**Mobilize.Net SnowConvert for Oracle**

***App Version Not Provided***

The purpose of this document is to summarize the technical considerations and code analysis in migrating SQL to Snowflake from Oracle that either have an impact on the automated code conversion or cannot be handled by automated code conversion, as well as providing a high-level inventory and automation capability of the code that will need to be addressed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OVERALL CONVERSION SUMMARY** | |  | **SQL CONVERSION SUMMARY** | |
| Total Files | 1 |  | Code Conversion Rate | 99.27 % |
| Total Files Not Generated | 0 |  |  |  |
| Conversion Speed | 15 lines/sec | |  | |
| Conversion Time | 00:00:03 |  | **OBJECT CONVERSION SUMMARY** | |
| Total Conversion Errors | 0 |  | Identified Objects | 4 |
| Total Parsing Errors | 0 |  | Object Conversion Rate | 50% |
| Total Warnings | 1 |  | Fully Converted Objects | 2 |
| Total Lines of Code (LOC) | 58 |  | Unrecognized Elements  Lines of Code in Unrecognized Elements  Wrapped Objects | 0  0  0 |

**TABLE OF CONTENTS**

[**FILE AND OBJECT LEVEL BREAKDOWN** 3](#_Toc95288203)

[**ISSUES BREAKDOWN** 5](#_Toc95288204)

[**ENVIRONMENT SETTINGS** 7](#_Toc95288205)

[**SCHEMAS** 7](#_Toc95288206)

[**TABLES: 100 Identified Object Conversion Rate** 7](#_Toc95288207)

[**VIEWS: 100 Identified Object Conversion Rate** 10](#_Toc95288208)

[**FUNCTIONS:** 10](#_Toc95288209)

[**PROCEDURES:** 11](#_Toc95288210)

[**SYNONYMS: - Identified Object Conversion Rate** 12](#_Toc95288211)

[**SEQUENCES: - Identified Object Conversion Rate** 12](#_Toc95288212)

[**PACKAGES: - Identified Object Conversion Rate** 13](#_Toc95288213)

[**PACKAGE BODIES: - Identified Object Conversion Rate** 13](#_Toc95288214)

[**STATEFUL PACKAGES: 0** 13](#_Toc95288215)

[**TRIGGERS: (*Work in progress*) % Identified Object Conversion Rate** 13](#_Toc95288216)

[**TYPES: (*Work in progress*) - Identified Object Conversion Rate** 14](#_Toc95288217)

[**NUMERIC DATA OPERATIONS CONSIDERATIONS** 14](#_Toc95288218)

[**TERMINOLOGY (CONCEPTS EXPLAINED)** 15](#_Toc95288219)

## **FILE** **AND OBJECT LEVEL BREAKDOWN**

***SQL – Files***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **File** | **Code Conversion Rate** | **Lines of Code** | **Total Object**  **Quantity** | **Parsing**  **Errors** |
| SQL | **99.27 %** | 58 | 4 | 0 |

* *The SQL Conversion Rate is considering both identified objects and unrecognized elements to determine its value. As a result, the average for the conversion rates of Identified Objects given below may differ from the overall rate given above.*
* *Total object quantity does not consider the objects inside packages, please refer to the “Packages bodies – Identified Objects” table for more information.*
* *Parsing errors are also considering the unrecognized objects, so this is not a sum of the parsing errors per object presented below.*

***SQL – Identified Objects***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Object** | **Conversion Rate** | | **Lines of Code** | **Total Object Quantity** | **Parsing**  **Errors** |
| **Object** | **Code** |
| Tables | **100 %** | **100 %** | 10 | 1 | 0 |
| Views | **100 %** | **100 %** | 5 | 1 | 0 |
| Materialized Views | **- %** | **- %** | - | - | - |
| Procedures | **0 %** | **97.9 %** | 29 | 2 | 0 |
| Functions | **- %** | **- %** | - | - | - |
| Packages | **- %** | **- %** | - | - | - |
| Packages Bodies | **- %** | **- %** | - | - | - |
| Sequences | **- %** | **- %** | - | - | - |
| Synonyms | **- %** | **- %** | - | - | - |
| Types | **- %** | **- %** | - | - | - |
| DB Link | N/A | N/A | - | - | - |
| Index | N/A | N/A | - | - | - |
| Triggers | N/A | N/A | N/A | N/A | N/A |

* *The Object Conversion Rate is considering only identified objects to determine its value. Objects with errors are not considered fully converted objects.*
* *The data inside the “SQL – Identified Objects” table shows only information about identified objects. Parsing errors that could not be attributed to an object category are shown in the “SQL – Files” table.*
* *If a hyphen (‘-’) is listed in the table above, it means no objects of that kind were found in the input folder.*
* *If N/A is listed in the table above, SnowConvert is currently not converting that object type. This is either listed for future conversion support or because the object type is not supported in Snowflake.*
* *Wrapped objects are not counted in the table of Identified Objects. These objects are considered as not converted. A solution to exclude these objects from the code conversion rate is being developed.*
* *Types are currently a work in progress and are not being fully converted. See the “Types” section below for more details.*

***Packages bodies – Identified Objects***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Object** | **Conversion Rate** | | **Lines of Code** | **Total Object Quantity** | **Parsing**  **Errors** |
| **Objects** | **Code** |
| Procedures | **- %** | **- %** | - | - | - |
| Functions | **- %** | **- %** | - | - | - |

* *The conversion rates given above, considers only the functions and procedures inside package bodies.*

## **ISSUES BREAKDOWN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Low** | **Medium** | **High** | **Critical** |
| # of issues | 1 | 2 | 0 | 0 |
| # of unique issues | 1 | 1 | 0 | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Error Code** | **Description** | **Instances** | **Level** |
| [MSCEWI3111](https://docs.mobilize.net/snowconvert/for-oracle/issues/mscewi3111) | Transaction control statement requires additional intervention | 2 | Medium |
| [MSCEWI1048](https://docs.mobilize.net/snowconvert/general/issues/mscewi1048) | Sequence warning | 1 | Low |

*If you are using the full version of SnowConvert, you can find out the exact file and location of each error in the Issues Report in the output reports folder created by SnowConvert.*

**Issue Categorization**

|  |  |  |  |
| --- | --- | --- | --- |
| LOW | MEDIUM | HIGH | CRITICAL |
| Warnings for source code that there was no direct conversion to Snowflake. The code has been converted to a functionally equivalent output but should still be reviewed. | Errors that produce a functional or runtime difference. The user may have to invest a *low amount* of manual effort to complete the conversion. | Errors that produce a functional or runtime difference. The user may have to invest a *high amount* of manual effort to complete the conversion. | Parsing errors or errors that cause significant conversion exceptions in SnowConvert. If the source code is correct, a significant intervention will be required. |

*For more information on the error codes and their severity, please visit our documentation page on errors:* [*https://docs.mobilize.net/snowconvert/oracle-issues-and-troubleshooting*](https://docs.mobilize.net/snowconvert/oracle-issues-and-troubleshooting)

## **ENVIRONMENT SETTINGS**

These are key settings that need to be considered in the Oracle migration.

*Parameters* - TIMESTAMP\_TYPE\_MAPPING (Snowflake) - Depending on customer requirements this alias mapping will need to be set to TIMESTAMP\_LTZ for the default behavior of the TIMESTAMP function to match the default behavior in Oracle.

## **SCHEMAS**

In Oracle, schemas are owned by users, while in Snowflake schemas are owned by roles. When a user is created in Snowflake, it doesn’t have a schema created for it, and only gets access to databases and schemas by being added to a role with database and schema permissions.

*Number of Schemas Containing Objects* - 1

## **TABLES: 100 % Identified Object Conversion Rate**

Code Conversion Rate: 100 %

Number of Tables: 1

Lines of Code: 10

Total Parsing Errors: 0

Successfully converted objects have been converted to an equivalent table and data types capable of storing the same data values as Oracle, but some functionality below needs to be further analyzed and considered based upon specific use cases.

*Constraints / Data Values* - Snowflake does not enforce or allow constraints. If your data input/ingestion process relies on these capabilities to validate data prior to being inserted into a table, these processes will need to be rearchitected to prevent duplicate or bad data.

* CHECK (**0**) - Not supported in Snowflake, checks the value of data being inserted to be a specific list of values or meet certain conditions.
* UNIQUE PRIMARY INDEX / PRIMARY KEY (**0** /**1**) - Not enforced in Snowflake. Definition is allowed for documentation purposes only.
* FOREIGN KEY (**0**) - Not enforced in Snowflake. Definition is allowed for documentation purposes only.

*Object Naming*

Invalid column names (**0**) - These column names are not supported in Snowflake and are converted to lowercase and added double quotes. This could require changes to external reporting/ETL systems. Examples include ‘current\_date’, ‘localtimestamp’.

* + Identifiers Requiring Double Quotes (**0**) - These objects require double quotes for defining in Snowflake and as a result are case sensitive unless the parameter QUOTED\_IDENTIFIERS\_IGNORE\_CASE has been set to True.

*Data Types*

* DATE (**0**) - Dates in Snowflake do not support an optional time part. In Oracle this can occur. Currently DATE is converted to DATE but could also be converted to TIMESTAMP\_NTZ to support cases where customer data is using the DATE datatype as a DATETIME.
* ROWID (**0**) - While Oracle stores the address of every row in the database, Snowflake does not have a similar concept, so these columns will need to be rearchitected.
* ANYDATA (**0**) - It is a self-describing type, containing an instance of a given type along with a description of the type it contains. Snowflake does not have an equivalent, but their behavior can be mimicked by changing them to VARIANT.
* ANYTYPE (**0**) – It is a type designed to store any named SQL type or unnamed transient type. Snowflake does not have an equivalent and these will need to be rearchitected.
* ANYDATASET (**0**) - It is a type containing a type description and a set of data instances of that type. Snowflake does not have an equivalent and these will need to be rearchitected.
* NUMBER without precision (**5**) - Number when defined without a scale/precision in Oracle allows for a flexible scale value for any given record from 0 to 38 provided the total precision never exceeds 38. Snowflake does not allow this and always has a fixed scale/precision. Data profiling of these columns showed that a max scale of **0** and max precision of **0** is needed for the current data, so there is no current issue to support these columns.
* INTERVAL (**0**) - Interval data types are not supported in Snowflake and are converted to varchars and manipulated as such using custom functions. Heavy use of intervals can cause performance issues and may require some re-architecture of the DDL and SQL.
* BLOB (**0**) - BINARY can be used instead but has a maximum of 8,388,608 bytes.
* CLOB/NCLOB (**5**) - VARCHAR can be used instead but has a maximum of 16,777,216 bytes (for single-byte).
* SPATIAL (**0**) - Spatial types are supported. Specific use cases must be evaluated.
* XMLTYPE (**0**) – XmlType columns can be migrated to Snowflake using the Variant Datatype.
* CUSTOM (**0**) - While Oracle allows the creation of user defined data types, Snowflake does not have a similar concept, so these columns will need to be rearchitected.

*Data Values*

* BINARY Defaults (**0**) - Binary data types do not currently support a default value being defined.
* TIME Defaults (**0**)
  + - Time with TimeZone is not a supported type in Snowflake.
  + FLOAT Default (**0**)
    - Timestamp can be defined as a default value for a FLOAT column in Oracle but is not supported in Snowflake.
  + IDENTITY Columns (**1**) - These are converted to sequences and can represent an area that requires additional testing.
  + SYS\_CONTEXT Defaults (**0**) - When passing 'USERENV' and 'SESSIONID'

parameters to SYS\_CONTEXT function, the column must be varchar in Oracle

in order to hold the Snowflake current\_session value.

*PARTITION BY* (**0**) - Partitions are not supported in Snowflake but have a similar functionality called CLUSTER BY. By default, the PARTITION BY statements are converted to CLUSTER BY but are commented out as CLUSTER BY does not support all the same cases that can be defined in Partitions, so some must be modified before enabling. Each CLUSTER BY statement will need to be individually evaluated based upon the necessity for performance.

*UPPER FUNCTION* (**0***)*

## **VIEWS: 100 % Identified Object Conversion Rate**

Code Conversion Rate: 100 %

Number of Views: 1

Lines of Code: 5

Total Parsing Errors: 0

KEEP Function (**0**) – Not supported.  
XML Function (**0**) – Not supported.

JSON Function (**0**) – Not supported.

ORA\_HASH (**0**) - converted to HASH but generates a different value because of different algorithms. Should only be an issue if data is being exported to other systems and matched on these values hash values.

ROWID() (**0**)

ROWNUM() (**0**)

EXTRACTVALUE for environment variables (**0**)

NLSSORT() (**0**)

OLAP FUNCTION REFERENCES (**0**)

CUSTOM TYPES (**0**) – SnowConvert counts the custom type columns selection only if the column name is explicitly specified in the select list.

## **MATERIALIZED VIEWS: - % Identified Object Conversion Rate**

Code Conversion Rate: - %

Number of Views: -

Lines of Code: -

Total Parsing Errors: -

## **FUNCTIONS:**

Snowflake does not support the same features in functions as Oracle. Not all possible logic and database access can be consistently replicated with Snowflake JavaScript or SQL functions. Some of them are transformed to Snowflake procedures to keep the functional equivalence, but this kind of functions cannot be called from queries due to procedures behavior. These functions are currently transformed to dummy user defined functions.

**Identified Functions Object Conversion Rate: - %**

Code Conversion Rate: - %

Number of Functions: -

Lines of Code: -

Total Parsing Errors: -

**Identified Functions Object Conversion Rate (inside packages): -%**

Code Conversion Rate (inside packages): -%

Number of Functions (inside packages): -

Lines of Code (inside packages): -

Total Parsing Errors (inside packages): -

Total Functions: -

Wrapped Functions (**0**): Wrapped functions in Oracle are “encrypted” and will not export correctly.

Functions by Language Type:

* SQL - 0
* JAVA - 0 (Not addressed by converter)
* C - 0 (Not addressed by converter)

*Note*: Only functions written by SQL are evaluated by SnowConvert and appear in the code conversion rate, number of functions, lines of code, and total parsing errors listed above.

SQL Function Conversion Breakdown:

Total of Functions transformed to Snowflake procedures: 0

Functions transformed to Snowflake JavaScript functions: 0

Functions transformed to Snowflake SQL functions: 0

Function calls inside queries: 0

Function calls outside queries: 0

If you want to know more about this transformation you can follow this link: <https://docs.mobilize.net/snowconvert/for-oracle/sql-translation-reference>

## **PROCEDURES:**

**Identified Procedure Object Conversion Rate: 0 %**

Code Conversion Rate: 97.9 %

Number of Procedures: 2

Lines of Code: 29

Total Parsing Errors: 0

**Identified Procedure Object Conversion Rate (inside packages): -%**

Code Conversion Rate (inside packages): -%

Number of Procedures (inside packages): -

Lines of Code (inside packages): -

Total Parsing Errors (inside packages): -

Total procedures: 2

Wrapped Procedures (**0**): Wrapped procedures in Oracle are “encrypted” and will not export correctly.

Procedures that use Oracle Built-In Packages (**0**)

PRAGMA AUTONOMOUS\_TRANSACTION (**0**): Allows a subprogram to work in an independent transaction from the main/parent transaction.

ROWID (**0**)

ROWNUM (**0**)

Cursors (**-**)

Loops (While, For, and Loop) (**-**)

Dynamic SQL Statements (**0**)

DBMS\_SQL Package Occurrences (**0**): DBMS\_SQL package provides subprograms to use dynamic SQL using cursors.

SQLCODE Occurrences (**0**)

## **SYNONYMS: - % Identified Object Conversion Rate**

Snowflake does not have synonyms as a database object, so synonyms are converted to views or the synonym is replaced with the underlying referenced object from that synonym.

Code Conversion Rate: - %

Number of Synonyms: -

Lines of Code: -

Total Parsing Errors: -

## **SEQUENCES: - % Identified Object Conversion Rate**

Code Conversion Rate: - %

Number of Sequences: -

Lines of Code: -

Total Parsing Errors: -

## **PACKAGES: - % Identified Object Conversion Rate**

Packages are a set of related functions and procedures that are bundled together. While Snowflake does not currently support converting packages, the conversion capability is in progress as an addition to the conversion tool.

Code Conversion Rate: - %

Number of Packages: -

Wrapped Packages (**0**): Wrapped packages in Oracle are “encrypted” and will not export correctly.

## **PACKAGE BODIES: - % Identified Object Conversion Rate**

Package bodies have the implementation of functions and procedures that were declared in the package. While Snowflake does not currently support converting package bodies, functions and procedures inside package bodies are being extracted.

Code Conversion Rate: - %

Number of Package Bodies: -

Wrapped Package Bodies (**0**): Wrapped packages in Oracle are “encrypted” and will not export correctly.

### **STATEFUL PACKAGES: 0**

Packages or package bodies that include constants and variables. This includes packages without a package body or vice versa. Also package and package bodies with the same name are only counted once.

Number of Stateful Packages: 0

## **TRIGGERS: (*Work in progress*) % Identified Object Conversion Rate**

Snowflake does not have triggers but depending on the requirements of the use case for triggers and how frequently they need to run, this functionality could be converted to tasks in Snowflake.

*Note*: Transformation and parsing for TRIGGERS is still a work in progress, therefore no information is shown above.

## **TYPES: (*Work in progress*) - % Identified Object Conversion Rate**

Snowflake does not have user defined types but depending on the requirements of the use case for types, a VARIANT that contains JSON is being used in Snowflake.

* Standalone Custom Types: These were declared on a CREATE TYPE statement. Their usage is mostly oriented to DDL definitions.
  + References changed to Variant (0): References to types in column definitions or parameters that were successfully resolved and changed to Variant.
  + References to not found types (0): References to types that were not found, but changed to Variant nevertheless.
  + Object Types (0): Types that are defined as an object and are being partially transformed.
  + Types without Body (0): Types that are defined without a body will not be transformed.
  + Subtype Types (0): Types that are defined as a subtype of another type will not be transformed.
  + Array Types (0): Types that are defined as an array of another type will not be transformed.
  + Nested Table Types (0): Types that are defined as a nested table of another type will not be transformed.
  + Wrapped Types (0): Wrapped types in Oracle are “encrypted” and will not export correctly.
  + Wrapped Type Bodies (0): Wrapped type bodies in Oracle are “encrypted” and will not export correctly.
* Embedded Custom Types: These were declared as a member of another statement, like a Package, Function or Procedure Declaration. Their usage is mostly to moving data from one variable to another and storing temporal, or internal, data and will be migrated depending on their usages.
  + Record Types (0): Types defined as a structure of multiple fields.
  + Associative Arrays (0): Types defined as a dictionary-like collection of elements.
  + Varrays (0): Types defined as a static collection of elements.
  + Nested Tables (0): Types defined as a dynamic collection of elements.
  + Cursor Types (0): Types defined as a cursor.

*Notes*:

* Transformation and parsing for TYPES is still a work in progress, therefore the data shown above may change in the future.
* Method members and type options are currently not being supported by SnowConvert.
* Usages of types in PL/SQL are still a work in progress and therefore are still not being converted.

**NUMERIC DATA OPERATIONS CONSIDERATIONS**

* Scales of intermediate values: Sometimes, when performing an operation between two Numbers with floating point values it is possible to incur on issues that are related to the way Snowflake internally handles these operations, such as:

SELECT 0.00000000000000000001 / 0.0000000000000000001;

# **GLOSSARY**

In this section, we try to explain concepts used in multiple report documents generated by SnowConvert. You can also read more about the terminology used in SnowConvert on our [documentation page](https://docs.mobilize.net/snowconvert/general/reports#glossary).

* **Total Conversion Issues,** the total count of conversion issues that happened during the conversion process. A conversion issue is an error that happened during the conversion process related to file I/O, memory management, or any abnormal situation that could not be handled. These are unhandled code exceptions and are considered critical issues
* **Total Parsing Errors,** the total count of parsing errors that occurred during the code analysis process. A parsing error occurs when the parser (the component that reads the source code files) encounters something unexpected. This usually means a syntax error, which refers to a code element in the file that did not match the SQL grammar specification that the parser was expecting. In other cases, these errors can also occur because the parser is not yet ready to support a specific grammar. Parsing errors are also considered critical issues. If this number is high in relation with the migration workload size, input code revision is advised.
* **Total Warnings,** the total count of warnings that SnowConvert generated for the given input. A warning is inserted when the translation of a specific element is mostly functionally equivalent but there are some corner cases in which some user intervention might be required. They have low severity because their intention is to provide information that can be reviewed if the code shows any kind of functional difference when executed on the target platform.
* **Total Lines of Code (LOC),** the total number of lines in the text of the source code files, that are not comments or blank lines, that were processed by the conversion tool.
* **Identified Objects**, in general, this refers to TABLES, VIEWS and PROCEDURES. These objects are classified as top-level because they are usually the “root” elements for a database dialect DDL, and they can contain other “smaller” definitions. The top-level objects vary from one SQL dialect to another (Oracle, Teradata, SqlServer, etc). For more information about top-level objects in each SQL dialect SnowConvert supports, you can check [this link](https://docs.mobilize.net/snowconvert/general/reports/top-level-objects-report) in our documentation. Parsing errors might cause SnowConvert to **not** be able to properly count all top-level objects.

In Oracle translation, identified objects count is **not** considering the functions and procedures inside packages, for a detailed information on those objects, please check the *Packages bodies – Identified Objects* table, or the *Top Level Objects report* in the output.

* **Unrecognized Elements,** any code element (or parts of them) such as DML, DDL, control statements, with parsing errors that SnowConvert was unable to process.
* **Lines of Code in Unrecognized Elements,** the total lines of code in all the unrecognized elements. This is a good indicator of how much code SnowConvert was **not** able to process.
* **Fully Converted Objects,** the count of top-level objects that were fully converted without any error in any of their sub-parts. They are considered ready for deployment.
* **Code Conversion Rate (SQL), t**his metric is calculated using as reference the source code of both recognized and unrecognized elements in the given workload. The conversion rate is the percentage of source code that was successfully converted by SnowConvert into functionally equivalent Snowflake code. Take into consideration that unrecognized elements (because of parsing issues) will affect this metric, as their source code will be counted as not converted. Furthermore, elements that do not have equivalence in Snowflake will also punish the conversion rate, because even though SnowConvert is able to process them (frequently by removing them or commenting them) the translation does not represent a functionally equivalent code. For example, if there is a lot of code of indexes (which do not have an equivalent in Snowflake) the conversion rate can be highly punished because the source code of those unsupported elements might represent a significant part of the overall workload.
* **Object Conversion Rate,** the percentage between **fully** converted objects and identified objects.