RETURNING option (p. 367)
DELETE FROM table_name WHERE condition	Yes	
DELETE FROM ONLY table [[AS] alias]	No	
DELETE FROM table [[AS] alias]	Yes	
DELETE FROM table * [[AS] alias]	No	Issue 6603: Can't use function in from clause (p. 368)

SELECT

Clause	Automatically Converted	Details
[existing_window_name] [PARTITION BY expression [, ...]] [ORDER BY expression [ASC DESC USING operator] [NULLS { FIRST LAST }] [, ...]] [frame_clause]	No	Issue 6678: MySQL doesn't support function %s with OVER/ORDER BY/WITHIN GROUP/FILTER clause (p. 363)

UPDATE

Clause	Automatically Converted	Details
UPDATE ONLY table_name SET { column_name = { expression DEFAULT }} [, ...]	No	
UPDATE table_name * SET { column_name = { expression DEFAULT }} [, ...]	No	
UPDATE table_name [[AS] alias] SET { column_name = { expression DEFAULT }} [, ...]	Yes	
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...]	Yes	
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...] WHERE CURRENT OF cursor_name	No	Issue 6672: MySQL does not support delete by cursor (p. 367)
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...] FROM from_list WHERE condition	Partial	Issue 6669: FROM clause was rewritten (p. 382)
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...] RETURNING {output_expression [AS output_name] [, ...]}	No	Issue 6066: MySQL doesn't support the UPDATE statement with the RETURNING option (p. 382)
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...] RETURNING *	No	Issue 6066: MySQL doesn't support the UPDATE statement with the RETURNING option (p. 382)
UPDATE table_name SET { column_name = { expression DEFAULT }} [, ...] WHERE condition	Yes	
UPDATE table_name SET {(column_name [, ...]) = ({ expression DEFAULT } [, ...]) } [, ...]	Yes	

Clause	Automatically Converted	Details
UPDATE table_name SET {(column_name [, ...]) = (query) }	Yes	
WITH RECURSIVE with_query [, ...] UPDATE table_name SET { column_name = { expression DEFAULT } [, ...]	No	Issue 6127: MySQL doesn't support the WITH RECURSIVE clause (p. 375)
WITH with_query [, ...] UPDATE table_name SET { column_name = { expression DEFAULT } [, ...]	Yes	

PL/pgSQL

Basic Statements

Assignment

Clause	Automatically Converted	Details
Variable assignment	Partial	Issue 6668: MySQL doesn't support ROW/RECORD type (p. 374)

Executing a Command with No Result

Clause	Automatically Converted	Details
PERFORM	Yes	

Executing a Query with a Single-Row Result

Clause	Automatically Converted	Details
INTO	No	Issue 6668: MySQL doesn't support ROW/RECORD type (p. 374)

Executing Dynamic Commands

Clause	Automatically Converted	Details
EXECUTE, INTO, STRICT, USING	No	Issue 6334: MySQL doesn't support the dynamic SQL statement RETURN QUERY EXECUTE (p. 367)

Category

Simple Structure of PL/pgSQL

Clause	Automatically Converted	Details
DECLARE, BEGIN, END	Yes	

Conditionals

IF-THEN

Clause	Automatically Converted	Details
IF, THEN, END IF	Yes	

IF-THEN-ELSE

Clause	Automatically Converted	Details
IF, THEN, ELSE, END IF	Yes	

IF-THEN-ELSIF

Clause	Automatically Converted	Details
IF, THEN, ELSE, ELSIF, END IF	Yes	

Searched CASE

Clause	Automatically Converted	Details
CASE, WHEN, THEN, ELSE, END CASE	Yes	

Simple CASE

Clause	Automatically Converted	Details
CASE, WHEN, THEN, ELSE, END CASE	Yes	

Cursors

CLOSE

Clause	Automatically Converted	Details
CLOSE	Yes	

Declaring Cursor Variables

Clause	Automatically Converted	Details
NO, SCROLL, CURSOR, FOR	No	Issue 6337: MySQL doesn't support a variable of REFCURSOR type (p. 365)

FETCH

Clause	Automatically Converted	Details
FETCH [direction { FROM IN }] cursor INTO target;	Partial	Issue 6639: MySQL doesn't support the direction clause (p. 365)

MOVE

Clause	Automatically Converted	Details
MOVE [direction { FROM IN }] cursor;	No	Issue 6640: MySQL doesn't support the MOVE option (p. 365)

OPEN FOR EXECUTE

Clause	Automatically Converted	Details
OPEN, NO, SCROLL, FOR, EXECUTE, USING	No	Issue 6337: MySQL doesn't support a variable of REFCURSOR type (p. 365)

OPEN FOR Query

Clause	Automatically Converted	Details
OPEN, NO, SCROLL, FOR	No	Issue 6638: MySQL doesn't support the SCROLL option in cursors (p. 365)

Opening a Bound Cursor

Clause	Automatically Converted	Details
OPEN bound_cursorvar [[[argument_name :=] argument_value [, ...]]];	Yes	

Declarations

ALIAS

Clause	Automatically Converted	Details
ALIAS FOR	Yes	

Collation of PL/pgSQL Variables

Clause	Automatically Converted	Details
COLLATE	No	Issue 6343: MySQL doesn't support the COLLATE option; automatic conversion cannot be performed (p. 375)

Copying Types

Clause	Automatically Converted	Details
%TYPE, %ROWTYPE	Yes	

Declaring Function Parameters

Clause	Automatically Converted	Details
Declaring function parameters	Yes	

Record Types

Clause	Automatically Converted	Details
Record types	Yes	Issue 6613: Unable to convert a function without OUT parameters returning a RECORD type (p. 369)

Variable Declaration

Clause	Automatically Converted	Details
Variable declaration	Yes	

Errors and Messages

CLOSE

Clause	Automatically Converted	Details
RAISE ;	No	Issue 6329: MySQL doesn't support the RAISE exception (p. 368)
RAISE [level] condition_name [USING option = expression [, ...]];	No	Issue 6329: MySQL doesn't support the RAISE exception (p. 368)
RAISE [level] 'format' [, expression [, ...]] [USING option = expression [, ...]];	Partial	Issue 6332: MySQL doesn't support the RAISE statement to report messages (p. 372)
RAISE [level] SQLSTATE 'sqlstate' [USING option = expression [, ...]];	No	Issue 6329: MySQL doesn't support the RAISE exception (p. 368)
RAISE [level] USING option = expression [, ...];	Partial	Issue 6332: MySQL doesn't support the RAISE statement to report messages (p. 372)

Obtaining the Result Status

Clause	Automatically Converted	Details
GET [CURRENT] DIAGNOSTICS variable = item [, ...];	No	Issue 6665: MySQL doesn't support the GET [CURRENT] DIAGNOSTICS command (p. 367)

Options

Clause	Automatically Converted	Details
EXCEPTION WHEN condition [OR condition ...] THEN handler_statements [..] END;	Partial	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
MESSAGE, DETAIL, HINT, ERRCODE,	No	Issue 6332: MySQL doesn't support the RAISE statement to report messages (p. 372)

Result Status Parameters

Clause	Automatically Converted	Details
COLUMN_NAME	Yes	
CONSTRAINT_NAME	Yes	
MESSAGE_TEXT	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
PG_DATATYPE_NAME	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
PG_EXCEPTION_CONTEXT	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
PG_EXCEPTION_DETAIL	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
PG_EXCEPTION_HINT	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
RETURNED_SQLSTATE	Yes	
SCHEMA_NAME	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)
TABLE_NAME	No	Issue 6664: MySQL doesn't support the condition information %s (p. 368)

Loops

CONTINUE

Clause	Automatically Converted	Details
CONTINUE, WHEN	Yes	

EXIT

Clause	Automatically Converted	Details
EXIT, WHEN	Yes	

FOR (Integer Variant)

Clause	Automatically Converted	Details
FOR, IN, LOOP, END LOOP, REVERSE, BY	Yes	

LOOP

Clause	Automatically Converted	Details
LOOP, END LOOP	Yes	

Looping Through Arrays

Clause	Automatically Converted	Details
FOREACH, IN, LOOP, END LOOP, SLICE, IN ARRAY	No	Issue 6608: MySQL doesn't support arrays (p. 364)

Looping Through Query Results

Clause	Automatically Converted	Details
FOR, IN, LOOP, END LOOP, EXECUTE, USING	Yes	

WHILE

Clause	Automatically Converted	Details
WHILE, LOOP, END LOOP	Yes	

RETURN Statement

NULL; Command

Clause	Automatically Converted	Details
RETURN	Yes	
RETURN NEXT	No	Issue 6604: MySQL doesn't support RETURN NEXT clause (p. 369)
RETURN QUERY	Yes	
RETURN QUERY EXECUTE	No	Issue 6334: MySQL doesn't support the dynamic SQL statement RETURN QUERY EXECUTE (p. 367)

PostgreSQL to MySQL Conversion Reference

AGGREGATE FUNCTION

Item	Issue	Resolution
bit_and, bit_or	Issue 6680: Check working of function with null values (p. 363)	Perform a manual conversion.
OVER/ORDER BY/ WITHIN GROUP/ FILTER	Issue 6678: MySQL doesn't support function %s with OVER/ORDER BY/WITHIN GROUP/ FILTER clause (p. 363)	Perform a manual conversion.

ALTER

Item	Issue	Resolution
ALTER	Issue 6676: Automatic conversion of the ALTER statement isn't supported (p. 363)	Perform a manual conversion.

Arithmetic Operators

Item	Issue	Resolution
Arithmetic operators and string type	Issue 6681: Unable to perform an automated migration of arithmetic operations with string and other types (p. 364)	Make cast operands to the expected type.

ARRAY

Item	Issue	Resolution
ARRAY	Issue 6608: MySQL doesn't support arrays (p. 364)	Perform a manual conversion.

Buildin function

Item	Issue	Resolution
DateTime	Issue 6677: MySQL does not support time zone (p. 364)	Perform a manual conversion.

BUILT-IN SQL FUNCTIONS

Item	Issue	Resolution
BUILT-IN SQL FUNCTIONS	Issue 6667: MySQL doesn't support the %s function (p. 364)	Create a user-defined function.

CREATE

Item	Issue	Resolution
CREATE	Issue 6094: Unable to convert object due to %s not created (p. 365)	Review the %s object.

CURSOR

Item	Issue	Resolution
CURSOR with clause NEXT, PRIOR, FIRST, LAST, ABSOLUTE,	Issue 6639: MySQL doesn't support the direction clause (p. 365)	Perform a manual conversion.

Item	Issue	Resolution
RELATIVE, FORWARD or BACKWARD		
CURSOR with MOVE option	Issue 6640: MySQL doesn't support the MOVE option (p. 365)	Perform a manual conversion.
CURSOR with options SCROLL or NO SCROLL	Issue 6638: MySQL doesn't support the SCROLL option in cursors (p. 365)	Revise your code to eliminate cursors with the SCROLL option.
REFCURSOR	Issue 6337: MySQL doesn't support a variable of REFCURSOR type (p. 365)	Try using temporary tables.

Databases

Item	Issue	Resolution
ALTER DATABASE	Issue 6101: MySQL doesn't support this clause in an ALTER DATABASE statement (p. 365)	Perform a manual conversion.
CREATE DATABASE, DROP DATABASE	Issue 6100: The database-to-database migration is not provided (p. 366)	Use the schema-to-database migration plan.

DATA TYPES

Item	Issue	Resolution
CAST(... AS BIT BIT VARYING VARBIT DECIMAL NUMERIC REAL FLOAT4 DOUBLE PRECISION FLOAT8 CHARACTER CHAR CHARACTER VARYING TEXT TIME WITH TIME ZONE TIME(p) WITH TIME ZONE TIMESTAMP WITH TIME ZONE TIMESTAMP(p) WITH TIME)	Issue 6673: MySQL doesn't support %s type in the CAST function, and a loss of precision or accuracy of data is possible (p. 366)	Check if the data violates these restrictions. If so, perform a manual migration.
CAST(... AS BOX POINT LINE LSEG PATH POLYGON CIRCLE TXID_SNAPSHOT)	Issue 6674: MySQL doesn't support %s type in the CAST function (p. 366)	Check if the data violates these restrictions. If so, perform a manual migration.

DELETE

Item	Issue	Resolution
RETURNING	Issue 6069: MySQL doesn't support the DELETE statement with the RETURNING option (p. 367)	To perform this operation, divide the DELETE statement with the RETURNING clause into a DELETE statement with following INSERT statements and use the same key conditions in each SELECT.
WHERE CURRENT OF	Issue 6672: MySQL does not support delete by cursor (p. 367)	Perform a manual conversion.

Domains

Item	Issue	Resolution
CREATE DOMAIN, ALTER DOMAIN, DROP DOMAIN	Issue 6050: MySQL doesn't support domains (p. 367)	Perform a manual conversion.

DYNAMIC SQL

Item	Issue	Resolution
RETURN QUERY EXECUTE	Issue 6334: MySQL doesn't support the dynamic SQL statement RETURN QUERY EXECUTE (p. 367)	Review and modify the execution string.

Error Handling

Item	Issue	Resolution
GET [CURRENT] DIAGNOSTICS ... PG_ CONTEXT	Issue 6665: MySQL doesn't support the GET [CURRENT] DIAGNOSTICS command (p. 367)	Perform a manual conversion.
GET STACKED DIAGNOSTICS PG_DATATYPE_NAME TABLE_NAME SCHEMA_NAME PG_EXCEPTION_DETAIL PG_EXCEPTION_HINT PG_EXCEPTION_CONTEXT	Issue 6664: MySQL doesn't support the condition information %s (p. 368)	Perform a manual conversion.

Item	Issue	Resolution
RAISE [exception]	Issue 6329: MySQL doesn't support the RAISE exception (p. 368)	Review the RAISE exception used, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.

EXCEPTIONS

Item	Issue	Resolution
EXCEPTIONS	Issue 6342: MySQL doesn't support the %s condition name (p. 368)	Review the exception used, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.

FROM

Item	Issue	Resolution
FROM	Issue 6603: Can't use function in from clause (p. 368)	Perform a manual conversion.
QUERY	Issue 6682: Can't use the table-value function in the from clause (p. 368)	MySQL does not support the table-value function.

FUNCTION

Item	Issue	Resolution
language	Issue 6675: Automatic conversion function on %s language isn't supported (p. 369)	Perform a manual conversion.
RECORD	Issue 6613: Unable to convert a function without OUT parameters returning a RECORD type (p. 369)	Perform a manual conversion.

Functions

Item	Issue	Resolution
RETURN NEXT	Issue 6604: MySQL doesn't support RETURN NEXT clause (p. 369)	Perform a manual conversion.
VARIADIC	Issue 6605: MySQL doesn't support VARIADIC mode of an argument (p. 369)	Perform a manual conversion.

Indexes

Item	Issue	Resolution
CREATE INDEX: GiST, SPGiST, Gin methods	Issue 6091: MySQL doesn't support %s index method (p. 369)	Perform a manual conversion.
Expression-based indexes	Issue 6090: MySQL doesn't support indexes on expressions (p. 369)	Perform a manual conversion.
Ordered indexes	Issue 6092: MySQL doesn't support explicit sort ordering in the index definition (p. 369)	If default (ascending) sort order in the index might cause performance degradation, perform a manual conversion or rewrite statements supposed to use it.
Partial indexes	Issue 6093: MySQL doesn't support partial indexes (p. 370)	If the full index might cause performance degradation, perform a manual conversion.

INSERT

Item	Issue	Resolution
RETURNING	Issue 6172: MySQL doesn't support the INSERT statement with the RETURNING option (p. 370)	To perform this operation, divide the INSERT statement with the RETURNING clause into an INSERT statement with following SELECT statements and use the same key conditions in each SELECT. You can also use the last value that was generated by AUTO_INCREMENT column in the key condition.

INTERSECT, EXCEPT

Item	Issue	Resolution
INTERSECT, EXCEPT	Issue 6612: Query with INTERSECT/EXCEPT ALL operator was transformed (p. 370)	Perform a manual conversion.

LABEL

Item	Issue	Resolution
LABEL	Issue 6344: MySQL doesn't support labeled variables (p. 371)	Try rewriting variables without using labels.

LIMIT

Item	Issue	Resolution
LIMIT, OFFSET	Issue 6611: LIMIT/OFFSET option was omitted (p. 371)	Perform a manual conversion.

Materialized Views

Item	Issue	Resolution
CREATE MATERIALIZED VIEW	Issue 6130: MySQL doesn't support materialized views; the source materialized view is converted to a table with triggers (p. 372)	If this migration approach is unsuitable for any reason, perform a manual conversion.

Message from Stored Routines

Item	Issue	Resolution
RAISE DEBUG LOG INFO NOTICE WARNING	Issue 6332: MySQL doesn't support the RAISE statement to report messages (p. 372)	Try using INSERT in the log table. To do this, you must add code into AWS_POSTGRESQL_EXT.RAISE_DEBUG, AWS_POSTGRESQL_EXT.RAISE_LOG, AWS_POSTGRESQL_EXT.RAISE_INFO, AWS_POSTGRESQL_EXT.RAISE_NOTICE, and AWS_POSTGRESQL_EXT.RAISE_WARNING.

Operators and Date Functions

Item	Issue	Resolution
Date/Time Operators	Issue 6660: Unable to convert the %s operation with the interval value as one or more arguments (p. 372)	Perform a manual conversion.
Date/Time Operators	Issue 6661: The operation will return the number of days instead of the interval value (p. 372)	Check if the resulting type meets your requirements. If it doesn't, perform a manual conversion.
Date/Time Operators	Issue 6662: Unsupported format for the interval's value (p. 373)	Currently, the only supported format is a SQL ANSI format for interval literals. Perform a manual conversion.

ORDER BY

Item	Issue	Resolution
NULLS FIRST LAST	Issue 6610: All nulls first/last transformation was performed (p. 373)	Perform a manual conversion.
USING	Issue 6609: ORDER option was omitted because MySQL doesn't support key word USING in order by clause (p. 373)	Perform a manual conversion.

Parser Error

Item	Issue	Resolution
Parser Error	Issue 6663: Unable to resolve the object (p. 373)	Verify if object %s is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

Patterns

Item	Issue	Resolution
SIMILAR TO	Issue 6607: Could not convert pattern to MySQL version; check pattern manually (p. 374)	Perform a manual conversion.

ROW/RECORD Type

Item	Issue	Resolution
ROW()	Issue 6606: MySQL doesn't support ROW() function with any, some, or all predicates (p. 374)	Perform a manual conversion.
ROW/RECORD	Issue 6668: MySQL doesn't support ROW/RECORD type (p. 374)	Perform a manual conversion.

Select

Item	Issue	Resolution
DISTINCT ON		Perform a manual conversion.
ONLY *		Perform a manual conversion.

Item	Issue	Resolution
DateTime	Issue 6679: Value don't fixed on start transaction (p. 374)	Review your code, and if needed use a local variable

SELECT INTO

Item	Issue	Resolution
SELECT INTO	Issue 6666: Transformation of not strict into clause was performed (p. 375)	Perform a manual conversion.

SELECT, UPDATE, INSERT, DELETE

Item	Issue	Resolution
WITH	Issue 6127: MySQL doesn't support the WITH RECURSIVE clause (p. 375)	Use a stored procedure to prepare data, or rewrite your query to avoid the WITH clause.

Sequences

Item	Issue	Resolution
CREATE SEQUENCE, ALTER SEQUENCE, DROP SEQUENCE	Issue 6060: MySQL doesn't support sequences (p. 375)	Perform a manual conversion.

SQL Functions

Item	Issue	Resolution
COLLATION	Issue 6343: MySQL doesn't support the COLLATE option; automatic conversion cannot be performed (p. 375)	You must use the COLLATION settings that were assigned when the database was created.

Table Columns

Item	Issue	Resolution
Application data types: CIDR, INET, MACADDR, UUID,	Issue 6006: MySQL doesn't support %s type (p. 376)	Check the target data type and correct it if inappropriate.

Item	Issue	Resolution
XML, JSON, JSONB, TSVECTOR, TSQUERY		
ARRAY	Issue 6008: MySQL doesn't support array types for table column definition (p. 376)	Perform a manual conversion.
CIRCLE	Issue 6011: MySQL doesn't support CIRCLE as a spatial type; the approximated results of BUFFER function of POLYGON data type is used to perform a conversion (p. 376)	If the approximated representation is not suitable for any reason, perform a manual conversion.
DECIMAL types	Issue 6002: MySQL doesn't support %s with precision more than 65 digits and scale more than 30 digits; the loss of precision or accuracy of data is possible (p. 376)	Check, if the data violates these restrictions. If so, perform a manual migration.
INTERVAL	Issue 6005: MySQL doesn't support INTERVAL type (p. 376)	Perform a manual conversion.
LINE	Issue 6010: MySQL doesn't support infinite lines; a LINESTRING with two points on the line is used for conversion (p. 376)	If the two-pointed representation is not suitable for any reason, perform a manual conversion.
Range types	Issue 6009: MySQL doesn't support range types (p. 376)	Perform a manual conversion.
Serial types: BIGSERIAL / SERIAL8 / SERIAL / SERIAL4 / SMALLSERIAL / SERIAL2	Issue 6001: Unable to provide full migration for auto-increment table columns (p. 376)	MySQL supports only one auto-increment column per table and it must be a key. Decide whether the column should be an auto-incremented key or not, and if it should, alter it manually.
TIME WITH TIME ZONE	Issue 6003: MySQL doesn't support time zone information for %s type (p. 376)	Perform a manual conversion.
Timestamps with time zone	Issue 6004: MySQL doesn't support time zone information for %s type (p. 376)	Perform a manual conversion. If the time zone information has no value, you can choose DATETIME(6) as a target data type. Otherwise, try to convert the data into TIMESTAMP(6), taking into account the value of time_zone server setting.

Table Columns, Function Arguments, Local Variables

Item	Issue	Resolution
Bit strings	Issue 6012: Converted bit strings are right-padded by zero bits to full-byte length (p. 377)	If the approximated representation is not suitable

Item	Issue	Resolution
		for any reason, perform a manual conversion.

Tables

Item	Issue	Resolution
Constraints	Issue 6111: MySQL doesn't support deferrable constraints (p. 377)	Check your schema and ensure that non-deferrable constraints can provide the necessary level of integrity. Otherwise, perform a manual conversion on the constraints.
Constraints	Issue 6112: MySQL doesn't support exclusion constraints (p. 377)	Perform a manual conversion of the exclusion constraint to provide the same level of data integrity.
Constraints	Issue 6113: MySQL doesn't support check constraints; the source constraint is converted to a conditional error raising in triggers (p. 377)	If the approach with triggers is unsuitable, perform a manual conversion of the check constraint to provide the same level of data integrity.
Constraints	Issue 6114: MySQL doesn't support SET DEFAULT as a referential action in foreign keys (p. 377)	Check that SET NULL is an appropriate referential action, otherwise perform a manual conversion of the constraint.
Constraints	Issue 6115: MySQL doesn't support expressions in DEFAULT constraints, so they are converted to a BEFORE INSERT trigger (p. 377)	Right now, explicitly inserted NULL values are not distinguished from those specially omitted to use default values. Also, updating values to DEFAULT is not supported at the present time. If you need that functionality, perform a manual conversion.
Constraints	Issue 6117: Unable to keep the NOT NULL constraint for %s column while converting its DEFAULT constraint based on an expression (p. 378)	Due to the transformation of the DEFAULT constraint based on an expression into the part of a trigger, the NOT NULL constraint is converted as another part of the same trigger.
Tables without columns	Issue 6110: MySQL doesn't support tables without columns (p. 378)	Perform a manual conversion.

Item	Issue	Resolution
Typed tables	Issue 6107: MySQL doesn't support unlogged tables (p. 378)	If a regular table is unsuitable, perform a manual conversion.
Typed tables	Issue 6108: MySQL doesn't support inherited tables (p. 378)	Perform a manual conversion.
Typed tables	Issue 6109: MySQL doesn't support typed tables (p. 379)	Perform a manual conversion.

Triggers

Item	Issue	Resolution
Conditional triggers	Issue 6083: MySQL doesn't support conditional triggers that fire on updates of particular column(s); a conversion will omit such a condition (p. 379)	If a resulting trigger must fire for updates of only the particular column(s), perform a manual conversion.
Constraint triggers	Issue 6082: MySQL doesn't support constraint triggers (p. 379)	Perform a manual conversion.
Disabled triggers	Issue 6087: MySQL doesn't support disabling triggers (p. 379)	Perform a manual conversion.
DROP TRIGGER	Issue 6086: Unable to automatically transform dropping a trigger (p. 379)	Perform a manual conversion.
Instead-of triggers	Issue 6084: MySQL doesn't support INSTEAD OF triggers (p. 379)	Perform a manual conversion.
Multiple triggers on the same event	Issue 6085: MySQL doesn't support multiple triggers on the same event; triggers are merged into one in alphabetical order by trigger name (p. 379)	If a result of merging is inappropriate, perform a manual conversion.
One trigger on multiple events	Issue 6081: MySQL doesn't support triggers firing on multiple events; such a trigger is converted to multiple single-event triggers with the same body (p. 380)	If this migration approach is not suitable for the trigger, perform a manual conversion.
Statement-level triggers	Issue 6080: MySQL doesn't support statement-level triggers (p. 381)	Perform a manual conversion.
Trigger functions	Issue 6650: Unable to convert the statement with TG_ARGV automatically (p. 381)	Perform a manual conversion.
Trigger functions	Issue 6651: Unsupported using of RETURN statement in the trigger converted to LEAVE statement (p. 381)	Check flow of execution and perform a manual conversion if necessary.
Trigger functions	Issue 6652: MySQL doesn't support OLD record in triggers on INSERT or NEW record in triggers on DELETE (p. 381)	Check, if NULL instead of a field of the unsupported record is appropriate.

Item	Issue	Resolution
Trigger on TRUNCATE event	Issue 6088: MySQL doesn't support TRUNCATE event for triggers (p. 381)	Perform a manual conversion.

UDT

Item	Issue	Resolution
CREATE TYPE, ALTER TYPE, DROP TYPE	Issue 6120: MySQL doesn't support user-defined data types (p. 381)	Perform a manual conversion.

Unknown

Item	Issue	Resolution
Unknown	Issue 6013: This syntactic element conversion is not supported yet (p. 382)	Perform a manual conversion.

UPDATE

Item	Issue	Resolution
FROM	Issue 6669: FROM clause was rewritten (p. 382)	Perform a manual conversion.
RETURNING	Issue 6066: MySQL doesn't support the UPDATE statement with the RETURNING option (p. 382)	To perform this operation, divide the UPDATE statement with the RETURNING clause into an UPDATE statement with following INSERT statements that have the specified key conditions in the SELECT part.
WHERE CURRENT OF	Issue 6670: MySQL does not support update by cursor (p. 382)	Perform a manual conversion.

User-Defined Aggregate Functions

Item	Issue	Resolution
CREATE AGGREGATE, ALTER AGREGGATE, DROP AGREGGATE	Issue 6140: Unable to convert user-defined aggregate statements (p. 383)	Perform a manual conversion.

VIEW

Item	Issue	Resolution
QUERY	Issue 6076: The SELECT statement cannot contain a subquery in the FROM clause (p. 383)	Rewrite the view query without a subquery, or create an auxiliary table for the subquery.

Views

Item	Issue	Resolution
Default values for the view's columns	Issue 6072: MySQL doesn't support default values for the view's column (p. 383)	If the same functionality is necessary, perform a manual conversion.
Row-level secured views	Issue 6071: MySQL doesn't support row-level security (p. 383)	If the same functionality is necessary, perform a manual conversion.
Temporary views	Issue 6070: MySQL doesn't support temporary views or views on temporary tables (p. 383)	Perform a manual conversion.

Conversion Issues with AGGREGATE FUNCTION

Issue 6680: Check working of function with null values

Perform a manual conversion.

Issue 6678: MySQL doesn't support function %s with OVER/ORDER BY/WITHIN GROUP/FILTER clause

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with ALTER

Issue 6676: Automatic conversion of the ALTER statement isn't supported

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Arithmetic Operators

Issue 6681: Unable to perform an automated migration of arithmetic operations with string and other types

Arithmetic operations that use a string type as one of the operands can't be automatically converted. Modify the code to cast values as the intended type.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with ARRAY

Issue 6608: MySQL doesn't support arrays

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Buildin Function

Issue 6677: MySQL does not support time zone

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Built-In SQL Functions

Issue 6667: MySQL doesn't support the %s function

Create a user-defined function instead.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CREATE

Issue 6094: Unable to convert object due to %s not created

Review the %s object.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with CURSOR

Issue 6639: MySQL doesn't support the direction clause

MySQL doesn't support the following direction clauses when using FETCH with cursors:

- ABSOLUTE 2
- BACKWARD
- FIRST FROM
- FORWARD
- LAST FROM
- PRIOR FROM
- RELATIVE 2
- RELATIVE 3

Perform a manual conversion in these cases.

Issue 6640: MySQL doesn't support the MOVE option

Revise your code to eliminate cursors with the MOVE option.

Issue 6638: MySQL doesn't support the SCROLL option in cursors

Revise your code to eliminate cursors with the SCROLL option.

Issue 6337: MySQL doesn't support a variable of REFCURSOR type

Try revising the code to use temporary tables instead.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Databases

Issue 6101: MySQL doesn't support this clause in an ALTER DATABASE statement

Perform a manual conversion.

Issue 6100: The database-to-database migration is not provided

Use the schema-to-database migration plan instead.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DATA TYPES

Issue 6673: MySQL doesn't support %s type in the CAST function, and a loss of precision or accuracy of data is possible

Statements that cast values to any of the following data types are converted to statements that cast values to CHAR:

- BYTEA
- CIDR
- JSON
- int4range

For example, take the following PostgreSQL statements.

```
select '1'::BYTEA
select CAST('1' AS BYTEA)
```

These are both converted to the following MySQL statement.

```
select CAST('1' AS CHAR)
```

Review the converted code and revise it if necessary.

Issue 6674: MySQL doesn't support %s type in the CAST function

Statements that cast a value to the TXID_SNAPSHOT data type are converted to statements that cast a value to CHAR. For example, take the following PostgreSQL statements.

```
select '2628:2628:'::TXID_SNAPSHOT
select CAST('2628:2628:' AS TXID_SNAPSHOT)
```

These are both converted to the following MySQL statement:

```
select CAST('2628:2628:' AS CHAR)
```

Review the converted code and revise it if necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with DELETE

Issue 6069: MySQL doesn't support the DELETE statement with the RETURNING option

To convert this operation, change the DELETE statement with the RETURNING clause into a DELETE statement with following INSERT statements, and use the same key conditions in each INSERT.

Issue 6672: MySQL does not support delete by cursor

MySQL doesn't support using a DELETE statement within a cursor. Perform a manual conversion in this case.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Domains

Issue 6050: MySQL doesn't support domains

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Dynamic SQL

Issue 6334: MySQL doesn't support the dynamic SQL statement RETURN QUERY EXECUTE

Review and modify the execution string.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Error Handling

Issue 6665: MySQL doesn't support the GET [CURRENT] DIAGNOSTICS command

Perform a manual conversion.

Issue 6664: MySQL doesn't support the condition information %s

Some of the condition information items that are used with the GET STACKED DIAGNOSTICS statement in PostgreSQL are not supported by MySQL. These include the following:

- PG_DATATYPE_NAME
- TABLE_NAME
- SCHEMA_NAME
- PG_EXCEPTION_DETAIL
- PG_EXCEPTION_HINT
- PG_EXCEPTION_CONTEXT

Manually convert the code wherever you use these condition information items.

Issue 6329: MySQL doesn't support the RAISE exception

Review the RAISE exception used, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Exceptions

Issue 6342: MySQL doesn't support the %s condition name

Review the exception used, and if possible convert it to an exception using the SIGNAL or RESIGNAL statement.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with FROM

Issue 6603: Can't use function in from clause

Code that selects from a function, for example `SELECT * FROM function_name;`, isn't converted. Perform a manual conversion in this case.

Issue 6682: Can't use the table-value function in the from clause

MySQL does not support the table-value function. Perform a manual conversion in this case.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with FUNCTION

Issue 6675: Automatic conversion function on %s language isn't supported

Perform a manual conversion.

Issue 6613: Unable to convert a function without OUT parameters returning a RECORD type

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Functions

Issue 6604: MySQL doesn't support RETURN NEXT clause

Perform a manual conversion.

Issue 6605: MySQL doesn't support VARIADIC mode of an argument

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Indexes

Issue 6091: MySQL doesn't support %s index method

The InnoDB engine in MySQL doesn't support HASH and ON *table_name* indexes. Perform a manual conversion in these cases.

Issue 6090: MySQL doesn't support indexes on expressions

Perform a manual conversion.

Issue 6092: MySQL doesn't support explicit sort ordering in the index definition

Indexes that use explicit sort ordering are converted to use the default (ascending) sort order instead. If this conversion causes performance degradation, revise the converted code as necessary.

Issue 6093: MySQL doesn't support partial indexes

Partial indexes are converted to full indexes instead. For example, take the following PostgreSQL statement.

```
CREATE INDEX i_indexed_partial
ON test_pg_mysql.indexed_partial USING btree
( n )
WHERE ( n < 100 );
```

The preceding statement is converted to the following MySQL statement:

```
CREATE INDEX i_indexed_partial
USING BTREE ON test_pg_mysql.indexed_partial
( n );
```

If this conversion causes performance degradation, revise the converted code as necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with INSERT

Issue 6172: MySQL doesn't support the INSERT statement with the RETURNING option

To convert this operation, change the INSERT statement with the RETURNING clause into an INSERT statement with following SELECT statements, and use the same key conditions in each SELECT. You can also use the last value that was generated by any AUTO_INCREMENT column in the key condition.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with INTERSECT, EXCEPT

Issue 6612: Query with INTERSECT/EXCEPT ALL operator was transformed

The EXCEPT ALL and INTERSECT ALL options of the SELECT statement can't be directly converted, so they are converted to other language options that should produce the same results. For example, take the following PostgreSQL statements.

```
SELECT * FROM customers
except all
SELECT * FROM customers;

SELECT * FROM customers
intersect all
SELECT * FROM customers;
```


The preceding statements are converted to the following MySQL statements.

```
SELECT customers.id, customers.name, customers.surname, customers.postcode FROM customers
where (customers.id, customers.name, customers.surname, customers.postcode)
not in (SELECT customers.id, customers.name, customers.surname, customers.postcode FROM
customers);

SELECT customers.id, customers.name, customers.surname, customers.postcode
FROM customers inner join (SELECT customers.id, customers.name, customers.surname,
customers.postcode FROM customers) nn
on (customers.id=nn.id or (customers.id is null and nn.id is null))
and customers.name=nn.name
and (customers.surname=nn.surname or (customers.surname is null and nn.surname is null))
and (customers.postcode=nn.postcode or (customers.postcode is null and nn.postcode is
null));
```

Review the converted code, and revise it if necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with LABEL

Issue 6344: MySQL doesn't support labeled variables

Try rewriting variables without using labels.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with LIMIT

Issue 6611: LIMIT/OFFSET option was omitted

MySQL doesn't support use of the LIMIT or OFFSET options with the SELECT statement. These options are dropped during conversion. For example, take the following PostgreSQL statement.

```
SELECT * FROM customers c offset 3;
```

This statement is converted to the following MySQL statement:

```
SELECT * FROM customers c;
```

Review the converted code, and revise it as necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Materialized Views

Issue 6130: MySQL doesn't support materialized views; the source materialized view is converted to a table with triggers

Review the converted code to make sure it meets your needs, and revise it if necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Messages from Stored Routines

Issue 6332: MySQL doesn't support the RAISE statement to report messages

Try using INSERT to write a message to a log table instead. To do this, you must add code in one of the following tables in the AWS_POSTGRESQL_EXT schema (the temporary work schema used for PostgreSQL):

- RAISE_DEBUG
- RAISE_LOG
- RAISE_INFO
- RAISE_NOTICE
- RAISE_WARNING

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Operators and Date Functions

Issue 6660: Unable to convert the %s operation with the interval value as one or more arguments

Some operations with arguments that use an interval data type can't be converted, for example `SELECT INTERVAL '1-1' YEAR TO MONTH * 2;`. Perform a manual conversion in this case.

Issue 6661: The operation will return the number of days instead of the interval value

After conversion, some calculations that use date or time data types return the number of days rather than an interval value as they did previously, for example `SELECT TIMESTAMP '2012-01-01`

09:30:15.01' - DATE '2010-01-01';. Check to see if the new return type meets your requirements, and revise the code if necessary.

Issue 6662: Unsupported format for the interval's value

Currently, the only supported format for a literal value assigned to an interval data type is SQL ANSI. Perform a manual conversion in the case of an unsupported format.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with ORDER BY

Issue 6610: All nulls first/last transformation was performed

MySQL doesn't support the NULLS FIRST or NULLS LAST options with the SELECT statement. Instances of this language are converted to use the isnull() function instead. For example, take the following PostgreSQL statements.

```
SELECT * FROM customers c order by id nulls last;  
SELECT * FROM customers c order by id desc nulls first;
```

These statements are converted to the following MySQL statements.

```
SELECT * FROM customers c order by isnull(id), id;  
SELECT * FROM customers c order by isnull(id) desc, id desc;
```

Review the converted code, and revise it if necessary.

Issue 6609: ORDER option was omitted because MySQL doesn't support key word USING in order by clause

MySQL doesn't support the ORDER BY *field_name* USING clause, for example `SELECT * FROM customers c order by id using >;`. During conversion, the ORDER BY clause is dropped. Review this code and revise it if necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Parser Error

Issue 6663: Unable to resolve the object

Verify if object %s is present in the database. If it isn't, check the object name or add the object. If the object is present, transform the code manually.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Patterns

Issue 6607: Could not convert pattern to MySQL version; check pattern manually

MySQL doesn't support some pattern matching syntax, for example `SELECT * FROM customers where name similar to 'name'`; Perform a manual conversion in this case.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with ROW, RECORD type

Issue 6606: MySQL doesn't support ROW() function with any, some, or all predicates

MySQL doesn't support the ROW() function with the any, some, or all predicates, for example `SELECT * FROM customers c where ROW(c.id, c.postcode) <= all (select id, order_sum from orders where id = c.id);`. Perform a manual conversion instead.

Issue 6668: MySQL doesn't support ROW/RECORD type

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT

Issue 6679: Value don't fixed on start transaction

Review your code, and revise it to use a local variable if necessary.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT INTO

Issue 6666: Transformation of not strict into clause was performed

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SELECT, UPDATE, INSERT, DELETE

Issue 6127: MySQL doesn't support the WITH RECURSIVE clause

Use a stored procedure to prepare data, or rewrite your query to avoid the WITH RECURSIVE clause.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Sequences

Issue 6060: MySQL doesn't support sequences

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with SQL Functions

Issue 6343: MySQL doesn't support the COLLATE option; automatic conversion cannot be performed

You must use the COLLATION settings that were assigned when the database was created.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Table Columns

Issue 6006: MySQL doesn't support %s type

Check the target data type and correct it if necessary.

Issue 6008: MySQL doesn't support array types for table column definition

MySQL doesn't support using an array data type for either a table column or a function variable, so this code isn't automatically converted from PostgreSQL. Perform a manual conversion instead.

Issue 6011: MySQL doesn't support CIRCLE as a spatial type; the approximated results of BUFFER function of POLYGON data type is used to perform a conversion

Check the converted code, and revise it if the approximated representation doesn't meet your needs.

Issue 6002: MySQL doesn't support %s with precision more than 65 digits and scale more than 30 digits; the loss of precision or accuracy of data is possible

Check to see if you are converting columns whose numeric data types violate these restrictions. If so, perform a manual migration instead.

Issue 6005: MySQL doesn't support INTERVAL type

Perform a manual conversion of any code that uses this data type.

Issue 6010: MySQL doesn't support infinite lines; a LINESTRING with two points on the line is used for conversion

Any instance of the LINE data type is converted to a LINESTRING data type. If the two-pointed representation is not suitable for your purposes, revise this code.

Issue 6009: MySQL doesn't support range types

MySQL doesn't support range data types like LONGTEXT or VARCHAR(*length*). Manually convert code that uses these data types instead.

Issue 6001: Unable to provide full migration for auto-increment table columns

The PostgreSQL auto-increment data types SMALLSERIAL, SERIAL and BIGSERIAL are converted to the MySQL SMALLINT, INT, and BIGINT data types, respectively. If you want to enable auto-incrementing, MySQL supports only one auto-increment column per table and it must be a key. Choose an appropriate column and manually update it to meet these requirements.

Issue 6003: MySQL doesn't support time zone information for %s type

Perform a manual conversion.

Issue 6004: MySQL doesn't support time zone information for %s type

Perform a manual conversion instead. If the time zone information isn't important, you can choose DATETIME(6) as a target data type. Otherwise, try to convert the data into TIMESTAMP(6), taking into account the value of time_zone server setting.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Table Columns, Function Arguments, and Local Variables

Issue 6012: Converted bit strings are right-padded by zero bits to full-byte length

Review the converted code, and revise it if the approximated representation is not suitable for your needs.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Tables

Issue 6111: MySQL doesn't support deferrable constraints

Check your schema and ensure that nondeferrable constraints can provide the necessary level of integrity. Otherwise, perform a manual conversion on the constraints.

Issue 6112: MySQL doesn't support exclusion constraints

Perform a manual conversion of the exclusion constraint to provide the same level of data integrity.

Issue 6113: MySQL doesn't support check constraints; the source constraint is converted to a conditional error raising in triggers

In MySQL, check constraints are parsed but ignored. Instead, PostgreSQL check constraints are converted to triggers that fire before INSERT and UPDATE events. Review this code to see if it meets your needs, and if necessary perform a manual conversion of the check constraint to provide the expected level of data integrity.

Issue 6114: MySQL doesn't support SET DEFAULT as a referential action in foreign keys

MySQL doesn't support using SET DEFAULT when creating a foreign key. Any SET DEFAULT statements are converted to SET NULL statements. Review this code and modify it if necessary.

Issue 6115: MySQL doesn't support expressions in DEFAULT constraints, so they are converted to a BEFORE INSERT trigger

MySQL doesn't support using expressions as default values when creating a table, so during conversion a BEFORE INSERT trigger is created to insert the default values instead. CHARACTER VARYING columns are converted into TEXT or LONGTEXT columns (depending on the length of the source column), which can't have default values in MySQL. Therefore, any default values in such columns are treated as expressions and are also included in the BEFORE INSERT trigger.

For example, take the following PostgreSQL code.

```
CREATE TABLE IF NOT EXISTS test_pg_mysql.constraint_default_expression
( n numeric( 10, 0 ) DEFAULT 12345+98765
, d date DEFAULT now()
, d2 date DEFAULT '2016-01-06'::date
, s character varying (10) DEFAULT 'A string'
);
```

This code is converted into the following MySQL code.

```
CREATE TRIGGER constraint_default_expression_bir
BEFORE INSERT
ON test_pg_mysql.constraint_default_expression
FOR EACH ROW
BEGIN
    default_values_expr: BEGIN
-- default value for n column
        IF NEW.n IS NULL THEN
            SET NEW.n = 123456 + 98765;
        END IF;
-- default value for d column
        IF NEW.d IS NULL THEN
            SET NEW.d = now( 6 );
        END IF;
-- default value for d2 column
        IF NEW.d2 IS NULL THEN
            SET NEW.d2 = CAST( '2016-01-06' AS DATE );
        END IF;
-- default value for s column
        IF NEW.s IS NULL THEN
            SET NEW.s = 'A string';
        END IF;
    END default_values_expr;
END;
```

Issue 6117: Unable to keep the NOT NULL constraint for %s column while converting its DEFAULT constraint based on an expression

MySQL doesn't support using expressions as default values when creating a table, so during conversion a BEFORE INSERT trigger is created to insert the default values instead. For more details, see [Issue 6115: MySQL doesn't support expressions in DEFAULT constraints, so they are converted to a BEFORE INSERT trigger \(p. 377\)](#).

If a column that has a default value and so is converted in this way also has a NOT NULL constraint, the NOT NULL constraint is converted as part of the same trigger rather than remaining as part of the CREATE TABLE statement.

Issue 6110: MySQL doesn't support tables without columns

MySQL doesn't support using the CREATE TABLE statement without specifying at least one column. This usage is something you might do when creating a table based on a user-defined type, for example. Perform a manual conversion instead.

Issue 6107: MySQL doesn't support unlogged tables

Unlogged tables in PostgreSQL are converted to regular tables in MySQL. Review this code to see if it meets your needs, and revise it if necessary.

Issue 6108: MySQL doesn't support inherited tables

Perform a manual conversion.

Issue 6109: MySQL doesn't support typed tables

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Triggers

Issue 6083: MySQL doesn't support conditional triggers that fire on updates of particular column(s); a conversion will omit such a condition

MySQL doesn't support conditional triggers that rely on the update of a specific column. For example, take the following statement in PostgreSQL.

```
CREATE TRIGGER triggering_cond_bur_of
BEFORE UPDATE OF n
```

This statement is converted to the following statement in MySQL:

```
CREATE TRIGGER triggering_cond_bur
BEFORE UPDATE
```

If you require this functionality, perform a manual conversion instead.

Issue 6082: MySQL doesn't support constraint triggers

MySQL doesn't support constraint triggers, so any CREATE CONSTRAINT TRIGGER statements are converted to CREATE TRIGGER statements. Review this code and modify it if necessary.

Issue 6087: MySQL doesn't support disabling triggers

PostgreSQL allows users to disable table triggers so that they aren't fired, and then re-enable them at a later time. MySQL doesn't support this, so you must perform a manual conversion of this code.

Issue 6086: Unable to automatically transform dropping a trigger

DROP TRIGGER statements aren't automatically converted. Perform a manual conversion instead.

Issue 6084: MySQL doesn't support INSTEAD OF triggers

Perform a manual conversion.

Issue 6085: MySQL doesn't support multiple triggers on the same event; triggers are merged into one in alphabetical order by trigger name

Multiple triggers on one event are automatically merged into a single trigger, with their functionality inserted into the new trigger in alphabetic order by trigger name. Review this trigger code after conversion and revise it if necessary.

For example, take the following PostgreSQL code.

```
CREATE TRIGGER triggering_multiple_inc
  BEFORE INSERT
  ON test_pg_mysql.triggering_multiple
  FOR EACH ROW
  EXECUTE PROCEDURE test_pg_mysql.fu_triggering_simple_inc();

CREATE TRIGGER triggering_multiple_inc_2
  BEFORE INSERT
  ON test_pg_mysql.triggering_multiple
  FOR EACH ROW
  EXECUTE PROCEDURE test_pg_mysql.fu_triggering_simple_inc();
```

This code is converted into the following MySQL code.

```
CREATE TRIGGER triggering_multiple_bir
  BEFORE INSERT
  ON test_pg_mysql.triggering_multiple
  FOR EACH ROW
  BEGIN
    tr1: BEGIN
      -- triggering_multiple_inc
      SET NEW.n = NEW.n + 1;
    END tr1;

    tr2: BEGIN
      -- triggering_multiple_inc_2
      SET NEW.n = NEW.n + 1;
    END tr2;
  END;
```

Issue 6081: MySQL doesn't support triggers firing on multiple events; such a trigger is converted to multiple single-event triggers with the same body

PostgreSQL supports specifying multiple events with the CREATE TRIGGER statement, for example CREATE TRIGGER *trigger_name* BEFORE INSERT OR UPDATE. However, MySQL only supports specifying one event, for example CREATE TRIGGER *trigger_name* BEFORE UPDATE. PostgreSQL triggers that specify multiple events are converted to several triggers with the same trigger body, one for each event specified. Review this trigger code after conversion and revise it if necessary.

For example, take the following PostgreSQL code.

```
CREATE TRIGGER
  triggering_simple_inc BEFORE INSERT
  OR UPDATE ON
  test_pg_mysql.triggering_simple FOR EACH
  ROW EXECUTE PROCEDURE
  test_pg_mysql.fu_triggering_simple_inc();
```

This code is converted into the following MySQL code:

```
CREATE TRIGGER triggering_simple_bir
  BEFORE INSERT
  ON test_pg_mysql.triggering_simple
  FOR EACH ROW
  BEGIN
    tr1: BEGIN
      SET NEW.n = NEW.n + 1;
    END tr1;
  END;
```

```
CREATE TRIGGER triggering_simple_bur
  BEFORE UPDATE
  ON test_pg_mysql.triggering_simple
  FOR EACH ROW
  BEGIN
    tr1: BEGIN
      SET NEW.n = NEW.n + 1;
    END tr1;
  END;
```

Issue 6080: MySQL doesn't support statement-level triggers

MySQL doesn't support the FOR EACH STATEMENT clause for CREATE TRIGGER statements. Perform a manual conversion of this code.

Issue 6650: Unable to convert the statement with TG_ARGV automatically

CREATE TRIGGER statements that use TG_ARGV to work with trigger arguments can't be converted. Perform a manual conversion instead.

Issue 6651: Unsupported using of RETURN statement in the trigger converted to LEAVE statement

Check flow of execution and perform a manual conversion if necessary.

Issue 6652: MySQL doesn't support OLD record in triggers on INSERT or NEW record in triggers on DELETE

MySQL doesn't support the NEW variable in triggers that use DELETE, and it also doesn't support the OLD variable in triggers that use INSERT. In these cases, the variable reference is replaced with NULL. For example, `NEW.n > 0` is replaced by `NULL > 0`. Review this code to see if using NULL instead of a field of the unsupported record is appropriate, and revise as necessary.

Issue 6088: MySQL doesn't support TRUNCATE event for triggers

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with User-Defined Types

Issue 6120: MySQL doesn't support user-defined data types

Perform a manual conversion of any code involving user-defined types. This conversion should include any references to user-defined types, and also any instances of the CREATE TYPE, ALTER TYPE, or DROP TYPE statements.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Unknown

Issue 6013: This syntactic element conversion is not supported yet

Perform a manual conversion.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with UPDATE

Issue 6669: FROM clause was rewritten

The FROM clause in UPDATE statements is rewritten during PostgreSQL to MySQL conversion. For example, take the following PostgreSQL statement.

```
update test_pg_mysql.customers c set id = c.id +1, name = o.name,  
    postcode = o.order_sum from test_pg_mysql.orders o where o.id = c.id;
```

This statement is converted to the following MySQL statement.

```
update customers c  
    set id = c.id +1,  
    name = (select name from orders where id = c.id),  
    postcode = (select order_sum from orders where id = c.id)  
    where c.id in (select id from orders);
```

Review this code to make sure it produces the results that you expect.

Issue 6066: MySQL doesn't support the UPDATE statement with the RETURNING option

To perform this operation, convert the UPDATE statement with the RETURNING clause into an UPDATE statement with following INSERT statements that have the specified key conditions in the SELECT clause.

Issue 6670: MySQL does not support update by cursor

MySQL doesn't support updating a table from within a cursor, as shown in the following example.

```
DECLARE c1 CURSOR FOR SELECT id FROM test_pg_mysql.customers;  
val numeric(14,0);  
BEGIN  
    OPEN c1;  
    fetch c1 into val;  
    update test_pg_mysql.customers set name = 'nfhr' where current of c1;  
    CLOSE c1;  
END
```

Perform a manual conversion instead.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with User-Defined Aggregate Functions

Issue 6140: Unable to convert user-defined aggregate statements

Perform a manual conversion of any user-defined aggregate functions.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with VIEW

Issue 6076: The SELECT statement cannot contain a subquery in the FROM clause

Either rewrite the query that the view is based on so that it doesn't use a subquery, or create an auxiliary table for the subquery.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Conversion Issues with Views

Issue 6072: MySQL doesn't support default values for the view's column

MySQL doesn't support specifying a default value when creating or altering a view, as shown in the following example.

```
ALTER VIEW test_pg_mysql.v_default_values  
ALTER n SET DEFAULT 10;
```

If you need this functionality, perform a manual conversion instead.

Issue 6071: MySQL doesn't support row-level security

If the same functionality is necessary, perform a manual conversion instead.

Issue 6070: MySQL doesn't support temporary views or views on temporary tables

Perform a manual conversion instead.

Related Topics

- [PostgreSQL to MySQL Conversion Reference \(p. 350\)](#)
- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Related Topics

- [AWS Schema Conversion Tool Reference \(p. 141\)](#)
- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)

Related Topics

- [What Is the AWS Schema Conversion Tool? \(p. 1\)](#)
- [Getting Started with the AWS Schema Conversion Tool \(p. 12\)](#)
- [Installing and Updating the AWS Schema Conversion Tool \(p. 7\)](#)

Document History

The following table describes the important changes to the documentation since the last release of the *AWS Schema Conversion Tool (AWS SCT) User Guide*.

Version	Change	Description	Date Changed
1.0.607	FIPS endpoint support for Amazon S3	You can now request AWS SCT to connect to Amazon S3 and Amazon Redshift by using FIPS endpoints to comply with Federal Information Processing Standard security requirements. For more information, see Storing AWS Credentials (p. 135) and Using Data Extraction Agents (p. 115) .	October 30, 2017
1.0.607	Data extraction tasks can ignore LOBs	When you create data extraction tasks, you can now choose to ignore large objects (LOBs) to reduce the amount of data that you extract. For more information, see Managing Data Extraction Tasks with the AWS Schema Conversion Tool (p. 123) .	October 30, 2017
1.0.605	Data extraction agent task log access	You can now access the data extraction agent task log from a convenient link in the AWS Schema Conversion Tool user interface. For more information, see Managing Data Extraction Tasks with the AWS Schema Conversion Tool (p. 123) .	August 28, 2017
1.0.604	Converter enhancements	The AWS Schema Conversion Tool engine has been enhanced to offer improved conversions for heterogeneous migrations.	June 24, 2017
1.0.603	Data extraction agents support filters	You can now filter the data that the extraction agents extract from your data warehouse. For more information, see Creating Data Extraction Filters in the AWS Schema Conversion Tool (p. 122) .	June 16, 2017
1.0.603	AWS SCT supports additional data warehouse versions	You can now use the AWS Schema Conversion Tool to convert your Teradata 13 and Oracle Data Warehouse 10 schemas to equivalent Amazon Redshift schemas. For more information, see Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool (p. 88) .	June 16, 2017
1.0.602	Data extraction agents support additional data warehouses	You can now use data extraction agents to extract data from your Microsoft SQL Server data warehouses. For more information, see Using Data Extraction Agents (p. 115) .	May 11, 2017
1.0.602	Data extraction agents can copy data to Amazon Redshift	Data extraction agents now have three upload modes. You can now specify whether to just extract your data, to extract your data and just upload it to Amazon S3, or to extract, upload,	May 11, 2017

Version	Change	Description	Date Changed
		and copy your data directly into Amazon Redshift. For more information, see Managing Data Extraction Tasks with the AWS Schema Conversion Tool (p. 123) .	
1.0.601	AWS SCT supports additional data warehouses	You can now use the AWS Schema Conversion Tool to convert your Vertica and Microsoft SQL Server schemas to equivalent Amazon Redshift schemas. For more information, see Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool (p. 88) .	April 18, 2017
1.0.601	Data extraction agents support additional data warehouses	You can now use data extraction agents to extract data from your Greenplum, Netezza, and Vertica data warehouses. For more information, see Using Data Extraction Agents (p. 115) .	April 18, 2017
1.0.601	Data extraction agents support additional operating systems	You can now install data extraction agents on computers running the macOS and Microsoft Windows operating systems. For more information, see Installing Extraction Agents (p. 118) .	April 18, 2017
1.0.601	Data extraction agents upload to Amazon S3 automatically	Data extraction agents now upload your extracted data to Amazon S3 automatically. For more information, see Data Extraction Task Output (p. 125) .	April 18, 2017
1.0.600	Data Extraction Agents	You can now install data extraction agents that extract data from your data warehouse and prepare it for use with Amazon Redshift. You can use the AWS Schema Conversion Tool to register the agents and create data extraction tasks for them. For more information, see Using Data Extraction Agents (p. 115) .	February 16, 2017
1.0.600	Customer Feedback	You can now provide feedback about the AWS Schema Conversion Tool. You can file a bug report, you can submit a feature request, or you can provide general information. For more information, see Providing Customer Feedback (p. 2) .	February 16, 2017
1.0.502	Integration with AWS DMS	You can now use the AWS Schema Conversion Tool to create AWS DMS endpoints and tasks. You can run and monitor the tasks from AWS SCT. For more information, see Working with the AWS Database Migration Service Using the AWS Schema Conversion Tool (p. 113) .	December 20, 2016

Version	Change	Description	Date Changed
1.0.502	Amazon Aurora with PostgreSQL compatibility as a target database	The AWS Schema Conversion Tool now supports Amazon Aurora with PostgreSQL compatibility as a target database. For more information, see Converting Database Schema to Amazon RDS by Using the AWS Schema Conversion Tool (p. 69) .	December 20, 2016
1.0.502	Support for profiles	You can now store different profiles in the AWS Schema Conversion Tool and easily switch between them. For more information, see Storing AWS Profiles in the AWS Schema Conversion Tool (p. 135) .	December 20, 2016
1.0.501	Support for Greenplum Database and Netezza	You can now use the AWS Schema Conversion Tool to convert your data warehouse schemas from Greenplum Database and Netezza to Amazon Redshift. For more information, see Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool (p. 88) .	November 17, 2016
1.0.501	Redshift optimization	You can now use the AWS Schema Conversion Tool to optimize your Amazon Redshift databases. For more information, see Optimizing Amazon Redshift by Using the AWS Schema Conversion Tool (p. 111) .	November 17, 2016
1.0.500	Mapping rules	Before you convert your schema with the AWS Schema Conversion Tool, you can now set up rules that change the data type of columns, move objects from one schema to another, and change the names of objects. For more information, see Creating Mapping Rules in the AWS Schema Conversion Tool (p. 70) .	October 4, 2016
1.0.500	Move to cloud	You can now use the AWS Schema Conversion Tool to copy your existing on-premises database schema to an Amazon RDS DB instance running the same engine. You can use this feature to analyze potential cost savings of moving to the cloud and of changing your license type. For more information, see What Is the AWS Schema Conversion Tool? (p. 1) and Creating and Using the Assessment Report in the AWS Schema Conversion Tool (p. 76) .	October 4, 2016
1.0.400	Data warehouse schema conversions	You can now use the AWS Schema Conversion Tool to convert your data warehouse schemas from Oracle and Teradata to Amazon Redshift. For more information, see Converting Data Warehouse Schema to Amazon Redshift by Using the AWS Schema Conversion Tool (p. 88) .	July 13, 2016

Version	Change	Description	Date Changed
1.0.400	Application SQL conversions	You can now use the AWS Schema Conversion Tool to convert SQL in your C++, C#, Java, or other application code. For more information, see Converting Application SQL by Using the AWS Schema Conversion Tool (p. 128) .	July 13, 2016
1.0.400	New feature	The AWS Schema Conversion Tool now contains an extension pack and a wizard to help you install, create, and configure AWS Lambda functions and Python libraries to provide email, job scheduling, and other features. For more information, see The AWS Schema Conversion Tool Extension Pack and AWS Services for Databases (p. 85) and The AWS Schema Conversion Tool Extension Pack and Python Libraries for Data Warehouses (p. 109) .	July 13, 2016
1.0.301	SSL Support	You can now use Secure Sockets Layer (SSL) to connect to your source database when you use the AWS Schema Conversion Tool. For more information, see Connecting to Your Source Database (p. 14) .	May 19, 2016
1.0.203	New feature	Adds support for MySQL and PostgreSQL as source databases for conversions.	April 11, 2016
1.0.202	Maintenance release	Adds support for editing the converted SQL that was generated for the target database engine. Adds improved selection capabilities in the source database and target DB instance tree views. Adds support for connecting to an Oracle source database using Transparent Network Substrate (TNS) names.	March 2, 2016
1.0.200	Maintenance release	Adds support for PostgreSQL as a target database engine. Adds the ability to generate converted schema as scripts and to save the scripts to files prior to applying the schema to the target DB instance.	January 14, 2016
1.0.103	Maintenance release	Adds offline project capability, the ability to check for new versions, and memory and performance management.	December 2, 2015
1.0.101	Maintenance release	Adds the Create New Database Migration Project wizard. Adds the ability to save the database migration assessment report as a PDF file.	October 19, 2015
1.0.100	Preview release	Provides the user guide for the AWS Schema Conversion Tool preview release.	October 7, 2015