ORACLE DATABASE 12c

An Oracle White Paper
June 2013

Migrating Applications and Databases with Oracle Database 12*c*

ORACLE

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Introduction

Migrating applications and data from one database to another is often a high-risk, expensive, and time-consuming process. However, Oracle provides products that reduce the time, risk, and financial barriers involved in migrating non-Oracle databases to the Oracle platform.

Oracle Database 12*c* introduces significant new features designed to lower the cost and time required to migrate non-Oracle database to the Oracle platform. These features include: enhanced SQL Developer, enhanced SQL Developer Migration Workbench, auto-increment IDENTITY columns, Implicit Result Sets, 32K VARCHARs, SQLTranslation Framework, Driver for MySQL Applications, and FETCH FIRST ROWS. This white paper outlines the new database features which assist in migrations.

.

Oracle SQL Developer

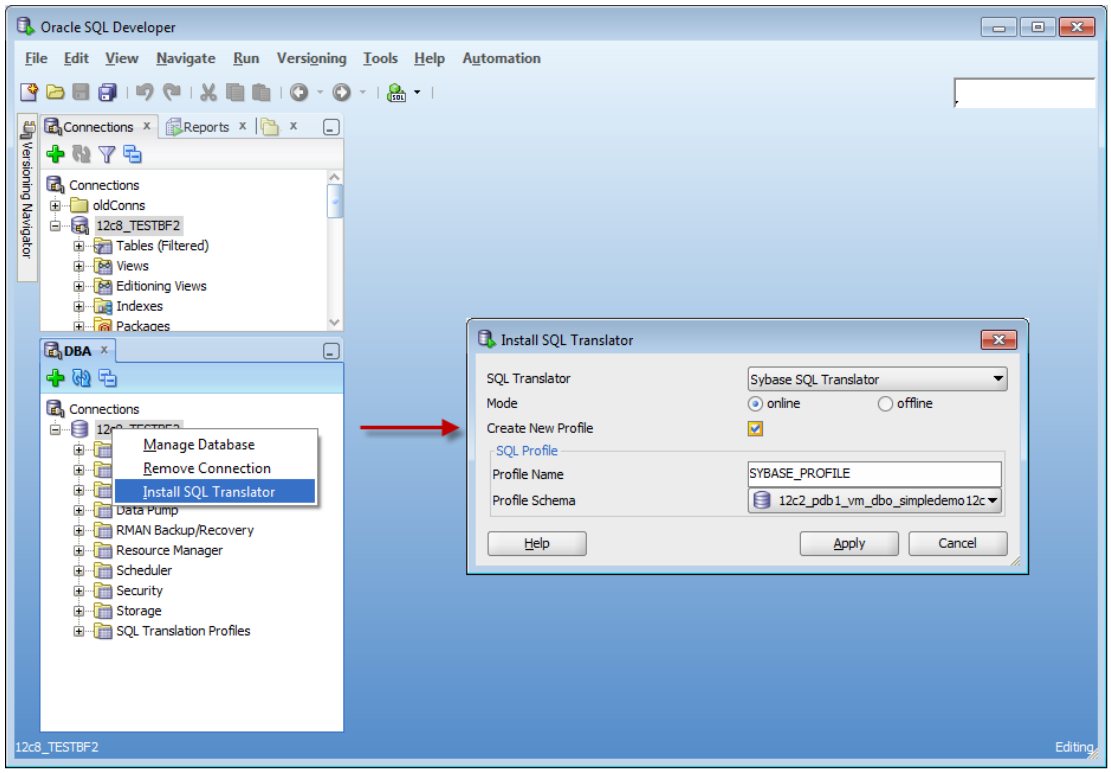
Introduction

Included with Oracle Database 12*c* is Oracle SQL Developer. SQL Developer is a graphical tool that enhances productivity and simplifies database development tasks. Using SQL Developer, users can browse, create and modify database objects, run SQL statements, edit and debug PL/SQL and have access to an extensive list of predefined reports or create their own.

Oracle SQL Developer is also the primary third-party database migration platform for Oracle database. Oracle SQL Developer provides an integrated migration tool for migrating Microsoft SQL Server, Sybase, MySQL, Microsoft Access, IBM DB2 LUW and Teradata to Oracle Database.

With SQL Developer, users can create connections to non-Oracle databases for object and data browsing. Once a connection is created, the tool provides utilities to migrate any of these databases to Oracle. Depending on the database in question, SQL Developer automatically converts the tables, triggers, stored procedures and all other relevant objects to an Oracle database. Once the target Oracle database has been generated, SQL Developer assists in the migration of the data from the non-Oracle database to the target Oracle database.

When the target database for an Oracle database migration is version 12*c*, Oracle SQL Developer will automatically migrate the objects and stored procedures using the new database 12*c* features discussed below. Included in the descriptions are some examples of how the code or objects are migrated to Oracle in 11g and older databases versus going forward in 12*c* and newer databases.



Oracle Database 12*c* Application Migration Enhancements

Introduction

The following new features directly address the challenges our customers face when planning migrations to Oracle database. This white paper will provide a brief description and example of each of the following features:

- Identity Columns
- 32k Varchar2s
- FETCH FIRST rows (SQL)
- Implicit Cursors
- Oracle Multitenant
- SQL Translation Framework
- Driver for MySQL Applications

Each of these features will dramatically decrease the amount of work required to migrate your applications to Oracle Database, thus saving you time and money..

Identify Columns

Primary key constraints define a table column or columns that will serve to uniquely identify records in that table. A common programming technique is to have a value automatically generated and assigned as rows in a table are generated and inserted. In prior versions of Oracle Database, this was frequently accomplished by creating a SEQUENCE and a TRIGGER. The sequence would define the values to be generated, and the trigger would fire on an insert and would feed the value of the sequence to the table.

In other third party relational database management systems this can also be accomplished by using an Identity column. This allows the sequence logic to be directly embedded into the definition of the table, bypassing the need to create a sequence and a trigger to handle the generation and population of the primary key value of the table records.

This represents significant cost savings for customers migrating to Oracle Database. Instead of having to generate two additional database objects for each table making use of an identify column, this can now be defined in the table itself. This also lowers the cost of maintenance going forward as there are fewer database objects to manage and support.

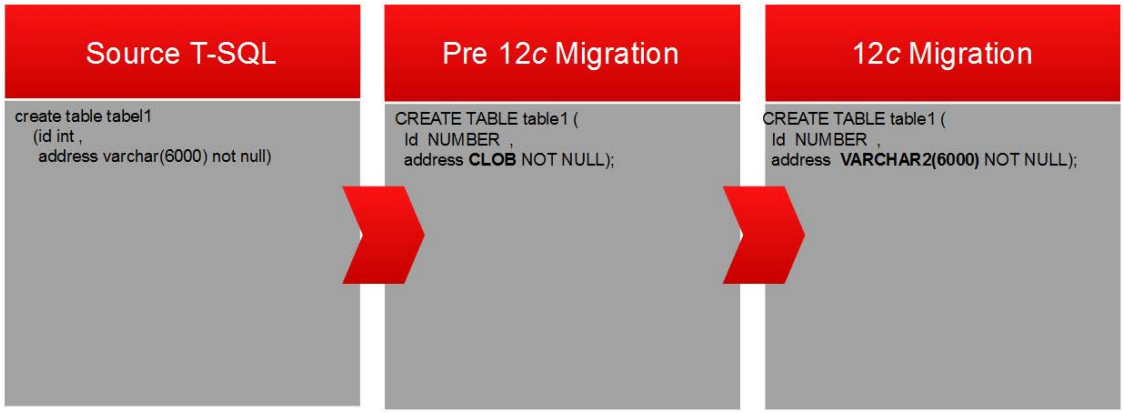
Fewer objects, less code, less work – all handled automatically when migrating to Oracle Database 12*c* using Oracle SQL Developer.

32K VARCHAR2's

Since its introduction, the VARCHAR2 datatypes ((including VARCHAR2, NVARCHAR2, and RAW) have had a max size of 4,000 bytes, which equates to 4,000 characters in single byte charactersets.

Table column definitions exceeding this size would be migrated as CLOBs or BLOBs. This presented a challenge for many customers migrating from non-Oracle database environments as other third party databases supported much larger strings in their native datatypes. With the introduction of Oracle Database 12*c*, VARCHAR2, NVARCHAR2, and RAW now supports up to 32,768 bytes.

CLOBs can present optimization and flexibility challenges for developers. Offering an extended size VARCHAR2 means that in most cases migrations can continue with no requirement to switch to CLOBs in the column definition for tables containing large strings. In addition, indexes can be built on top of such columns, unlike CLOBS or BLOBS.



The default Oracle Database 12*c* parameters must be updated to allow the new 32k VARCHAR2 size.

To enable the increased size limits for these datatypes, the following database parameters are required:

MAX_SQL_STRING_SIZE controls the maximum size of the extended data types in SQL, where:

LEGACY means the length limit of the data types used prior to Oracle 12*c*.

EXTENDED means the 32767 bytes limit in Oracle Database 12*c*.

You must set the COMPATIBLE initialization parameter to 12.0.0.0 or higher to set MAX_SQL_STRING_SIZE = EXTENDED.

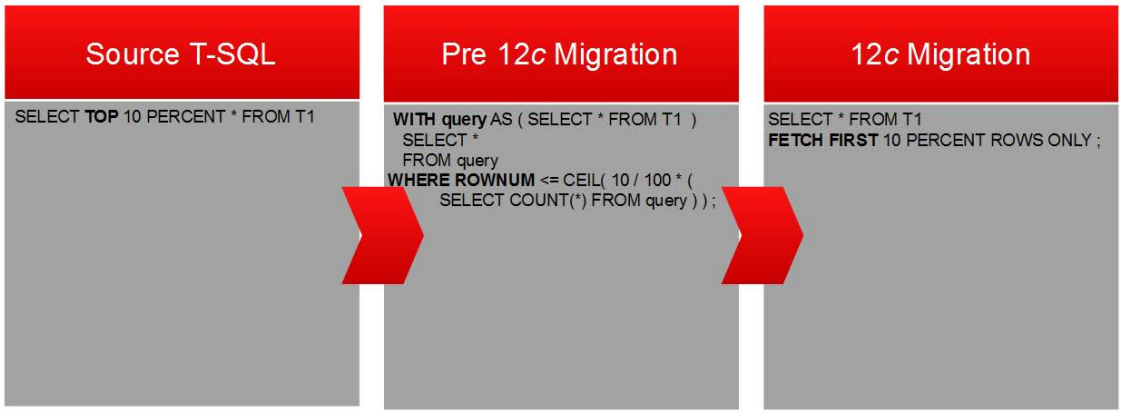
FETCH FIRST ROWS

Queries that order data and then limit row output are widely used and are often referred to as `Top-N` queries. Prior to Oracle Database 12*c*, developers would attempt to implement this ANSI SQL feature using the pseudo-column 'ROWNUM' to limit the number of rows returned in a query.

In Oracle Database 12*c* Release 1, SQL SELECT syntax has been enhanced to allow a row_limiting_clause, which limits the number of rows that are returned in the result set. The row_limiting_clause provides both easy-to-understand syntax and expressive power. Limiting the number of rows returned can be valuable for reporting, analysis, data browsing, and other tasks.

You can specify the number of rows or percentage of rows to return with the FETCH_FIRST keywords. You can use the OFFSET keyword to specify that the returned rows begin with a row after the first row of the full result set. The WITH TIES keywords includes rows with the same ordering keys as the last row of the row-limited result set (you must specify ORDER BY in the query).

The row_limiting_clause follows the ANSI SQL international standard for enhanced compatibility and easier migration.

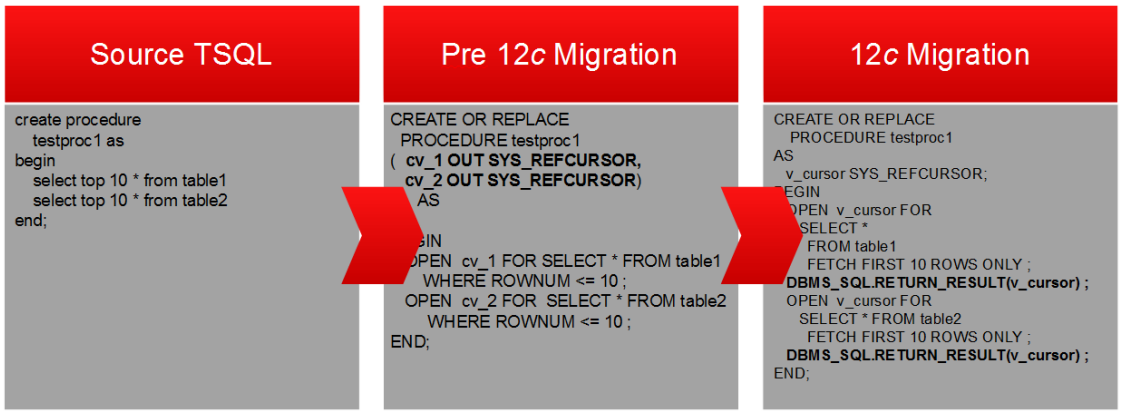


The new FETCH FIRST SQL is powerful and easy to read.

Implicit Cursors

A common programming practice in Microsoft SQL Server and SAP's Sybase ASE database's extended SQL language, T-SQL, is to write SQL statements directly in their stored procedures. Calling these stored procedures would make the resultset for the one or more queries immediately available to the calling user or program.

Prior to Oracle Database 12*c*, migrating these stored procedures to Oracle Database PL/SQL equivalent procedures would require changing the procedure header to include one or more SYS_REFCURSORs as OUT or RETURN parameters, then subsequently retrieve the result sets via the Ref Cursors.



With Oracle Database 12*c*, stored procedures can use the DBMS_SQL package's RETURN_RESULT() function to make the query results available to the calling user or program.

JDBC Example

New Oracle JDBC methods

getMoreResults() or getMoreResults(int): checks if there are more results available in the result set.

The int parameter that can have one of the following values:

KEEP_CURRENT_RESULT, CLOSE_ALL_RESULTS, CLOSE_CURRENT_RESULT

getResultSet(): iteratively retrieves each implicit result

The following Java code migrated from a third-party database will work with Oracle Database without any changes

```
CallableStatement cstmt = null;
ResultSet rs = null;
cstmt = conn.prepareCall("{call testproc1()}");
cstmt.execute();
boolean resultsAvailable = cstmt.getMoreResults();
```

Oracle Multitenant

With Oracle database 12c, multitenancy for applications can now be achieved at the database level with Oracle Multitenant. A single Oracle Database 12*c* Container Database (CDB) can service one or more Pluggable Databases (PDBs). Instead of migrating multiple third-party databases to a single Oracle Database using schemas as a 'container' for each migrated database, Oracle Database 12*c* now allows for individual PDBs to be used.

SQL Translation Framework

Introduction

While migrating database objects and data is a major undertaking, migrating database applications is no less critical or time consuming. Each of the relational database management systems has its own implementation of the SQL standard. SQL that may run in Sybase ASE may not run in Oracle database. The amount of custom SQL present in an application can largely define the amount of time required to fully migrate a database and its applications.

Database application migrations are assisted by Oracle SQL Developer and its application scanner scripts. SQL Developer can parse and document SQL statements that require translation before running against an Oracle Database. The task of doing the actual translations is left to the end user, or can be attempted one at a time using SQL Developer's SQL Translation Scratch Editor.

The SQL Translation Scratch editor is an ad hoc translation engine that allows users to connect to 3rd party databases, run their SQL statements, translate them to Oracle, run the statements again in Oracle Database, and compare the results. Developers can then manually update their applications to run the modified code. This works fine for static SQL, but there is no solution for dynamically generated SQL statements.

This time consuming and error-prone process has been greatly enhanced with the introduction of the SQL Translation Framework in Oracle 12*c*.. The framework allows for the translators in Oracle SQL Developer to be loaded directly into the database as a collection of Java stored classes and procedures. Available with Oracle Database 12*c* are translators for Sybase ASE and SQL Server.

Once installed from SQL Developer to the database, the translator can be activated at the session or service level. Statements sent to the database will be parsed as non-Oracle SQL, translated, and executed. A collection of these translations are stored in a SQL Translation Profile. The contents of the profile can be reviewed, modified, and approved by the migration team to ensure the translations are accurate. Profiles can be created for each application to be migrated and can then be transferred between databases so that the translations are transportable.

SQL Translation Framework Workflow

1. Framework receives SQL call
2. Performs lookup in SQL translation dictionary (Profile)
3. When not found it fingerprints the statement and adds it to the dictionary
4. It then processes the template with the values supplied

An Example

- Framework receives

  ```
  SELECT TOP 2 * FROM T1
  ```

- Performs static lookup of a conversion in the SQL Translation Dictionary

- Not Available: Generate the Fingerprint
- Select Top <ora:literal type=integer order=1> * From T1

  Note: literals are mapped in translations such that 'select 1; select 2 select 3;' are treated as a single statement, where the literal (1, 2, or 3) is stored as <ora:literal type=integer order=1>) in the Fingerprint.

- Lookup fingerprint in the SQL Translation Dictionary

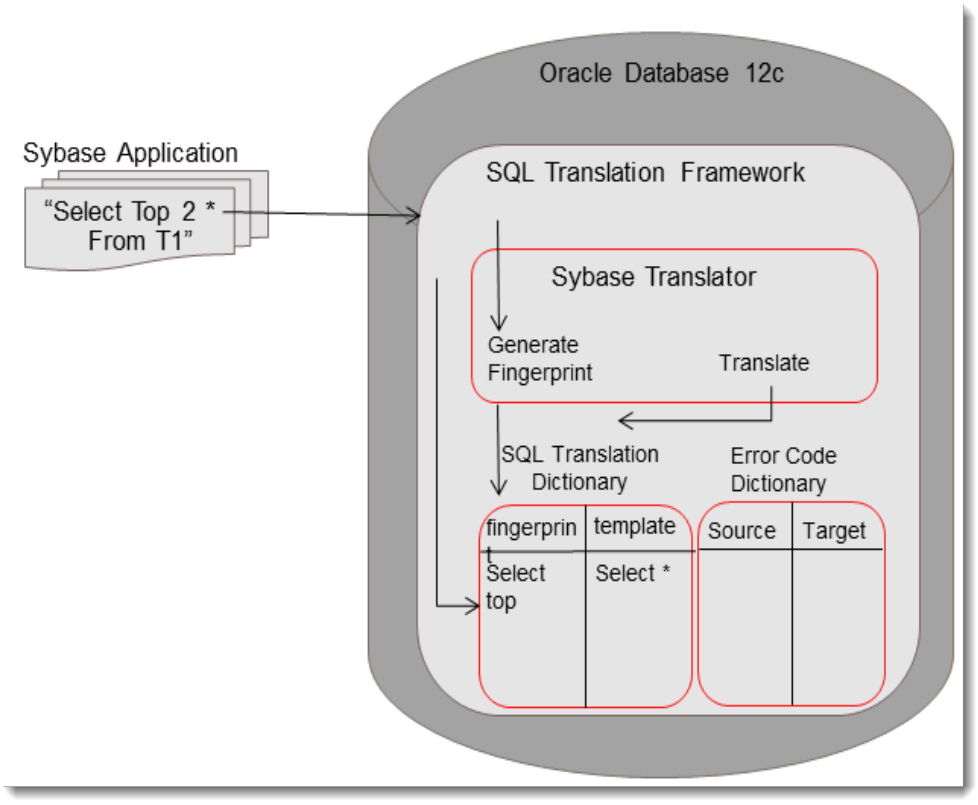
  Available: Gets the Fingerprint

  Select * From T1 FETCH FIRST <ora:literal type=integer order=1> ROWS ONLY

- Processes the Template with values acquired

  Select * From T1 FETCH FIRST 2 ROWS ONLY

- Returns the translated SQL to the Framework
- SQL Translation Framework handles binds if necessary



SQL Translation Framework Diagram: A Sybase application connects to Oracle and runs, having its statements translated and executed on-the-fly for Oracle.

Conclusion

Oracle Database 12*c* helps customers lower IT costs and delivers a higher quality service by enabling consolidation onto database clouds and engineered systems like Oracle Exadata and Oracle Database Appliance. It's proven to be fast, reliable, secure and easy to manage for all types of database workloads including enterprise applications, data warehouses and bid data analysis.

Moving your database and database powered applications to Oracle Database often requires significant application and data model updates as non-Oracle technologies must be implemented to work with existing Oracle structures, data types, proprietary SQL and procedural languages (PLSQL.) Oracle Database 12*c* includes many new features which minimize database and application changes to accommodate applications not originally developed for Oracle Database.

Application Development with
Oracle Database 12*c*
June 2013
Author: Jeff Smith
Contributing Authors: Ashley Chen, David Gambino, Kuassi Mensah

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com

Oracle is committed to developing practices and products that help protect the environment

Hardware and Software, Engineered to Work Together