2024년 09월 06일

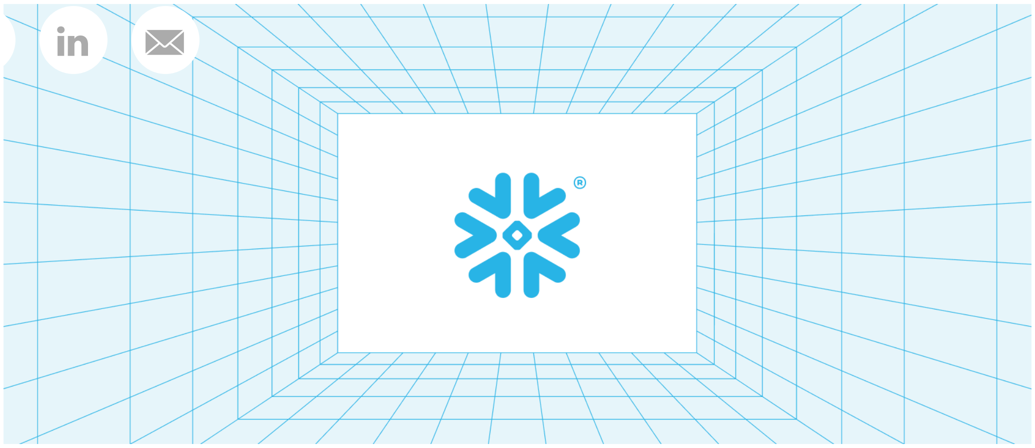
AUTHOR

Federico Zoufaly

Accelerate Code Conversion for Migrations to Snowflake with SnowConvert: Now Available For Free Table Conversion & Assessments

Core Platform

SHARE



At Snowflake, we are committed to helping customers migrate their workloads to the Snowflake Data Cloud with ease. Since 2020, SnowConvert has converted over 1.7 billion lines of code for over 45 million database objects, achieving an average conversion rate greater than 90%+, and resulting in saving 40–70% of the equivalent manual estimated conversion effort. Now, we are launching free table conversions and assessments via SnowConvert, for both customers and partners. Our goal is to help you and your organization accelerate code conversion when migrating a legacy database to Snowflake.

As someone interested in the mechanics of a migration, you likely already understand the value of the Snowflake Data Cloud and are looking for ways to speed up the conversion, so your team can realize the value of the Snowflake platform sooner rather than later.

A legacy RDBMS migration can be a complex project that involves many different stages, from planning, code conversion, testing, data migration to final cutover. Each stage is critical to the success of the project. Code conversion tends to be one of the most common challenges, given it tends to touch business logic that has been in place for many years. Typically, the documentation of legacy code is out of date, and expert coders that developed it have probably long left your company — or have maybe even forgotten details about the code. Snowflake makes SnowConvert available precisely to help simplify code conversion, from both an assessment and an automatic conversion perspective.

AUTHOR

Federico Zoufaly

WHAT IS SNOWCONVERT?

SnowConvert is a high-fidelity legacy database source code conversion tool. Currently, it supports code conversion from Teradata, Oracle, and SQL Server. The focus of SnowConvert is to accelerate the conversion of SQL DDL and DML, as well as scripts like stored procedures, functions, report generation and ETL scripts supported by the legacy RDBMS vendor (e.g., Teradata BTEQ scripting language or Oracle PL/SQL). More precisely, SnowConvert is designed to convert tables, views, stored procedures, macros, and join indexes, as well as DML like complex queries, inserts, updates, and proprietary scripts from your legacy data warehouse to Snowflake.

SHARE

SnowConvert is a client-side tool that is installed locally and will never access the legacy database directly or share any proprietary information. It can be run on Windows, Mac and Linux. If you want to get started with SnowConvert immediately, feel free to jump ahead, or you can read on to learn more about how exactly it works.

HOW DOES SNOWCONVERT WORK?

From a technical perspective, SnowConvert's initial stages are very similar to those of a compiler. Behind the scenes, SnowConvert parses your code into an Abstract Syntax Tree (AST) and creates an in-memory Symbol Table. Once this detailed representation of the code is created, SnowConvert's rule inference engine is activated and the translation of the code to the equivalent syntax supported by Snowflake occurs. At the last stage, the transformed AST is re-converted to source code format and saved together with all the reports that will aid the rest of the migration process.

AUTHOR

Federico Zoufaly

One big difference between a compiler and a code conversion tool is in the completeness of the output code. A compiler always takes code written at a certain level of abstraction and generates code at a lower level of abstraction that is consumable by the machine; the target being 100% translation. In a code conversion scenario, readability and maintainability of the output code are paramount. This is especially true when generating code that requires manual intervention, particularly when the source and target platforms do not have the exact same functionality. For example, in Teradata there is a Table type called SET that does not allow duplicate records. Since this functionality is not directly supported in Snowflake, the best approach is to convert a SET table to a standard Table.

SHARE

In the code example above, (Teradata on the left, Snowflake on the right), a message is generated as part of the converted code to guide the database architect about potential functional differences between the input and the converted code. In most scenarios, this difference can be ignored, but having business context is important to determine if additional consideration is required.

Additionally, some table parameters were simply eliminated in the conversion. These parameters in Teradata are used to describe physical properties of the table, but in Snowflake these are not relevant since Snowflake manages all physical infrastructure under the hood. The same situation applies for the INDEX, which is not necessary in Snowflake.

The example also includes a message related to a potential performance impact due to the encoding of the original tables. Through messages embedded in the output code, SnowConvert provides guidance to the database architect performing the migration.

Another scenario where there might be a functional difference is when converting data types. SnowConvert will output the most similar data type and adjust the code as much as possible to follow the rules of the original code. A message will always suggest a human review, and there should always be a customer business analyst available to understand the implications of the platform data type differences, decide whether to adopt SnowConvert's suggestion, and explain the differences to the ultimate users of the data.

AUTHOR

Federico Zoufaly

Whenever you are moving platforms, there is always the need to adapt some of the business rules to the new platform behavior.

While SnowConvert is designed to create a functionally equivalent output, there are cases where manual conversion is required. Let's see a stored procedure snippet example: (Teradata on the left, Snowflake on the right).

SHARE

In the above example, the variable declarations are converted first, including a cursor. The next instruction is a dynamic SQL instruction where the source code to be executed is inside a string. In this case, SnowConvert outputs a message that must be resolved by the database architect.

This is an ongoing process: we regularly update SnowConvert, continually adding and refining our translation rules to enhance the quality of code output.

HOW TO GET STARTED WITH SNOWCONVERT?

When starting a code conversion project, you first need to determine which code you need to convert. The best repository for all DDL and code stored in the database is the database itself. Snowflake provides a set of scripts that can be used to extract the DDL. When using these scripts, ensure that only the schemas that you need to convert are included, and exclude system catalogs, testing environments, backups, etc.

In addition to the DDLs, you may also have DML or other scripts that are typically included by the legacy vendor and are stored as files. Obtain the latest versions of these files and ensure they are exactly the same as what is used in production.

AUTHOR

Federico Zoufaly

Once you have all of the code that needs to be converted in a unified repository, you can start the code conversion project.

SnowConvert has two main operational modes: Assessment and Conversion.

In Assessment mode, covered below, SnowConvert will scan the code and provide a high-level inventory of all the different code units that are found. It will also provide a report of the conversion issues found in the code, allowing the migration team to understand the complexity of the code conversion project. This report is valuable during the planning phase of the project, providing input to design the timeline. Other reports generated by SnowConvert that help during the conversion process are the Missing Object References and the Object References. SnowConvert builds an in-memory symbol table, allowing us to report which objects are referenced by the source code but not included in the conversion set, and which objects depend on others, aiding in ordering the deployment in Snowflake.

SHARE

We’ve also included some sample projects to allow prospects and partners to see examples of converted code to assess the automation capability of SnowConvert. Each supported platform includes a sample project that can be run as a conversion. It includes original platform code and translated Snowflake output for both complex and simple scenarios, including scenarios that cannot be fully automated.

While in Assessment mode, you will also be able to see sample output for a few of your objects just to get a high-level idea of SnowConvert capabilities.

AUTHOR

Federico Zoufaly

SHARE

In addition, as part of the Assessment, SnowConvert will generate all of the DDL for TABLE objects that are part of the input. You don't need anything extra to be able to convert all of your TABLE DDLs — just convert, deploy to Snowflake, and see the results!

Once the assessment is complete, the next step is to actually execute the conversion. This step will take all of the source code and build a complete AST and symbol table that will be used to drive the conversion process. Each input file will be converted to a corresponding output file with the best possible converted code. As explained before, the converted code will contain a set of embedded messages as comments. These messages are specifically designed to help the conversion engineer finalize the conversion process. Remember, SnowConvert will typically convert most of the code without issues; however, you will typically have to do some manual work before you can deploy the converted code to Snowflake. SnowConvert is not magical (even though at times it may seem magical!) and can't do all of the work for you, but it greatly reduces and accelerates the conversion work required.

HOW TO ACCESS SNOWCONVERT?

Snowflake provides free training on how to use SnowConvert before we share access to the tool. Training is required given operating SnowConvert can seem deceptively simple. You point to some legacy code, you click convert, you obtain converted code. However, in order to obtain the best possible quality and avoid typical mistakes when using this powerful tool, it's important to understand its advantages and limitations. Once you have completed the training, you will get access to SnowConvert in Assessment mode, enabling you to assess all of your legacy code and generate converted code for all your TABLE DDLs.

If you want to utilize SnowConvert in Conversion mode, and convert objects other than just tables, you will need to enter into a support relationship with Snowflake Professional Services. At Snowflake, we want to make sure you have the best experience using the tool and share all of our lessons learned from converting millions upon millions of lines of code for hundreds of enterprises around the world.

In summary, SnowConvert is a sophisticated source code translation engine (not a “glorified search and replace tool”) that dramatically reduces the code conversion effort of a legacy database migration project. Since 2020, it has converted over 1.7 billion lines of code for over 45 million database objects with average automatic conversion rate greater than 90%, saving 40–70% of the equivalent manual estimated conversion effort on highly complicated code bases.

Learn more about SnowConvert, happy code conversion! 🙂

AUTHOR

Federico Zoufaly

SHARE

SHARE Snowfl Inc