SSMA issues and possible remedies when migrating Oracle databases

Prepared by

DM Jumpstart Engineering Team (askdmjfordmtools@microsoft.com)

Note: The detail provided in this document has been harvested as part of a customer engagement sponsored through the Azure Data Migration Jumpstart Program.

Table of Contents

Introduction

This document was written in a context of an Oracle Assessment to Azure SQL DB migration project. It contains the most common error messages that were found during the assessments, a brief description and a possible remedy that can help you to solve the issue. Of course, not all possible error messages are in the document. If you have feedback or suggestions for improving this data migration asset, please contact the Data Migration Jumpstart Team (askdmjfordmtools@microsoft.com).

Thanks for your support!

Note: For additional information about migrating various source databases to Azure, see the Azure Database Migration Guide.

O2SS0001: Unsupported SQL clause

Oracle allows you to assign a non-scalar condition in WHERE clause. The problem is that SQL Server doesn't support conditions of this type. Thus, the SQL Server Migration Assistant (SSMA) for Oracle doesn't convert queries with a a non-scalar condition in WHERE clause. So, when you try to convert the source code that includes a non-scalar condition in a WHERE clause, SSMA will generate an error O2SS0001.

Possible Remedies

To resolve this issue, you should rewrite the converted query using EXISTS condition. This condition tests for the existence of rows in a subquery. Then you should substitute the AND condition with IN condition.

More information on https://www.dbbest.com/blog/ssma-convert-unsupported-sql-clause/

O2SS0004: Unparsed SQL

Possible Remedies:

- Replace Pivot with exists, aliases or case;
- Create Linked server for the Oracle Database Links

https://www.dbbest.com/blog/oracle-sql-server-migration-ssma-convert-pivot-operator/

O2SS0005: Source datatype not recognized

Oracle has an object data type called *anydata*. This data type supports wide range of data types. For example, when creating a table with a column defined as anydata type, the column can store many types of data from string to numeric. SSMA does not support migration of anydata and when migrating Oracle database containing the type, SSMA raise a migration error O2SS0005: Source datatype not recognized.

MSSQL has a similar data type called sql_variant.

MSSQL Server what can come close to TABLE OF TableName%RowType is Declare a table type variable (in case it is collection). Another way may be creating a #temp table. But temp table method may involve IO which need to be aware of and also in there are limitation where you can create #temp table. For a single row, it is better to declare separate scalar variable for each column.

Table variable Method

Declare @Dept Table (Department_id Int, Name Varchar(100), Description Varchar(1000)) The below statement will insert all the rows from Department to @Dept variable.

```
Select *INTO @Dept
FROM
Department
```

Temp table method

```
Select * into #Temp_Dept
FROM Department
```

The above statement will create a temp table on the fly. This can be used in Procedure but not in Function.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/21/migrating-oracle-anydata-to-sql-server/

O2SS0006: Type 'INTERVAL DAY(5) TO SECOND(3)'

Oracle uses specific interval expressions to store a period. SQL Server does not have a direct analogue for this data type. Thus, when you try to convert interval expressions using SQL Server Migration Assistant (SSMA) for Oracle, you will get an error O2SS0006.

Possible Remedies

To resolve this issue, you should change the data type for [INT_DTS] column from varchar (8000) to [INT_DTS] varchar (100). SSMA converts all unsupported types to varchar (8000), but in that case, you do not need to define that huge column size.

More information on https://www.dbbest.com/blog/ssma-convert-interval-expressions/

O2SS0007: Check constraint condition not parsed

SQL Server Migration Assistant (SSMA) for Oracle is not able to convert a BINARY_FLOAT column with a constraint for checking a NAN (Not a Number) condition. In Oracle, the BINARY_FLOAT data type allows an application to store number underflow or overflow values. SQL Server generates an error as soon as a number overflow or underflow is computed and thus can't store the value in a FLOAT data type. When you try to convert the schema of a table having a constraint condition which is checking for NAN values, SSMA generates an error "Error O2SS0007 Check constraint condition not parsed" because SQL Server doesn't support floating point conditions like NAN.

Possible Remedies

Create a new table in SQL without including the check constraint.

Modify the application code for inserting the values in this table. Create a TRY… CATCH block which would be restricting the users to insert an unsupported value and generate an error message similar to an error generated from a check constraint.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/15/migrating-oracle-to-sql-server-using-ssma-error-o2ss0007-check-constraint-condition-not-parsed/

O2SS0013: EXECUTE IMMEDIATE not converted

In Oracle PL/SQL, you can use EXECUTE IMMEDIATE statement to execute a dynamic SQL statement. In Microsoft SQL Server Transact-SQL, you can use EXECUTE statement or EXECUTE sp_executesql stored procedure to execute dynamic SQL statements.

Possible Remedies

You have to use EXECUTE sp_executesql if you need to execute a dynamic SQL with parameters.

More information on

http://www.sqlines.com/oracle-to-sql-server/execute_immediate
https://channel9.msdn.com/Events/Ignite/Microsoft-Ignite-Orlando-2017/BRK3356

O2SS0029: Cannot convert EXIT statement

SQL Server Migration Assistant (SSMA) for Oracle doesn't convert the EXIT statement when an outer loop is exited from inside the scope of inner loop. The LOOP statement is used to define a loop with an indeterminate number of iterations. The EXIT statement exits a loop and transfers control immediately to the end of the loop. Whenever an outer loop is exited from inside the scope of inner loop, SSMA cannot always convert the exit statement and generates following error: 'Error O2SS0029 Cannot convert EXIT statement'.

Possible Remedies

Use the GOTO approach in SQL Server. The GOTO statement is used to transfer control of program execution to the statement that has a specified statement label.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/01/migrating-oracle-to-sql-server-using-ssma-error-o2ss0029-cannot-convert-exit-statement/

O2SS0050: Conversion of identifier is not supported

Oracle allows you to create a query with aggregate functions referring to a remote table. For example, if your query includes two tables: one from the local schema and another from the remote database. You can simply use a database link in Oracle to create that construction. The problem is that SQL Server Migration Assistant (SSMA) for Oracle doesn't convert queries that refer to the remote objects through the database link. So, when you try to convert a query with aggregate functions referring to a remote table, SSMA will generate an error O2SS0050.

Possible Remedies

To resolve this issue, you should create the linked server on the SQL Server and convert all aggregate expressions manually. Consider using the same name for the linked server as Oracle's database link. Finally, you will have to put the converted code in the appropriate place.

More information on https://www.dbbest.com/blog/ssma-convert-aggregate-functions/

O2SS0083: Unresolved identifier

Error is commonly related to the existence of database links, create a linked server or use openrowset.

Possible Remedies

See O2SS0050: Conversion of identifier is not supported

O2SS0086: The literal ‘INTERVAL YEAR TO MONTH'

In Oracle, you may use specific interval literals that indicate a certain period. SQL Server does not have a direct analogue for interval literals. Thus, when trying to convert interval literals using SQL Server Migration Assistant (SSMA) for Oracle, you will get an error O2SS0086. The term literal refers to a fixed data value just as the constant value term. You can specify interval literals in terms of years and months or in terms of days, hours, minutes, and seconds. Oracle Database supports two types of interval literals, YEAR TO MONTH and DAY TO SECOND. SSMA can convert neither of them, so it generates one of the following error messages:

> O2SS0086: The literal ‘INTERVAL YEAR TO MONTH’ was not converted.
> O2SS0086: The literal ‘INTERVAL DAY TO SECOND’ was not converted.

Possible remedies

To fix this, you should use DATEADD SQL Server function with the relevant datepart (e.g. year – yy, month – mm etc.) and the appropriate sign "+" or "—".

More information on https://www.dbbest.com/blog/ssma-convert-interval-literals/

O2SS0157: Dynamic string for OPEN...FOR not converted

SQL Server Migration Assistant (SSMA) for Oracle doesn’t convert the dynamic string within an OPEN...FOR statement and presents a natural alternative to returning cursors by using a result set returned by Transact-SQL procedures. The OPEN-FOR statement implements the query associated with a cursor variable and assigns database resources to process the query and recognizes the result set. A CURSOR is a mechanism by which you can assign a name to a "SELECT statement" and manipulate the information within that SQL statement

If you have any dynamic string in your code, the SSMA tool will generate the following error message: “OPEN ... FOR statement will be converted, but the dynamic string must be converted manually”.

Possible Remedies

- Remove the single quotes from the dynamic query to make it a static query and run SSMA against the code again
- Use the natural approach followed in SQL Server which is – returning the result set directly from executing stored procedures.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/20/migrating-oracle-to-sql-server-using-ssma-error-o2ss0157-dynamic-string-for-open-for-not-converted/

O2SS0179: Packaged variable (constant) data type not recognized

Some of the reasons why SQL Server Migration Assistant (SSMA) for Oracle doesn't convert few of the Oracle data types (like MLSLABEL) when a variable of that type is used in a package. MLSLABEL data type is used by Trusted Oracle and is used to store the binary format of an operating-system label. Trusted Oracle uses a label to control access to information. The maximum width of an MLSLABEL column is 255 bytes. Any labels that are valid on your operating system can be inserted into an MLSLABEL column. When you insert a label into an MLSLABEL column, Trusted Oracle implicitly converts the data into the binary format of the label.

To store packaged variables, SSMA for Oracle uses special procedures (like sysdb.ssma_oracle.set_pv_varchar) that reside in a SysDB database in a ssma_oracle.db_storage table. Whenever SSMA tries to convert unsupported data types it gives an error because SSMA cannot find its corresponding object in the SQL Server.

Possible Remedies

You should consider replacing the unsupported data types with any other supported data types (like VARCHAR) as shown below.

More information on https://blogs.msdn.microsoft.com/ssma/2011/05/12/migrating-oracle-to-sql-server-using-ssma-error-o2ss0179-packaged-variable-constant-data-type-not-recognized/

O2SS0231: Foreign key cannot be converted

Oracle allows you to create foreign key for table using columns with different data types. But SQL Server Migration Assistant (SSMA) for Oracle cannot convert them to SQL Server correctly because SQL Server doesn't support the foreign keys that use columns with different data types. So, when you try to convert the original code that includes the foreign key with the columns of different data types, SSMA will generate the following error message: "Error O2SS0231: Foreign keys with different types of columns and referenced columns cannot be converted".

Possible Remedies

Alter table's columns in such a way that avoid data truncation during data migration from Oracle to SQL Server>

More information on https://www.dbbest.com/blog/oracle-sql-server-convert-foreign-keys/

O2SS0245: Cursor conversion in return statements not supported

SQL Server Migration Assistant (SSMA) for Oracle cannot convert some of the statements that have cursor as a return type in any function. A cursor is a mechanism by which you can assign a name to a "SELECT statement" and manipulate the information within that SQL statement. Database programmers use cursors to process individual rows returned by database system queries. In SSMA, the conversion of cursors in return statements is not supported. So, if a function is defined with a cursor as return type, then SSMA generates O2SS0245 error.

Possible Remedies

One possible remediation is to create and use temporary table inside the function of SQL Server instead of using cursor. And then the reference of this temporary table is returned by the function.

More information on https://blogs.msdn.microsoft.com/ssma/2011/05/19/migrating-oracle-to-sql-server-using-ssma-error-o2ss0245-cursor-conversion-in-return-statements-not-supported/

O2SS0260: Wrapped package cannot be converted

Wrapping (encrypting) is a process of hiding the source code. Wrapping helps to protect the source code from the competitors and others who might misuse it. In Oracle, one can wrap the source code using DBMS_DDL subprograms. Whenever you try to convert the WRAPed (encrypted) PL/SQL code, SSMA gives a warning message saying that the “Required code is wrapped”. When you skip this warning by clicking Continue button, SSMA generates an error message, “Error O2SS0260 Wrapped package cannot be converted”. Also, you will be able to view only the Conversion statistics and not the code in the Assessment Report window, as the code is wrapped.

Possible Remedies

Try to get the PL/SQL Code and manually convert it to the corresponding SQL Server code and add the option “WITH ENCRYPTION”

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/17/migrating-oracle-to-sql-server-using-ssma-error-o2ss0260-wrapped-package-cannot-be-converted/

O2SS0264: Unable to convert cursor or cursor variable as a function or procedure call parameter

SQL Server Migration Assistant (SSMA) for Oracle does not convert the PL/SQL block when a cursor or cursor variable is passed as a parameter to a function or procedure call. A cursor is a mechanism by which you can assign a name to a "SELECT statement" and manipulate the information within that SQL statement. Database programmers use cursors to process individual rows returned by database system queries. In Oracle, SYS_REFCURSOR is used to pass cursors from and to a stored procedure. When a cursor or cursor variable is passed as a parameter to a function or procedure call, SSMA cannot convert that statement and generates following error "Error O2SS0264: Unable to convert cursor or cursor variable as a function or procedure call parameter"

Possible Remedies

When SSMA convert the Oracle procedure in SQL Server, it converted the CURSOR type to varchar (8000). But in SQL server we can declare cursor data type in an OUTPUT Parameter. After changing its type, we have to remove this initialization

More information can be found on https://blogs.msdn.microsoft.com/ssma/2011/06/20/migrating-oracle-to-sql-server-using-ssma-o2ss0264-unable-to-convert-cursor-or-cursor-variable-as-a-function-or-procedure-call-parameter/

O2SS0265: Unable to convert condition

SQL Server Migration Assistant (SSMA) for Oracle doesn’t convert the statement having Cursor attributes with any conditional operator. A CURSOR is a mechanism by which you can assign a name to a "SELECT statement" and manipulate the information within that SQL statement. Database programmers use cursors to process individual rows returned by database system queries. Although SSMA facilitates emulation of various Cursor attributes, it generates Error O2SS0265 when it encounters Cursor statement with some conditional logic, like IS NULL, etc.

Possible Remedies

The solution is to rewrite the conditional block in the SQL Server Metadata Explorer after converting the code using SSMA. Majority of the code gets converted.

More information on https://blogs.msdn.microsoft.com/ssma/2011/05/19/migrating-oracle-to-sql-server-using-ssma-error-o2ss0265-unable-to-convert-condition-with-cursor-attributes/

O2SS0293: Columns list in set clause cannot be converted

Oracle PL/SQL allows you to perform multi column update through sub-query. Consider the following example:

```
CREATE TABLE ACCOUNT
   (
       ACCOUNT_ID NUMBER NOT NULL,
       ACCOUNT_OWNER VARCHAR2(30) NOT NULL
   );

UPDATE ACCOUNT
    SET (ACCOUNT_ID, ACCOUNT_OWNER) = (SELECT 1, 2 FROM dual)
    WHERE ACCOUNT_ID = 10;
```

SSMA does not support converting UPDATE statement with sub-query.

Possible Remedies

Convert the update statement to a T-SQL update statement

```
UPDATE    acct
SET            acct.ACCOUNT_ID = updtqry.col1,
                  acct.ACCOUNT_OWNER = updtqry.col2
FROM       ACCOUNT acct, (SELECT 1 col1, 2 col2) updtqry
WHERE    acct.ACCOUNT_ID=10;
```

More information on https://blogs.msdn.microsoft.com/ssma/2011/05/10/migrating-oracle-to-sql-server-using-ssmaerror-o2ss0293-columns-list-in-set-clause-cannot-be-converted/

O2SS0302: Unable to convert recursive call in inline call conversion

The issue is caused by the fact that SQL doesn’t support calling a function from the function itself. In pseudo code your issue looks like:

```
Function couper_texte(string){
PerformOperationSubstring
Do While (NOTOK){
Couper_texte(string) }
Return string
}
```

Possible Remedies

A solution here should be implemented from the Oracle side to be a quick win:

```
Function couper_texte(string){
Do While (NOTOK){
Couper_texte_2(string) }
Return string }
Function couper_texte_2(string){ PerformOperationSubstring}
```

By creating a new function with the original name, we make sure that not a single call to it in other functions/procedures must be adjusted.

O2SS0339: Cannot convert usage of standalone user-defined type

A User Defined Type (UDT) is an entitled data type that is made in the database by the user. A UDT can be a distinct type which segments a common representation with a built-in data type. Whenever you create a User Defined data type and make a reference of that data type in a packaged query to declare a variable, SSMA does not recognize this data type and hence does not convert it to corresponding SQL Server code. This is because the user defined data type is not defined in the scope of the code where the variable is being declared and hence generates error "Error O2SS0339 Cannot Convert Usage of Standalone User-Defined Types".

Possible Remedies

Create the user defined data types inside the package specification keeping the remaining code as is. When you define the UDTs inside the package specification, the UDTs comes in the scope of the Package and hence the error is resolved.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/29/migrating-oracle-to-sql-server-using-ssma-error-o2ss0339-cannot-convert-standalone-user-defined-types/

O2SS0343 FORALL statement with SAVE EXCEPTION clause is not supported

SQL Server Migration Assistant (SSMA) for Oracle doesn't support the SAVE EXCEPTION clause in the FORALL statement. Exception handling is a programming language construct or mechanism designed to handle the occurrence of exceptions, special conditions that change the normal flow of program execution. In Oracle, FORALL statement lets you run multiple DML statements very efficiently and can only repeat a single DML statement, unlike a general-purpose FOR loop. SAVE Exception statement is an optional keyword that cause the FORALL loop to continue even if some DML operations fail.

The Oracle exception model differs from Microsoft SQL Server both in exception raising and exception handling. It is preferable to use the SQL Server exceptions model as part of the Oracle PL/SQL code migration. Whenever FORALL statement used with SAVE exception clause, SSMA doesn't support it and generate following error message: Error O2SS0343 FORALL statement with SAVE EXCEPTION clause is not supported.

Possible Remedies

One possible remediation is to use the try and catch block to handle the exceptions in T-SQL using ERROR_NUMBER and ERROR_MESSAGE functions instead of the SQLCODE and SQLERRM Oracle functions.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/01/migrating-oracle-to-sql-server-using-ssma-error-o2ss0343-forall-statement-with-save-exception-clause-is-not-supported/

O2SS0346: Referencing columns not nullable

In Oracle we have a column which is marked for accepting Null (can accept Null) and also a part of Referential integrity. SSMA raises this as an error because, if the column is part of Referential Integrity, it cannot be NULL.

Possible Remedies

Change the project settings int the SSMA to accept NULL values.

More information on http://sqlthreads.blogspot.com/2014/09/how-to-resolve-o2ss0346-sql-server.html

O2SS0351: Conversion of collection method not supported

A collection is an ordered group of elements, all of the same type. It is a general concept that encompasses lists, arrays, and other familiar data types. You can use the methods EXISTS, COUNT, LIMIT, FIRST, LAST, PRIOR, NEXT, EXTEND, TRIM, and DELETE to manage collections in Oracle whose size is unknown or varies. Whenever you make a reference of Limit method in collection query, SSMA is not able to resolve that method. Hence, usage of this collection method results in error O2SS0351. In Oracle, LIMIT returns the maximum number of elements that array can contain (which you must specify in its type definition).

Possible remedies

Replace the Limit method with Count method of SQL server to get the number of elements in collection

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/29/migrating-oracle-to-sql-server-using-ssma-error-o2ss0351-conversion-of-collection-method-not-supported/

O2SS0352: BULK COLLECT INTO clause in SELECT statement not converted

A collection is an ordered group of elements, all of the same type. It is a general concept that encompasses lists, arrays, and other familiar data types. Each element has a unique subscript that determines its position in the collection. The DISTINCT clause specifies that only unique value can appear in the result set. The BULK COLLECT INTO clause binds the output of the query to the collection, resulting in less communication between the PL/SQL and SQL engines. SSMA provides for a migration path for Oracle collections. However, if the SELECT statement used to populate the collection uses a DISTINCT clause, SSMA generates the O2SS0352 error.

Possible Remedies

Replace DISTINCT clause with UNIQUE clause (a synonym of DISTINCT)

More information on https://blogs.msdn.microsoft.com/ssma/2011/05/12/migrating-oracle-to-sql-server-using-ssma-error-o2ss0352-bulk-collect-into-clause-in-select-statement-not-converted/

O2SS0359: Cannot get description for return type of function call expression

SQL Server Migration Assistant (SSMA) for Oracle is not able to convert the record set which is returned as the return type of the function. A Function is a block of code that performs a task and then returns control to the calling code. When it returns control, it also returns a value to the calling code. When a function returns a value, the value is returned via a return statement to the caller of the function, after being implicitly converted to the return type of the function in which it was defined. Whenever you call a function that returns a record in another procedure, SSMA is not able to resolve that data type. Hence, when you try to use this record set in your called procedure, SSMA generates an error.

Possible Remedies

The solution is to rewrite the code in SQL Server. As SQL Server supports scalar functions, inline table-valued functions and multi statement table-valued functions, you can declare a temporary Table (@my table) within the T_SQL Code of called function. In the code you fill this table using your same business Logic and then return this table back to the calling environment. In the calling function, you also must use table variable to store the return value (record set in our case) of called function.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/20/migrating-oracle-to-sql-server-using-ssma-o2ss0359-cannot-get-description-for-return-type-of-function-call-expression/

O2SS0376: Unable to convert record variable in this context

The %ROWTYPE attribute in Oracle defines the particular record type of a row in a table. A common usage of %ROWTYPE attribute is to have variables declared as ROWTYPE of a table to transfer data in and out of a procedure or function. An IN ROWTYPE parameter of a function or procedure can be set with a default value. Often, the IN ROWTYPE parameter is defaulted to NULL.

Possible Remedies

Rewrite the stored procedure manually.

More information on https://blogs.msdn.microsoft.com/ssma/2012/02/15/proceduresfunctions-with-rowtype-parameters-defaulted-to-null/

O2SS0408: Collection or record type is not supported

A collection is an ordered group of elements, all the same type. It is a general concept that encompasses lists, arrays, and other familiar data types. Each element has a unique subscript that determines its position in the collection. Whenever you make a reference of a user defined data type in collection query, SSMA is not able to resolve that data type. Hence, when you try to insert the values in table, SSMA generates an error. SSMA provides direct emulation of only the Integer and String data type and rest of the things are treated as Records.

Possible Remedies

Rewrite the code in Oracle. Instead of creating a user defined data type separately, just declare a data type as a record of a non-user defined data type and make its reference while assigning the values.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/01/migrating-oracle-to-sql-server-using-ssma-error-o2ss0408-collection-or-record-type-is-not-supported/

O2SS0418: Failed to parse statement batch with package definition

SQL Server Migration Assistant (SSMA) for Oracle doesn't convert the PL/SQL block when using invalid syntax while creating a package in Oracle code. Whenever we used the invalid syntax while creating a package in Oracle code, SSMA generates following error: "Error O2SS0418 Failed to parse statement batch with package definition"

Possible Remedies

Correct the source code and delete the invalid syntax in Oracle.

Verify if any application code calls the package containing the invalid syntax. If not, consider removing the package from your migration project.

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/15/migrating-oracle-to-sql-server-using-ssma-error-o2ss0418-failed-to-parse-statement-batch-with-package-definition/

O2SS0457: User defined type column not converted

Oracle database can contain tables with nested tables with more than one column. Nested table is not supported in SQL Server. To convert the nested tables in Oracle to tables in SQL Server, the Migration Assistant is attempted to store the sub-tables in the form of nvarchar data type, which is an expected result.

Possible remedies

If you want to convert them to standard tables in SQL Server database, you may need to manually create different tables for representing different entities and link them with primary and foreign key relationships.

O2SS0474: User defined type variable not converted

Oracle’s table functions allow you to define a set of PL/SQL statements that will, when queried, behave just as a regular query to the table. You may use the table function to manipulate the individual elements of a collection (user-defined object types) in your SQL code.

The problem is that SQL Server Migration Assistant (SSMA) for Oracle cannot convert table function and generates a set of errors:

> O2SS0474: Conversion of user defined type variable is not supported and is converted to type of VARCHAR(8000).
> O2SS0339: Cannot convert usage of standalone user-defined type.
> O2SS0482: Conversion of following TABLE expression is not supported: TABLE().

Possible remedies

> Declare a table variable with the same structure as the Oracle’s object type.
> Verify the SQL statement that should use the table variable

More information can be found on https://www.dbbest.com/blog/oracle-sql-server-migration-ssma-convert-unsupported-table-expressions/

O2SS0482: Conversion of following TABLE expression is not supported: TABLE()

Same as O2SS0474: User defined type variable not converted

O2SS0490: Conversion of identifier for CURRVAL is not supported

SQL Server does not support obtaining current value of a sequence. You can lookup current_value attribute from sys.sequences. However, note that the current_value represent global value of sequence (across all sessions). Oracle's CURRVAL returns the current value for each session. Consider the following example:

Session 1	Session 2
SELECT MySequence.nextval FROM dual; NEXTVAL ------------ 1	
SELECT MySequence.currval FROM dual; CURRVAL ------------ 1	
	SELECT MySequence.nextval FROM dual; NEXTVAL ------------ 2
	SELECT MySequence.currval FROM dual; CURRVAL ------------ 2
SELECT MySequence.currval FROM dual; CURRVAL ------------ 1	

The current_value from sys.sequence record the latest value across all session. Hence, in the example above, the current_value is the same for any sessions (current_value = 2).

When converting PL/SQL statement containing CURRVAL to SQL Server, SSMA generates the following error: O2SS0490: Conversion of identifier <sequence name> for CURRVAL is not supported.

Possible Remedies

You should consider rewriting your statement by storing next sequence value into a variable then read the variable value as current value.

More information on https://blogs.msdn.microsoft.com/ssma/2011/07/12/converting-oracle-sequence-to-sql-server-denali/

O2SS0502: Materialized view with other view

Oracle supports database object called Materialized View, which unlike a normal view, stores the result set of the query defining the view. The tables/views referred in the query are collectively called master tables. A materialized view can be described as a replica of the master table(s) from a single point in time. The data in the materialized view can be updated from the master tables through a process called Refresh. Oracle provides different refresh modes and interval options like automatic periodic refresh and on demand refresh for this purpose.

A materialized view can be read-only, updateable and writable. Based on the environment they are used in, Oracle offers different types of materialized views including Primary Key materialized view (including sub-query materialized view), Object materialized view, ROWID materialized view, Complex Materialized view (like materialized join view and materialized aggregate view) etc.

SQL Server has Indexed view that can provide similar functionalities as a Materialized view. SQL Server supports creating an index on a view. Creating an index on a view, results in storage of logical data from the view into physical index files thereby materializing the query results.

Possible Remedies

Materialized View should be converted to Index View during migration to SQL Server. In addition to providing dynamic data refresh advantage, creating an indexed view can also improve query against the underlying table(s).

More information on https://blogs.msdn.microsoft.com/ssma/2011/06/20/migrating-oracle-materialized-view-to-sql-server/

O2SS0522: Materialized view with float

SQL Server has several different ways of implementing the same behaviour as an Oracle materialized view. SSMA cannot understand the original optimization reason for why the application needed the materialized view, so SSMA flags certain conditions as an error.

Currently SQL Server Migration Assistant (SSMA) for Oracle converts materialized views to SQL Server indexed views. While converting a materialized view, SSMA creates necessary unique clustered index on the view in SQL Server. Also, SSMA adds WITH SCHEMABINDING option to the CREATE VIEW statement. So, if the source Oracle's materialized view contains elements that are prohibited in the SELECT statement of the indexed view, the conversion will fail with an error.

Possible Remedies

Materialized View should be converted to Index View during migration to SQL Server.

More information on https://www.dbbest.com/blog/ssma-convert-materialized-view-float-type/

Feedback and suggestions

If you have feedback or suggestions for improving this data migration asset, please contact the Data Migration Jumpstart Team (askdmjfordmtools@microsoft.com). Thanks for your support!

Note: For additional information about migrating various source databases to Azure, see the Azure Database Migration Guide.