Migrating to Oracle Database

Your Guide to a Successful Migration, On-Premises or in the Cloud

Introduction

This paper provides an in-depth look at the process of migrating third-party databases to Oracle Database 12*c* using Oracle SQL Developer tools. It will help you determine what's involved with converting a third-party database to an Oracle Database instance, and also explain how to make your existing applications work with the finished product – the migrated Oracle instance. You will learn how to create a migration repository, capture the source database, convert it to the Oracle data type, and copy the data. The examples apply to on-premises and cloud scenarios, with attention to two cloud environments: Oracle Database Cloud Service (DBaaS) and Oracle Database Exadata Cloud Services.

Generally speaking, the process of migrating data from a third-party RDBMS to Oracle Database is straightforward. Changing the architecture of your application so it can communicate with Oracle Database can be as simple as dropping in a new JDBC or ODBC driver. Moving stored procedures to Oracle's PL/SQL ecosystem is slightly more complicated. Every migration is different, and some applications require changes at the lowest levels of your technology stack. Asking the right questions up front is critical so you can identify potential roadblocks, and plan accordingly.

This paper assumes you have access to Oracle Database 12.1.0.2 or higher. If you have questions regarding Oracle Database licensing and pricing, please contact your Oracle account manager.

Benefits of Migrating to Oracle Database Cloud

Oracle Database 12*c* introduces several significant capabilities that significantly lower the cost and time required to migrate non-Oracle databases to the Oracle platform. Oracle Database Cloud Service lets you choose pay-per-use metered services, or a regular monthly subscription of Oracle 11g, 12.1 and 12.2 database services. You can automatically provision high availability configurations to elevate service levels for maintenance or unexpected failure scenarios. Provisioning choices include pre-packaged bundles of Oracle Database options for DevOps, Performance, Security, Analytics and Monitoring. The service comes with an integrated Oracle Application Express environment, allowing you move development activities to the cloud. Oracle Database 12*c* Multitenant makes it easy to consolidate many databases quickly and manage them as a cloud service.

Identifying the Scope of Your Migration

Discussions regarding the migration to Oracle Database usually start with a series of questions:

- What database are you migrating from? (For example, SQL Server or DB2)

    - What version of this database?

        - If it's the *latest* version, e.g. SQL Server 2016, are you using any version-specific features?

- How many databases are you migrating?

    - If more than one, are they essentially copies of each other, or is each one a different migration?

- How large are your databases (amount of disk space for the data itself in GB/TB)?

- Do your database applications rely heavily on stored procedures?

    - How many stored procedures and triggers need to be migrated?

    - Do they predominately fall under the '*select * from table return result set to application*' category?

- What is your migration project timeline, and when do you need to "go live?"

- Will you be running the Oracle and third-party database systems concurrently, with plans to fail over to the Oracle implementation?

- Will your new database instance reside on-premises or in the cloud?

- What is the downtime/maintenance window, if any, allowed for this migration?

- What is the nature of your homegrown applications (.NET, Java, etc.)?

    o Applications provided by an ISV generally have their own migration path

If downtime isn't an option or you will be required to replicate database changes to the original system for some time after your migration, consider using Oracle GoldenGate 12*c* to shorten the amount of time required to migrate the data.

Database Migration Tools

Oracle SQL Developer can help you identify the physical characteristics of the database(s) you are migrating. It is included with every copy of Oracle Database and also available for download on the Oracle Technology Network. This utility serves as a graphical user interface for Oracle Database and an integrated development environment for PL/SQL.
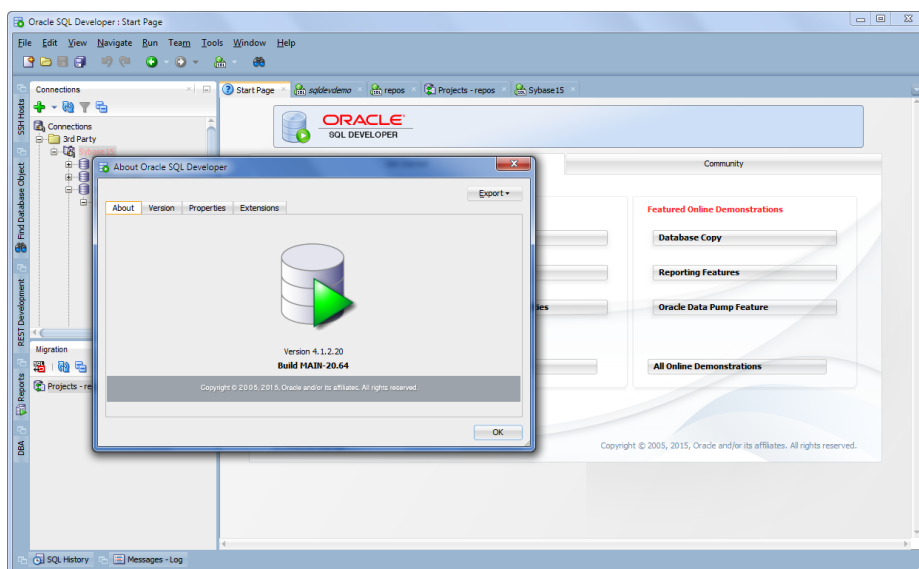
Figure 1: SQL Developer version 4.1.2

SQL Developer is also the official third-party database migration solution for Oracle Database. It currently supports SQL Server, Sybase ASE, DB2, and Teradata. It also assists with migrations from MySQL. For exact versions of supported databases, please consult the latest information on OTN.

SQL Developer will capture the third-party database definitions, analyze and convert those definitions to their Oracle equivalents, and provide the tools you need to move and verify the data. Here is a link to the supported JDBC drivers.

SQL Developer provides a 9-step wizard that you can use to create a migration project, including setting up and running the capture and conversion processes of the source third-party database.

Environment Setup

You may be migrating your data to Oracle Cloud or installing an in-house Oracle Database. Whether you are migrating 1,000 databases or just one, it's best to start small to get a feel for the migration process and the tools. You will need the following:

- Oracle Database 12*c*, set up with a repository to store the project metadata (as described below)
- The latest version of SQL Developer
- The correct JDBC driver to connect to your third party database

Oracle provides a VirtualBox appliance, which includes Oracle Database 12*c*, the latest version of SQL Developer, and hands-on labs that present examples of how to migrate a database. The virtual machine is not certified or available for commercial use, but it is helpful for learning the ins and outs of performing an Oracle Database migration.

Add Third-Party JDBC Drivers and Connect to your Source Database

From the supported JDBC driver list, download the appropriate driver for your database type. In SQL Developer, go to Preferences. On the Database page, select the Third Party JDBC Drivers page. Click 'Add Entry…' and navigate to the appropriate jar file, as shown below.
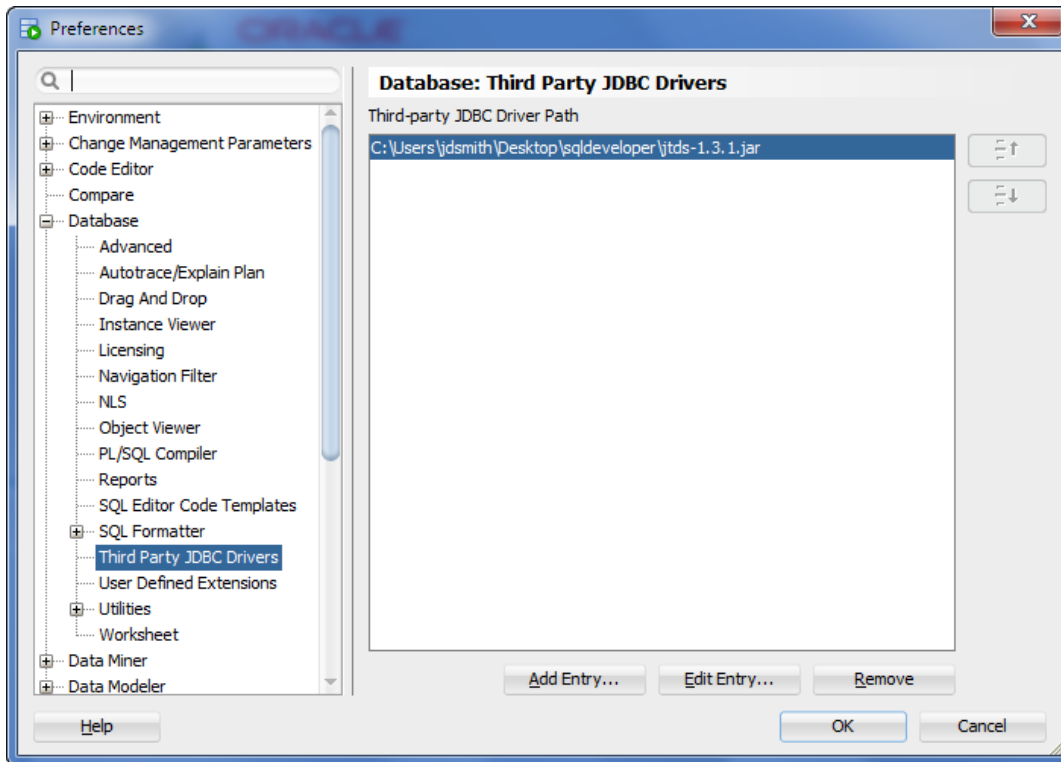
Figure 2: jtds v1.3 is recommended for Sybase ASE and Microsoft SQL Server migrations.

Once the driver has been successfully added, you can define a connection to databases that are compatible with that driver.

Create a connection to your source database. Verify that you can browse the objects and data to be migrated. In your target Oracle Database, create a new user/schema for the migration repository. Define a SQL Developer connection to this schema so you can use it to create the repository.

Launching the Wizard in SQL Developer

In SQL Developer, go to the Tools menu and select 'Migration -> Migrate…'

This will launch a wizard that guides you through the process of creating a migration repository, capturing the objects, and performing the move to Oracle Database.

You can create the repository in advance or you can do so via the wizard.
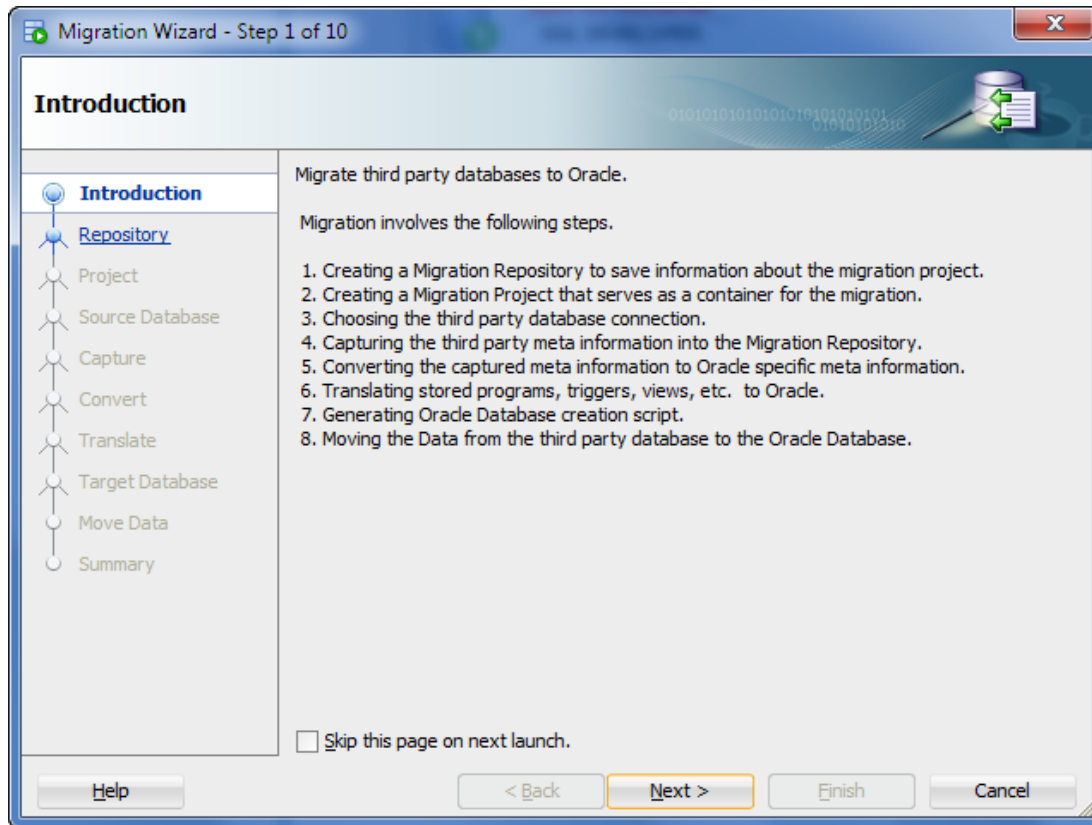

Figure 3: The Migration Wizard

The next page will ask you to pick your migration repository connection. Select the connection previously created, or create one now using the 'Create new connection' button.

The migration repository can be created in any Oracle instance. It does not have to be the same instance that you plan to migrate to.

Next you will be prompted to name your project and select a directory for migration project output. Select the third-party database to be migrated. You have two options to access the data model – a 'live' connection and an offline model that is captured via shell scripts. The default selection is 'Online.'

Offline is for cases in which the engineer performing the migration does not have access to the source third-party database. In this scenario, use the 'Tools > Migration > Create Database Capture Scripts…' interface.

This command will generate either a Windows Batch file or a Linux Shell script for an administrator to run. It will capture the data model of the source database and write it to files that can then be referenced in the 'Source Database' page of the Migration Wizard.

4
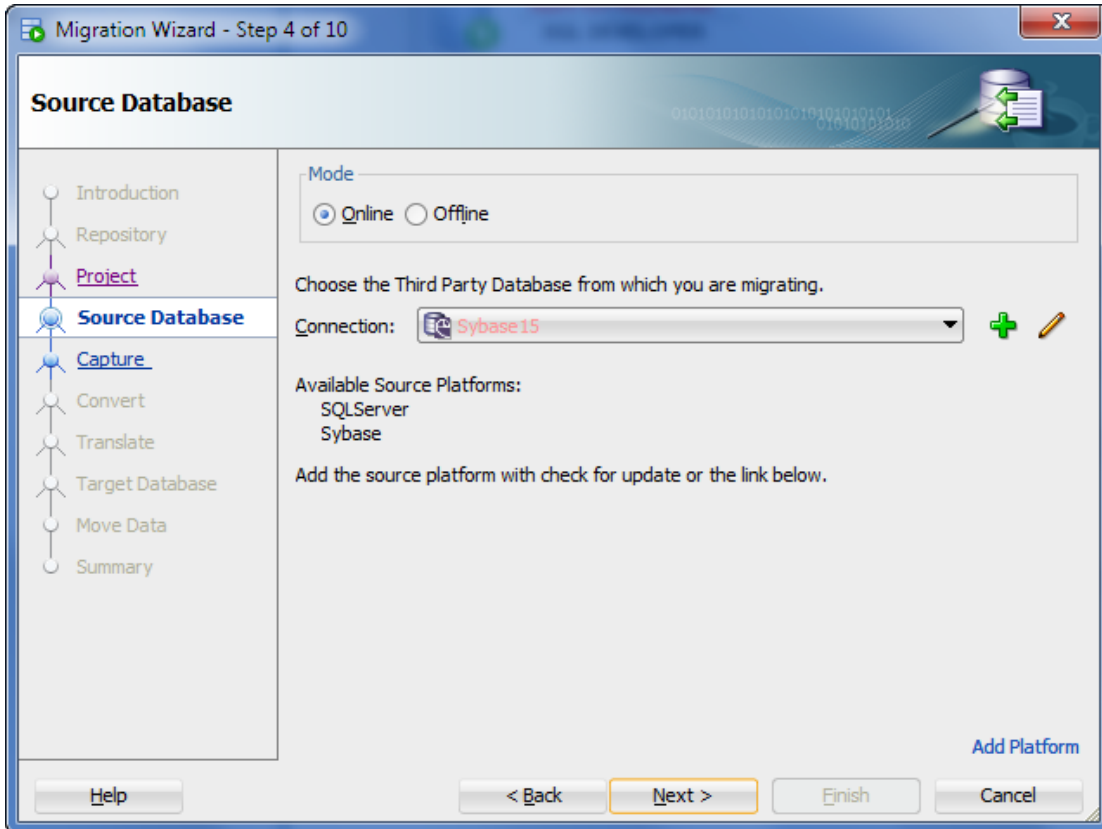
Figure 4: Use Offline mode when you do not have access to the source third-party database.

Once a connection has been established, select one or more databases to migrate. Determine the appropriate Oracle data types for the conversion. For example, the Sybase ASE data type 'DateTime' will be stored as a 'Date' in Oracle. Similarly, 'Time' will be stored as 'Timestamp.' You can override the defaults if you prefer to define your own data definition rules.

Objects will be renamed in the migration process to avoid reserved word collisions and object name length limitations, such as the 30-character limitation in Oracle Database 12*c* Release 1. Apart from the data objects, SQL Developer will also translate constraints, views, functions, procedures, and triggers. The list of supported objects to be translated for each third-party databases can be found here.

By default, SQL Developer will attempt to translate all available 'SQL' objects – that is, objects with embedded native SQL or T-SQL.

Specifying the Target Database

The Target Database Step allows you to specify how you would like to generate the migrated Oracle users and objects. There are two Modes, online and offline.

5
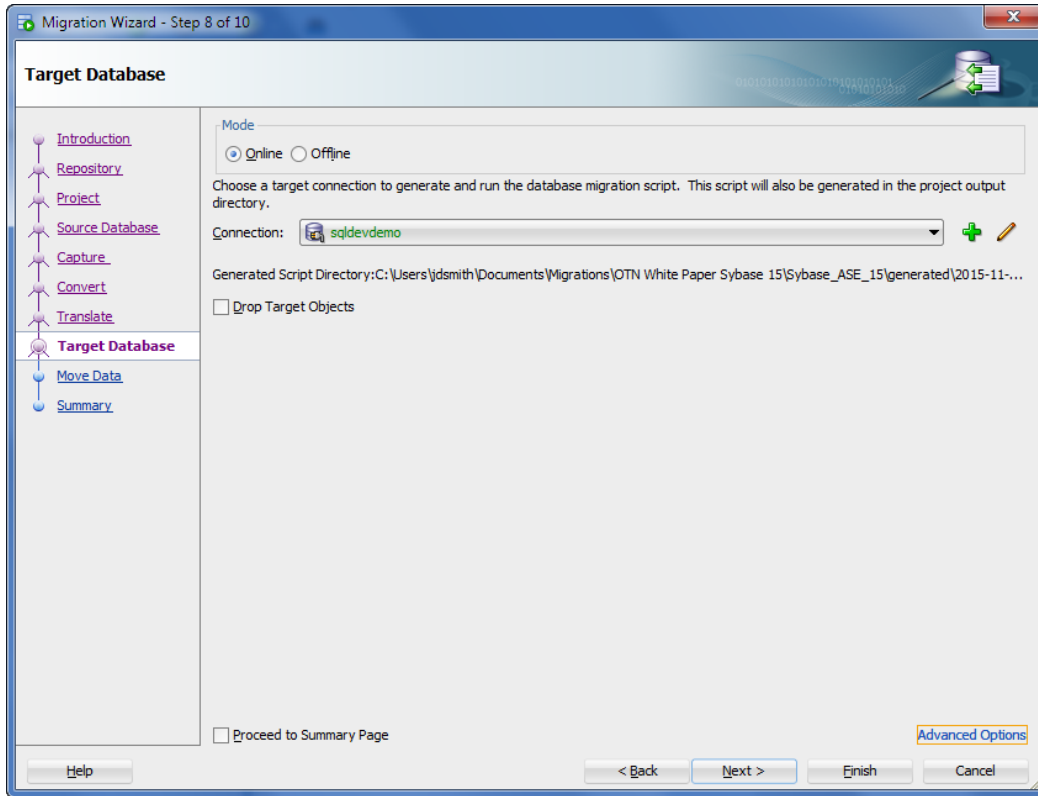
Figure 5: You can send the migrated objects to any Oracle instance. Do not use the repository connection.

Selecting Target Locations in Oracle Cloud

Oracle SQL Developer facilitates connections to Oracle Database Cloud Service and Oracle Database Exadata Cloud Service. However, both services restrict traffic from the open Internet to port 22 via SSH. Before connecting to your target Oracle Database running in these Cloud environments, use the SQL Developer SSH Tunneling feature to connect to your database.
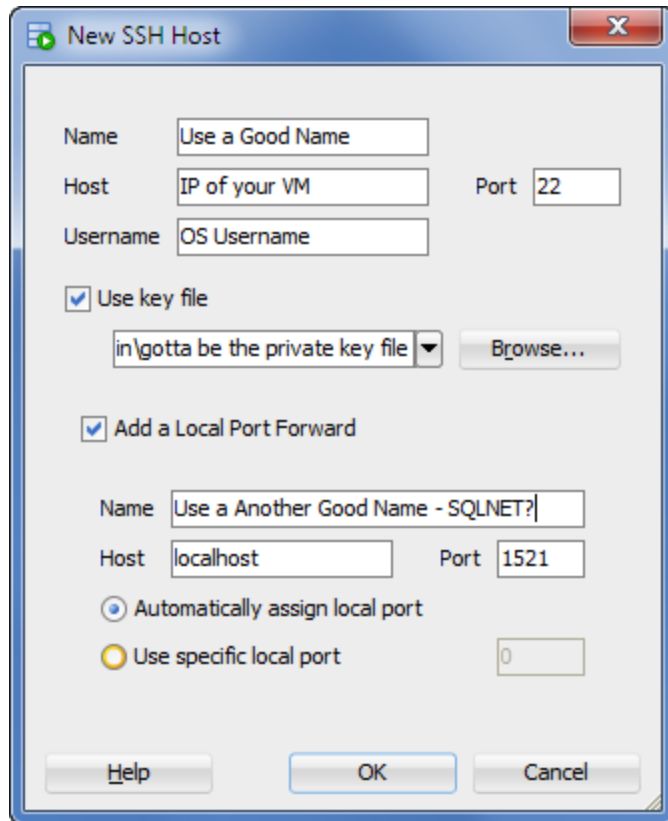
Figure 6: When you create a tunnel, you can also set up a port forward.

The host is the network location of your DBaaS instance. The username is the OS user on the DBaaS server you are connecting to via SSH. The *port forward* screen allows you to accept traffic from port 22 to the DBaaS Listener port of 1521. Once this path is defined, your DBaaS connection will connect via this tunnel, and it can be used for your migration target.
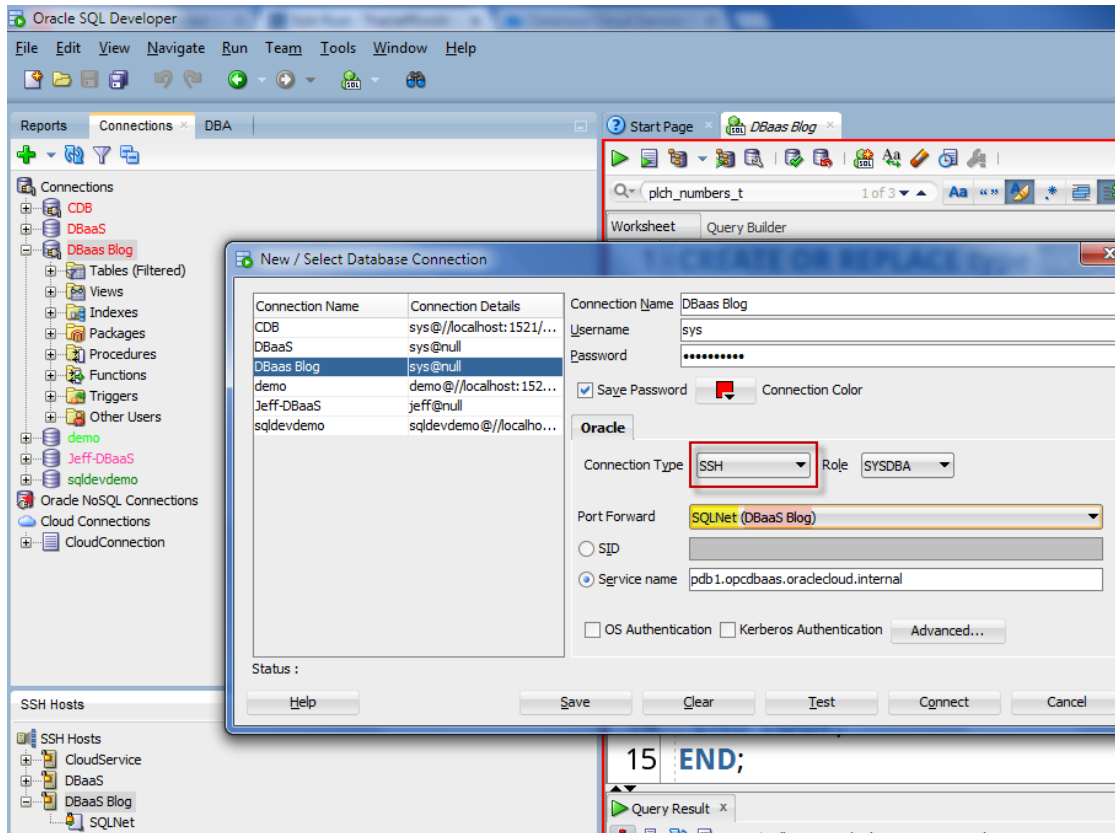
Figure 7: A DBaaS connection via an SSH tunnel in Oracle SQL Developer.

Once connected, you can migrate to a cloud-based or on-premises instance of Oracle Database by following the process outlined below. In both cases, SQL Developer will create a "generation script" that defines a new Oracle user, connects to that user, and creates the necessary tables, indexes, views, procedures, and so forth. No data movement will occur during this step.

In Online Mode, use the wizard to choose the target Oracle Database, and then automatically run the generation script. The connection you select must have the necessary privileges to create a new user and migrate all the associated objects.

In Offline Mode, you don't have to choose the Target Oracle Database. Instead you can open the generation script in a SQL Developer SQL Worksheet, so it can be inspected first. You can then choose the target Oracle database in the SQL Worksheet and execute the script against that worksheet.

Loading Data into Oracle Tables

Always use the 'offline' mode to migrate your data. This mode will create SQL*Loader scripts to load the data into the source Oracle tables. The 'online' mode is good for test runs, especially when the amount of data is small. Moving a large instance in online mode can take several days. Oracle does not advise executing this function from within the SQL Developer interface.
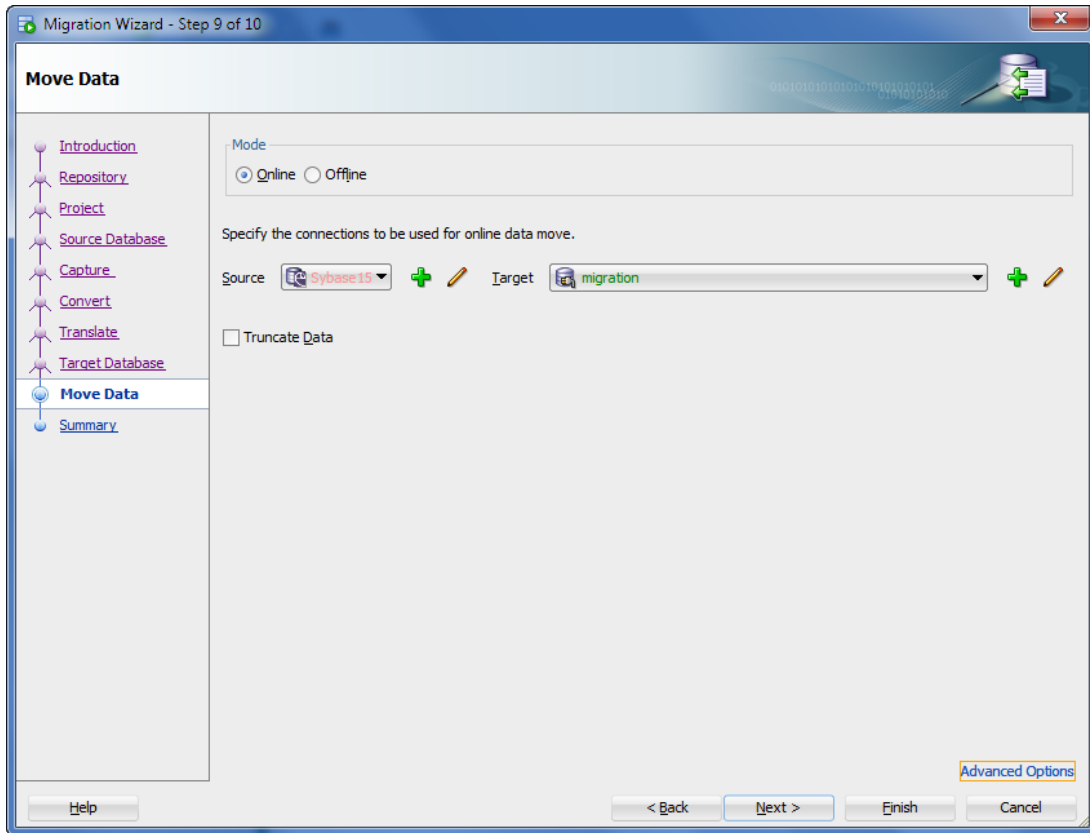
Figure 8: Use 'online' mode for test runs and small databases, 'offline' mode for everything else.

Inspecting the Migration Results

After the migration has finished, you can inspect the details of how objects have been captured and converted by opening the migration project. Several reports are available that summarize issues, sorted by object. Most of the reports have a top-to-bottom structure. Selecting an item at the top of the page shows information of that type on the bottom of the page.

Each object is 'hot-linked' so you can immediately drill in to the object, inspect problems, and fix them.

Figure 9: This report shows a group of objects with statuses. Hot-links let you drill into individual objects.

Oracle Database migrations are highly automated. Most customers experience conversion rates as high as 80 percent. However, each migration is different, and your results may vary. For example, stored procedures may compile with errors that must be addressed. In general, Oracle's automated procedures dramatically reduce the overall migration effort.

Dealing with Conversion Issues

The translators that SQL Developer employs to convert non-Oracle stored procedures to Oracle PL/SQL objects do not translate 100 percent of these procedures. Unique coding practices in each RDBMS prevent some of these database artifacts from being automatically translated to Oracle. These coding gaps also vary from site to site, since each development team has its own approach to utilizing these RDBMS features.

Even when all the RDBMS code is translated, make sure you thoroughly test the resulting Oracle Database for accuracy and performance. Some of the programming structures and techniques employed by third-party databases do not have direct equivalents in Oracle. Conversion issues are flagged during the migration process. In most cases, comments are inserted to advise whether additional coding is required before the migration can be considered complete.

Opening the invalid objects in SQL Developer's PL/SQL editor makes it much easy to analyze the code and correct the issues since you can take advantage of built-in error message look-ups, a profiler, and a debugger.
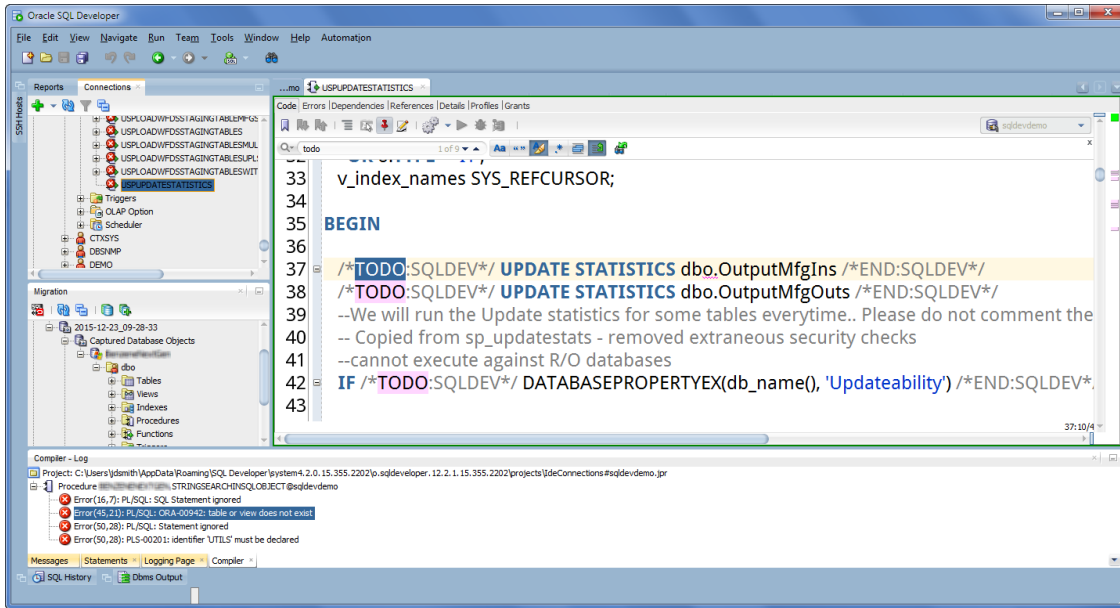
Figure 10: Procedure Editor showing compiler errors and code to be translated for the migrated T-SQL Function.

For example, the above figure reveals stored procedures that require attention at lines 37 and 38. In this case, fixing the problem could involve creating a stored procedure to collect statistics on tables, instead of inserting the DBMS_STATS package call directly in the stored procedure. Oracle allows you to generate statistics for all or part of a table (for example, you can use 25 percent of the table data to generate the optimizer statistics). This decision requires in-depth knowledge of the application and the data model—in other words, human intervention.

How much intervention is required? In one recent migration project, 256 "To-Do" items were logged for 52 PL/SQL objects and 132,000 lines of code. While the number of items will vary depending on the nature of the code, you should budget sufficient time to test and fix code issues after each automated migration procedure. Perform the 'capture' process early in the project so you can adequately assess the scope of work ahead.

Moving the Data – Offline Versus Online Methods

Unless you have a small test database to migrate, use the OFFLINE method. To start the migration from your project, right-click on your project capture and select 'Move data…'

This command will advance the migration wizard to the data step, where you can choose online or offline. Choosing ONLINE will connect to both the source and destination databases, read the data from the source tables, and insert the data into the target tables. This method is only suggested for test environments with small amounts of data. Use the OFFLINE mode for large databases and production databases.

The OFFLINE method captures the data to local delimited text files, which feed the Oracle SQL*Loader utility. SQL*Loader is designed to load significant amounts of data very quickly. It automatically logs rejected records so that they can be fixed and moved in a subsequent run. SQL*Loader is included with every copy of Oracle Database on the target Oracle server.

To perform the capture, execute the scripts generated by SQL Developer. For SQL Server and Sybase, these scripts use the BCP utility, a Microsoft Command Line Utility that comes with each SQL Server download.

Figure 11: Running the MicrosoftSQLServer_data.bat file to export data to local flat files.

Once the script has finished you will see a series of scripts and SQL*Loader control and data files in your project directory's 'datamove' and 'data' subdirectories.

Figure 12: Directory listing showing the SQL*Loader data files.

Running SQL*Loader requires either the Oracle Client or an Oracle Database install. You can run the oracle_loader.bat file on your local machine if you have a Client, or you can move the files directly to your Oracle Database server.

If your target database is a DBaaS cloud, move the SQL*Loader files to your DBaaS machine and run the scripts directly from there.

Verifying Data Quality

Once the data migration has completed, you can perform a data quality assessment, in which raw record counts are compared with the original and converted tables, and deltas are highlighted. Please note that this can take a long time for very large datasets.

Figure 13: A Data Quality report, with original and converted tables shown side-by-side.

Advanced Data Migrations with Oracle GoldenGate 12*c*

Some migrations require advanced tools. Depending on your particular application needs and service level agreements, you might need to execute a faster data migration to minimize downtime. Similarly, if you need to run a third-party database application side-by-side with your new Oracle application to permit data synching, then you should consider using replication technology such as Oracle GoldenGate 12*c*, a comprehensive software package for real-time data integration and replication in heterogeneous environments.

Oracle GoldenGate 12*c* is ideal for scenarios requiring high availability, real-time data integration, transactional change data capture, data replication, transformations, and verification between operational and analytical enterprise systems. It boosts performance and simplifies configuration and management, with innate support for cloud environments, expanded heterogeneity, and enhanced security. Talk to your Oracle account manager to discuss licensing, since Oracle GoldenGate 12*c* is not included with your Oracle Database license.

Application Migrations

Connectivity between your applications and the database should be addressed at the outset of your migration project. If you are using a .NET or Java application, you will need to switch your ODBC or JDBC driver to an Oracle driver. After that you'll want to make sure your applications can connect and interact with your new Oracle Database via the Oracle SQL dialect. While Oracle SQL supports much of the ANSI SQL standard, Oracle's interpretation of this standard differs from SAP's interpretation for Sybase, Microsoft's interpretation for SQL Server, IBM's interpretation for DB2, and so forth.

SQL Developer includes application scanner scripts that allow you to audit the database code to determine what might need to be changed in your database applications. It also offers a Scratch Editor that can translate SQL and stored procedures to their Oracle equivalents, on an ad-hoc basis.

A much more powerful interface is the Oracle Database 12*c* SQL Translation Framework. For Sybase ASE and SQL Server T-SQL, this framework can receive queries in these SQL dialects, translate them to their Oracle equivalents, and execute them on the fly.

In addition, Oracle Database 12*c* includes many other capabilities that decrease the time and cost of migrating applications to Oracle Database. To learn more, please refer to this whitepaper: Migrating Applications and Databases with Oracle Database 12*c*.

Some Practical Advice

Start with a small database or test database. The wizard allows you to roll through the entire process in a single step, but it makes more sense to break it into these fundamental pieces:

- Perform the capture and translate

    - Verify that all objects are captured and translations are acceptable

- Deploy the new Oracle objects

    - Verify that the objects are compiled without errors, all objects exist, and that those objects are in the expected Oracle schema

- Move the data

    - Make sure the time required to copy the data to Oracle meets your project requirements. You can accelerate and fine-tune this process using techniques such as External Tables as well as with licensed products such as Oracle GoldenGate 12*c*

- Run a sample application or report to ensure the output is as expected

The wizard allows you to initiate, stop, and resume the process at any step.

Resources

Migrating to Oracle Database 12*c* is easier with Oracle SQL Developer's Migration feature. Successful migrations start by identifying the project requirements, analyzing the data and applications that need to be moved, and adopting a mindset of trial and error. To learn more about migrating to Oracle Database 12*c* and Oracle Database Cloud, please take a look at the SQL Developer Migrations page and review the information on these Migrations Forums.

ORACLE®

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

Integrated Cloud Applications & Platform Services

16