# Contents

**Preface** ........................................................................................................................................................................... ix

## Part I  Installing Oracle Warehouse Builder

### 1  Installation Overview and Requirements

- **What’s New in Installing Warehouse Builder** ................................................................. 1-1
- **Warehouse Builder Architecture and Components** ...................................................... 1-2
  - Design Center ................................................................................................................. 1-3
  - Control Center Manager ............................................................................................... 1-3
  - Target Schema .............................................................................................................. 1-3
  - Warehouse Builder Repository ..................................................................................... 1-4
  - Repository Browser ...................................................................................................... 1-4
  - Control Center Service ................................................................................................. 1-4
  - Implementation Strategies ............................................................................................ 1-4
- **General Steps for Installing Warehouse Builder** ......................................................... 1-7
- **Understanding the Installation Requirements** ............................................................ 1-8
- **Preparing the Server** .................................................................................................... 1-10
  - UNIX Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2 .......... 1-10
  - Windows Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2 .... 1-11
- **Preparing the Oracle Database** .................................................................................. 1-12
  - Database Configuration Settings for the Warehouse Builder Repository ..................... 1-13
  - Configuring the Target Data File Path for Flat File Targets ......................................... 1-13
  - Configuring Oracle 10g Databases for Design and Runtime Repositories (Optional) .... 1-13
- **Preparing Client Computers** ........................................................................................ 1-17
- **Downloading and Installing the Standalone Warehouse Builder Software** ................ 1-17
  - About the Oracle Universal Installer ........................................................................... 1-18
  - Installing the Warehouse Builder Software ................................................................. 1-18
  - Launching Warehouse Builder Components ............................................................... 1-19
- **Hosting the Repository on Oracle Database 10g Release 2** ..................................... 1-21
  - Running Scripts to Create a Warehouse Builder Repository Schema ......................... 1-21
  - Enabling Access to Workspaces Hosted on Oracle 10g Databases ............................ 1-22
- **Steps for Installing Warehouse Builder in Oracle RAC Environments** ...................... 1-22
  - Installing Warehouse Builder on Each Node of a Cluster ........................................... 1-24
  - Ensuring the Availability of Service Names for Oracle RAC Nodes ............................ 1-25

## Part II  Managing Workspaces and Workspace Users

### 2  Managing Workspaces and Workspace Users

- **Using the Repository Assistant** .................................................................................. 2-1
- **Connecting to the Oracle Database** ........................................................................... 2-1
  - Defining Workspace Users ......................................................................................... 2-1
- **Choosing Workspace Operations** ............................................................................. 2-2
  - Managing Workspaces ............................................................................................... 2-2
  - Selecting the Base Language for the Repository ....................................................... 2-3
Managing Workspace Users................................................................. 2-5
Implementing a Remote Runtime (Optional)........................................ 2-6
Remote Runtime Scenarios .................................................................. 2-6
Setting the Security Policy for the Repository .................................. 2-10

3 Upgrading to Oracle Warehouse Builder 11g Release 1 (11.1)
General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1) ......................................................... 3-1
Migrating An Oracle Database Environment ..................................... 3-2
Migrating a Complete Database ........................................................ 3-3
Migrating an Oracle Workflow Schema ............................................. 3-3
Selectively Migrating a Warehouse Builder Environment to a New Database .................................................. 3-4
Upgrading a Design Repository .......................................................... 3-7
Migrating All Design Metadata from Warehouse Builder 10g Release 2 (10.2) .................................................. 3-8
Selectively Migrating Design Metadata ............................................. 3-9
Upgrading Existing Control Centers or Runtime Repositories ........... 3-11
Using the Control Center Upgrade Assistant ................................... 3-12
Connecting to a New Control Center ................................................ 3-12
Connecting to an Existing Runtime Repository Or Control Center .... 3-13
Choosing An Upgrade Operation ....................................................... 3-13
Reviewing Selections in the Upgrade Assistant .................................. 3-16
Upgrading Locations in the Design Repository .................................. 3-16
Reusing and Redeploying Specific Objects ........................................ 3-17
Reusing Dimensions and Cubes ......................................................... 3-17
Reusing Advanced Queues .............................................................. 3-18
Reusing Oracle Workflow Locations ................................................. 3-18
Reusing Process Flows and Schedules .............................................. 3-18
Reusing Flat Files and External Directories from a Different Database Instance .................................................. 3-18
Reusing Data Profiles ...................................................................... 3-19

4 Deinstalling Oracle Warehouse Builder
General Steps for Deinstalling Oracle Warehouse Builder .................. 4-1
Deleting the Workspace Users ......................................................... 4-1
Deleting the Workspace Owner ........................................................ 4-2
Deinstalling the Oracle Warehouse Builder Software ....................... 4-3
Deleting the Schema Objects (Optional) .......................................... 4-3
Deleting a Repository from an Oracle 10g Database ......................... 4-3

5 Installing and Enabling Optional Components
Enabling Integration with Oracle E-Business Suite .......................... 5-1
Configuring Repository Browser Environments ................................ 5-2
Enabling Integration with Third-Party Name and Address Data Libraries .......................... 5-2
Enabling Integration with Oracle Workflow ..................................... 5-3

6 Troubleshooting a Warehouse Builder Installation
General Steps for Troubleshooting Warehouse Builder ....................... 6-1
Inspecting Logs Files in Warehouse Builder ...................................... 6-1
Part II  Administering Oracle Warehouse Builder

7 Implementing Security in Warehouse Builder

About Metadata Security .................................................................................................................. 7-1
About the Security Interface ............................................................................................................. 7-1
Metadata Security Strategies .......................................................................................................... 7-2
  Minimal Metadata Security Strategy (Default) ........................................................................... 7-3
  Multiuser Security Strategy ......................................................................................................... 7-3
  Full Metadata Security Strategy ................................................................................................ 7-3
Registering Database Users ........................................................................................................... 7-4
  Selecting Existing or Creating New Database Users ................................................................. 7-4
Editing User Profiles ....................................................................................................................... 7-6
  Roles ............................................................................................................................................... 7-6
  Default Object Privilege ............................................................................................................... 7-6
  System Privileges ......................................................................................................................... 7-8
Support for a Multiple-user Environment .................................................................................... 7-9
  Read/Write Mode ......................................................................................................................... 7-9
  Read-Only Mode ........................................................................................................................... 7-9
Defining Security Roles .................................................................................................................. 7-9
  Everyone Role ............................................................................................................................. 7-10
  Administrator Role ..................................................................................................................... 7-10
Editing Role Profiles ....................................................................................................................... 7-10
  Users ............................................................................................................................................... 7-11
Applying Security Properties on Specific Metadata Objects ......................................................... 7-11
  Security Tab .................................................................................................................................. 7-11
Security Enforcement ...................................................................................................................... 7-12
Managing Passwords in Warehouse Builder ................................................................................ 7-13
  Changing Passwords that Access Warehouse Builder ............................................................. 7-13
  Encrypting Passwords to Warehouse Builder Locations ............................................................ 7-13

8 Creating Custom Objects and Properties

Extending the Workspace With User Defined Objects .................................................................... 8-1
  About Oracle Metabase (OMB) Plus ........................................................................................... 8-1
Adding New Properties to Workspace Objects ................................................................................ 8-3
  Creating UDPs: An Example ....................................................................................................... 8-4
Adding UDOs to the Workspace ...................................................................................................... 8-5
  Writing Scripts to Define UDOs ................................................................................................... 8-6
  Creating UDOs: An Example ....................................................................................................... 8-6
  Associating UDOs with Objects .................................................................................................. 8-9
Working with UDOs and UDPs ....................................................................................................... 8-12
  Propagating UDOs and UDPs to Other Workspaces ................................................................. 8-13
9 Managing Metadata Dependencies

Introduction to the Metadata Dependency Manager ........................................... 9-1
Usage Scenario ......................................................................................... 9-2
What are Lineage and Impact Analysis Diagrams? ..................................... 9-3
Generating an LIA Diagram ........................................................................ 9-4
Modifying the Display of an LIA Diagram .................................................... 9-4
Using Groups in an LIA Diagram ................................................................. 9-5
Displaying an Object’s Attributes ............................................................... 9-5
Propagating Metadata Changes Throughout Warehouse Builder ................ 9-6
Propagating Changes in the Dependency Manager ....................................... 9-7
About the Metadata Dependency Manager User Interface ......................... 9-7

10 Version and History Management

Snapshots Versus the Metadata Loader ......................................................... 10-1
About Snapshots ....................................................................................... 10-2
Creating Snapshots .................................................................................... 10-3
Adding Components to a Snapshot ............................................................. 10-3
Managing Snapshots .................................................................................. 10-4
Comparing Snapshots ............................................................................... 10-5
Converting a Full Snapshot to a Signature Snapshot .................................... 10-6
Restoring Workspace Objects From Snapshots .......................................... 10-7
Exporting and Importing Snapshots ............................................................ 10-7
Deleting Snapshots .................................................................................... 10-8
Version and History Management with the Metadata Loader (MDL) .......... 10-8
Accessing the Metadata Loader ................................................................. 10-9
Using Metadata Loader in the Design Center ............................................ 10-9
Exporting Metadata from the Design Center .............................................. 10-9
Importing Metadata Using the Design Center ............................................ 10-12
Upgrading Metadata from Previous Versions ............................................ 10-19
Metadata Loader Utilities ......................................................................... 10-22
Multiple Session Concurrency and MDL ..................................................... 10-23
Metadata Loader Log File ....................................................................... 10-23
About Metadata Loader Results ............................................................... 10-24

11 Managing Multiple Environments from Development to Production

Strategies for Managing Multiple Environments .......................................... 11-1
Multiple Environments Based on a Single Design Environment ................ 11-1
Using Configurations to Manage Multiple Environments ......................... 11-3
About Configurations ............................................................................... 11-4
Deploying a Design to Multiple Environments ........................................... 11-5
Multiple Environments Based on Multiple Design Environments .............. 11-7
Using Snapshots to Manage Multiple Environments .................................. 11-8
## 12 Moving Large Volumes of Data with Transportable Modules

### About Transportable Modules
- About Transportable Modules and Oracle Database Technology
- About Transportable Modules and Oracle Database Technology

### Benefits of Using Transportable Modules

### Instructions for Using Transportable Modules
- Verifying the Requirements for Using Transportable Modules
- Specifying Locations for Transportable Modules
- Creating a Transportable Module
- Configuring a Transportable Module
- Generating and Deploying a Transportable Module
- Designing Mappings that Access Data through Transportable Modules

### Editing Transportable Modules
- Name
- Source Location
- Tablespaces
- Target Locations
- Viewing Tablespace Properties
- Reimporting Metadata into a Transportable Module
This preface includes the following topics:

- **Audience** on page ix
- **Documentation Accessibility** on page ix
- **Conventions** on page x
- **Related Publications** on page x

## Audience

This manual is written for those responsible for installing Oracle Warehouse Builder, including:

- Data warehouse administrators
- System administrators
- Data warehouse and ETL developers
- Other MIS professionals

To install Oracle Warehouse Builder, you must be familiar with installing Oracle Database.

## Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

http://www.oracle.com/accessibility/

### Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.
TTY Access to Oracle Support Services
Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Conventions
The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><code>monospace</code></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>

Related Publications
The Warehouse Builder documentation set includes these manuals:

- *Oracle Warehouse Builder User’s Guide*
- *Oracle Warehouse Builder API and Scripting Reference*

To access to documentation set, including the latest version of the release notes, refer to the following Web site:

http://www.oracle.com/technology/documentation/warehouse.html

In addition to the Warehouse Builder documentation, you can reference *Oracle Database Data Warehousing Guide*. 
Part I Installing Oracle Warehouse Builder

This part contains the following chapters:

- Chapter 1, "Installation Overview and Requirements"
- Chapter 2, "Managing Workspaces and Workspace Users"
- Chapter 3, "Upgrading to Oracle Warehouse Builder 11g Release 1 (11.1)"
- Chapter 4, "Deinstalling Oracle Warehouse Builder"
- Chapter 5, "Installing and Enabling Optional Components"
- Chapter 6, "Troubleshooting a Warehouse Builder Installation"
1 Installation Overview and Requirements

This chapter outlines the installation process, discusses the hardware and software requirements, and introduces the Oracle Warehouse Builder architecture and its components. This chapter includes the following topics:

- What’s New in Installing Warehouse Builder on page 1-1
- Warehouse Builder Architecture and Components on page 1-2
- General Steps for Installing Warehouse Builder on page 1-6
- Understanding the Installation Requirements on page 1-8
- Preparing the Server on page 1-10
- Preparing the Oracle Database on page 1-11
- Preparing Client Computers on page 1-16
- Downloading and Installing the Standalone Warehouse Builder Software on page 1-17
- Hosting the Repository on Oracle Database 10g Release 2 on page 1-20
- Steps for Installing Warehouse Builder in Oracle RAC Environments on page 1-22

What’s New in Installing Warehouse Builder

This section includes important tips for users who are familiar with installing previous version of Warehouse Builder. If you are new to Warehouse Builder, please skip to “Warehouse Builder Architecture and Components” on page 1-2.

Oracle Warehouse Builder Embedded in Oracle Database 11g

Beginning in this release, Warehouse Builder is automatically installed when you install the Oracle Database 11g Release 1 (11.1).

To install Warehouse Builder on client computers, download and install the standalone Warehouse Builder software. The standalone software is also required for certain cases such as installing Warehouse Builder 11g onto Oracle Database 10g. For more information, see "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

SYSDBA Privileges No Longer Required for Installation

Previously, installing a Warehouse Builder repository on an Oracle Database required a user with SYSDBA privileges. Beginning in this release, this is no longer required. A schema OWBSYS is created while installing Oracle Database 11g Release 1. OWBSYS holds the metadata which is divided into workspaces. To start using Warehouse Builder, you create a new workspace. You do not need SYSDBA privileges.

Unified Warehouse Builder Repository with Multiple Workspaces

Previously, users accessed the repository as a whole. Beginning with this release, users are assigned to workspaces within the repository. Thus, instead of granting access to a repository, you grant access to a workspace.
Starting and Stopping the Browser Listener

In previous releases, Windows users could start and stop the Browser Listener from the program group. You could go to Start, Programs, owb home, Warehouse Builder and click Start Browser Listener and Stop Browser Listener. Beginning in this release, you can only stop and start the Browser Listener from a command line. This change improves usability and facilitates new password requirements.

Oracle Workflow Embedded within Warehouse Builder

Previously, if you wanted to utilize Oracle Workflow to manage job dependencies or if you wanted to deploy process flows, it was necessary to install Oracle Workflow. Beginning in this release, these additional installation steps are no longer required as Oracle Workflow components are embedded within Warehouse Builder.

Security Based on PL/SQL Package No Longer Supported

Beginning with Warehouse Builder11g Release 1 (11.1), the preferred method of implementing metadata security is through the user interface available in the Design Center and described in the Oracle Warehouse Builder User’s Guide. If, in a previous release, you implemented security using a PL/SQL package, Warehouse Builder 11g Release 1 (11.1) does support that implementation.

Warehouse Builder Architecture and Components

Oracle Warehouse Builder is an information integration tool that leverages the Oracle Database to transform data into high-quality information. The Oracle Database is a central component in the Warehouse Builder architecture because the Database hosts the Warehouse Builder repository and the code generated by Warehouse Builder.

Figure 1–1 illustrates the interaction of the major components of the Warehouse Builder software.

The Design Center is the user interface for designing, managing, scheduling, and deploying ETL processes for moving and transforming data. All metadata associated with the work done in the Design Center is stored in the Warehouse Builder Repository. The repository is hosted on an Oracle Database and you can use the Repository Browser to report on the metadata in the repository. Also hosted on an Oracle Database is the Target Schema to which Warehouse Builder loads data resulting from the ETL processes that you run through the Control Center Service.
Design Center

The Design Center provides the graphical interface for defining sources, designing targets, and designing ETL and other data transformation processes. As you create a design in the Design Center, you are working with logical designs only, not physical implementations.

Control Center Manager

The Control Center Manager is the console for managing deployment and job execution. Previously, in the Design Center, you created a logical design for transforming data. Your logical design may have introduced objects that do not yet exist, such as staging tables. Now in the Control Center Manager, you deploy the design. That is, you instruct Warehouse Builder to create the necessary physical objects such as the staging tables, for example. Subsequently, you execute the design. During execution, Warehouse Builder runs the code associated with extracting, transforming, and loading the data.

Target Schema

The target schema is the target to which you load your data and the data objects that you designed in the Design Center such as cubes, dimensions, views, and mappings. The target schema contains Warehouse Builder components such as synonyms that enable the ETL mappings to access the audit/service packages in the repository. The repository stores all information pertaining to the target schema such as execution and deployment information.

Notice that the target schema is not a Warehouse Builder software component but rather an existing component of the Oracle Database. As such, you can associate multiple target schemas with a single Warehouse Builder repository. You can have a 1 to 1 relationship or many target schemas to a single repository.
Warehouse Builder Repository

The repository schema stores metadata definitions for all the sources, targets, and ETL processes that constitute your design metadata. In addition to containing design metadata, a repository can also contain the runtime data generated by the Control Center Manager and Control Center Service.

As part of the initial installation of Warehouse Builder, you use the Repository Assistant to define the repository in an Oracle Database. You can host a Warehouse Builder 11g repository either on Oracle Database 10g Release 2 (10.2) or 11g.

About Workspaces

In defining the repository, you create one or more workspaces with each workspace corresponding to a set of users working on related projects. A common practice is to create separate workspaces for development, testing, and production. Using this practice, you can allow users such as your developers access to the development and testing workspaces but restrict them from the production workspace.

Later in the implementation cycle, you also use the Repository Assistant to manage existing workspaces or create new ones.

For examples of the options for implementing repositories, see "Implementation Strategies" on page 1-4.

Repository Browser

The Repository Browser is a web browser interface for reporting on the repository. You can view the metadata, create reports, audit runtime operations and perform lineage and impact analysis. The Repository Browser is organized such that you can browse design-specific and control center-specific information.

Control Center Service

The Control Center Service is the component that enables you to register locations. It also enables deployment and execution of the ETL logic you design in the Design Center such as mappings and process flows.

Implementation Strategies

This section provides an overview of the various choices and considerations for implementing Warehouse Builder. Detailed instructions on how to implement each option are provided in subsequent sections.

The choices for implementing Warehouse Builder include the following:

- Basic Implementation
- Traditional Client/Server Implementation
- Repository on Oracle Database 10g Release 2
- Separate Design and Runtime Environments Implementation
- Development, Test, and Production Environments
- Remote Runtime Environment Implementation
Basic Implementation
The simplest option is to host the client and server components on a single local computer, which is suitable if you are performing a proof of concept or launching a pilot program.

If you install Oracle Database 11g, the most commonly used Warehouse Builder components are also installed for you. The next step is to start the Warehouse Builder Repository Assistant to define a workspace and workspace user.

Two Warehouse Builder components not included in the Oracle Database 11g installation are deployment to Discoverer and execution of runtime scripting commands. To access these components, see "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

To implement a basic implementation with a pre-existing Oracle Database 10g Release 2 installation, see "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

Traditional Client/Server Implementation
This is the most commonly implemented strategy with client components residing on multiple client computers and server components residing on a single server.

When you install the Oracle Database 11g, the Database installation includes all the Warehouse Builder components necessary for the server. You need only start the Warehouse Builder Repository Assistant to define workspaces and workspace users. Subsequently, you download and install the Warehouse Builder software on the client machines.

Repository on Oracle Database 10g Release 2
When you install Oracle Database 11g, the Warehouse Builder server components are also installed.

However, you may choose to host the repository on Oracle Database 10g Release 2. There are no known limitations or restrictions for hosting a Warehouse Builder 11g repository on Oracle Database 10g Release 2, other than the fact that you will not have access to functionality new in Oracle Database 11g.

For instructions, see "Hosting the Repository on Oracle Database 10g Release 2" on page 1-20.

Separate Design and Runtime Environments Implementation
In an implementation such as shown in Figure 1–2, one repository stores the metadata necessary for designing in Warehouse Builder. This includes the metadata for sources, targets, and ETL processes that users access through the Design Center.

You can define a separate environment dedicated to job execution. Refer to this environment as the runtime environment. Notice that each runtime environment requires a single Control Center Service to manage the control center and its deployment and execution activities.

Runtime data is stored in audit tables. You can access the audit tables through the control center specific reports in the Repository Browser.

The only communication between the design repository and the control center repository occurs when you deploy objects to the target schema.
Development, Test, and Production Environments

Organizations typically maintain separate environments for developing and testing prior to releasing a project into a production environment. You can have multiple physical environments based on a common logical design in a single Warehouse Builder repository. Or, you can associate each physical environment with its own Warehouse Builder repository.

To implement either strategy, refer to Chapter 11, "Managing Multiple Environments from Development to Production".

Remote Runtime Environment Implementation

The Control Center Service is the Warehouse Builder server component that governs the deployment of objects to target schemas. Most commonly, the Control Center Service is installed on the computers hosting the target schemas.

In some cases, though, it may be desirable to run the Control Center on a computer that does not host an Oracle Database. You may wish to implement a remote runtime for purposes of load balancing. Also consider remote runtime if your company security policies restrict you from installing additional software on the computer hosting the target schema.

To implement any of these scenarios, refer to "Implementing a Remote Runtime (Optional)" on page 2-6.

General Steps for Installing Warehouse Builder

If you want to upgrade an existing installation, then refer to Chapter 3, "Upgrading to Oracle Warehouse Builder 11g Release 1 (11.1)".

If you want to begin a new installation, then use the following instructions to determine which topics to reference.

To begin a new installation:
1. Determine your implementation strategy.

To accommodate a variety of environments and customer needs, Warehouse Builder offers you flexibility in where you install server and client components.

Review "Warehouse Builder Architecture and Components" on page 1-2 and "Understanding the Installation Requirements" on page 1-8 to develop an implementation strategy.


3. Preparing the Server on page 1-10

4. For Oracle RAC environments only, proceed to "Steps for Installing Warehouse Builder in Oracle RAC Environments" on page 1-22. Otherwise, continue with the next step in these instructions.

5. Preparing the Oracle Database on page 1-11

6. If necessary, install the Warehouse Builder standalone software.

Skip this step if the Warehouse Builder repository is hosted on Oracle Database 11g and you do not intend to integrate with Oracle Discoverer or utilize runtime scripting commands.

Otherwise, refer to the instructions in "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

7. To utilize the default Warehouse Builder schema installed in Oracle Database 11g, first unlock the schema.

Connect to SQL*Plus as the SYS or SYSDBA user. Run the following commands:

```
SQL> ALTER USER OWBSYS ACCOUNT UNLOCK;
SQL> ALTER USER OWBSYS IDENTIFIED BY owbsyspasswd;
```

8. Define Warehouse Builder workspaces and workspace users on the Oracle Database.

Start the Warehouse Builder Repository Assistant on the computer hosting the Oracle Database.

To start the Repository Assistant on Windows, from the Windows Start menu, select Programs and navigate to the Oracle product group you installed in the previous step. Select Warehouse Builder, Administration, and then Repository Assistant.

To start the Repository Assistant on UNIX, run

```
owb home/owb/bin/unix/reposinst.sh
```

Follow the prompts in the Repository Assistant. Or, for detailed instructions, see Chapter 2, "Managing Workspaces and Workspace Users".

9. Setting the Security Policy for the Repository on page 2-10

When you install a repository, Warehouse Builder enforces a default metadata security policy. The default policy is a minimal security policy appropriate for proof-of-concept or pilot projects.

You can override the default by selecting a maximum security policy. Alternatively, you can use the security interface in Warehouse Builder to design your own security policy. In either of these two cases, ensure that repository database has the Advanced Security Option (ASO) enabled.
10. **Installing and Enabling Optional Components** (Optional) on page 5-1

    See Chapter 5 for instructions on enabling optional components such as browsers, third party tools, and related Oracle products.

11. Install the Warehouse Builder software on the client computers.

    Repeat the steps in "Installing the Warehouse Builder Software" on page 1-17 on each computer to be used as a client.

12. When you complete the installation process, verify that the Warehouse Builder components can be successfully launched as described in "Launching Warehouse Builder Components" on page 1-19.

---

**Understanding the Installation Requirements**

Refer to this section as you develop your implementation strategy.

Table 1–1 lists the components required in an Oracle Warehouse Builder environment. Optionally, you can also integrate with the components listed in Table 1–2.

### Required Components

Table 1–1 lists the components required in an Oracle Warehouse Builder environment. The table summarizes important considerations for installing each component and identifies where to look for further details.

<table>
<thead>
<tr>
<th>Components</th>
<th>Important Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server</strong></td>
<td>For Windows, both 32-bit and 64-bit architectures are supported. Ensure a minimum of 850 MB disk space, 768 MB available memory, and 768 MB of page file size, TMP, or swap space. All UNIX platforms require 768 MB available memory and 1100 MB of page file size, TMP, or swap space. For Linux, ensure a minimum of 1100 MB disk space. More disk space is required for all other UNIX platforms. See &quot;Preparing the Server&quot; on page 1-10.</td>
</tr>
</tbody>
</table>
| **Oracle Database** | Ensure that DB_BLOCK_SIZE is set to the most optimal value of 16384 or the largest block size the server allows. Optionally, you may need to change additional configuration settings as described in:  
  * Setting the Security Policy for the Repository on page 2-10  
  * Steps for Installing Warehouse Builder in Oracle RAC Environments on page 1-22  
  * Configuring the Target Data File Path for Flat File Targets on page 1-12  
  See "Preparing the Oracle Database" on page 1-11. |
| **Client computer** | For Windows, ensure that the computer has a minimum of 850 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space. For Linux 32-bit, ensure that the computer has a minimum of 1100 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space.  
  See "Preparing Client Computers" on page 1-16. |
Understanding the Installation Requirements

Installation Overview and Requirements

Table 1–2 Compatible Components

<table>
<thead>
<tr>
<th>Components</th>
<th>Important Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Universal Installer</td>
<td>Start the Universal Installer as described in &quot;Installing the Warehouse Builder Software&quot; on page 1-17. Be sure to specify a separate home directory for Warehouse Builder. For Windows, ensure that you log onto the Windows system as an Administrator and then start the Universal Installer.</td>
</tr>
<tr>
<td>Oracle Warehouse Builder Components</td>
<td>For an overview, see &quot;Warehouse Builder Architecture and Components&quot; on page 1-2.</td>
</tr>
<tr>
<td>Oracle Discoverer</td>
<td>See &quot;Downloading and Installing the Standalone Warehouse Builder Software&quot; on page 1-17.</td>
</tr>
<tr>
<td>Oracle E-Business Suite</td>
<td>See &quot;Enabling Integration with Oracle E-Business Suite&quot; on page 5-1. You can also integrate with other applications such as PeopleSoft, SAP, and Siebel. Integrating with these products is discussed in Oracle Warehouse Builder User's Guide.</td>
</tr>
<tr>
<td>Oracle Workflow</td>
<td>Beginning with Oracle 11g Release 1, Oracle Workflow is shipped with the Warehouse Builder software and licensed for using Warehouse Builder with the Oracle 11g Database. If the Warehouse Builder repository is hosted on Oracle 10g Release 2, you need to install an appropriate version of Oracle Workflow 2.6.4 and follow &quot;Enabling Integration with Oracle Workflow&quot; on page 5-3.</td>
</tr>
<tr>
<td>Third Party Name and Address Data</td>
<td>Requires the following from one of the certified vendors listed on Oracle Technology Network: Regional data libraries Name and Address adapter software See &quot;Enabling Integration with Third-Party Name and Address Data Libraries&quot; on page 5-2.</td>
</tr>
</tbody>
</table>

Compatible Components

Table 1–2 lists some of the optional components that are compatible with an Oracle Warehouse Builder environment. The table does not list all compatible components, but does list those components that either effect how you install Warehouse Builder or require intervention by an Oracle Database Administrator.

The table summarizes important considerations for each optional component and identifies where to look for further details.
Preparing the Server

If you have yet to install an Oracle Database on the server, then consult the Oracle Database Installation Guide for your operating system. Be sure to install the required operating system patches before installing the Oracle Database.

Review the certification matrix on the Oracle Metalink Web site for the most up-to-date list of certified hardware platforms and operating system versions. This Web site also provides compatible client and database versions, patches, and workaround information for bugs. The Oracle Metalink Web site is available at the following URL:

http://metalink.oracle.com/

If you intend to host the Warehouse Builder repository on Oracle Database 11g, no additional steps are required. Proceed with the next topic, "Preparing the Oracle Database" on page 1-11.

If you intend to host the Warehouse Builder repository on Oracle Database 10g Release 2, proceed with either "UNIX Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2" or "Windows Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2".

UNIX Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2

On all UNIX platforms other than Linux, only the Warehouse Builder server components are supported. For Linux 32-bit platforms, however, you can install both server and client components. That is, you can install the Repository and Control Center Service on a UNIX server but the Design Center and Repository Browser require either a Windows or Linux 32-bit platform.

If you are installing only the server components, then ensure that the UNIX operating system meets the requirements listed in Table 1–3. If you are also installing the client components to be accessed by Linux, then see the additional hardware requirements listed in "Preparing Client Computers" on page 1-16.

Table 1–3  UNIX Operating Environment Software Requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk Space</td>
<td>1100 MB for Linux. All other UNIX platforms require more disk space.</td>
</tr>
<tr>
<td>Available Memory</td>
<td>768 MB for Linux. Memory requirements increase depending on the functions being performed and the number of users connected.</td>
</tr>
<tr>
<td>Page File Size, TMP</td>
<td>1 GB for Linux. Swap Space</td>
</tr>
</tbody>
</table>

Setting Environmental Variables on a UNIX Server

When installing on UNIX, you must specify the environmental variable for the Oracle home, that is, the directory in which Warehouse Builder is to be installed.

Use the UNIX commands listed in Table 1–4 where full_path is the path into which you install Warehouse Builder.
Preparing the Oracle Database

Windows Servers Hosting a Warehouse Builder Repository on Oracle 10g Release 2

On Windows platforms, you can install either the Warehouse Builder server or client components or both components on the same computer. Table 1–5 contains the Windows operating system requirements. These requirements are in addition to the requirements of any other Oracle products you are installing on the same computer. Refer to the documentation for each Oracle product you are installing to determine complete system requirements.

Table 1–5 Windows Operating Environment Software Requirements

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disk Space</td>
<td>850 MB</td>
</tr>
<tr>
<td>Available Memory</td>
<td>768 MB</td>
</tr>
<tr>
<td>Page File Size, TMP, or Swap Space</td>
<td>1 GB</td>
</tr>
<tr>
<td>System Architecture</td>
<td>32-bit and 64-bit</td>
</tr>
<tr>
<td>Operating System</td>
<td>Warehouse Builder is supported on the following operating systems:</td>
</tr>
<tr>
<td></td>
<td>Windows 2000 with Service Pack 1 or higher. All editions, including Terminal Services and Windows 2000 MultiLanguage Edition (MLE), are supported.</td>
</tr>
<tr>
<td></td>
<td>Windows Server 2003</td>
</tr>
<tr>
<td></td>
<td>Windows NT Server 4.0, Windows NT Server Enterprise Edition 4.0, and Terminal Server Edition with Service Pack 6a or higher are supported. Windows NT Workstation is no longer supported.</td>
</tr>
<tr>
<td></td>
<td>Windows XP Professional</td>
</tr>
</tbody>
</table>

Preparing the Oracle Database

Warehouse Builder 11g Release 1 (11.1) is supported and certified for use with the following releases of the Oracle Database:

- Oracle Database 11g Standard Edition Release 1 (11.1.x)
- Oracle Database 11g Enterprise Edition Release 1 (11.1.x)
- Oracle Database 10g Standard Edition Release 2 (10.2.x)
- Oracle Database 10g Enterprise Edition Release 2 (10.2.x)
**Note:** Warehouse Builder has not been tested or certified and therefore is not supported for use with the Personal or Express Editions of the Oracle Database.

When you install Enterprise or Standard Editions of Oracle Database 11g, the installation provides you with an unpopulated schema, `OWB_SYS`, for use in Oracle Warehouse Builder 11g.

You must install the Oracle Database on any computer that you intend to create a Warehouse Builder design repository or a target schema as described in subsequent chapters.

The size requirements for the repository varies according to the character set. The tablespace usage of an empty repository with an AL32UTF8 character set, for example, is approximately 90 MB. To accommodate an average usage of the Warehouse Builder repository with a single-byte character set, Oracle recommends an additional 1330 MB for a total of 1420 MB recommended. For multibyte character sets, extrapolate a larger tablespace requirement.

To successfully create a Warehouse Builder repository, the following Oracle Database components are required:

- JServer JAVA Virtual Machine 10.2.0.1
- Oracle XDK 10.2.0.1
- Oracle Database Java Packages 10.2.0.1

If you intend to implement one of the metadata security options available in Warehouse Builder, then enable the Oracle Advanced Security option in the database. See "Setting the Security Policy for the Repository" on page 2-10 for an overview of the metadata security options.

### Database Configuration Settings for the Warehouse Builder Repository

#### Oracle 11g Database Configuration Settings

The Oracle 11g Database self tunes its configuration settings to optimize server resources for hosting both the design and runtime components. The only additional step you may need is "Configuring the Target Data File Path for Flat File Targets" on page 1-12.

#### Oracle 10g Database Configuration Settings

As with Oracle 11g Database, the Oracle 10g Database also self tunes with the possible exception of "Configuring the Target Data File Path for Flat File Targets" on page 1-12.

Additionally, you have the option of maintaining the Warehouse Builder design and runtime components in separate repositories. If so, then refer to "Configuring Oracle 10g Databases for Design and Runtime Repositories (Optional)" on page 1-13.

### Configuring the Target Data File Path for Flat File Targets

To configure the Target Data File Path for Flat file Targets, you set this path in the `init.ora` file of the warehouse instance. Set the `UTL_FILE_DIR` parameter to the directory for the flat file targets so that the database has access to it.

For example, for the output file location `D:\Data\FlatFiles\File1.dat`, set the `UTL_FILE_DIR` parameter in your `init.ora` file to:

```
UTL_FILE_DIR = D:\Data\FlatFiles
```
For multiple valid file locations, such as both D:\Data\FlatFiles and E:\OtherData, set the parameter in init.ora to:

**UTL_FILE_DIR = D:\Data\FlatFiles**
**UTL_FILE_DIR = E:\OtherData**

These lines must be consecutive in the init.ora file.

You can bypass this checking of directories by using the following command:

**UTL_FILE_DIR = * **

### Configuring Oracle 10g Databases for Design and Runtime Repositories (Optional)

This section lists the configuration parameters that ensure performance when using Oracle 10g databases to separately host a design repository and a runtime repository for Warehouse Builder.

#### Parameters for the Design Repository Database Instance

The Oracle Database self tunes its configuration settings to optimize server resources. To ensure that Warehouse Builder performs effectively, verify that DB_BLOCK_SIZE is set to its optimal value.

Table 1–6 lists the initialization parameters for a Warehouse Builder design repository.

#### Table 1–6 Initialization Parameters for the Design Repository Instance

<table>
<thead>
<tr>
<th>Initialization Parameter</th>
<th>Set to Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPATIBLE</td>
<td>db value</td>
<td>Set this to value to equal the release number of the Oracle Database. For example, specify 10 for 10g. If this parameter is not in the initialization file, then add it to the end of the file.</td>
</tr>
<tr>
<td>DB_BLOCK_SIZE</td>
<td>8192</td>
<td>This parameter is set when the database is created. It cannot be changed. Warehouse Builder does not recommend a value higher than 8192 for a design repository.</td>
</tr>
<tr>
<td>DB_CACHE_SIZE</td>
<td>104877600</td>
<td>This is 100 MB.</td>
</tr>
<tr>
<td>LOCK_SGA</td>
<td>TRUE</td>
<td>Oracle recommends locking the design SGA in physical memory.</td>
</tr>
<tr>
<td>O7_DICTIONARY_ACCESSIBILITY</td>
<td>TRUE</td>
<td>Set this to TRUE as an alternative to setting REMOTE_LOGIN_PASSWORDFILE parameter to EXCLUSIVE.</td>
</tr>
<tr>
<td>OPEN_CURSORS</td>
<td>300</td>
<td>You may specify a higher value.</td>
</tr>
<tr>
<td>REMOTE_LOGIN_PASSWORDFILE</td>
<td>EXCLUSIVE</td>
<td>To enable the user sys to connect as sysdba, set this parameter to EXCLUSIVE. If, however, this parameter must be set to NONE, then set O7_DICTIONARY_ACCESSIBILITY to TRUE.</td>
</tr>
</tbody>
</table>

#### Parameters for the Runtime Repository Database Instance

To support the Warehouse Builder runtime component, you may need to modify the Oracle Database instance. Table 1–7 lists the database configuration parameters.
### Initialization Parameters for the Runtime Instance

<table>
<thead>
<tr>
<th>Initialization Parameter</th>
<th>Set to Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>AQ_TM_PROCESSES</td>
<td>1</td>
<td>This parameter is required for the Warehouse Builder and Oracle Workflow advanced queuing system.</td>
</tr>
<tr>
<td>COMPATIBLE</td>
<td>db value</td>
<td>Set this to value to equal the release number of the Oracle Database. If this parameter is not in the initialization file, then add it to the end of the file.</td>
</tr>
<tr>
<td>DB_BLOCK_SIZE</td>
<td>16384</td>
<td>This parameter is set when the database is created. Do not change it. The recommended value is 16384. If your server does not allow a block size this large, then use the largest size available. If your computer has less than 512 MB of RAM, then a value of 9600 is recommended.</td>
</tr>
<tr>
<td>DB_CACHE_SIZE</td>
<td>314632800</td>
<td>Set this value to 300 MB or as high as the system permits. You may need to adjust operating system parameters to allow larger shared memory segments. Do not set any value for the DB_CACHE_SIZE parameter if you set a value for the SGA_TARGET parameter.</td>
</tr>
<tr>
<td>DB_FILE_MULTIPLE_BLOCK_READ_COUNT</td>
<td>16</td>
<td>A value of 16 is recommended, but 32 is preferred. If you have fewer than 8 CPUs, then set DB_WRITER_PROCESSES to 1. Increase this parameter value by 2 for every additional 8 CPUs.</td>
</tr>
</tbody>
</table>
| DB_WRITER_PROCESSES                     | n            | n is the number of CPUs. Disable this parameter by setting it to 0 if:  
- DB_WRITER_PROCESSES has a value greater than 1. In this case, tuning the DBWR_IO_SLAVES parameter has no effect.  
- there is only 1 CPU, and the platform does not support asynchronous I/O. |
| DBWR_IO_SLAVES                           | n            | n is the number of CPUs. Disable this parameter by setting it to 0 if:  
- DB_WRITER_PROCESSES has a value greater than 1. In this case, tuning the DBWR_IO_SLAVES parameter has no effect.  
- there is only 1 CPU, and the platform does not support asynchronous I/O. |
<p>| DISKASYNCHIO                             | TRUE         | If the platform does not support asynchronous I/O, then set DBWR_IO_SLAVES to a positive number, such as 4, to simulate asynchronous I/O. |
| ENQUEUE_RESOURCES                        | 3000 or higher if you are importing large MDL files. | A minimum setting of ‘1’ is required for the install to complete without error. |
| JAVA_POOL_SIZE                           | 20 MB        | The minimum recommended value is 20 MB. Do not set any value for the JAVA_POOL_SIZE parameter if you set a value other than 0 for the SGA_TARGET parameter. |
| JOB_QUEUE_PROCESSES                      | greater than 10 | Optimal setting is 10. If JOB_QUEUE_PROCESSES is set to 0, then the Control Center Service does not run and produces error messages. |</p>
<table>
<thead>
<tr>
<th>Initialization Parameter</th>
<th>Set to Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>LARGE_POOL_SIZE</td>
<td>0</td>
<td>Do not set any value for this parameter if you set a value for the SGA_TARGET parameter. This parameter enables the server to set the LARGE_POOL_SIZE automatically. Prerequisite: PARALLEL_AUTOMATIC_TUNING must be set to TRUE.</td>
</tr>
<tr>
<td>LOG_BUFFER</td>
<td>See comments</td>
<td>Set the value to larger than 512K and must be 128K times the number of CPUs.</td>
</tr>
<tr>
<td>LOG_CHECKPOINT_TIMEOUT</td>
<td>3000</td>
<td>This setting increases the timeout for performing checkpoints from the default 3 minutes to 5 minutes.</td>
</tr>
<tr>
<td>MAX_COMMIT_PROPAGATION_DELAY</td>
<td>0</td>
<td>This is only required when installing on Oracle RAC systems. If it is not set to 0, then data propagation delays may cause NO_DATA_FOUND errors in the Control Center Service.</td>
</tr>
<tr>
<td>OPEN_CURSORS</td>
<td>500</td>
<td>You may specify a higher value if you start multiple sessions or if you run multiple or complicated mappings in one session.</td>
</tr>
<tr>
<td>OPTIMIZER_MODE</td>
<td>all_rows</td>
<td>For other possible optimizer modes, see Oracle Designing and Tuning for Performance, Oracle Database Performance Tuning Guide and Reference, and Oracle Data Warehousing Guide.</td>
</tr>
<tr>
<td>PARALLEL_ADAPTIVE_MULTI_USER</td>
<td>TRUE</td>
<td>Set PARALLEL_AUTOMATIC_TUNING to TRUE as a prerequisite for this parameter.</td>
</tr>
<tr>
<td>PARALLEL_AUTOMATIC_TUNING</td>
<td>TRUE</td>
<td>This setting delegates the task of tuning parallel processing to the server. Set this parameter for Oracle9i or Oracle8i databases only. For Oracle 10g and later, this parameter is not available and setting SGA_TARGET to a nonzero value is recommended.</td>
</tr>
<tr>
<td>PGA_AGGREGATE_TARGET</td>
<td>314572800</td>
<td>This is 300 MB. If you perform frequent sorting and aggregation, then you can increase this value. However, PGA_AGGREGATE_TARGET must be smaller than the available physical memory size.</td>
</tr>
<tr>
<td>PLSQL_OPTIMIZE_LEVEL</td>
<td>2</td>
<td>The PL/SQL compiler in Oracle Database can perform more elaborate optimization on PL/SQL code.</td>
</tr>
<tr>
<td>QUERY_REWRITE_ENABLED</td>
<td>TRUE</td>
<td>Set this parameter to TRUE if you plan to generate materialized views with the QUERY_REWRITE option.</td>
</tr>
<tr>
<td>REMOTE_LOGIN_PASSWORDFILE</td>
<td>EXCLUSIVE</td>
<td>You must use the SYS account with SYSDBA privileges to access or create a runtime schema. The workspace user requires access to certain v_$ tables. These grants are made by the SYSDBA account when you create the workspace. This setting ensures that the SYSDBA privilege is granted to SYS.</td>
</tr>
<tr>
<td>RESOURCE_MANAGER_PLAN</td>
<td>plan_name</td>
<td>Oracle strongly recommends creating a resource plan for managing resource usages for Warehouse Builder runtime. Refer to the Oracle Database Administration Guide for information on resource plans.</td>
</tr>
</tbody>
</table>
Preparing Client Computers

For Windows, ensure that the computer has a minimum of 850 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space.

For Linux, ensure that the computer has a minimum of 1100 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space. Ensure that you set the ORACLE_HOME variable.

If you previously deinstalled Warehouse Builder and the path \owb\home\owb\j2ee\owbb\owbb remains, then delete the owbb directory before installing Warehouse Builder again.

Table 1–7 (Cont.) Initialization Parameters for the Runtime Instance

<table>
<thead>
<tr>
<th>Initialization Parameter</th>
<th>Set to Value</th>
<th>Comments</th>
</tr>
</thead>
</table>
| SGA_TARGET               | 500 MB to 1 GB | The larger value, or as close to it as possible, is recommended if computer memory allows it. If you set the SGA_TARGET parameter, do not set these following parameters which the server automatically adjusts:  
  ■ JAVA_POOL_SIZE  
  ■ DB_CACHE_SIZE  
  ■ LARGE_POOL_SIZE  
  ■ SHARED_POOL_SIZE  
  Alternatively, you can set the SGA_TARGET parameter to 0, which turns off the automatic sizing feature. In that case, follow the recommendations on sizing the preceding four parameters.  
  **Note:** For Oracle 10g and later, setting SGA_TARGET is recommended. |
| SHARED_POOL_SIZE         | 419430400    | The recommended minimum value is 400 MB. Do not set any value for the SHARED_POOL_SIZE parameter if you set a value for the SGA_TARGET parameter. |
| STATISTICS_LEVEL         | TYPICAL      | With this setting, you do not have to create rollback segments. |
| UNDO_MANAGEMENT          | AUTO         | Specifies the directories that PL/SQL can use for file input and output. UTL_FILE_DIR = * specifies that all directories can be used for file input and output. If you want to specify individual directories, then repeat this parameter on contiguous lines for each directory. If you use flat file targets in Warehouse Builder, then set this parameter to the directory where you want to create the flat file target so that your database engine has access to it. Refer to "Configuring the Target Data File Path for Flat File Targets". |
| UTL_FILE_DIR             | *            |  
| WORKAREA_SIZE_POLICY     | AUTO         | . |

Preparing Client Computers

For Windows, ensure that the computer has a minimum of 850 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space.

For Linux, ensure that the computer has a minimum of 1100 MB disk space, 768 MB available memory, and 1GB of page file size, TMP, or swap space. Ensure that you set the ORACLE_HOME variable.

If you previously deinstalled Warehouse Builder and the path \owb\home\owb\j2ee\owbb\owbb remains, then delete the owbb directory before installing Warehouse Builder again.
Setting Environmental Variables on the Linux Client
When installing client components on Linux, you must specify the environmental variable for the Oracle home.

Use the UNIX commands listed in Table 1–8 where full_path is the path into which you install Warehouse Builder.

<table>
<thead>
<tr>
<th>Environmental Variable</th>
<th>C Shell Command</th>
<th>Korn Shell Command</th>
<th>Bourne Shell Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORACLE_HOME</td>
<td>setenv ORACLE_</td>
<td>export ORACLE_</td>
<td>ORACLE_HOME=full_</td>
</tr>
<tr>
<td></td>
<td>HOME full_path</td>
<td>HOME=full_path</td>
<td>path; export ORACLE_</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>HOME</td>
</tr>
</tbody>
</table>

Downloading and Installing the Standalone Warehouse Builder Software
Download the Warehouse Builder standalone software to complete any of the following tasks:

- Installing the software on a client computer
- "Hosting the Repository on Oracle Database 10g Release 2" on page 1-20
- Enabling integration with Oracle Discoverer
- Enabling the use of runtime scripting commands

To download the standalone software, locate the software from the following link:


About the Oracle Universal Installer
When installing the standalone software, Oracle Warehouse Builder utilizes the Oracle Universal Installer to install components and to configure environment variables. The installer guides you through each step of the installation process.

About Oracle Home and Warehouse Builder
Oracle home is the top-level directory into which you install Oracle software. Some Oracle products enable you to share the same Oracle home. Or you can create separate homes and assign names to each home as you install each product.

Warehouse Builder, however, cannot share its home directory with any other Oracle product. When the Oracle Universal Installer prompts you to specify a home directory for Oracle Warehouse Builder, specify a directory different from the Oracle Database or any other Oracle product.

This separate directory is designated by the term owb home in the Warehouse Builder documentation.

For Linux, in addition to specifying the owb home, you also need to set the ORACLE_HOME variable.

Installing the Warehouse Builder Software
Use the Oracle Universal Installer to install Warehouse Builder components.

To install the software, complete the following:

1. Review and complete the Checklist: Before You Start the Universal Installer.
2. Run the installer following either the instructions Running the Oracle Universal 
Installer for Warehouse Builder on Windows or Running the Oracle Universal 
Installer for Warehouse Builder on UNIX.

3. If the Warehouse Builder repository is hosted on an Oracle 10g Database, complete 
the instructions "Hosting the Repository on Oracle Database 10g Release 2" on 
page 1-20.

Next
When the software installation completes successfully, you can continue with the next 
step in General Steps for Installing Warehouse Builder on page 1-6.

Checklist: Before You Start the Universal Installer
This section contains additional points to address before launching the Universal 
Installer:
■ If you have not already done so, review the Oracle Warehouse Builder Release 
Notes either on the Oracle Warehouse Builder CD-ROM or, for the latest version, 
■ Close all other open applications.

Running the Oracle Universal Installer for Warehouse Builder on Windows
To run the Oracle Universal Installer on Windows:
1. Ensure that you are logged on to your system as a member of the Administrators 
group.
2. Insert the Oracle Warehouse Builder CD-ROM.
3. If your computer supports the autorun feature, then the autorun window launches 
the Oracle Warehouse Builder installation.
   If your computer does not support the autorun feature, then locate the executable 
setup.exe in the root directory of the CD-ROM or downloaded software. Start 
the installer by launching the setup.exe program.
4. When prompted, specify a home directory to be used only for the Warehouse 
Builder installation.
   For example, you could specify a directory such as C:\oracle\owb11_1.
   For the sake of brevity, the directory you specify in this step is referred to as the 
owb home throughout this guide.
5. Follow the on screen instructions.
When the software installation completes successfully, you can continue with the next 
step in General Steps for Installing Warehouse Builder on page 1-6.

Running the Oracle Universal Installer for Warehouse Builder on UNIX
To run the Oracle Universal Installer on UNIX:
You can run Oracle Universal Installer from the CD-ROM. Do not run the Installer 
while the CD-ROM directory is the current directory or you will be unable to unmount 
the next CD-ROM when prompted to do so.
1. If you have not already done so, you must set the ORACLE_HOME environmental 
variable as described in "Setting Environmental Variables on a UNIX Server" on 
page 1-10 and "Setting Environmental Variables on the Linux Client" on page 1-17.
2. Log in as the operating system user of the Oracle Database.

For example, log in as the oracle user. If you choose to log in as the oracle user, you must configure the user environment by setting the default file mode creation mask (umask) to 022 in the shell startup file.

Be sure you are not logged in as the root user when you start the Oracle Universal Installer. If you are, then only the root user would have permissions to manage Oracle Warehouse Builder.

3. Start the installer by entering the following at the prompt:

   ```
   cd mount_point
   ./runInstaller
   ```

4. As the installation proceeds, the Oracle Universal Installer prompts you to run several scripts. You need to switch users and run the script as root.

### Launching Warehouse Builder Components

The Oracle Warehouse Builder CD installs the client and server-side software at the same time. After you complete the installation, you can start the Warehouse Builder components listed in Table 1–8.

The components in Table 1–8 are listed in the order that you are likely to use the components directly after installation:

<table>
<thead>
<tr>
<th>Warehouse Builder Component</th>
<th>Windows: Select Start, Programs, Oracle, Warehouse Builder and then...</th>
<th>Linux: Locate owb_home/owb/bin/unix and then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository Assistant</td>
<td>Select Administration, and then Repository Assistant.</td>
<td>Run reposinst.sh</td>
</tr>
<tr>
<td>Design Center</td>
<td>Select Design Center.</td>
<td>Run owbclient.sh</td>
</tr>
<tr>
<td>Start Control Center Service</td>
<td>Select Administration then Start Control Center Service.</td>
<td>Run local_service_login.sh as follows:</td>
</tr>
<tr>
<td>Control Center Manager</td>
<td>Start the Design Center. From the Tools menu, select Control Center Manager.</td>
<td>Run local_service_login.sh as follows:</td>
</tr>
</tbody>
</table>

After you complete the installation, you can start the Warehouse Builder components listed in Table 1–8. The components in Table 1–8 are listed in the order that you are likely to use the components directly after installation:

<table>
<thead>
<tr>
<th>Warehouse Builder Component</th>
<th>Windows: Select Start, Programs, Oracle, Warehouse Builder and then...</th>
<th>Linux: Locate owb_home/owb/bin/unix and then...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository Assistant</td>
<td>Select Administration, and then Repository Assistant.</td>
<td>Run reposinst.sh</td>
</tr>
<tr>
<td>Design Center</td>
<td>Select Design Center.</td>
<td>Run owbclient.sh</td>
</tr>
<tr>
<td>Start Control Center Service</td>
<td>Select Administration then Start Control Center Service.</td>
<td>Run local_service_login.sh as follows:</td>
</tr>
<tr>
<td>Control Center Manager</td>
<td>Start the Design Center. From the Tools menu, select Control Center Manager.</td>
<td>Run local_service_login.sh as follows:</td>
</tr>
</tbody>
</table>
Hosting the Repository on Oracle Database 10g Release 2

To host a Warehouse Builder 11g repository on Oracle Database 10g Release 2, complete the following steps:

1. Install Warehouse Builder 11g on to the computer hosting Oracle Database 10g Release 2 as described in "Installing the Warehouse Builder Software" on page 1-17.

   You can locate the software from the following link:


2. For additional considerations for hosting the repository on Oracle 10g Database, refer to "Preparing the Oracle Database" on page 1-11.

3. Complete the steps in "Running Scripts to Create a Warehouse Builder Repository Schema" on page 1-21.

4. Start and complete the Repository Assistant.

   Use the Repository Assistant to create a Warehouse Builder 11g repository and workspace in the Oracle Database 10g.

5. Complete the steps in "Enabling Access to Workspaces Hosted on Oracle 10g Databases" on page 1-21.
Running Scripts to Create a Warehouse Builder Repository Schema

To create a Warehouse Builder repository on an Oracle 10g Release 2 Database, complete the following steps:

1. Change the current directory to the \owb\home\owb\UnifiedRepos directory.
   For example:
   ```
   C:\> cd \owb\home\owb\UnifiedRepos\n   ```

2. Run the version of SQL*Plus provided with Warehouse Builder, with SYSDBA privilege. This executable is located in the \owb\home\bin directory. For example, type the following:
   ```
   C:\owb home\owb\UnifiedRepos> owb home\bin\sqlplus sys/sys_password as sysdba;
   ```

3. Run the cat_owb.sql script stored in the \owb\home\owb\UnifiedRepos directory.
   The script creates the repository and sets up the required roles and privileges on the 10g Release 2 Database.
   The script prompts you for the name of the default tablespace in which to create OWBSYS schema. For example, to install the OWBSYS schema into the USERS tablespace in a 10g Release 2 Database hosted on Windows, type the following:
   ```
   SQL> @cat_owb.sql
   ```
   Enter Tablespace Name for OWBSYS user:
   ```
   USERS
   ```

4. Unlock the OWBSYS user and assign it a password. For example:
   ```
   SQL> alter user OWBSYS account unlock identified by owbsys_password;
   ```

5. Run the script \owb\home\owb\UnifiedRepos\reset_owbccc_home.sql
   Use this script to ensure that the Control Center runs correctly from the Warehouse Builder 11.1 home. When prompted for the \owb\home, type the directory carefully. The entry is case-sensitive, does not accept a trailing slash, and requires forwards slashes only, regardless of the operating system. For example, for Windows, if the \owb\home is
   ```
   C:\Oracle\My_OWB_Home\>
   ```
   then type the following:
   ```
   C:\Oracle\My_OWB_Home
   ```

Enabling Access to Workspaces Hosted on Oracle 10g Databases

Warehouse Builder 11g clients connect to workspaces on an Oracle 11g Database by default. To access workspaces on an Oracle 10g Release 2 Database, you must take additional steps.

To enable access to workspaces on a 10g Release 2 Database:

1. On each client computer, locate the file \owb\home\owb\bin\admin\Preference.properties.
   If the file does not exist, you can create it based on the example file Preference.properties.tmp in the same directory.
Steps for Installing Warehouse Builder in Oracle RAC Environments

The overall process for installing on an Oracle RAC environment is similar to the "General Steps for Installing Warehouse Builder". However, there are a few specific details to observe as noted in the following instructions:

To install in an Oracle RAC environment:

1. Create the Oracle RAC environment as described in the Oracle Clusterware and Oracle Real Application Clusters Installation Guide specific to your platform.

2. If you have not already done so, review the most recent Oracle Warehouse Builder Release Notes available at http://otn.oracle.com.

3. Preparing host computers

   For each computer to host Warehouse Builder components, configure the tnsnames.ora file located in the ${owb_home}/owb/network/admin directory.

4. Preparing the Oracle Database on page 1-11

   For Oracle 10g RAC environments, be sure to set the initialization parameter MAX_COMMIT_PROPAGATION_DELAY to a value of zero.

   Also configure tnsnames.ora for each Oracle Database server that will be a Warehouse Builder data source or target. If you fail to configure tnsnames.ora for any host or database server, you may encounter a repository connection error.

5. If necessary, install the Warehouse Builder standalone software.

   If the Warehouse Builder repository is hosted on Oracle Database 11g and you do not intend to integrate with Oracle Discoverer, skip to the next step.

   Otherwise, complete the steps in "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

6. Managing Workspaces and Workspace Users on page 2-1

   The Repository Assistant prompts you to define users and an owner for the repository.

   To start the Repository Assistant on Windows, from the Windows Start menu, select Programs and navigate to the Oracle product group you installed in the previous step. Select Warehouse Builder, Administration, and then Repository Assistant.

   To start the Repository Assistant on UNIX, locate owb home/owb/bin/unix and run reposinst.sh.

7. Register each Oracle RAC node.

2. Edit Preference.properties, add a property REPOS_DB_VERSION_ALLOWED and set its value to one of the following:

   - Oracle 10g
   - Oracle 10g, Oracle 11g

   For example:

   REPOS_DB_VERSION_ALLOWED=Oracle 10g,Oracle 11g

   After you save the file, the client can access repositories stored in Oracle Database 10g Release 2.
For each node, start the Repository Assistant and select the Advanced Set up option. Connect to the node using Host:Port:Sid, where Host is the physical node name. Select the option for registering the Oracle RAC instance.

8. If the software is installed on separate disks, copy rtrepos.properties to each node in the cluster.

   If you did not install to a shared disk, then you must manually copy the file owb home/owb/bin/admin/rtrepos.properties from the primary node to each node in the cluster.


10. Installing and Enabling Optional Components (Optional).

   Consider performing the optional step "Configuring Repository Browser Environments" on page 5-2. The Repository Browser lets you nominate a node and register other nodes.

11. Install the Warehouse Builder software on the client computers.

    Repeat "Installing the Warehouse Builder Software" on page 1-17 for each computer to be used as a client.

12. When complete the installation process, you can start all the Warehouse Builder components.

    When "Launching Warehouse Builder Components" on page 1-19 such as the Design Center, Control Center Manager, and Repository Assistant, select the log on option SQL*Net connection and specify the net service name you assigned in the tnsnames.ora file.

    Because you can connect to Warehouse Builder repositories using a net service name, you can embed Oracle RAC properties into the connect string to utilize Oracle RAC capabilities such as connect time failover, load balancing on server and load balancing of connections.

13. "Ensuring the Availability of Service Names for Oracle RAC Nodes" on page 1-25

    The Control Center Service requires that service names for the individual nodes in the cluster be available. If these are not present after the Oracle RAC installation, you must manually ensure the availability.

**Installing Warehouse Builder on Each Node of a Cluster**

Whether you are installing Warehouse Builder components onto a server or a client computer, you use the Oracle Universal Installer to install Warehouse Builder components.

For Oracle RAC, it is recommended that you install the Warehouse Builder components on each node of the cluster. The Control Center Service is required on to each node of the Oracle RAC cluster.You can achieve this in a single installation of the Warehouse Builder software if you install on a shared disk such as an OCFS or NTS shared disk.

Before launching the Universal Installer, review and complete the Checklist: Before You Start the Universal Installer.

**Checklist: Before Using the Universal Installer in an Oracle RAC Environment**

This section contains additional points to address before launching the Universal Installer:
If you have not already done so, review the latest version Oracle Warehouse Builder Release Notes at http://otn.oracle.com.

The installed location must be the same directory path if using separate owb home installed disks, that is, local physical disks on each server.

Close all other open applications.

For Windows Users Installing in an Oracle RAC Environment

To run the Oracle Universal Installer in an Oracle RAC environment:

1. Ensure that you are logged on to your system as a member of the Administrators group.
2. Insert the Oracle Warehouse Builder CD-ROM.
3. If your computer supports the autorun feature, the autorun window will automatically start the Oracle Warehouse Builder installation.
   
   If your computer does not support the autorun feature, locate the executable setup.exe in the root directory of the CD-ROM or downloaded software. Start the installer by launching the setup.exe program.
4. When prompted to specify the cluster node, you can select all hosts or the local node.
   
   If you select local node, then you must install Warehouse Builder separately for each system unless installing to a shared disk.
5. When prompted, specify a home directory to be used only for the Warehouse Builder installation.
   
   For example, specify a path such as C:\oracle\owb11_1.
   
   For the sake of brevity, the path you specify in this step is referred to as owb home throughout this guide.
6. Follow the on screen instructions.

When the software installation completes successfully, you can continue with the next step in "Steps for Installing Warehouse Builder in Oracle RAC Environments" on page 1-22.

For UNIX Users Installing in an Oracle RAC Environment

To run the Oracle Universal Installer in an Oracle RAC environment:

You can run Oracle Universal Installer from the CD-ROM. Do not run the Installer while the CD-ROM directory is the current directory or you will be unable to unmount the next CD-ROM when prompted to do so.

1. If you have not already done so, you must set the ORACLE_HOME environmental variable as described in "Setting Environmental Variables on a UNIX Server" on page 1-10 and "Setting Environmental Variables on the Linux Client" on page 1-17.
2. To enable clusterware installation, ensure that you are running an interactive secure shell such as /bin/ssh and have host user equivalency to all nodes.
3. Log in as the operating system user of the Oracle Database.
   
   For example, log in as the oracle user.
   
   Be sure you are not logged in as the root user when you start the Oracle Universal Installer. If you are, then only the root user will have permissions to manage Oracle Warehouse Builder.
4. Start the installer by entering the following at the prompt:

```
cd mount_point
./runInstaller
```

5. When prompted to specify the cluster node, you can select all hosts or the local node.

   If you select local node, then you must install Warehouse Builder separately for each system unless installing to a shared disk.

6. As the installation proceeds, the Oracle Universal Installer prompts you to run several scripts. You need to switch users and run the script as root.

When the software installation completes successfully, you can continue with the next step in General Steps for Installing Warehouse Builder on page 1-6.

**Ensuring the Availability of Service Names for Oracle RAC Nodes**

1. List all of the instance or node names in the cluster. Issue the following command:

```
srvctl config database -d dbname
```

   where dbname is the unique database name as specified by the init parameter db_name.

2. For a given instance, instn, add a service with the following command:

```
srvctl add service -d dbname -s instn -r instn
```

   The resulting service name is instn.clusterdomainname. For example, if the instance name is owbrac1, then the service name could be owbrac1.us.oracle.com.

3. For a given instance, instn, start the service with the following command:

```
srvctl start service -d dbname -s instn
```

4. For a given instance, instn, verify the service is running with the following command:

```
srvctl status service -d dbname -s instn
```

5. Complete steps 2 through 4 for each node.
2 Managing Workspaces and Workspace Users

This chapter includes the following topics:

- Using the Repository Assistant on page 2-1
- Connecting to the Oracle Database on page 2-1
- Choosing Workspace Operations on page 2-2
- Implementing a Remote Runtime (Optional) on page 2-6
- Setting the Security Policy for the Repository on page 2-10

Using the Repository Assistant

Use the Repository Assistant to define the repository on an Oracle Database.

Note: Before proceeding with this wizard, first complete steps 1 through 6 in General Steps for Installing Warehouse Builder on page 1-6.

As an alternative to using the wizard, you can install a repository using the OMB Plus scripting language. The default settings for creating a repository are the same whether you use the OMB Plus or the Repository Assistant. For example both methods assign USERS as the default tablespace for indexes.

For more information installing through the scripting language, read about the OMBINSTALL command in Oracle Warehouse Builder Scripting Reference.

Connecting to the Oracle Database

The assistant prompts you for connection information to the Oracle Database.

In a RAC environment, in addition to typing the host name, port number, and Oracle service name, also select the SQL*NET Connection option. Type the net service name, defined in owb home\network\admin\tnsnames.ora.

To proceed with the next step, Defining Workspace Users, note that the Database must be running and you must enter a database user with SYSDBA privileges.

Defining Workspace Users

Specify a workspace user name and password based on the "Guidelines for User Names and Passwords". The Repository Assistant assigns the user as a deployment target. In other words, that user can access both the Design Center for designing ETL processes and the Control Center Manager for deploying and auditing.

Guidelines for User Names and Passwords

The Repository Assistant prompts you to create user names and confirm passwords.

To specify a valid user name and password, adhere to the security standard implemented on the Oracle Database. The default minimum requirement is that both the user name and password be a VARCHAR(30). Also, do not include any special characters. Your database may have more requirements if a password complexity verification routine was applied. For more information about user names, passwords,
Choosing Workspace Operations

Choose one of the following options:

- **Managing Workspaces**: Select this option if you want to create, delete, or alter a workspace owner.
- **Managing Workspace Users**: Select this option to create or remove the registration for one or more workspace users.
- **Registering a Real Application Cluster (RAC) instance**: This option is only available for local installations. To register a RAC instance, select this option, click Next and then Finish on the Summary page.

For additional information, see "Steps for Installing Warehouse Builder in Oracle RAC Environments" on page 1-22.

Managing Workspaces

Managing a workspace includes:

- **Creating a new workspace**
- **Dropping Workspaces**

Dropping Workspaces

When you drop a workspace owner, be aware that each workspace owner can be associated with multiple workspace users. After you drop a workspace owner, any remaining users become orphans and you cannot use the Repository Assistant to delete them. Therefore, use the Repository Assistant to delete associated users prior to dropping a workspace owner. Alternatively, you can delete orphan users through the SQL Plus.

Selecting the Default Tablespaces

The assistant recommends a set of default tablespaces. You can accept the recommendations or specify new tablespaces.
Figure 2–1  Dialog: Selecting Default Tablespaces

Selecting the Base Language for the Repository

The repository has a single base language. That is, the physical name for each repository object is assumed to be written in the base language.

The Repository Assistant assumes a default base language contingent upon the locale of the computer from which you launched the Repository Assistant. For example, when you run the assistant from a computer with its locale set to British English, the default base language for the repository is en_GB for British English.

You can accept the default or select from a list of base languages. Note, however, that you can define the base language only once. That is, after you create the repository, you cannot change the base language and you cannot add another base language. This means that when creating a new repository object, the business name can be in any language, but the physical name should always be written in the base language.

However, to enable users to use different languages for the physical names of objects, you have the option to support multiple display languages.

Adding Display Languages

Unlike the base language, you can have multiple display languages and you can add display languages after initially defining the repository.

Display languages are associated with business names only. For a given object, its physical name corresponds to the base language. However, for each display language you enable, users can create a corresponding business name.

Table 2–1 lists the International Organization for Standardization (ISO) IDs for each display language supported in Warehouse Builder.
<table>
<thead>
<tr>
<th>ISOID</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>sq_AL</td>
<td>Albanian</td>
</tr>
<tr>
<td>en_US</td>
<td>American English</td>
</tr>
<tr>
<td>ar_AE</td>
<td>Arabic</td>
</tr>
<tr>
<td>ar_EG</td>
<td>Arabic Egypt</td>
</tr>
<tr>
<td>as_IN</td>
<td>Assamese</td>
</tr>
<tr>
<td>bn_IN</td>
<td>Bangla</td>
</tr>
<tr>
<td>pt_BR</td>
<td>Brazilian Portuguese</td>
</tr>
<tr>
<td>bg_BG</td>
<td>Bulgarian</td>
</tr>
<tr>
<td>fr_CA</td>
<td>Canadian French</td>
</tr>
<tr>
<td>ca_ES</td>
<td>Catalan</td>
</tr>
<tr>
<td>hr_HR</td>
<td>Croatian</td>
</tr>
<tr>
<td>cs_CZ</td>
<td>Czech</td>
</tr>
<tr>
<td>da_DK</td>
<td>Danish</td>
</tr>
<tr>
<td>nl_NL</td>
<td>Dutch</td>
</tr>
<tr>
<td>en_GB</td>
<td>English</td>
</tr>
<tr>
<td>et_EE</td>
<td>Estonian</td>
</tr>
<tr>
<td>fi_FI</td>
<td>Finnish</td>
</tr>
<tr>
<td>fr_FR</td>
<td>French</td>
</tr>
<tr>
<td>de_DE</td>
<td>German</td>
</tr>
<tr>
<td>el_GR</td>
<td>Greek</td>
</tr>
<tr>
<td>gu_IN</td>
<td>Gujarati</td>
</tr>
<tr>
<td>he_IL</td>
<td>Hebrew</td>
</tr>
<tr>
<td>hi_IN</td>
<td>Hindi</td>
</tr>
<tr>
<td>hu_HU</td>
<td>Hungarian</td>
</tr>
<tr>
<td>is_IS</td>
<td>Icelandic</td>
</tr>
<tr>
<td>in_ID</td>
<td>Indonesian</td>
</tr>
<tr>
<td>it_IT</td>
<td>Italian</td>
</tr>
<tr>
<td>ja_JP</td>
<td>Japanese</td>
</tr>
<tr>
<td>kn_IN</td>
<td>Kannada</td>
</tr>
<tr>
<td>ko_KR</td>
<td>Korean</td>
</tr>
<tr>
<td>es_US</td>
<td>Latin American Spanish</td>
</tr>
<tr>
<td>lv_LV</td>
<td>Latvian</td>
</tr>
<tr>
<td>lt_LT</td>
<td>Lithuanian</td>
</tr>
<tr>
<td>ms_MY</td>
<td>Malay</td>
</tr>
<tr>
<td>ml_IN</td>
<td>Malayalam</td>
</tr>
<tr>
<td>mr_IN</td>
<td>Marathi</td>
</tr>
</tbody>
</table>
Managing Workspace Users

All users and the workspace owner must first be defined as Oracle Database users. As a workspace owner, the actions you can take include adding workspace users or "Deleting Workspace Users" on page 2-6.

You cannot change user passwords from within the product. Change passwords directly in the Oracle Database as described in Oracle® Database Security Guide.

Selecting Workspace Users

The left panel in Figure 2–2 lists the existing Oracle Database users and schemas. Select existing database users from the list. Or, if adding new users, you can define and register a new user by clicking on Create New User located in the lower left corner.

If you select an existing user, you are prompted for the password before allowing you to proceed.

When selecting from the list, you can select one or more database users. Notice that, for security reasons, database administrator users such as SYSDBA are not available for registering as Warehouse Builder users.

### Table 2–1 (Cont.) ISO IDs for Supported Languages

<table>
<thead>
<tr>
<th>ISOID</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>es_MX</td>
<td>Mexican Spanish</td>
</tr>
<tr>
<td>no_NO</td>
<td>Norwegian</td>
</tr>
<tr>
<td>or_IN</td>
<td>Oriya</td>
</tr>
<tr>
<td>pl_PL</td>
<td>Polish</td>
</tr>
<tr>
<td>pt_PT</td>
<td>Portuguese</td>
</tr>
<tr>
<td>pa_IN</td>
<td>Punjabi</td>
</tr>
<tr>
<td>ro_RO</td>
<td>Romanian</td>
</tr>
<tr>
<td>ru_RU</td>
<td>Russian</td>
</tr>
<tr>
<td>zh_CN</td>
<td>Simplified Chinese</td>
</tr>
<tr>
<td>sk_SK</td>
<td>Slovak</td>
</tr>
<tr>
<td>sl_SI</td>
<td>Slovenian</td>
</tr>
<tr>
<td>es_ES</td>
<td>Spanish</td>
</tr>
<tr>
<td>sv_SE</td>
<td>Swedish</td>
</tr>
<tr>
<td>ta_IN</td>
<td>Tamil</td>
</tr>
<tr>
<td>te_IN</td>
<td>Telugu</td>
</tr>
<tr>
<td>th_TH</td>
<td>Thai</td>
</tr>
<tr>
<td>zh_TW</td>
<td>Traditional Chinese</td>
</tr>
<tr>
<td>tr_TR</td>
<td>Turkish</td>
</tr>
<tr>
<td>uk_UA</td>
<td>Ukrainian</td>
</tr>
<tr>
<td>vi_VN</td>
<td>Vietnamese</td>
</tr>
</tbody>
</table>
Implementing a Remote Runtime (Optional)

Recall that the Control Center Service is the Warehouse Builder server component that governs the deployment of objects to target schemas. Most commonly, the Control Center Service is installed on the computers hosting the target schemas. In limited cases, though, it may be necessary to run the Control Center on a computer that does not host an Oracle Database.

To achieve this, implement a remote runtime environment. That is, the target schemas are remote with respect to the Control Center Service running on another server.

Remote Runtime Scenarios

There are several scenarios for implementing a remote runtime. However, to deploy and run ETL processes in any of the scenarios, keep in mind that Control Center Service must be running. Remote runtime environments include the following scenarios:

- Control Center Service Installed on a Dedicated Computer
- Control Center Service Installed on a Local Server
- A Standalone Target Schema

Regardless of the specific scenario that you choose, follow the instructions "Steps for Installing and Testing a Remote Runtime" on page 2-9.
Control Center Service Installed on a Dedicated Computer
Oracle Database is not required to be installed on the computer hosting the Control Center Service. You can deploy all types of mappings to the remote target without restriction.

![Control Center Service Installed on a Dedicated Computer](image)

Control Center Service Installed on a Local Server
You can deploy all types of mappings to the remote target. However, if the mapping calls external programs such as SQL Loader, then these programs run on the local server.
A Standalone Target Schema

Although it is not a preferred scenario, it is possible to implement a remote target without any Warehouse Builder components installed on the target computer. This scenario has a significant restriction. Because the remote target schema and the repository are in two different databases, you cannot deploy PL/SQL mappings to the standalone target schema.
Steps for Installing and Testing a Remote Runtime
To implement a remote runtime environment, complete the following steps:

1. If necessary, install the client components including the Design Center and Control Center Service as depicted in "Remote Runtime Scenarios" on page 2-6.
   For Oracle Database 11g Release 1, the necessary components are already installed. Skip to the next step.
   For Oracle Database 10g Release 2, see "Downloading and Installing the Standalone Warehouse Builder Software" on page 1-17.

2. From the client computer, start the Repository Assistant.
   Select Start, Programs, owb home, Warehouse Builder, Administration, then Repository Assistant.

3. Select the Advanced Setup option.

4. Connect to the server that will host the standalone target schema.
   The Repository Assistant displays a dialog as shown in Figure 2–6. Enable Allow the Local Control Center Service and click OK.

5. Follow the prompts in the Repository Assistant.

6. From the client computer, start the Control Center Service.
   Select Start, Programs, owb home, Warehouse Builder, Administration then Start Control Center Service. When prompted, connect to the workspace you previously created. Use the workspace owner user name and password.

7. Start the Control Center Manager to deploy and then run ETL processes on the target schema.
   To start the Control Center Manager, navigate to the Tools menu in the Design Center and select Control Center Manager.

8. Start the Repository Assistant again to create additional target users. (Optional)
Setting the Security Policy for the Repository

When you install a repository, Warehouse Builder enforces a default metadata security policy. The default policy is a minimal security policy appropriate for proof-of-concept or pilot projects. With minimal security, Oracle Database security policies keep data in the design repository secure, and metadata is available to anyone who knows the workspace owner log-on information.

You can change the default by selecting a maximum security policy. Alternatively, you can use the security interface in Warehouse Builder to design your own security policy as described in Oracle Warehouse Builder User’s Guide. In either of these two cases, ensure that the repository database has the Oracle Advanced Security option enabled.

To change the default metadata security policy:

1. Start the Warehouse Builder Design Center.
   
   In Windows, from the Start menu select Programs and navigate to the Oracle product group you installed in the previous step. Select Warehouse Builder and then Design Center.

2. Log in as the workspace owner.

3. From the main menu, select Tools and then Preferences.


5. For the parameter Default Metadata Security Policy, specify the security policy to be applied.
   
   Minimum security enables all users to have full control over objects that any newly registered user creates. However, maximum security restricts access so that only the registered user that created an object and the Warehouse Builder repository administrators have access to an object.

   For a description of the additional security parameters, refer to the online help.

   For instructions on designing your own security policy or for more information about the maximum and minimum security policies, see Chapter 7, "Implementing Security in Warehouse Builder".

---

Note: Only the Repository Assistant can create users for the target schemas. The security interface in the Design Center can only create users that are local to the workspace. Therefore, it cannot be used to create users for the target schemas.
3 Upgrading to Oracle Warehouse Builder 11g Release 1 (11.1)

This chapter includes the following topics about upgrading Warehouse Builder:

- General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1) on page 3-1
- Migrating An Oracle Database Environment on page 3-2
- Upgrading a Design Repository on page 3-7
- Upgrading Existing Control Centers or Runtime Repositories on page 3-11
- Reusing and Redeploying Specific Objects on page 3-16

General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1)

As a general rule, you can upgrade directly to the current release of Warehouse Builder from subsequent releases, beginning with Warehouse Builder 9.2.0.4 and onwards. You may encounter exceptions to this rule if you decide to migrate runtime audit data from early releases, as described in "Using the Control Center Upgrade Assistant" on page 3-12.

To upgrade to Warehouse Builder 11g Release 1 (11.1), refer to the following sections:

1. Understanding the Installation Requirements on page 1-8
   Verify that your environment meets the new minimum requirements for computers hosting client and server components.

2. Understand changes to the product architecture described in "Warehouse Builder Architecture and Components" on page 1-2.

3. Identify and complete any changes necessary to the database environment.
   If you determine that you must upgrade to a new Oracle Database version, then complete the steps in "Migrating An Oracle Database Environment" on page 3-2.
   If you determine that the new control center is to be hosted on a database or server that is different from the existing one, then do not continue with these instructions. Instead, refer to "Selectively Migrating a Warehouse Builder Environment to a New Database" on page 3-4.

   Any instructions in the Release Notes supersede the instructions in this guide.

5. If you want to continue to host the repository on Oracle Database 10g, see "Repository on Oracle Database 10g Release 2" on page 1-5.
   Otherwise, proceed to the next step.

6. Install a new repository.
   From the Windows Start menu, select Programs and navigate to the Oracle product group. Select Warehouse Builder, Administration, and then Repository Assistant.
   On UNIX, locate owb home/owb/bin/unix and run reposinst.sh.
   Follow the instructions Chapter 2, "Managing Workspaces and Workspace Users".
If in the pre-existing repository you created multiple users through the optional PL/SQL package, then take note of the instructions for "Selecting Workspace Users" on page 2-5.

7. **Upgrading a Design Repository** on page 3-7
8. **Upgrading Existing Control Centers or Runtime Repositories** on page 3-11
9. **Reusing and Redeploying Specific Objects** on page 3-16

To reuse most objects designed in a previous release, you need to take additional steps.

10. **Installing and Enabling Optional Components** (Optional) on page 5-1
11. **Installing the Warehouse Builder Software** on the Client computers on page 1-17

Repeat the instructions **Installing the Warehouse Builder Software** on page 1-17 for each computer to be used as a client.

12. When complete the installation process, you can start all the components as described in "Launching Warehouse Builder Components" on page 1-19.

13. **Deinstalling Oracle Warehouse Builder** (Optional) on page 4-1

You have the option of deinstalling the existing components from the previous release. Or, you can deinstall pre-existing components at a later date.

---

**Migrating An Oracle Database Environment**

If your current version of Oracle Database is compatible with Warehouse Builder, then you can keep the current version or optionally choose to upgrade to a higher compatible version of the database. You can migrate your Oracle environment to a different instance of Oracle Database, or you can continue on the same instance.

**Before You Begin**

Before you upgrade the Oracle Database, stop the Control Center Service if it is running.

Log on to the host as the repository owner. Run the `owb home\rtp\sql\stop_service.sql` script.

Note: In Warehouse Builder 10.1 and earlier versions, the Control Center Service was known as the Runtime Platform Service. The `stop_service.sql` is located in the same directory.

**Upgrading to Oracle Database 11g (11.x)**

If you intend to upgrade to Oracle Database 11g while maintaining the same database instance, then all relevant instructions are detailed in the Oracle Database 11g Upgrade Guide. Continue with step 4 in "General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1)" on page 3-1.

If you intend to create a new database instance, then you must take the additional steps of either

"Migrating a Complete Database" (Recommended)  

or  

"Selectively Migrating a Warehouse Builder Environment to a New Database" on page 3-4
Migrating a Complete Database

If you are upgrading from Warehouse Builder 10g Release 2 (10.2), you can migrate a complete Oracle Database as described in the upgrade documentation for Oracle Database 11g.

If you want to migrate an Oracle Workflow Schema, then see "Migrating an Oracle Workflow Schema" on page 3-3.

Migrating an Oracle Workflow Schema

Oracle Workflow Schemas Used in Warehouse Builder 10g Release 2

Take the following precautions now, that is, prior to upgrading runtime metadata.

To upgrade Oracle Workflow on a new Oracle Database instance:

1. In the new Database instance, locate and run the Oracle Workflow installation software.

   For Oracle Database 11g, the installation software is located at owb home/wf/install. Use wfinstall.csh for UNIX or wfinstall.bat for Windows platforms.

2. Use Oracle Database export and import utilities to move an Oracle Workflow schema from one database to another.

   This may result in invalid PL/SQL packages due to missing privileges. To resolve this, use the scripts owf_grants.sql and tsupgrade_compile_pkg.sql. Both scripts are at available at owb home/owb/rtasst/upgrade.

   Logon as SYS and use owf_grants.sql to grant the privileges.

3. Continue with steps 4 through 6 in "General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1)" on page 3-1.

   Following the general steps, you install the Warehouse Builder software, install a repository, and register users for Warehouse Builder. When prompted to register user, ensure that the OWF_MGR user is registered as a user of your workspace.

4. Continue with "Upgrading a Design Repository" on page 3-7 followed by "Upgrading Existing Control Centers or Runtime Repositories" on page 3-11.

5. The final step is "Reusing Process Flows and Schedules" on page 3-18.

   Manually re-register the workflow locations and redeploy the process flow packages from within the new version of Warehouse Builder.

Oracle Workflow Schemas Used in Warehouse Builder 10g Release 1 and Prior

After you move an Oracle Database instance, Warehouse Builder process flows remain registered to the Oracle Workflow installed on the pre-existing Oracle Database instance. This condition occurs only when the Warehouse Builder installation from which you are upgrading is Warehouse Builder 10g Release 1 (10.1) or prior.

Take the following precautions now, that is, prior to upgrading runtime metadata.

To upgrade Oracle Workflow on a new Oracle Database instance:

1. In the new Database instance, locate the Oracle Workflow installation software.

   For Oracle Database 11g, the installation software is located at owb home/wf/install. Use wfinstall.csh for UNIX and wfinstall.bat for Windows platforms.
2. Run the Oracle Workflow assistant in **Upgrade** mode on the new database instance.

   The Oracle Workflow schema is now upgraded. However, in Warehouse Builder, the associated locations remain registered to the pre-existing instance of Oracle Workflow.

3. Continue with steps 4 through 6 in "General Steps for Upgrading to Warehouse Builder 11g Release 1 (11.1)" on page 3-1.

   Following the general steps, you install the Warehouse Builder software, install a repository, and register users for Warehouse Builder.

4. Register the Workflow user in the new repository.

   Start the new Design Client and navigate to the security interface. In the **Global Explorer** on the lower right window, expand the **Security** node and right-click the **Users** node to create a new user. Add a new user with the same name as your workflow user in the pre-existing database from which you are migrating. Deselect the user as target.

5. Continue with "Upgrading a Design Repository" on page 3-7 followed by "Upgrading Existing Control Centers or Runtime Repositories" on page 3-11.

6. The final step is "Reusing Process Flows and Schedules" on page 3-18.

   Manually re-register the workflow locations and redeploy the process flow packages from within the new version of Warehouse Builder.

### Selectively Migrating a Warehouse Builder Environment to a New Database

Use this option to selectively move a Warehouse Builder environment from one Oracle Database to another. You must use this option if the new control center is to be hosted on a database or server different from the existing runtime repository or control center.

As this is the most challenging migration and upgrade scenario, avoid this scenario if possible. Do not use this option if you intend to either keep the same database instance or migrate the full database.

#### Steps for Migrating Warehouse Builder to an Oracle Database 11g

1. Review the *Oracle Warehouse Builder Release Notes*.

   Any instructions in the Release Notes supersede the instructions in this guide.

2. Install a new repository on Oracle Database 11g.

   From the Windows **Start** menu, select **Programs** and navigate to the Oracle product group. Select **Warehouse Builder, Administration**, and then **Repository Assistant**.

   On UNIX, locate `owb home/owb/bin/unix` and run `reposinst.sh`.

3. **Upgrading a Design Repository** on page 3-7

4. **Exporting Target Schemas from the Existing Runtime Environment**

5. **Creating the Target Schemas and Users in the New Database**

6. **Copying External Directory References to the New Database Instance** (Optional)

7. **Importing Target Schemas to the New Database**

8. **Upgrading Existing Control Centers or Runtime Repositories** on page 3-11

9. Take manual steps to reuse specific types of objects.
If the existing Warehouse Builder environment included flat files and external tables, then "Reusing Flat Files and External Directories from a Different Database Instance" on page 3-18.

See Reusing and Redeploying Specific Objects on page 3-16 for additional steps to reuse certain objects such as Advanced Queues that you designed in a previous release.

10. Installing and Enabling Optional Components (Optional) on page 5-1

See Chapter 5 for instructions on installing and configuring optional components.

11. Installing the Warehouse Builder Software on the Client computers on page 1-17

Repeat the instructions Installing the Warehouse Builder Software on page 1-17 for each computer to be used as a client.

12. When complete the installation process, you can start all the Warehouse Builder components as described in "Launching Warehouse Builder Components" on page 1-19.

13. Deinstalling Oracle Warehouse Builder (Optional) on page 4-1

You have the option of deinstalling the existing components from the previous release. Or, you can deinstall pre-existing components at a later date.

Exporting Target Schemas from the Existing Runtime Environment

Pre-create the tablespaces in the new Database environment to exactly match the tablespaces in the existing version of Oracle Database.

1. Use Oracle Export in the existing version of the Oracle Database to export the existing target schemas into a DMP file with the following command for each schema:

   exp $OWBTargetUserName/$OWBTargetUserPassword@Old_DBTNSConnection Owner=$OWBTargetUserName FILE=OldOWBTarget.dmp LOG=OldOWBTarget.log

   $OWBTargetUser stands for the Warehouse Builder target schema user from the existing version of Warehouse Builder.

   For example, type:

   exp owb_target/owb_target owner=owb_target FILE=owb_target.dmp LOG=owb_target.log

2. Identify all the tablespaces for each of the existing Warehouse Builder target schema users.

   Connect to SQL*Plus in the existing version of the Oracle Database as the Warehouse Builder target schema user, and then enter the following command:

   select distinct TABLESPACE_NAME from user_segments;

   Enter the following to check the default and temporary tablespaces for existing Warehouse Builder target schema users:

   select DEFAULT_TABLESPACE, TEMPORARY_TABLESPACE from user_users;

Creating the Target Schemas and Users in the New Database

The steps you take here depend on the version from which you are upgrading.
When migrating from Warehouse Builder 9.2 or 10.1, complete the following instructions:

1. In the new Database instance, create the tablespaces you listed from the existing instance in “Exporting Target Schemas from the Existing Runtime Environment”.

2. In the new Database, connect as a SYS user to SQL*Plus to create each target schema and grant privileges to it.

   For each target schema you create, enter the following commands in SQL*Plus:
   
   ```sql
   connect SYS/SYS as sysdba;
   create user OldOWBTargetSchemaUser identified by OldOWBTargetSchemaPassword default tablespace users temporary tablespace temp;
   SET DEFINE %
   define user=OldOWBTargetSchemaUser
   @new owb home\owb\rtasst\upgrade\preowb10_2\warehouse_system_rights.sql
   @new owb home\owb\rtasst\upgrade\preowb10_2\xmltk_grant.sql
   ```

When migrating from Warehouse Builder 10.2, complete the following instructions:

1. In the new Database instance, create the tablespaces you listed from the existing instance in “Exporting Target Schemas from the Existing Runtime Environment”.

2. In the new Database, connect as a SYS user to SQL*Plus to create each target schema and grant privileges to it.

   For each target schema you create, enter the following commands in SQL*Plus:
   
   ```sql
   connect SYS/SYS as sysdba;
   create user OldOWBTargetSchemaUser identified by OldOWBTargetSchemaPassword default tablespace users temporary tablespace temp;
   SET DEFINE %
   define user=OldOWBTargetSchemaUser
   @new owb home\owb\rtasst\upgrade\owb10_2\warehouse_system_rights.sql
   ```

Copying External Directory References to the New Database Instance

Complete this section if in the pre-existing Warehouse Builder environment includes external directories which are used by external tables and flat files.

External directories have two elements: the logical and the physical. The logical element is the reference residing in the database to a directory located outside the database. These instructions migrate the logical elements. You migrate the physical elements in a later step subsequent to upgrading the runtime environment.

To migrate the external directories for each target schema users:

1. Create a copy of the script gen_ext_dirs.sql.

   Locate owb home\owb\mig\gen_ext_dirs.sql on the new host, that is, the computer hosting the new Warehouse Builder installation.
Copy the file to a temporary directory on the original host, that is, the computer hosting the pre-existing version of Warehouse Builder from which you are migrating.

2. In SQL*Plus, connect as a Warehouse Builder target schema user and run TEMP\gen_ext_dirs.sql on the original host computer.

3. Locate the ext_dirs.sql file generated in the SQL*Plus default directory. Typically, this default directory is owb home\bin.

4. Rename the generated script.
   As you complete these instructions, you generate a separate script for each Warehouse Builder target schema use. Consider renaming the file to indicate the target schema.

5. Transfer the generated and renamed ext_dirs_TargetUserName.sql file to a temporary location on the new host.

6. On the new host, use SQL*Plus to connect as the Warehouse Builder 11g Release 1 (11.1) target schema user and run ext_dirs.sql.

7. Repeat steps 2 through 6 for each target schema user you intend to migrate.

Importing Target Schemas to the New Database

Use Oracle Import to import the target schema files into the new user you created.

To import a target schema:

1. To import the target schema DMP file you created in "Exporting Target Schemas from the Existing Runtime Environment" on page 3-5, enter the following command:
   imp OldOWBTargetUserName/OldOWBTargetUserPassword@New_DBTNSConnection FILE=OldOWBTarget.dmp LOG=NewOWBTarget.log

   OldOWBTargetUser stands for the Warehouse Builder target schema user from the existing version of Warehouse Builder.

   For example, enter:
   imp owb_target/owb_target@New10gConnection FILE=owb_target.dmp LOG=c:\temp\owb_target_import.log

2. Examine the import log file, whose name and location you specified in the import command.
   Proceed to the next step only if the last line of the log file indicates a successful completion.
   If the last line of the log file indicates an unsuccessful completion, you must fix all import errors before proceeding.

3. Repeat these instructions for each target schema you want to migrate.

Upgrading a Design Repository

You have the following options when upgrading the design metadata in a repository:

- Migrating All Design Metadata from Warehouse Builder 10g Release 2 (10.2)
- Selectively Migrating Design Metadata
Migrating All Design Metadata from Warehouse Builder 10g Release 2 (10.2)

Follow these instructions if you want to export the entire design metadata from repository in Warehouse Builder 10g Release 2 and import the entire design metadata into the current release.

Exporting All Design Metadata from Warehouse Builder 10g Release 2 (10.2)

In these instructions, you modify and run a tcl script using Warehouse Builder 10g Release 2 (10.2). The script exports the entire design metadata from a Warehouse Builder 10.2 repository.

All the design metadata is exported as a single unit.

To export all design metadata:

1. Copy the tcl script from the home directory of the current release into a temporary directory.
   Locate `ExportEntireRepos.tcl` in `owb 11.1 home/owb/bin/upgrade/`.

2. Use a text editor to update the copy of `ExportEntireRepos.tcl` that you saved in a temporary directory.
   Specify the Warehouse Builder 10g Release 2 repository owner for the connection information. This includes the repository owner user name, password, host, port and service.
   Specify the export and import directory. These directories must specify the same directory location. The export and import directory will contain the MDL data files, log files and generated scripts.

3. Make sure no one is logged into the Warehouse Builder 10g Release 2 repository.
   The export script requires exclusive access to the repository for the upgrade process.

4. Run OMB Plus using the tcl script.
   For Unix, type the following command on the same line:
   ```bash
   owb 10.2 home/owb/bin/unix/OMBPlus.sh temp directory/ExportEntireRepos.tcl
   ```
   For Windows, type the following command on the same line:
   ```bash
   owb 10.2 home\owb\bin\win32\OMBPlus.bat temp directory\ExportEntireRepos.tcl
   ```

5. Verify that the export completed successfully by reviewing `ExportEntireRepos.log` located in the export directory.

Importing All Design Metadata into the Current Release

In these instructions, you modify and run a tcl script. The script imports the entire design metadata as a single unit from Warehouse Builder 10.2 into a Warehouse Builder 11.1 repository.

To import all design metadata:

1. If you have not already done so, create a workspace and a workspace owner in the new release.
   Refer to Chapter 2, "Managing Workspaces and Workspace Users" for additional information.
2. If the new repository resides on the same Database as the previous repository, skip to the next step.

If the new repository resides on a different Database from the previous repository, then create the same Warehouse Builder users from the previous repository in the new database, e.g., use SQL*Plus.

SQL> CREATE USER owb User Name IDENTIFIED BY password DEFAULT TABLESPACE debasing...

Refer to Oracle SQL Language Reference documentation for details for CREATE USER command.

Note: This step is important if you want to migrate security information into Warehouse Builder 11g Release 1. The upgrade process creates the Warehouse Builder users if the database users exist. So there is no need to create the Warehouse Builder users explicitly.

3. Use a text editor to update the script in owb 11.1 home/owb/bin/upgrade/ImportEntireRepos.tcl.

Specify the Warehouse Builder 11g Release 1 workspace owner and workspace. For the connection information, include the workspace owner user name, password, host, port and service.

Specify the export and import directory. The export and import directory must be the same directory specified in ExportEntireRepos.tcl. The import directory will contain the log files of the import.

4. Run OMB Plus using the tcl script. Enter the commands on the same line.

For UNIX, type the following command on the same line:

owb 11.1 home/owb/bin/unix/OMBPlus.sh
owb 11.1 home/owb/bin/upgrade/ImportEntireRepos.tcl

For Windows, type the following command on the same line:

owb 11.1 home\owb\bin\win32\OMBPlus.bat
owb 11.1 home\owb\bin\upgrade\ImportEntireRepos.tcl

5. Verify that the Import completed successfully by reviewing ImportEntireRepos.log located in the import directory.

Selectively Migrating Design Metadata

Follow these instructions if you want to migrate only a portion of an existing repository. For example, use these instructions to export and import selected projects or collections.

Create a full database backup before you begin. Additionally, create metadata export (MDL) files for all Warehouse Builder projects. Keep these backups until you have completed and tested the entire upgrade process.

These instructions apply whether or not you upgraded your Oracle Database.

1. Export design metadata from the existing version of Warehouse Builder into an MDL file using the Metadata Loader.

2. Use the new version of Warehouse Builder to create a new repository.

3. Import design metadata into the new repository.
Exporting Design Metadata from a Prior Release of Warehouse Builder

Export each project, collection, or public transformation you want to migrate to Metadata Loader (MDL) files using the Metadata Loader. If you have created any user-defined definitions, then export these objects too. For more information on exporting metadata, see Oracle Warehouse Builder User’s Guide.

Note: You must export and import metadata using the Metadata Loader. Warehouse Builder upgrade does not support files that were exported or imported using back end database commands.

To export existing metadata into an MDL file:

1. Use the prior version of the Warehouse Builder client to select the project, collection, or public transformation you want to export.

   For information about exporting user-defined definitions, refer to Oracle Warehouse Builder User’s Guide.

2. From the Project menu, choose Export Metadata, then File.

   The Metadata Loader assigns a path and file name to the exported MDL file. Make a note of the path and filename for all data you export. For more information on exporting metadata, refer to Oracle Warehouse Builder User’s Guide.

Importing Design Metadata to Warehouse Builder

After having installed the new software, you must import and upgrade design metadata into the new version of Warehouse Builder. Ensure that you first import custom public transformations, if any. Use the Metadata Import utility to import design metadata. For more information on importing metadata, see Oracle Warehouse Builder User’s Guide.

Note: Warehouse Builder upgrade does not support files that were exported or imported using back end database commands.

To import and upgrade metadata into the new Warehouse Builder repository:

1. From the new Warehouse Builder Design Center, select the Design menu, Import, and Warehouse Builder Metadata.

   The Metadata Import dialog is displayed.

2. In the File Name field, specify the path and file name of the exported data from the former repository.

3. In the Log File field, specify the path and file name of the log file or click Browse to locate a directory and file name. Warehouse Builder records information about the import in this log file.

4. In the Import Option section, select the import option used while importing metadata. The options available are:

   - Create new metadata only: Adds new metadata to a repository.

   - Update metadata (replace existing objects and create new metadata): Adds new objects and replaces existing objects with those in the MDL file.

   - Merge metadata (merge existing objects and create new metadata): Adds new objects and overwrites existing objects only if they are different from those in the MDL file. Existing objects are not deleted.

   - Replace existing objects only: Replaces existing objects in the repository.

5. In the Match By section, select Universal Identifier.
6. (Optional) If the MDL file contains additional languages or user-defined definitions, click the Advanced button to select the options that include them in the import.

Because the MDL file being imported was created using an earlier version of Warehouse Builder, the Metadata Upgrade dialog is displayed. Click Upgrade to upgrade the MDL file to the current version. Click Cancel if you do not want to upgrade the MDL file.

If the MDL file is upgraded, then the Import Advanced Options dialog displays. Use this dialog to import the following:

- Additional language metadata
- User-defined definitions

Click OK to save your selections and close the Import Advanced dialog. For more information about the advanced import options, refer Oracle Warehouse Builder User’s Guide.

7. (Optional) To view a detailed summary of the contents of the export MDL file, click Show Summary.

Because the MDL file being imported was created using an earlier version of Warehouse Builder, the Metadata Upgrade dialog is displayed. Click Upgrade to upgrade the export MDL file to the current version. Click Cancel if you do not want to upgrade the MDL file.

If the MDL file is upgraded, the Show Summary dialog is displayed. This dialog provides a brief summary of the contents of the export MDL file.

8. Click Import to import the MDL file.

If the MDL file was not previously upgraded in step 6 or step 7, the Metadata Upgrade dialog is displayed. Click Upgrade to upgrade the export MDL file to the current version. Click Cancel if you do not want to upgrade the MDL file.

If you click Upgrade, then the Metadata Progress dialog displays the progress of the upgrade and import operation. After the upgrade completes, click Close to return to the Design Center.

In prior versions of Warehouse Builder, locations and runtime repository connections were owned by individual projects. Beginning in Oracle Warehouse Builder 10g Release 2, locations and connections are owned by a project called PUBLIC_PROJECT. If locations or runtime repository connections with the same names as the ones being upgraded exist in the repository, then Warehouse Builder generates unique names when they are imported for the upgrade. You may need to manually clean up location associations after the upgrade is complete.

For more information about the changes made to repository objects after an upgrade and import operation, see Oracle Warehouse Builder User’s Guide.

### Upgrading Existing Control Centers or Runtime Repositories

In previous releases, runtime repositories were managed by the interface known as the Deployment Manager. Beginning in Warehouse Builder 10.2 release, the Control Center Manager replaced the Deployment Manager. The term runtime repository from previous releases is replaced by the term control center.

In previous releases, you could have multiple runtime repositories or control centers associated with a single Warehouse Builder installation. Beginning with Warehouse Builder 11g Release 1 (11.1), a repository is owned by a database user called OWBSYS.
This repository can contain one or more workspaces. Each Workspace is equivalent to a repository from any previous release. That is, it can contain design metadata, runtime audit data or both. Therefore, when upgrading to this release, the upgrade assistant re-creates each runtime repository or control center as a workspace within the new, unified repository.

To upgrade existing control centers or runtime repositories, start the Control Center Upgrade Assistant.

For Windows, start owb home\owb\bin\win32\cc_migrate.bat.
For UNIX, start owb home\owb\bin\UNIX\cc_migrate.sh.

**Using the Control Center Upgrade Assistant**

Use the Control Center Upgrade Assistant to move audit data from a runtime repository or control center that you created in a previous release. After you use the Control Center Upgrade Assistant, use the Control Center Manager to manage runtime repositories, now referred to as control centers.

The Control Center Upgrade Assistant directs you through the steps needed to migrate an existing Warehouse Builder environment into one that can be further designed and managed from the current release of Warehouse Builder. These steps involve migrating audit-data into a workspace and upgrading target schemas to fulfill requirements in the current release.

If you want upgrade your Oracle Database, do so before running this assistant. Also, ensure that target schemas that contain PL/SQL mappings are in the same database instance as their owning Control Center.

**Supported Upgrade Scenarios**

Use this assistant to complete any of the scenarios illustrated in Figure 3–1. You can migrate audit data directly to Warehouse Builder release 11.1 from Warehouse Builder releases 10.2, 10.1.0.5, and 10.1.0.4.

**Figure 3–1 Valid Paths for Migrating and Upgrading Runtime Audit Data**

| Release 10.2 | → | Release 11.1 |
| Release 10.1.0.5 | → | Release 11.1 |
| Release 10.1.0.4 | → | Release 11.1 |
| Release 10.1.0.2 | → | Release 10.1.0.2 | → | Release 11.1 |
| Release 9.2.0.4 | → | Release 10.1.0.2 | → | Release 10.2 | → | Release 11.1 |

**Connecting to a New Control Center**

Connect to the repository created in the Oracle Database when you recently installed the latest version of this product.

Connect to the repository as the repository owner.
Choose a Workspace
From the newly created repository, select the workspace you created during the installation process.

Connecting to an Existing Runtime Repository Or Control Center
As you use the Control Center Upgrade Assistant, the assistant prompts you for the connection details to the previous runtime repository or control center.

By default, the assistant assumes that the new control center and existing one share the same host name, port number, and Oracle service name. This is the case for the most common upgrade scenarios including the following scenarios:

- Since the time you created the runtime repository or control center, you did not upgrade the Oracle Database to a new version.
- You did upgrade the Oracle Database but performed a full database migration such as described in "Migrating a Complete Database" on page 3-3.

If you want to upgrade to a control center on a different host or database other than the existing one, then complete the steps in "Selectively Migrating a Warehouse Builder Environment to a New Database" on page 3-4, start the Control Center Upgrade Assistant again, and then enter the correct connection information.

Choosing An Upgrade Operation
Use the upgrade operations in the following order:

1. Select Move to transfer location registration information and audit data from the pre-existing runtime repository or control center to the new control center.
   After you successfully move location registration information, you access the other options for Upgrade and Generate.

2. Select Upgrade to upgrade the locations details that you previously moved for use in the new control center.

3. Select Generate to create a Tcl script that you can later apply to update a design repository.


Move
When you select the Move operation, the assistant prompts you to connect to the original runtime repository or control center as a repository owner.

In this step, you upgrade runtime audit data so that the new control center displays the correct status, history, location details, and version numbers for objects you deployed and ran in the runtime repository or control center from the previous release.

This option is only available if the new control center does not contain any registered locations or audit data.

Upgrade
The assistant lists the locations registered in the new control center and displays whether they are valid or not. Figure 3–2 displays the Upgrade Registered Locations dialog.
Upgrading Existing Control Centers or Runtime Repositories

Figure 3–2  Upgrade Registered Locations Dialog

[Image]

Upgrade Operation: Location Details
Select a location and use this button to check and fix locations for possible errors. Ensure that all locations are valid before proceeding. Figure 3–3 displays the Registered Locations Details dialog.

Figure 3–3  Registered Locations Details Dialog

[Image]

This dialog box displays the registration details of the currently selected location. Select Get Status to evaluate its status and view the results of various tests.
For each location, the assistant checks that the password is valid and that it is the same as the registered password. The assistant verifies that the version registered for the location is the same as the actual version. If the assistant detects any errors, then you can select **Upgrade** to correct the errors.

**Generate**

Use this operation to generate a TCL script of OMB Plus commands that you can use to alter the location information stored against locations in the design repository. Such information was not stored against locations in Warehouse Builder 10.1.x. Also, depending on your migration path, it may have been altered as part of migration.

**To generate a Tcl script that updates location information:**

1. Ensure that you previously imported MDL from the previous design repository to the new Warehouse Builder 11.1 repository as described in "Upgrading a Design Repository" on page 3-7.

2. Click **Generate**.

3. Edit the script to change the value of the variable **CC_NAME**.
   
   Either edit the script in the user interface and select save again. Or edit the script in Notepad.

   Set **CC_NAME** to the name of the control center object in the new design repository. For example, assume that you migrated a runtime repository called **MY_DEV_RUNTIME** from a previous of Warehouse Builder. Therefore, in the Tcl script, change the line
   
   `CC_NAME cc_name`
   
   to
   
   `CC_NAME MY_DEV_RUNTIME`

4. Save the edited script in a temporary directory.

5. Click OK in the Generate dialog and complete the Control Center Upgrade Assistant.


**Reviewing Selections in the Upgrade Assistant**

Verify your selections in the assistant before clicking **Finish**.

On the summary page, control center refers to the new control center to which you are upgrading. Runtime repository refers to the repository from which you are upgrading.

**Upgrading Locations in the Design Repository**

1. After you generate a Tcl script using the "Generate" on page 3-15, use an OMB Plus session to connect to the new design repository using the OMBCONNECT command.

   For example, enter:

   ```
   OMBCONNECT <repos>/<password>@<host>:<port>:<service_name>
   USE WORKSPACE <wokspace_name>
   ```

   For more information on OMB Plus commands, see the Oracle Warehouse Builder Scripting Reference.
2. Run the Tcl script against the new design repository.
   For example, enter at the OMB Plus prompt:
   
   ```
   source/temp_path/my_generated_script.tcl
   ```
   
   When you run the script, the details for the control center and the location address are seeded into the design repository. Each location is associated with the control center. The registration details are also added to the logical location because, beginning in Warehouse Builder 11.1, these details can be stored in the Design Center.

3. For each project that you migrated, ensure that a configuration object uses the control center object you specified for the `CC_NAME` variable previously in these instructions.

   In the Design Center, expand the project node, and expand the `Configurations` node. Right-click DEFAULT_CONFIGURATIONS and select **Open Editor**.

   On the Name page, enable the Set and Save option.

   On the Details tab, select the incoming control center which is the same control center you set for `CC_NAME`.

4. Save the changes you made in the Design Center by selecting **Save All** from the Design menu.

5. To view and re-deploy the migrated objects, start the Control Center Manager from the Tools menu.

6. Register all of the locations and provide any passwords.

   For security purposes, the location passwords are not saved. In the Control Center Manager, right click each location and select **Register**. Enter the password and optionally test the connection.

---

**Reusing and Redeploying Specific Objects**

As a general rule, when migrating from the previous release to Warehouse Builder 11g, you can run migrated mappings immediately without redeploying them in the new environment. For all other jobs, however, you must take further action before running jobs for the first time. The objects that require further action include, but are not limited to the following: dimensions and cubes, Discoverer integration, process flows, schedules, and data profiles.

This section includes the following additional instructions for reusing objects you created in a previous release:

- Redeploying Dimensions and Cubes
- Reusing Advanced Queues
- Reusing Oracle Workflow Locations
- Reusing Process Flows and Schedules
- Reusing Flat Files and External Directories from a Different Database Instance
- Reusing Data Profiles
Reusing and Redeploying Specific Objects

Redeploying Dimensions and Cubes

**Migrating Type 2 Slowly Changing Dimensions**
Beginning in Warehouse Builder 11g, hierarchy versioning is available and enabled as a default for type 2 slowly changing dimensions. If you migrate a type 2 slowly changing dimension, verify that the settings are set correctly after migration. For these dimensions to behave in the same way as originally designed in a previous release, you must first deselect hierarchy versioning and then deploy the dimension.

To disable hierarchy versioning, go to the Type 2 Slowly Changing Dimension Settings tab. Each level is listed together with its level attributes. With this new version of Warehouse Builder, each level except the highest has an attribute representing the parent ID. The name of this attribute is similar to the parent level name and has the suffix "_.ID".

If **Record History** for this ID is set to **Trigger History**, then hierarchy versioning is turned on for this level. To turn off hierarchy versioning, remove the entry from **Record History** and leave it blank.

**Migrating Dimensions from Warehouse Builder 10g Release 2**
When you migrate a target schema from Warehouse Builder 10g Release 2 (10.2), you must redeploy any dimensions that used a ROLAP implementation to the OLAP catalog. To do this, you set the configuration property called **Deployment Options** to **Deploy to Catalog Only** and deploy the dimensions.

**Migrating Dimensions and Cubes from Warehouse Builder 10g Release 1**
Warehouse Builder 10g Release 2 (10.2) introduced significant changes to the logical model for dimensions and cubes. After upgrading from Warehouse Builder 10g Release 1 or prior, dimensions and cubes appear as new objects in the Control Center Manager.

New validation rules may cause errors or warnings that did not exist in your previous installation. This is expected and does not indicate problems with the migrated data. Validate the cubes and dimensions and fix errors that may prevent you from redeploying the objects.

Redeploy the objects as this is necessary for updating the audit history.

**Reusing Advanced Queues**
In a previous release you may have created a mapping with Advanced Queues. Only if you intend to redeploy such a mapping in this release, you must first create a separate table for each AQ with the following structure:

```
PAYLOAD SRC_TYPE107,
MSG_ID RAW(16),
CONSUMER_NAME VARCHAR2(30),
MSG_ORDER NUMBER,
CORR_ID VARCHAR2(128),
MSG_PRIORITY NUMBER
```

**Reusing Oracle Workflow Locations**
Re-register any Oracle Workflow locations created in a previous release.
Reusing Process Flows and Schedules

**Process Flows and Schedules from Warehouse Builder 10g Release 2**
Re-register the workflow locations and then select one process flow package to redeploy in the new environment. Preferably, redeploy a small process flow package with a single, simple process flow. Subsequently, other process flows created in this release can run without redeployment.

For schedules, redeploy all schedules from previous releases.

**Process Flows and Schedules from Warehouse Builder 10g Release 1 and Prior**
For process flows and schedules from release Warehouse Builder 10g Release 1 and earlier, you need redeploy each object separately in the new environment.

Reusing Flat Files and External Directories from a Different Database Instance
Complete the instructions in this section only if both of the following are true:

- If you migrated the Oracle Database by "Selectively Migrating a Warehouse Builder Environment to a New Database" on page 3-4.
- AND
- If you had flat files or external tables in the existing Warehouse Builder environment

If both of these points are true, you must copy the following objects from the computer hosting the existing instance of Oracle Database to the computer hosting the new instance:

- **Flat Files**: Copy any flat files used by SQL*Loader from the computer hosting the existing instance of Oracle Database to the computer hosting the new instance.

- **External Directories**: You must also copy all external directories from the computer hosting the existing instance of Oracle Database to the computer on which the new Database resides. Make sure to re-create identical file system directories.

Reusing Data Profiles

Migrating data profiles that you created in a previous release is not supported in Warehouse Builder 11g Release 1. You must manually recreate any data profile metadata you created in a previous release.
4 Deinstalling Oracle Warehouse Builder

This chapter contains the following topics for deinstalling Oracle Warehouse Builder components:

- General Steps for Deinstalling Oracle Warehouse Builder on page 4-1
- Deleting the Workspace Users on page 4-1
- Deleting the Workspace Owner on page 4-2
- Deinstalling the Oracle Warehouse Builder Software on page 4-3
- Deleting the Schema Objects (Optional) on page 4-3
- Deleting a Repository from an Oracle 10g Database on page 4-3

General Steps for Deinstalling Oracle Warehouse Builder

The steps you take to deinstall Warehouse Builder depends on if you want to remove only the client components from a computer or remove all server and client components from your environment.

To remove a client installation, follow the instructions in "Deinstalling the Oracle Warehouse Builder Software" on page 4-3. If you want to deinstall all Warehouse Builder components including the repository, then you must have SYSDBA privileges to the repository database. Follow the order of steps presented in this chapter to avoid the necessity of deleting components manually through a utility such as SQL Plus.

If you are deinstalling multiple or all components, then follow the order presented in this chapter.

To deinstall all components, complete the following steps:

1. **Deleting the Workspace Users** on page 4-1
   - Use the Advanced Setup option in the Repository Assistant to delete one or more users.
2. **Deleting the Workspace Owner** on page 4-2
   - Use the Advanced Setup option in the Repository Assistant to delete the workspace owner.
3. **Deinstalling the Oracle Warehouse Builder Software** on page 4-3
   - Start the Oracle Universal Installer to deinstall the software components.
4. **Deleting the Schema Objects (Optional)** on page 4-3
5. **Deleting a Repository from an Oracle 10g Database**

Deleting the Workspace Users

Before you can deinstall a workspace owner, you must first delete the associated workspace users. When you delete a workspace user, you unregister and remove the user from the repository. Deleting the user from the Warehouse Builder repository does not delete or alter the user account in the Oracle Database.

To delete workspace users:
Deleting the Workspace Owner

After deleting the workspace users, you can delete the workspace owner. When you delete a workspace owner, you unregister and remove the owner from the repository. Deleting the owner from the repository does not delete or alter the owner account in the Oracle Database.

To delete the workspace owner:

1. Start the Repository Assistant and navigate to the Choose Operation page.

   Repeat steps 1 through 3 in "Deleting the Workspace Users" on page 4-1.

2. In the Choose Operation page, select Manage a Warehouse Builder workspace owner option.

3. Click Next.

4. In the Manage Workspace Owner page, select Delete an existing Warehouse Builder Workspace owner option.

5. Click Next. The Workspace Owner Information page is displayed.

6. Enter the Workspace Owner Password.

7. On the Summary page, review your selections and click Finish.

   The Deinstallation Successful page appears once the workspace owner is deleted.
Deinstalling the Oracle Warehouse Builder Software

To deinstall the Oracle Warehouse Builder:

1. Start the Oracle Universal Installer.
   - For Windows, select Start, Programs, owb home, Oracle Installation Products, and then click Universal Installer.
   - For UNIX, locate owb home/oui/bin and run runInstaller.sh.

2. In the Oracle Universal Installer: Welcome page, click Deinstall Products.

3. In the Inventory page, on the Contents tab, in the You have the following Oracle products installed box, select the Oracle Warehouse Builder home.

4. Click Remove.

5. In the Confirmation page, click Yes to deinstall the Oracle Warehouse Builder. The deinstallation process begins.

6. After the deinstallation completes, click Close in the Inventory page.

7. In the Oracle Universal Installer: Welcome page, click Cancel to close the Oracle Universal Installer page.

Deleting the Schema Objects (Optional)

When you delete a workspace user or the workspace owner, you unregister and remove the owner from the repository. Deleting the user or owner from the Warehouse Builder repository does not delete or alter the owner account in the Oracle Database.

If you determine that it is necessary, then you can use Oracle Enterprise Manager to drop the workspace users, workspace owners, and the Warehouse Builder-related roles and synonyms permanently from Oracle Database.

Note: To grant permission to an OWB repository user to use Enterprise Manager for performing tasks, enter the following command in SQL*Plus:

```
GRANT SELECT any dictionary to "&OWB repository user";
```

Deleting a Repository from an Oracle 10g Database

You can delete a Oracle Warehouse Builder 11g repository and associated objects from a 10g Release 2 database using the SQL script `clean_owbsys.sql`. The script is stored in the owb home/owb/UnifiedRepos directory.

To delete an 11g Warehouse Builder repository from a 10g Release 2 database:

1. Change your working directory to owb home/owb/UnifiedRepos. For example:
   
   C:> cd owb home

2. Start SQL*Plus with SYSDBA privileges.

   Use the version of the SQL*Plus executable provided with Warehouse Builder 11g Release 1 (11.1). This executable is located in the owb home/bin directory.

   For example, enter the following command:

   ```
   C:>owb home\bin\sqlplus sys/sys_password as sysdba;
   ```
3. Run the `clean_owbsys.sql` script. For example:

```
SQL> @clean_owbsys.sql
```

The `OWBSYS` user and Warehouse Builder-related roles are dropped from the 10g Release 2 Database.

If you used Repository Assistant to create a workspace, a workspace owner DB user, and workspace user DB users, these objects will still exist in the database after you run `clean_owbsys.sql`. 
5 Installing and Enabling Optional Components

This chapter includes the following topics:

- Enabling Integration with Oracle E-Business Suite on page 5-1
- Configuring Repository Browser Environments on page 5-2
- Enabling Integration with Third-Party Name and Address Data Libraries on page 5-2
- Enabling Integration with Oracle Workflow on page 5-3

Enabling Integration with Oracle E-Business Suite

Warehouse Builder employs a design-deploy-run model as an ETL solution. To integrate with Oracle E-Business Suite (EBS), Warehouse Builder users must import metadata from EBS before designing mappings to move and transform data. Specifically, during the design phase, Warehouse Builder users require access to metadata in the APPS schema. Later, in the execution phase, Warehouse Builder users must access the data in that schema.

Because direct access to the APPS production schema is most likely limited and restricted, you may define a user on the EBS database through which Warehouse Builder users can access only the relevant metadata and data.

To enable access to EBS data and metadata:

1. Create a user on the database hosting EBS. This user needs at least CONNECT and RESOURCE roles.

2. Grant access to the relevant metadata by running the script `owb\home\owb\cmi\ebs\owbebs.sql`.

   This script grants access to the following tables in the APPS schema that contain metadata for EBS tables, views, sequences, and keys: FND_APPLICATION, FND_APPLICATION_VL, FND_TABLES, FND_VIEWS, FND_SEQUENCES, FND_COLUMNS, FND_PRIMARY_KEYS, FND_FOREIGN_KEYS, FND_PRIMARY_KEY_COLUMNS, FND_FOREIGN_KEY_COLUMNS.

   The script also creates a synonym in the user schema for each of the preceding objects.

3. Enable a user to extract data from the EBS database.

   You can create a new user or enable the same user you created in the previous steps. For each object that you want to enable data extraction, grant this user at least SELECT access to each object.

Warehouse Builder users can now import the E-Business Suite metadata as described in the importing section of Oracle Warehouse Builder User’s Guide.
Configuring Repository Browser Environments

The Repository Browser connects to Warehouse Builder repositories and enables you to view metadata, run Web reports, perform lineage and impact analysis on your metadata, and audit runtime executions.

When you install Warehouse Builder from the Oracle Universal Installer, the Repository Browser is also installed and available in the languages you selected in the for Product Languages in the Oracle Universal Installer.

To verify the installation, start the Repository Browser listener and then the Repository Browser. For information on how to use the Repository Browser, refer to Oracle Warehouse Builder User's Guide.

Making Additional Language Fonts Available

If end users need to view the Repository Browser in a language that you did not select when initially installing Warehouse Builder, then copy the additional language fonts from the Warehouse Builder CD. From the fonts directory, copy the following fonts to the JDK directory under the owb home:

- ALBANWTJ.TTF
- ALBANWTK.TTF
- ALBANWTS.TTF
- ALBANWTT.TTF
- ALBANYWT.TTF

Changing the Session Timeout

By default, Repository Browser sessions time out after 180 minutes, that is, 3 hours of inactivity.

To change this setting, update the session-config tag in web.xml located at owb home\owb\j2ee\owbb\WEB-INF\.  

By default, the tag displays as follows:

```
<session-config>
  <session-timeout>180</session-timeout>
</session-config>
```

Enabling Integration with Third-Party Name and Address Data Libraries

Warehouse Builder gives you the option to perform name and address cleansing on your data with the Name and Address operator. The Name and Address operator identifies and corrects errors and inconsistencies in name and address source data. The operator identifies inconsistencies by comparing input data to data libraries supplied by the third-party name and address cleansing software vendors. Purchase the data libraries directly from these vendors.

To install data libraries, refer to the installation instructions of the name and address cleansing software vendor of your choice. For the list of certified name and address cleansing software providers, refer to Oracle Technology Network at http://otn.oracle.com/products/warehouse/htdocs/OTN_Partners.html.

To integrate with third-party name and address data libraries:
1. Install Warehouse Builder as instructed in this guide.

2. Purchase data libraries from one of the certified vendors listed on OracleMetaLink at http://metalink.oracle.com.

3. Install and access a certified vendor’s data libraries and Name and Address adapter following the vendor’s instructions.

   If you are installing in a Real Application Cluster environment, then you may be able to install the name and address adapter on many nodes to benefit from the parallelism and failover enabled by the RAC architecture. Check with your vendor to see if your purchase license allows a multiple-node installation.

   You do not need to install the data libraries on multiple nodes. However, if you install all data libraries on one node, then performance may suffer due to file access time latency. Follow the recommendations of your name and address cleansing software vendor.

4. Design a mapping using the Name and Address operator to cleanse name or address data. Refer to Oracle Warehouse Builder User’s Guide for information on designing mappings using the Name and Address operator.

Enabling Integration with Oracle Workflow

If you plan to use Warehouse Builder process flows, then use Oracle Workflow to enable deployment. You can also deploy Warehouse Builder schedules to Oracle Workflow. For more information, read about schedules in Oracle Warehouse Builder User’s Guide.

To enable integration with Oracle Workflow:

1. Locate the Oracle Workflow installation program.

   For a Warehouse Builder 11g repository on an Oracle 11g Database, the installation script is provided with the Warehouse Builder 11g software at owb_home/owb/wf/install.

   For a Warehouse Builder 11g repository on an Oracle 10g Release 2 Database, you must download the Oracle Workflow 2.6.4 software.

2. Start the Oracle Workflow installation program.

   For Windows, type the following at the command prompt:
   
   C:\> cd owb_home/owb/wf/install
   
   C:\owb_home\owb\wf\install> wfinstall.bat

   For UNIX, type the following in a UNIX shell:
   
   $ cd owb_home/owb/wf/install
   
   $ wfinstall.csh

3. Complete the Workflow Configuration Assistant as follows:

<table>
<thead>
<tr>
<th><strong>Oracle Workflow Setting</strong></th>
<th><strong>Value</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Install Option</td>
<td>Server Only</td>
</tr>
<tr>
<td>Workflow Account</td>
<td>owf_mgr</td>
</tr>
<tr>
<td>Workflow Password</td>
<td>Specify a password for the account.</td>
</tr>
</tbody>
</table>
4. Click **Submit** to start the Workflow configuration process.

   The configuration process can take several minutes. Check `owb_home/owb/wf/install/wf.log` for messages to follow the progress of the configuration process.

   When the process is complete, the Workflow Configuration Assistant displays a message of completion.

5. **Install the Workflow Client. (Optional)**

   The installation of Oracle Workflow client is optional because the Process Flow Editor in Warehouse Builder replaces its functionality. However, install Oracle Workflow client if you want to view the deployed Warehouse Builder processes in Oracle Workflow.

   On the computer where you installed Warehouse Builder client, install the Oracle Workflow client from the CD for Oracle Workflow client.

6. **Create a Workflow Proxy User.**

   When the Workflow instance is remote from the database hosting the Warehouse Builder repository, you need to create a proxy-user.

   Within the database hosting the repository, use SQL Plus to create a user and grant it the OWB_USER role as a default. This enables the remote OWF instance to connect to the services provided by the Control Center.
Troubleshooting a Warehouse Builder Installation

Refer to this chapter in the event that you encounter errors or problems with an installation. This chapter includes the following topics:

- General Steps for Troubleshooting Warehouse Builder on page 6-1
- Inspecting Logs Files in Warehouse Builder on page 6-1
- Error Messages Related to Installation on page 6-6
- Troubleshooting Installation Problems That Do Not Display Error Messages on page 6-14
- Checking Java Virtual Machine (JVM) on page 6-15

General Steps for Troubleshooting Warehouse Builder

Take the following steps to troubleshoot errors in Warehouse Builder:

1. Review this chapter for a possible solution to the problem.

   If Warehouse Builder displays an error message during the installation process, then refer to "Error Messages Related to Installation" on page 6-6. If you did not note the error number, you can review the "Log Files for Installation Errors" on page 6-2.

   In the absence of an error message, refer to "Troubleshooting Installation Problems That Do Not Display Error Messages" on page 6-14.

2. Check for additional information about the problem by "Inspecting Logs Files in Warehouse Builder" on page 6-1.

3. If the problem remains unresolved, search for a possible solution at https://metalink.oracle.com/

4. Review the Oracle Warehouse Builder Release Notes for installation notes or known issues.

5. If you are unable to resolve the problem in the previous steps, contact Oracle Support.

   Oracle Support may ask you to complete the steps in "Generating Log Files for a Specific Warehouse Builder Component" on page 6-16.

Inspecting Logs Files in Warehouse Builder

This section outlines all the different types of error messages that are logged by Warehouse Builder and how to access them.

Warehouse Builder logs the following types of errors:

- Log Files for Installation Errors on page 6-2
- Log Files for Metadata Import and Export Errors on page 6-2
- Log File for Validation Errors on page 6-3
- Log File for Generation Errors on page 6-4
Log Files for Installation Errors
When you run the Oracle Universal Installer to install Warehouse Builder, the installation error logs are automatically stored in:
C:\ProgramFiles\Oracle\Inventory\logs\installActions<timestamp>.log
When you run the Warehouse Builder Repository Assistant, the workspace installation error logs are stored in:
\owb home\UnifiedRepos\log_timestamp.log
See "Error Messages Related to Installation" on page 6-6 for suggested actions for commonly encountered errors during installation.

Log Files for Metadata Import and Export Errors
Metadata Import: When you import a project or specific objects into your workspace using the Metadata Import Utility, Warehouse Builder records details of the import process in a log file. You can specify the name and location of this log file from the Metadata Import dialog box.

Metadata Export: When you export a Warehouse Builder project or specific objects using the Metadata Export Utility, Warehouse Builder records the details of the export in a log file. You can specify the name and location of this log file from the Metadata Export dialog box.

Metadata Import Using the Transfer Wizard
If you are importing design metadata using the Warehouse Builder Transfer Wizard, then you can view the log file after the import is complete. Warehouse Builder displays the My Metadata Transfer dialog box.

Click View Log File to view the log file, as shown in Figure 6–1. Click Save As to save the log file to your local system.
In Warehouse Builder, you can validate all objects by selecting the objects from the console tree and then selecting **Validate** from the Object menu. After the validation is complete, the validation messages are displayed in the Validation Results window, as shown in Figure 6–2.

**Figure 6–1  Metadata Import Log File**

**Log File for Validation Errors**

In Warehouse Builder, you can validate all objects by selecting the objects from the console tree and then selecting **Validate** from the Object menu. After the validation is complete, the validation messages are displayed in the Validation Results window, as shown in Figure 6–2.
You can also validate mappings from the Mapping Editor by selecting **Mapping**, then **Validate**. The validation messages and errors are displayed in the Validation Results window.

On the Validation tab of the Validation Results window, double-click an object name in the Object column to display the editor for that object. You can fix errors in the editor. Double-click a message in the Message column to display the detailed error message in a message editor window. To save the message to your local system, select **Code** in the menu bar, then select **Save as File**.

Warehouse Builder saves the last validation messages for each previously validated objects. You can access these messages at any time by selecting the object from the console tree in the Project Explorer, select **View** from the menu bar, and then click **Validation Messages**. The messages are displayed in the Validation Results window.

**Log File for Generation Errors**

After you generate scripts for Warehouse Builder objects, the Generation Results window displays the generation results and errors, as shown in **Figure 6–3**. Double-click an error under the Messages column on the Validation tab to display a message editor that enables you to save the errors to your local system.
Log Files for Deployment and Execution Errors

You can store execution or deployment error and warning message logs on your local system by specifying a location for them. In the Project Explorer, select the project. Then from the Tools menu, select Preferences. In the Preferences dialog box, click the Logging option in the object tree to the left. In the list box on the right, you can set the log file path, file name and maximum file size. You can also select the types of logs you want to store.

You can view this log of deployment and error messages from the Warehouse Builder console by selecting View from the menu bar, and then Messages Log. This Message Log dialog box is read-only.

Errors related to the Control Center Service are stored at the following path:

`owb home\log\Repository_Name\log.xx` on the Oracle Database server.

Errors related to transforming or loading data are stored in the runtime tables. You can access these error reports using the Repository Browser. The Browser provides detailed information about past deployments and executions. Click the Execution tab in the Execution reports to view error messages and audit details.

Log File for Name and Address Server Errors

If you are using the Name and Address cleansing service provided by Warehouse Builder, you can encounter related errors.

Name and address server start up and execution errors can be located at:

`owb home\owb\bin\admin\NASver.log`

If your Name and Address server is enabled in:
Error Messages Related to Installation

This section includes the following topics:

- No fonts were found in \<drive>\Program Files\Qarbon\viewletBuilder3jre\lib\fonts
- OWBSYS is not granted access to owb home/owb/bin/admin/rtrepos.properties: Please run UnifiedRepos/reset_owbcc_home.sql specifying the path of the Oracle Home from which the Control Center Service is being run.
- SYS user does not have SYSDBA privileges.
- RTC-5301: The Control Center Service is not currently available.
- API5022: Cannot Connect to the Specified Repository
- Runtime Assistant fails with LoadJava Error.
- Error when specifying a SYSDBA user.
- Regional Name and Address Data Libraries Are Not Available.
- Lineage and impact analysis reports: Extensive tablespace requirements for materialized views.
- Java out of memory error occurs during a batch operation.
- ORA-01925: Maximum of 30 enabled roles exceeded
- INS0009: Unable to connect to the database. Verify the connect information.
- INS0022: A spawned program error.
- ORA-12154: TNS: Could not resolve service name.
- ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.
- PL/SQL: ORA-04052: Error occurred when looking up remote object
- IMP-00003: ORACLE error 30371 encountered
- Unable to connect to SQL*Plus in <Oracle Database version>
- ORA-04020 deadlock detected while trying to lock object or ORA-04021 timeout occurred while waiting to lock object
- ORA-04088: error during execution of trigger 'DV_SYS.DV_BEFORE_DDL_TRG'
- DPF-0029: Source <Table_Name> must have less than 165 attributes

Causes and Actions

No fonts were found in \<drive>\Program Files\Qarbon\viewletBuilder3jre\lib\fonts

Cause: After installing Warehouse Builder client components, you installed another software program that relies on Jinitiator and overwrote Java objects necessary of Oracle products. This may prevent you from launching Warehouse Builder or any other Oracle product that depends on Java objects.

Action: Re-install Jinitiator.
**Error Messages Related to Installation**

**Troubleshooting a Warehouse Builder Installation**

**6-7**

**OWBSYS is not granted access to owb home/owb/bin/admin/rrepos.properties:**
Please run UnifiedRepos/reset_owbccc_home.sql specifying the path of the Oracle Home from which the Control Center Service is being run.

**Cause:** When running the script reset_owbccc_home.sql and prompted for the owb home, you typed an invalid path for owb home.

**Action:** Run the script again and enter the correct path.

On all platforms, including both Windows and Unix, the path you enter must use forward slashes, and is case-sensitive. The case of the path entered here must match exactly the case of the path for the Warehouse Builder home as known by the operating system.

On Unix, the correct path to enter is the path for the owb home directory. On Windows, to determine the correct path for the owb home directory, examine the path displayed as part of the default Windows command prompt, and replace the backslashes with Unix-style forward-slashes. Do not supply a trailing slash. For example, if the Windows command prompt is:

```
C:\Oracle\My_OWB_Home\>
```

then the text you would enter is:

```
C:/Oracle/My_OWB_Home
```

**SYS user does not have SYSDBA privileges.**

**Cause:** In a standard database installation, the SYS user has SYSDBA credentials and REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE. You can verify the credentials by issuing the following connect statement:

```
SQL> CONNECT sys@tns_name_of_db AS SYSDBA;
```

Enter password: sys_password

If your database is configured with REMOTE_LOGIN_PASSWORDFILE=NONE, then the statement fails.

**Action:** If the statement fails, then you have the following options:

- Reconfigure your database with
  ```
  REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE and create a password file if none exists.
  ```
- If the preceding is not an option, reconfigure your database with
  ```
  O7_DICTIONARY_ACCESSIBILITY=TRUE.
  ```

**RTC-5301: The Control Center Service is not currently available.**

**Cause:** A Control Center Service must be running to enable the Control Center to manage deployments and executions. The service connects to the Control Center using JDBC and can be run from any Warehouse Builder home. Normally the service is runs on the server host.

**Action:** You can start a service on the server host by using the script start_service.sql.

If it is not possible to run the service on the server host, then start the Control Center Service on the local computer using the script local_service_login.sh or local_service_login.bat as appropriate. Use this script as follows:

```
local_service_login.sh [-startup | -closedown] owb home
```
In this mode, the Control Center Service runs on the local computer and is available only when that computer is available and can connect to the Control Center.

Use the script `show_service.sql` to determine the status of the service.

**Control center service log file reports "DBMS_OBFUSCATION" or "No key is found."**

**Cause:** The encryption of the passwords is out of sync with the client.

**Action:** Reset the repository and restart the control center service. To reset the repository, run `owb/rtp/sql/reset_repository.sql`.

**API5022: Cannot Connect to the Specified Repository**

*This error occurs when you try to connect to the a repository after having performed a database export or import from the Warehouse Builder repository schema.*

**Cause:** The package `NAMESPACESERVICEIMPL` may be invalid. This occurs after a database export or import from the Warehouse Builder repository schema if the repository owner has no `SELECT` privilege on `SYS.V_$SESSION`. You can diagnose the cause as follows:

1. In SQL*Plus, connect to the Warehouse Builder repository schema.
2. Enter the following command at the SQL prompt:
   ```sql
   ALTER PACKAGE NAMESPACESERVICEIMPL compile body;
   ```
3. If **Warning: Package body altered with compilation errors** appears, enter the following command at the SQL prompt:
   ```sql
   show errors;
   ```
4. The following errors mean that the Warehouse Builder repository owner has no `SELECT` privilege on `SYS.V_$SESSION`.
   ```sql
   PL/SQL: SQL statement ignored
   PLS-00201: Identifier 'SYS.V_$SESSION' must be declared
   ```

**Action:** Complete the following steps:

1. In SQL*Plus, connect as the `SYS` user.
2. At the SQL prompt, enter the following command:
   ```sql
   grant SELECT on V_$SESSION to Warehouse_Builder_Repository_Owner;
   ```
3. Connect to the `Repository_Owner`.
4. Enter the following command at the SQL prompt:
   ```sql
   alter package NAMESPACESERVICEIMPL compile;
   ```

**Runtime Assistant fails with LoadJava Error.**

**Cause:** This can occur if the Oracle Database does not have the JServer option installed.

**Action:** Make sure that the Oracle Database has JServer option installed.

**Error when specifying a SYSDBA user.**

*Oracle Warehouse Builder Assistants require you to provide SYSDBA credentials when installing the Oracle Warehouse Builder Design Repository or Runtime components.*
**Cause:** In a standard database installation, the `SYS` user has `SYSDBA` credentials. You can verify this from SQL*Plus by issuing the following connect statement:

```
connect sys/sys_password@TNS_NAME_OF_DB as sysdba;
```

In a standard database installation, the preceding connect statement works because `REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE` and the default password file is created by the installation process.

If your database is configured with `REMOTE_LOGIN_PASSWORDFILE=NONE`, then the following statement fails:

```
connect sys/sys_password@TNS_NAME_OF_DB as sysdba;
```

In this case, you have two options.

**Action:** Reconfigure your database with

```
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
```

and create a password file if none exists.

**Action:** If the preceding is not an option, then reconfigure your database with

```
O7_DICTIONARY_ACCESSIBILITY=TRUE
```

With this setting, the statement

```
connect sys/sys_password@TNS_NAME_OF_DB
```

enables the Warehouse Builder Assistants to connect to `SYS` user.

**Regional Name and Address Data Libraries Are Not Available.**

**Cause:** The Name and Address regional data libraries may not have been installed in the correct location.

**Action:** Ensure that you have successfully extracted regional data to the `NAS_DATA` directory.

1. From the `owb` home, start the Name and Address Server:
   - **For Windows:** Run `owb\bin\win32\NAStart.bat`.
   - **For UNIX:** Run `owb/bin/unix/NASTART.sh`.

2. Open the log file: `owb\bin\admin\NASvr.log`.
   - The log contains a list of installed countries.
   - If there is no such list, then verify that you have extracted the regional library data to the correct location. If you have extracted the data to the wrong location, then you can either reinstall the data, or modify the `owb\bin\admin\NameAddr.properties` file to indicate the correct file path. If you modify the `NameAddr.properties` file, then stop and restart the Name and Address Server as follows:
     - **For Windows:** Start the server by running `owb\bin\win32\NAStart.bat`. Stop the server by running `owb\bin\win32\NAStop.bat`.
     - **For UNIX:** Start the server by running `owb/bin/unix/NASTART.sh`. Stop the server by running `owb/bin/unix/NAStop.sh`.

3. Once you have verified the installation, you can stop the Name and Address Server if you wish, because it is automatically started at the execution of any mapping that employs the Name and Address operator.

**Lineage and impact analysis reports:** Extensive tablespace requirements for materialized views.
The first time you refresh a materialized view, it is populated from the Oracle Warehouse Builder repository. The materialized view can occupy up to twice the amount of space allocated to the entire Warehouse Builder repository.

**Cause:** Insufficient space has been allocated to the Warehouse Builder repository schema.

**Action:** If the Warehouse Builder repository schema is created in a tablespace that is dedicated to its use, these issues are easier to monitor. Ensure that sufficient free space exists on the physical drive for tablespace expansion. Within Oracle Enterprise Manager, ensure that the tablespace is set to Autoextend On.

---

**Note:** To grant permission to an OWB repository user to use Enterprise Manager for performing tasks, enter the following command in SQL*Plus: `GRANT SELECT any dictionary to '&OWB repository user';`

---

**Java out of memory error occurs during a batch operation.**

Operations requiring large amounts of memory can result in a Java out of memory error, if the system resources (such as virtual memory) are constrained.

**Cause:** There is not enough virtual memory. The Warehouse Builder client runs with a maximum heap size, as defined by the -mx parameter in the owbclient.bat file. The -Dlimit parameter in the owbclient.bat file specifies the memory threshold (80% of Dlimit) at which the Warehouse Builder memory manager begins to assist Java garbage collection. If you change the -mx parameter value, set the -Dlimit parameter to the same value, or at least to 90% of the value. Note that setting the -Dlimit to a low value can have a negative impact on the performance of Warehouse Builder.

**Action:** Increase the -Dlimit parameter in Warehouse Builder as follows:

1. Exit Warehouse Builder.
2. Open this file in a text editor:
   - For Windows: Open the $OWBHOME\bin\win32\ombplus.bat.
   - For UNIX: Open the $OWBHOME/bin/win32/owbclient.sh.
3. Change the -Dlimit parameter to 334.
4. Save and close the file.
5. Restart Warehouse Builder.

**ORA-01925: Maximum of 30 enabled roles exceeded**

This error occurs when installing a repository or a target schema.

**Cause:** The maximum number of enabled roles in the database has been exceeded. When you create a repository or a target schema, new roles are created in the database assigned to the schema in question. When the number of roles exceeds the value of the MAX_ENABLED_ROLES parameter, this error occurs.

**Action:** Increase the value of the MAX_ENABLED_ROLES parameter in the init.ora file. When you deinstall a repository or a target schema, delete the associated roles as well.

**INS0009: Unable to connect to the database. Verify the connect information.**

This error occurs when you try to connect to a database.
Cause: See the cause for ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.

Action: Follow the instructions for ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.

INS0022: A spawned program error.

Cause: This error message can result from a server issue when installing Warehouse Builder runtime components on an HP-UX operating system.

Action: To identify the server issue, complete the following steps:

1. From SQL*Plus, connect to a SYS user.
   
   Create user test_lj identified by test_lj;
   Grant connect, resource to test_lj;

2. Create owb home/owb/bin/unix/test.sh with the following contents:
   
   ../unix/loadjava -thin -verbose -order -resolve -user 'test_lj/test_lj@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=hpdp3)(PORT=1522))(CONNECT_DATA=(SERVICE_NAME=dgpdw)))'
   ../../lib/int/rtpserver.jar

3. Change directory to owb home/owb/bin/unix/.

4. Run test.sh.

ORA-12154: TNS: Could not resolve service name.

This error occurs when you try to connect to a database.

Cause: You defined a Warehouse Builder location and specified connection information using SQL*Net. However, the required TNS name is not accessible.

Action: To set up a TNS name for use during deployment in general and for execution of mappings and process flows, the TNS name needs to be accessible from the owb home being used to run the control center service. To ensure access, run the Net Configuration Assistant from the owb home and then restart the control center service.

To set up a TNS name for use by database links, the TNS name needs to be accessible from the database server home. To ensure access, run the Net Configuration Assistant from the database server home.

Cause: See the cause for ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.

Action: Follow the instructions for ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.

ORA-12514: TNS: listener could not resolve SERVICE_NAME given in connect descriptor.

This error occurs when you try to connect to a database.

Cause: If you used Oracle Net Easy Configuration or Oracle Net Assistant to create the Net Service Name entry, and you used the default option (Service Name) on the newly created Net Service Name, then the parameter SERVICE_NAME is added to the TNSNAMES.ORA as a subclause to the CONNECT_DATA section in the Net Service Name entry. This replaces the (SID=SIDname) subclause in previous releases of the database, for example, Oracle Database8i (8.1.x).

Action: Implement the TNSNAMES.ORA file as follows:
1. Use the `GLOBAL_DBNAME` parameter in the `LISTENER.ORA` for each SID that you want to identify as a separate service. Use the value of this parameter as the value of the `SERVICE_NAME` parameter. You need to activate any changes that you make to `LISTENER.ORA` for this purpose by stopping and restarting the listener process.

2. Use the values of the parameters that exist in the `INIT.ORA`, namely `SERVICE_NAMES` and `DB_DOMAIN`, to determine the value of the `SERVICE_NAME` that you must use in `TNSNAMES.ORA`. The valid construction of this value is `SERVICE_NAMES.DB_DOMAIN` with the period separating the two `INIT.ORA` values. If your `SERVICE_NAMES` is `BIKES` and your `DB_DOMAIN` is `COM`, then your `SERVICE_NAME` is `BIKES.COM`.

3. If there is no `DB_DOMAIN` parameter set in your `INIT.ORA`, or if there is no `GLOBAL_DBNAME` in the `LISTENER.ORA`, then you can use the `SERVICE_NAMES` from the `INIT.ORA` in your `TNSNAMES.ORA` for the `SERVICE_NAME` parameter.

   For example, if `INIT.ORA` contains `SERVICE_NAMES = "TEST817"` and `db_domain` is not set, then the `TNSNAMES.ORA` entry is: `CONNECT_DATA = (SERVICE_NAME = "TEST817")`.

4. If you have multiple values specified in the `SERVICE_NAMES` parameter in the `init.ora`, then you can use one of them. If `SERVICE_NAMES` is not set, then you can use `DB_NAME.DB_DOMAIN` parameters from the `INIT.ORA` file.

5. If `SERVICE_NAMES` and `DB_DOMAIN` is not set in the `INIT.ORA` and there is no `GLOBAL_DBNAME` in the `LISTENER.ORA`, then your `SERVICE_NAME` in `TNSNAMES.ORA` file is `DB_NAME`.

**PL/SQL: ORA-04052: Error occurred when looking up remote object**

_This error occurs when you have upgraded to <Oracle Database version> and are trying to redeploy mappings without first redeploying connectors._

**Cause:** While upgrading the Oracle Database, you moved your database to a new computer. Your old and new database instances do not have the same domain name. You can verify the cause by logging in to SQL*Plus as a `SYS` user and entering the following command: `SELECT * FROM GLOBAL_NAME;` If the Global Name of the old database does not match that of the new database, then a domain mismatch is causing this error.

**Action:** Either add the domain name to the Global Name in your new database by issuing a command similar to the following statement: `ALTER DATABASE RENAME GLOBAL_NAME TO xxx10G.US.ORACLE.COM;` or redeploy your connectors.

Refer to _Oracle Warehouse Builder User’s Guide_ for information on deploying connectors.

**IMP-00003: ORACLE error 30371 encountered**

**ORA-30371: column cannot define a level in more than one dimension**

_This error occurred when you were importing your target schema during migration._

**Cause:** The Warehouse Builder target schema is created with the `select_catalog_role` privilege. If you have the same dimension object defined in multiple Warehouse Builder target schemas, then Oracle Export creates duplicates in the export file, and this error occurs when you import.

**Action:** Connect as a `SYS` user to the existing version of the Oracle Database from which you exported the target schemas. Enter the following statement in `SQL*Plus`:

```
revoke select_catalog_role from OLD_Target_Schema;
```
Export the target schema into an Oracle .DMP file again, and then import the file into the Oracle Database.

**Unable to connect to SQL*Plus in <Oracle Database version>**

**Cause:** Your Oracle home or Path is not set correctly, or your Net Service Names are not configured.

**Action:** Ensure your Oracle home and Path are set correctly, and your Net Service Names are configured in the Oracle Database.

- Ensure that ORACLE_HOME and PATH are set correctly. Your Oracle home directory must to point to the owb home. Set your PATH variable to include the owb home\bin directory before any other Oracle products.

- Ensure that the TNSNames.ora file is configured correctly:
  
  For Windows: From the Oracle Database program group, start Net Configuration Assistant and select Local Net Service Name Configuration to configure TNSNames.ora.

  For UNIX: Set ORACLE_HOME and PATH to the owb home for Warehouse Builder 11g Release 1 (11.1), then run owb home/bin/netca to start Net Configuration Assistant. Select Local Net Service Name Configuration to configure TNSNames.ora.

**ORA-04020 deadlock detected while trying to lock object or ORA-04021 timeout occurred while waiting to lock object**

When creating runtime objects, the Runtime Assistant halts and produces these errors in the error log when trying to lock sys.dbms_aq.

**Cause:** User sessions may be pinning Advanced Queue objects.

**Action:** First, log in to SQL*Plus as a SYS user and run a query to identify which user sessions are pinning the Advanced Queue packages, using the following query as an example:

```sql
column s.sid format a5;
column s.serial# format a8;
column s.username format a10;
column objectname format a10;
select distinct
s.sid,
s.serial#,
s.username,
x.kglnaobj as objectname
from
dba_kgllock 1,
v$session s,
x$kgl1lk x
where
1.kgl1lktype = 'Pin' and
s.saddr = 1.kgl1lkuse and
```
s.saddr = x.kgllkuse and
x.kglnaobj in ('DBMS_AQ', 'DBMS_AQADM');

The following is an example of the output you receive:

<table>
<thead>
<tr>
<th>SID</th>
<th>SERIAL#</th>
<th>USERNAME</th>
<th>OBJECTNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>29623</td>
<td>RTU_4942</td>
<td>DBMS_AQ</td>
</tr>
</tbody>
</table>

Noting the SID and Serial Number, issue the following command to kill the user sessions:

```
ALTER SYSTEM KILL SESSION 'SIDNoted, SerialNumberNoted';
```

For example, enter the following command to kill the session listed in the sample output for this error:

```
ALTER SYSTEM KILL SESSION '9,29623';
```

**ORA-04088: error during execution of trigger 'DV_SYS.DV_BEFORE_DDL_TRG'**

**Cause:** When you attempt to create a Warehouse Builder repository on an Oracle Database that includes the Database Vault option, you may encounter an error such as ORA-04088.

**Action:** Disable the triggers DV_BEFORE_DDL_TRG and DV_AFTER_DDL_TRG.

**Import wizard throws error ORA-00997 when importing a table.**

**Cause:** When you import table definitions from an Oracle Database, you may encounter an error such as "Repository Error Message: ORA-00997: illegal use of LONG datatype...". This occurs when the CURSOR_SHARING parameter is set to FORCE or SIMILAR.

**Action:** Set the database parameter CURSOR_SHARING to EXACT.

**DPF-0029: Source <Table_Name> must have less than 165 attributes**

**Cause:** Creating data profiling on a table having more than 165 columns.

**Action:** Decrease the number of columns on the table. This is a data profiling restriction.

---

**Troubleshooting Installation Problems That Do Not Display Error Messages**

This section includes causes and actions for the following installation problems:

- **Warehouse Builder Clients that Previously Launched Now Momentarily Display the Splash Screen and Fail to Start**
- **Newly Installed Warehouse Builder Clients Fail to Start and Previously Launched Oracle Products Fail to Start**
- **A Warehouse Builder Client Freezes or Hangs**

**Causes and Actions**

**Warehouse Builder Clients that Previously Launched Now Momentarily Display the Splash Screen and Fail to Start**
Cause: If you attempt to start a Warehouse Builder client such as the Design Center and the splash screen displays momentarily but the client fails to start, you may have overwritten required java objects during the subsequent installation of another software product.

If the client is installed on Windows and you launched the client from the Start menu, you may not see any error messages.

Action: Manually start the client by typing at the DOS prompt `run owb home\owb\owbclient.bat`. You are likely to encounter an error message such as No fonts were found in ‘<drive>:\Program Files\ Qarbon\viewlet Builder3jre\lib\fonts’ on page 6-6.

Newly Installed Warehouse Builder Clients Fail to Start and Previously Launched Oracle Products Fail to Start

Cause: After installing Warehouse Builder software, an error in the path variable can prevent you from launching Warehouse Builder clients and other Oracle products that previously launched without problems.

Action: Verify that the path for `owb home\bin` is listed correctly in the Environmental Variables.

A Warehouse Builder Client Freezes or Hangs

Cause: Client software may freeze or hang due to various causes.

Action: If a Warehouse Builder client appears to freeze or hang, perform a stack trace as follows:

1. At the DOS command prompt, enter:
   ```
   cd owb home\owb\bin\win32\n   ```
2. Run `owbclient.bat`.
3. When the program hangs, press Ctrl+Break.
   This produces the thread-dump. Contact Oracle Support and provide them with this information to help identify the problem.

Checking Java Virtual Machine (JVM)

To check, verify, or reinstall the Java Virtual Machine (JVM) server in the database, refer to Oracle MetaLink:

1. In your Web browser, go to: `http://metalink.oracle.com`.
2. Log in to OracleMetaLink, or register as a new user.
3. Enter the following terms into the Search field, separating each term by semicolons):
   ```
   INITJVM.SQL; INSTALL; JAVAVM; JVM; VERIFY; SERVER; INSTALL; CLEANUP
   ```
4. Press Enter.

This search returns the cleanup notes for the JVM. The number of available documents frequently changes because Oracle Support creates, merges, and deletes various cleanup notes. This string of search words returns the most current and pertinent documents.
Generating Log Files for a Specific Warehouse Builder Component

If Warehouse Builder is producing errors or exhibiting other unexpected results, additional error logging can help you and Oracle Support identify the cause.

For additional error logging:

1. At the command prompt, navigate to:
   - For Windows: owb home\owb\bin\win32
   - For UNIX: owb home/owb/bin/unix

2. Run one of the execution files as listed in Table 1–9 on page 1-19 and pipe the output to a log file.
   For example, enter: owbclient.bat 1>out.log 2>error.log

3. Examine the resulting log file.
   Use this log when contacting Oracle Support.
Part II  Administering Oracle Warehouse Builder

This part contains the following chapters:

- Chapter 7, "Implementing Security in Warehouse Builder"
- Chapter 8, "Creating Custom Objects and Properties"
- Chapter 9, "Managing Metadata Dependencies"
- Chapter 10, "Version and History Management"
- Chapter 11, "Managing Multiple Environments from Development to Production"
- Chapter 12, "Moving Large Volumes of Data with Transportable Modules"
About Metadata Security

Warehouse Builder enables you to define security on the metadata you store in the design repository. The design repository is an Oracle Database with users, roles, and access privileges already defined. Warehouse Builder metadata security operates in addition to the Oracle Database security. The Oracle Database provides security for data while Warehouse Builder provides security for the metadata.

In addition to being registered in the repository, all users must also be database users in the design repository database. Database users have access to the data in the database by using SQL Plus. However, they do not have access to Warehouse Builder and its metadata unless the users are also registered in Warehouse Builder.

Metadata security is optional and flexible. You can apply no metadata security controls or define a metadata security policy. You can define multiple users and apply full security. Or implement your own security strategy based on the security interface. Also, after you define a security strategy, you can later adapt the strategy to be more or less restrictive.

The following sections describe how to implement metadata security using the Design Center.

You can also implement security through OMB Plus. For more information, refer to the Oracle Warehouse Builder Scripting Reference.

About the Security Interface

Only users with administrative privileges can access the security interface and change security policy in Warehouse Builder.

When you install Warehouse Builder and then use the Repository Assistant to create a design repository, Warehouse Builder assigns the design repository owner you define to be the default administrator. The first time you start the Design Center after
installation, you must log in as the design repository owner. You can then define additional administrators or other users as necessary.

Log in to the Warehouse Builder Design Center as the design repository owner and Warehouse Builder displays the Global Explorer as shown in Figure 7–1, “Global Explorer" in the lower right corner of the Design Center.

Figure 7–1  Global Explorer

Under the security node, notice there are two predefined roles, ADMINISTRATOR and EVERYONE. The one predefined user is the design repository owner, REPOS_OWNER in this example, which is assigned the ADMINISTRATOR role by default.

To perform actions under the Security node, select an object and right-click to view all of the possible operations. Or select an object and select Edit from the menu bar. For a complete list of all the tasks administrators can perform, see "Administrator Role" on page 7-9.

**Metadata Security Strategies**

Warehouse Builder enables you to design a metadata security strategy that fits your implementation requirements. As you define your metadata security strategy, recognize that more restrictive policies are more time consuming to implement and maintain.

Consider modeling your strategy based on one of the following security strategies:

- **Minimal Metadata Security Strategy (Default)**
- **Multiuser Security Strategy**
- **Full Metadata Security Strategy**
Minimal Metadata Security Strategy (Default)

Minimal metadata security is the default security policy when you create a new design repository. As your project requirements change, you can apply other metadata security strategies at any time.

You may not want or need to apply extra metadata security if, for instance, you are implementing a pilot project or anticipate only one or a few users accessing Warehouse Builder.

All users log in to Warehouse Builder with the same user name and password—that of the design repository owner. In this case, Oracle Database security policies keep the data in the design repository secure and the metadata is available to anyone who knows the design repository owner logon information. All users can create, edit, and delete all objects and you cannot discern which user performed which operation.

Multiuser Security Strategy

Use this strategy if your implementation has multiple users and you want to track who performs which operations. Also, use this strategy to restrict to a single user the rights and access granted to the design repository owner. Although this strategy does not restrict user access to metadata objects, you can apply restrictions at a later date.

To implement security for multiple users, log on to Warehouse Builder as an administrator and complete the instructions in the following sections:

1. Registering Database Users on page 7-4
2. Editing User Profiles on page 7-5

Full Metadata Security Strategy

This section describes a process for applying all the metadata security options available in Warehouse Builder. You can enable all or some of these options. For instance, you could take steps one through three but ignore the remaining steps.

To implement full metadata security for multiple users, log on to Warehouse Builder as an administrator and complete the instructions in the following sections:

1. Set the parameter Default Metadata Security Policy to maximum.
   In the Design Center select Tools, Preferences, and then Security Parameters.
2. Registering Database Users on page 7-4
3. Editing User Profiles on page 7-5
   The Default Metadata Security Policy you set in step one of these instructions is not retroactive. It applies only to users you register after changing the setting. You must manually edit the profiles of preexisting users.
4. Defining Security Roles on page 7-9
5. Editing User Profiles on page 7-5
6. Applying Security Properties on Specific Metadata Objects on page 7-10

Important Note: Be sure to edit the security properties for all projects in the Project Explorer. By default, the EVERYONE role has its object privileges set to full control. Select each project, press F2, select the Security tab, and edit the privileges to the EVERYONE role to be more restrictive.
Registering Database Users

All Warehouse Builder users must also be Oracle Database users.

Use a wizard within Warehouse Builder to either register existing database users or create new database users and then register them in Warehouse Builder.

**Note:** Although, you are able to create users in SQL Plus, it is not advisable. Creating users through the Warehouse Builder interface ensures that users are assigned the necessary roles and privileges.

To start the registration wizard, go to the Security node in the Global Explorer, right-click **Users** and select **New**. Follow the prompts in the wizard as described in "Selecting Existing or Creating New Database Users" on page 7-4.

Selecting Existing or Creating New Database Users

The left panel in **Figure 7–2** lists the Oracle Database users defined for the design repository. Either select existing database users from the list or define and register a new user by clicking on **Create DB User**... located in the lower left corner.

When selecting from the list, you can select one or more database users. Notice that, for security reasons, you cannot register database administrator users such as SYS. The database default role settings must not be set to ALL. You can change the database default role settings from within the wizard as described in "Changing Database Default Roles" on page 7-5.

**Figure 7–2** Creating and Registering Users
Creating an Oracle Database User

If you have database system privilege CREATE USER, then you can create new database users. The Create Database User dialog prompts you to type a user name and password for the new user and assign the default table space and temporary table space.

To specify a valid user name and password, adhere to the security standard implemented on the Oracle Database. The default minimum requirement is that both the user name and password be a VARCHAR(30). Also, do not include any special characters. Your database may have more requirements if a password complexity verification routine was applied.

For more information about user names, passwords, and password complexity verification routines, refer to Oracle Database Security Guide.

Changing Database Default Roles

For security reasons, you cannot register database users that have default roles in the database set to ALL. You can, however, change the default setting. Correct the role assignment by selecting Fix Now. Or you can correct the role assignment yourself by selecting Fix Later.

Fix Now

If you select Fix Now, type the user name and password with SYSDBA privileges. The product registers the user and issues the necessary commands to the database. For example, when you register new users, the database role OWB_repository_name is assigned to each user. For security reasons, this role must not be the default role of any registered user. If you attempt to register a user U1 under these conditions and then select Fix Now, then the product registers the new user and issues a command such as the following:

```
alter user U1 default role all except OWB_repository_name
```

Fix Later

If you select Fix Later, then the product does not register the user. You must manually change the default role setting in the database and then return to the product to register the user. To manually change the setting, connect to the database as a user with the ALTER USER system privilege and issue the required commands.

Under the Fix Later option, notice a recommended SQL script for changing the default roles for the selected users. The script also changes the default role setting such that any role subsequently granted to the user cannot be the default role of the user. To change this, you can register the user and then issue a command such as the following:

```
alter user U1 default role all except OWB_repository_name
```

Editing User Profiles

For each user, you can enter an optional description, assign the user to existing Roles, specify the Default Object Privilege and the System Privileges.

Because these users are also defined as Oracle Database users, you cannot rename a user from within this product. Rename users through the Oracle Database.
Roles

You can assign a user to one or more roles. If you assign multiple roles with conflicting privileges, then the user is granted the more permissive privilege, which is the union of all the privileges granted to the multiple roles. For example, if you assign to the same user a role that allows creating a snapshot and a role that restricts it, then the user is allowed to create snapshots.

If you want to assign a user to a role that does not display on the Available Roles List, close the editor, create the new role, and then edit the user account. To create a new role, right-click Roles under the Security node in the Global Explorer and select New. For information on creating and editing roles, see Defining Security Roles on page 7-9 and Editing Role Profiles on page 7-10.

Default Object Privilege

Default object privileges define the access other users and roles have to objects the selected user creates. These privileges do not impact the privileges the user has for accessing objects that others create.

For example, Figure 7–3, "Default Object Privilege Settings for USER1" shows that for all objects that USER1 creates, USER1 and the ADMINISTRATOR and DEVELOPMENT roles have full access while the EVERYONE, PRODUCTION, and QA roles are restricted to read only.

If you are familiar with UNIX operating system security, then note that the default object privilege behaves similarly to the UMASK command. When you edit the default object privilege, the change only effects objects the user subsequently creates. There is no effect on previously created objects. Therefore, if you set default object privileges at the onset, then little or no additional object-level security setup is necessary.

To define the privileges other users have to objects the selected user creates, check the appropriate box for each role or user. You can grant the following privileges: FULL CONTROL, EDIT, COMPILE, and READ. All the privileges are additive. If you select COMPILE, then you apply both the compile and read privileges.
Figure 7–3, "Default Object Privilege Settings for USER1" shows access granted to roles. You can also grant access to individual users. However, when you grant access to a role, the privilege is also extended to all users in that role. Therefore, in Figure 7–3, even though USER2 is not specifically granted access, USER2 has read access through the EVERYONE role. Furthermore, if USER2 is a member of the DEVELOPMENT role, then that user has full control and access.

**Important:** By default, when you create a new user, the EVERYONE role has full control on all objects. To enable metadata security, be sure to edit all user profiles and restrict the access the EVERYONE role has to objects each user creates.

**Securing a Metadata Object Throughout its Lifecycle**

Default object privileges work in conjunction with object security properties to provide security options throughout the lifecycle of a given metadata object. Settings you specify on the Default Object Privilege tab persist until a qualified user overrides the restrictions on an object by object basis.

Assume that USER1 creates several mappings. When USER1 designs and develops those objects, the security policy shown in Figure 7–3 may be desirable. However, assume that USER1 completes the work on mappings and wants to release the objects to the quality assurance team for testing. The default object privilege therefore becomes too restrictive. To extend access to the QA role, USER1 can navigate to the mapping, right click, select Properties, and select the Security tab. For more details on overriding the default security on an object by object basis, see "Applying Security Properties on Specific Metadata Objects" on page 7-10.
Object Privileges
Object privileges apply to all metadata objects in the repository including projects, modules, and collections.

FULL CONTROL
Full control includes all the other privileges plus the ability to grant and revoke privileges on an object. Only users with full control over an object can override default security on an object-by-object basis as described in "Applying Security Properties on Specific Metadata Objects" on page 7-10.

EDIT
The edit privilege includes the compile, and read privileges. Additionally, edit allows users to delete, rename, and modify an object.

COMPILE
The compile privilege includes the read privilege and enables you to validate and generate an object.

READ
The read privilege enables you to view an object.

System Privileges
System privileges define user access to workspace-wide services. Use the System Privilege tab to allow or restrict users and roles from performing administrative tasks. You can control access to the following operations:

- ACCESS_PUBLICVIEW_BROWSER: Allows users to access the Repository Browser.
- CREATE_EXTENSIONMODEL: Allows users to create new object types in the workspace.
- CREATE_PROJECT: Allows users to create projects, which administrators create projects as a means of organizing metadata objects.
- CREATE_SNAPSHOT: Allows users to create snapshots which administrators use when backing up workspaces.
- CONTROL_CENTER_DEPLOYMENT: Allows users to deploy to the Control Center and then run those procedures. For security reasons, you can enable this privilege only on a user by user basis; that is, you cannot extend this privilege to roles.
- CONTROL_CENTER_EXECUTION: Allows users to run procedures from the Control Center. For security reasons, you can enable this privilege only on a user by user basis; that is, you cannot extend this privilege to roles.
- CONTROL_CENTER_VIEW: Allows users to view procedures from the Control Center. For security reasons, you can enable this privilege only on a user by user basis; that is, you cannot extend this privilege to roles.

Support for a Multiple-user Environment
Warehouse Builder enables multiple users to access the same Warehouse Builder repository at the same time by managing read/write privileges. Only one user is given write privileges to an object at any given time. All other users can have read-only
access. If a user has write access to an object, Warehouse Builder maintains a lock on the object while the object editor is open. If no changes were made to the object, then the lock is released as soon as the object editor is closed. If changes were made, then the lock is maintained until the user closes all editors associated with the object and either saves the changes or reverts to the last saved version. Other users cannot delete an object while it is in use.

**Read/Write Mode**

Whenever you open an editor, property sheet, or dialog box, you access objects in read/write mode by default. Your changes are available to other users only after you save them to the repository.

**Read-Only Mode**

If you attempt to open an object locked by another user, then Warehouse Builder displays a message that prompts you either to cancel the request or access the object in read-only mode. If you choose to continue in read-only mode, then the editor displays Read only in the title bar.

The user who is editing an object in read/write mode may save his or her changes while you have the object open in read-only mode. To update your view with the repository, click the Refresh button on the toolbar.

**Defining Security Roles**

You can use roles to represent groups of users with similar responsibilities and privileges. Unlike users which are also database users, these roles are not database roles. These roles are purely design constructs for implementing security within the product.

Roles enable you to more efficiently manage privileges because it is more efficient to grant or restrict privileges to a single role rather than multiple users.

The **Everyone Role** and the **Administrator Role** are predefined roles. You edit the privileges but cannot delete or rename the predefined roles.

**Everyone Role**

Use this role to easily manage privileges for all users. When you register new users, Warehouse Builder assigns those users to the Everyone role.

**Administrator Role**

 Administrators in Warehouse Builder can perform the security tasks described in Table 7–1.

<table>
<thead>
<tr>
<th>Task</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registering Database Users</td>
<td>From the Global Explorer, right-click Users and select New.</td>
</tr>
<tr>
<td>Editing User Profiles</td>
<td>When you register a user, Warehouse Builder assigns the user to the everyone role and grants access to metadata objects based on that role. To change a user profile, right-click the user and select Open Editor.</td>
</tr>
</tbody>
</table>
Editing Role Profiles

For each role that you create, you can edit the name, enter an optional description, assign the role to existing Users, and specify the system privilege. System privileges for roles behave the same as they do for users. For more information on system privilege, see System Privileges on page 7-8.

You cannot rename or edit the descriptions for the predefined roles Everyone and Administrator.

Users

You can assign multiple users to a role. If you want to assign a user that does not display on the Available Users list, then close the editor, create the user from the Security node in the Global Explorer, and then edit the role. To create a new user, right-click Users from the Security node and select New. For information on creating and editing users, see Registering Database Users on page 7-4 and Editing User Profiles on page 7-5.

Applying Security Properties on Specific Metadata Objects

You can grant or restrict access to metadata objects on an object-by-object basis.

View the Security Tab by right-clicking on any metadata object and selecting Properties.

---

Table 7–1 (Cont.) Administrator Security Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing User Passwords</td>
<td>You cannot change user passwords from within Warehouse Builder. Change passwords directly in the Oracle Database as described in Oracle® Database Security Guide.</td>
</tr>
<tr>
<td>Defining Security Roles</td>
<td>Warehouse Builder provides two roles, the administrator role and the everyone role. Define additional roles by right-clicking on Roles in the Global Explorer and selecting New.</td>
</tr>
<tr>
<td>Editing Role Profiles</td>
<td>Right-click a role and select Open Editor. You can add and remove users and change the system privileges for the role.</td>
</tr>
<tr>
<td>Deleting Users and Roles</td>
<td>From the Global Explorer, right-click a user or a role and select Delete. You can delete all Warehouse Builder users except for the design repository owner. Deleting the user from Warehouse Builder does not delete or alter the user account on the Oracle Database. You can delete all roles expect the predefined ADMINISTRATOR and EVERYONE roles. Warehouse Builder roles and database roles are separate constructs. Therefore deleting a Warehouse Builder has no effect on the database.</td>
</tr>
<tr>
<td>Renaming Roles</td>
<td>From the Global Explorer, right-click a role and select Rename. You can rename all roles expect the predefined administrator and everyone roles.</td>
</tr>
<tr>
<td>Applying Security Properties on Specific Metadata Objects</td>
<td>Right-click any metadata object in any of the three explorers on the Design Center, select Properties, and select the Security Tab. Or select any metadata object, press F2, and then select the Security tab.</td>
</tr>
</tbody>
</table>
Security Tab

Use the Security tab to define metadata security on an object-by-object basis. Only users that have full control privileges on an object can change the metadata access controls on the Security tab. Security properties are important in managing the lifecycle of your projects, as described in “Example: Using Security Properties to Freeze a Project Design” on page 7-11.

While the Default Object Privilege defines metadata security for objects a specific user creates, the Security tab overrides that metadata security policy on an object-by-object basis. Assume that USER1 is a developer that creates mappings and process flows. If you want all objects created by USER1 available to another developer, then use the Default Object Privilege. However, if you want to make only a few objects created by USER1 available to the QA group, then locate each object in the Design Center and use the Security tab.

Important: To enforce a full metadata strategy, edit the security properties for all projects in the Project Explorer. By default, the EVERYONE role has its object privileges set to full control. Change the EVERYONE role privilege to be more restrictive and select Propagate to apply the changes to all children.

Propagating Security Properties to Dependent Objects

You can apply security properties to an object and all its children by selecting Propagate on the Security tab. This option is disabled when you select an object that cannot have dependent objects.

Example: Using Security Properties to Freeze a Project Design

When users complete the design of a project, you may want to freeze the contents of the project. Once you complete the following steps, only administrators can change the objects in the project.

To freeze a project design:
1. Log on as an administrator.
2. In the Project Explorer, right-click the project node and select Properties.
3. On the Security tab, restrict the privileges for all user and roles other than the administrators as appropriate.
4. Click the Propagate button.

Security Enforcement

When any user attempts to perform an operation in Warehouse Builder, Warehouse Builder first verifies that the user has the required privileges to perform the operation. Table 7–2 lists the privileges required to run operations in Warehouse Builder.

<table>
<thead>
<tr>
<th>Warehouse Builder Operation</th>
<th>Security Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>User must have EDIT privilege on objects to be configured.</td>
</tr>
<tr>
<td>Copy</td>
<td>User must have READ privilege on the object to be copied.</td>
</tr>
</tbody>
</table>
### Table 7–2  (Cont.) Privileges Required for the Execution of Operations

<table>
<thead>
<tr>
<th>Warehouse Builder Operation</th>
<th>Security Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create object</td>
<td>User must have EDIT privilege on parent. For example, to create a mapping you must have Edit privilege on the module.</td>
</tr>
<tr>
<td>Cut</td>
<td>User must have EDIT privilege on the object to be cut.</td>
</tr>
<tr>
<td>Delete</td>
<td>User must have EDIT privilege on the object to be deleted.</td>
</tr>
<tr>
<td>Deploy</td>
<td>No security check is necessary on the Deploy operation because Warehouse Builder checks the previous Generate operation.</td>
</tr>
<tr>
<td>Edit</td>
<td>User must have EDIT privilege on the object to be edited.</td>
</tr>
<tr>
<td>Export</td>
<td>User must have READ privilege on objects to be exported. Administrative users can export security information such as roles, users, and privileges when Export security information is enabled.</td>
</tr>
<tr>
<td>Generate</td>
<td>User must have COMPILE privilege on object to be generated.</td>
</tr>
<tr>
<td>Import</td>
<td>User must have EDIT privilege on objects to be exported. Administrative users can import security information such as roles, users, and privileges when Import security information is enabled.</td>
</tr>
<tr>
<td>Move</td>
<td>User must have privileges listed for the Cut and Paste operations.</td>
</tr>
<tr>
<td>Paste</td>
<td>User must have EDIT privilege on the parent to receive the copied object.</td>
</tr>
<tr>
<td>Rename</td>
<td>User must have EDIT privilege on the object to be renamed.</td>
</tr>
<tr>
<td>Snapshot: compare snapshots</td>
<td>To compare with another snapshot or other repository object, user must have READ privilege on that snapshot and the snapshot or other repository object.</td>
</tr>
<tr>
<td>Snapshot: restore snapshot</td>
<td>To restore an object based on a snapshot, a user must have READ privilege on that object. To restore a folder, a user must have EDIT privilege on the folder and all of its children.</td>
</tr>
<tr>
<td>Snapshot: take snapshot</td>
<td>User must have the CREATE_SNAPSHOT system privilege to create snapshots.</td>
</tr>
<tr>
<td>Source import</td>
<td>User must have EDIT privilege on objects to be replaced by imported objects.</td>
</tr>
<tr>
<td>Synchronize inbound</td>
<td>User must have READ privilege on the object in the repository and EDIT privilege on the object in the editor.</td>
</tr>
<tr>
<td>Synchronize outbound</td>
<td>User must have EDIT privilege on the object in the repository.</td>
</tr>
<tr>
<td>Validate</td>
<td>User must have COMPILE privilege on object to be validated.</td>
</tr>
</tbody>
</table>

### Managing Passwords in Warehouse Builder

You can manage passwords within Warehouse Builder in the following ways:

- **Changing Passwords that Access Control Centers**
- **Encrypting Passwords to Warehouse Builder Locations**
Changing Passwords that Access Warehouse Builder

In keeping with standard security practices, you may want to periodically change the passwords used to access Warehouse Builder repositories.

Changing Passwords that Access Design Repositories
Manage the password to design repositories as you would any other Oracle Database.

Changing Passwords that Access Control Centers
To change the password for a repository that hosts a Control Center and is therefore a deployment environment, you must first stop the Control Center service, run a script to change the password, and restart the Control Center service.

To change the password for a repository that hosts a Control Center:

1. Log on to the Control Center as the repository owner.
2. Stop the Control Center by running the script
   
   `owb home/owb/rtp/sql/stop_service.sql`
   
   The script returns values of Unavailable or Available to indicate the status of Control Center.
3. Change the password by running the script
   
   `owb home/owb/rtp/sql/set_repository_password.sql`
   
   When prompted, specify the new password.
4. Restart the Control Center by running the script
   
   `owb home/owb/rtp/sql/start_service.sql`.

Encrypting Passwords to Warehouse Builder Locations

Warehouse Builder users create a location for each database, file server, or application that want to extract or load metadata and data. Locations include the user name and password used to access these various sources and targets. Warehouse Builder can store these passwords in the repository in an encrypted manner. The switch that turns on and off the password storage is Persist Location Password in Metadata, which is located in the Design Center under Tools, Preferences, Security Parameters.

The default encryption algorithm utilized is DES56C that is valid for Oracle Database 9i and subsequent versions. If the repository Database is version 10g or later, then you can set the encryption algorithm to 3DES168 or any other more powerful encryption by changing `owb home/owb/bin/admin/jdbcdriver.properties` file and specifying the following encryption parameters:

- `encryption_client; default = REQUIRED`
- `encryption_types_client; default = ( DES56C )`
- `crypto_checksum_client; default = REQUESTED`
- `crypto_checksum_types_client; default = ( MD5 )`

For the protocol to work, set the server to the default ACCEPTED mode. For more information, see the Oracle® Database JDBC Developer’s Guide and Reference.
This chapter describes how to extend the workspace by creating custom objects and custom properties. This chapter includes the following topics:

- Extending the Workspace With User Defined Objects on page 8-1
- Adding New Properties to Workspace Objects on page 8-3
- Adding UDOs to the Workspace on page 8-5
- Working with UDOs and UDPs on page 8-11
- Creating New Icons for Workspace Objects on page 8-13

Extending the Workspace With User Defined Objects

As an administrator, you may encounter scenarios that require you to add new types of objects or properties to an existing workspace. Any objects you add to the workspace are known as user defined objects (UDOs) and any properties you add are known as user defined properties (UDPs).

For example, as you use Warehouse Builder in conjunction with other applications, you may want to document how the data and metadata you manage in Warehouse Builder interacts with other applications. To facilitate this documentation, you can introduce new metadata objects and associate those objects with existing workspace objects. These custom objects appear in the Design Center with the icon of your choice. You can edit them with a basic editor and perform lineage and impact analysis.

Administration users can extend the workspace by adding new properties and objects.

- Adding New Properties to Workspace Objects: Each workspace object has a pre-defined property set to which you can add UDPs. Consider doing this, if, for example, you want to add a property to contain design notes for metadata objects. Once you define them, UDPs act like native properties.

- Adding UDOs to the Workspace: You can introduce new types of objects to the workspace by defining UDOs which follow the general rules for object locking, multiuser access, transactions, security, and snapshots. You can also import and export UDOs and UDPs using the Metadata Loader (MDL).

For the sake of clarity, this chapter refers to the objects native to the workspace as workspace objects. Any objects you introduce to the workspace are UDOs and any new properties are UDPs.

About Oracle Metabase (OMB) Plus

To define custom objects and properties, you must use the scripting utility Oracle Metabase (OMB) Plus. After you specify the UDO and UDP definitions through scripting, you can then create instances of those objects either through scripting or using the graphical user interface.

OMB Plus is the flexible, high-level command-line metadata access tool for Oracle Warehouse Builder. It is an extension of the Tcl programming language and you can use it on UNIX and Windows platforms. The section "Using OMB Plus Scripts to Specify UDOs and UDPs" describes how to write the syntactic constructs such as variable support, conditional and looping control structures, error handling, and
standard library procedures. Using OMB Plus, you can navigate through workspaces and manage and manipulate metadata in workspaces.

For more information about using OMB Plus to specify UDOs and UDPs, see Oracle Warehouse Builder API and Scripting Reference.

**Using OMB Plus Scripts to Specify UDOs and UDPs**

OMB Plus scripts enable you to define new objects, add and remove properties, as well as view the attributes for existing objects. The syntax is case sensitive and must be in upper case. While creating UDOs and UDPs, follow the guidelines in "Naming Conventions for UDOs and UDPs" on page 8-3.

**OMBDEFINE**

OMBDEFINE CLASS DEFINITION enables you to create new object definitions in the workspace.

To create a new module definition, use the following command:

OMBDEFINE MODULE CLASS DEFINITION 'UD_TABLEMODULE' SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) VALUES ('Table Module', 'Table Modules')

This creates a new module definition called UD_TABLEMODULE.

**OMBREDEFINE**

OMBREDEFINE CLASS DEFINITION enables you to redefine workspace objects to include custom properties.

To create a UDP on the Dimension object, use the following command:

OMBREDEFINE CLASS DEFINITION 'DIMENSION' ADD PROPERTY DEFINITION 'UD_DOCID' SET PROPERTIES (TYPE, DEFAULT_VALUE) VALUES ('INTEGER', '100')

This adds a new property definition called UD_DOCID to class definition DIMENSION.

The following command adds a new property definition for notes for the COLUMN type. Because columns exist in tables, views, materialized view, external tables and sequences, the following command adds the definition of this property to columns for all of those metadata objects:

OMBREDEFINE CLASS DEFINITION 'COLUMN' ADD PROPERTY DEFINITION 'UD_COL_NOTE' SET PROPERTIES (TYPE, DEFAULT_VALUE) VALUES ('STRING', 'notes')

When you create and save a new property definition, OMB Plus performs the following validations:

- A user access check ensuring that you have single-user access to the current workspace.
- A name space check ensuring that you did not define two identically named property definitions within the same class hierarchy.
- A property value check ensuring that you defined default values consistent with the data types that you specified.

To change the name or the default value of a given property definition, use a command as follows:

OMBREDEFINE CLASS DEFINITION 'TABLE' MODIFY PROPERTY DEFINITION 'UD_TBL_NOTE' SET PROPERTIES (DEFAULT_VALUE, BUSINESS_NAME)
VALUES ('99', 'Table Note')

To delete a UDP, use a command such as

OMBREDEFINE CLASS_DEFINITION 'TABLE' DELETE PROPERTY_DEFINITION 'UD_TBL_NOTE'

which deletes the UD_TBL_NOTE property definition from the Table class. Deleting a UDP is a destructive action and should generally be done with caution because it cannot be undone. It renders irretrievable all custom property values made for this property definition in your workspace.

OMBDESCRIBE

You can use OMBDESCRIBE on any class definition to view the attributes for a metadata element. Among other tasks, use OMBDESCRIBE to list the UDPs for a given object type. For instance, the following command lists the UDPs for dimensions:

OMBDESCRIBE CLASS_DEFINITION 'DIMENSION' GET PROPERTY_DEFINITIONS

You can also use OMBDESCRIBE to inspect the properties of a property definition. For instance, for a UDP called UD_DOCID, you can know its data type, default value, and business name using the following command:

OMBDESCRIBE CLASS_DEFINITION 'DIMENSION' PROPERTY_DEFINITION 'UD_DOCID' GET PROPERTIES (TYPE, DEFAULT_VALUE, BUSINESS_NAME)

Naming Conventions for UDOs and UDPs

It is mandatory to include the prefix UD_ while naming UDOs and UDPs. This ensures that the names of UDOs and UDPs are not identical to the names of predefined objects in the workspace.

Note: If in a previous release you named a UDP with the prefix UDP_, then it is still valid. However, for all subsequent UDOs and UDPs, use the UD_ prefix.

Adding New Properties to Workspace Objects

To define new properties on Workspace objects, complete the following steps:

1. Carefully plan the new additions to the workspace.
   If possible, you should define all user-defined properties into the workspace before enabling end users to access it. In doing so, you avoid the task of supplying values for UDPs on existing objects.

2. Log in as the workspace owner, in single user mode.
   If another user is logged on within the GUI or OMB Plus, then you will be prevented from running commands that alter the structure of the workspace.
   If already logged into the Design Center, then you can start OMB Plus from the Window option on the main menu of Design Center. To ensure that no other users access the workspace, connect to the workspace and issue the command OMBSWITCHMODE SINGLE_USER_MODE.

3. Use the command OMBREDEFINE on the workspace object to which you want to add a custom property definition.
   For examples on how to use this command, see the section OMBREDEFINE on page 8-2.

4. To view the changes in OMB Plus, use the command OMBDESCRIBE.
Adding New Properties to Workspace Objects

5. Once you create the UDP using scripting, users can view and assign values to the new property in the graphical user interface.

6. Notify users that they can log in to the Design Center.

Creating UDPS: An Example

To create UDPS for an object, complete the following steps:

1. Log in to the client as an administrator.

2. Open the OMB Plus client.

3. Ensure that you are in single user mode. You can verify this with the command OMBDISPLAYCURRENTMODE. If you are in multiple user mode, then switch to single user mode by using the command:

   OMBSWITCHMODE SINGLE_USER_MODE

4. In the OMB Plus client, enter the following command to create four UDPS for the object View:

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_OWNER' SET PROPERTIES \ 
   (TYPE, DEFAULT_VALUE, BUSINESS_NAME) VALUES \ 
   ('STRING', 'REP_OWNER', 'Object Owner')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_FILE' SET PROPERTIES \ 
   (TYPE, DEFAULT_VALUE) VALUES ('FILE', 'C:\vw.sql')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_LINK' SET PROPERTIES \ 
   (TYPE, DEFAULT_VALUE) VALUES ('URL', 'http://www.oracle.com')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_VERSION' SET PROPERTIES \ 
   (TYPE, DEFAULT_VALUE) VALUES ('DATE', '2006-1-7')

   OMBSAVE

   This creates the following UDPS: UD_OWNER, UD_FILE, UD_LINK, and UD VERSION.

   Note that the valid UDP types are: integer, string, float, double, date, timestamp, boolean, long, file, and url.

5. From the Project Explorer create a view in any module.

6. Open the property inspector for the view by right-clicking the view and selecting Properties.

7. Click the User Defined tab and select the view in the left-hand panel. The newly created UDPS are displayed.

   You can modify the values of any of the UDPS from the graphical user interface.

   Figure 8–1 displays the UDP that was created in the User Defined tab.
Adding UDOs to the Workspace

UDOs are objects that you define and add to the workspace in addition to existing workspace objects.

All UDOs must belong to a module, and the module itself is a UDO. This module acts as the topmost container holding other objects within it. A module can contain folders, first class objects (FCOs), and second class objects (SCOs). Similarly, a folder can contain other folders, FCOs, and SCOs. An FCO can contain one or more SCOs.

UDOs exhibit a parent-child relationship. The module is the topmost parent. An FCO within a module is a child element of the module. Similarly, an SCO within an FCO is a child element of the FCO. For example, an Oracle module is a parent module. A table within this module is an FCO and a column within the table is an SCO.

To define new objects for the workspace, complete the following steps:

1. Carefully plan the new additions to the workspace.
   Before you begin, fully review the remainder of this chapter and become familiar with the necessary scripting commands.

2. Log in to the client as an administrator and in single user mode.

3. Design the UDO based on the steps described in "Writing Scripts to Define UDOs".

4. Use the OMBDEFINE command to create a new module definition, and FCOs and SCOs within that module. Use the OMBREDEFINE command to make any changes to the UDOs or to set properties for these objects.

   Once you create the UDO through scripting, you can use the graphical user interface to create and edit the objects it contains.

5. Log in to the Design Center and view the new objects as described in "Working with UDOs and UDPs" on page 8-11.

   Verify that the new objects display as intended.

6. (Optional) Assign a new icon to the UDO, as described in "Creating New Icons for Workspace Objects" on page 8-13.

7. Notify users that they can log in to the client.

Writing Scripts to Define UDOs

To define a UDO, write a script that completes the following steps:
■ **Create a user defined module**: This will be the parent module.

■ **Define the object type**: Define the module as a folder that can contain other objects.

■ **Define FCOs and SCOs**: Create user-defined FCOs and SCOs for the UDO, and assign physical names to these objects. For example, UD_WORKBOOK is a valid physical name. You can also indicate a business name and plural name, both of which are displayed in the Design Center and in editors. Continuing the previous example, Workbook and Workbooks are likely entries for the business name and plural name, respectively. If you do not specify these values, then they default to the physical name.

■ **Define object properties**: Define the properties for all the objects you create. Some properties, such as Name, Business_Name, Plural_Name, and Description, are assigned automatically to any newly created object.

■ **Add component definition**: All parent objects need to be assigned a component definition. The child elements have to be added to the component definition. The component definition determines the lifetime of child elements. For example, a column cannot exist if the parent table is deleted.

■ **Define association types**: Create association types to indicate the types of relationships a UDO can have with workspace objects and other UDOs. You need to perform this step only if you want end users to later relate the UDO to specific instances of objects. For instance, in your script you could associate the UDO with tables and views. In the Design Center, end users could then relate instances of the UDO with specific tables and views. Warehouse Builder displays these relationships in impact and lineage analysis reports.

■ **Assign icons (optional)**: See “Creating New Icons for Workspace Objects” on page 8-13.

■ **Save the changes**.

### Creating UDOs: An Example

This section provides an example to create UDOs modeled on a Java application. The Java application acts as a module. This module contains classes (FCOs), and those classes contain methods (SCOs). Within a method, you can model the lines of code. From a business standpoint, this is of interest because a particular line of code in an application may be impacted by a change in a database table if it is used within a SQL (JDBC) statement.

Figure 8–2 displays the structure of the UDO.
To create the UDOs, perform the following steps:

1. Log in to the Warehouse Builder client as an administrator and open the OMB Plus window. Make sure that you are logged in single user mode.

2. First create a module definition and set properties for this module:

   ```
   OMBDEFINE MODULE_CLASS_DEFINITION 'UD.Java_APP' \
   SET_PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \ 
   VALUES ('Java Application', 'Java Applications')
   ```

   This defines the module definition and sets certain properties common to all objects. BUSINESS_NAME is the user-friendly name for an object. If the Naming mode preference for the Design Center is switched to Business mode, then the value set for BUSINESS_NAME is displayed for the object. PLURAL_NAME is the label that is used to show where multiple instances of an object are shown, such as the label used for a tree node in the Design Center that contains several instances of the object.

3. Now create a folder definition with the same name as the module so that the module assumes the role of a folder:

   ```
   OMBDEFINE FOLDER_DEFINITION 'UD.Java_APP'
   ```

4. Now create an FCO:

   ```
   OMBDEFINE FIRST_CLASS_OBJECT CLASS_DEFINITION \ 
   'UD.JCLASS' SET_PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \ 
   VALUES ('Java Class File', 'Java Class Files')
   ```

5. Add the FCO as a child of the folder class:

   ```
   OMBREDEFINE CLASS_DEFINITION 'UD.Java_APP' \ 
   ADD_CHILD_TYPE 'UD.JCLASS'
   ```
Adding UDOs to the Workspace

6. Create a component definition for the FCO:
   OMBDEFINE COMPONENT_DEFINITION 'UD_JCLASS'

7. Add the component definition to the folder definition:
   OMBREDEFINE FOLDER_DEFINITION 'UD_JAVA_APP' \\
   ADD 'UD_JCLASS'

8. Create an SCO and set its properties:
   OMBDEFINE SECOND_CLASS_OBJECT \\
   CLASS_DEFINITION 'UD_JMETHOD' \\
   SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \\
   VALUES ('Method', 'Methods')

9. Add the SCO as a child of the FCO:
   OMBREDEFINE CLASS_DEFINITION 'UD_JCLASS' \\
   ADD CHILD_TYPE 'UD_JMETHOD'

10. Add the SCO to the component definition:
    OMBREDEFINE COMPONENT_DEFINITION 'UD_JCLASS' \\
        ADD 'UD_JMETHOD'

11. Create an SCO and set its properties:
    OMBDEFINE SECOND_CLASS_OBJECT \\
        CLASS_DEFINITION 'UD_JMETHOD_LINE' \\
        SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \\
        VALUES ('Java Method Line', 'Java Method Lines')

12. Add this SCO as a child of the initially created SCO:
    OMBREDEFINE CLASS_DEFINITION 'UD_JMETHOD' \\
        ADD CHILD_TYPE 'UD_JMETHOD_LINE'

13. Add this SCO to the component definition:
    OMBREDEFINE COMPONENT_DEFINITION 'UD_JCLASS' \\
        ADD 'UD_JMETHOD_LINE'

This creates the following UDOs:
- A module folder called UD_JAVA_APP
- An FCO named UD_JCLASS, within the module
- An SCO named UD_JMETHOD, which is the child of UD_JCLASS
- Another SCO named UD_JMETHOD_LINE, which is the child of UD_JMETHOD

You can access the UDOs from the Project Explorer under the User Defined Modules icon. To create a new instance of the UDO, right-click the UDO and select New. You can create new modules and SCOs as well as edit these modules and SCOs.

Note: For the newly created UDO to be visible in the Project Explorer, you must shut down Warehouse Builder completely, and then start it up again, saving any changes if prompted.
Associating UDOs with Objects

UDO can be associated with other objects. By creating these associations, UDOs become a part of Lineage and Impact Analysis diagram just like any other object.

Associating a Java Application with a Table

This example associates the SCO, UD_JMETHOD, with one or more tables. This is modeling the fact that a method could be referencing tables in JDBC calls.

To associate the Java method to table, use the command:

```
OMBDEFINE ASSOCIATION_DEFINITION 'UD_XJMETHOD2TABLE' \
SET PROPERTIES (CLASS_1,CLASS_2,ROLE_1,ROLE_2 \
,ROLE_1_MAX_CARDINALITY,ROLE_1_NAVIGABLE) \
VALUES ('UD_JMETHOD','TABLE','TABLEUSED','JM2TABLE' \
,'INFINITE','true') ADD DEPENDENCY_DEFINITION 'DATAFLOW'
```

CLASS_1 and CLASS_2 can be any classes (FCO or SCO). At least one of the classes should be a user defined class. The other class can be either a user defined class or one of the main Warehouse Builder classes, such as table or column. In this example, the association is between the UDO, UD_JMETHOD, and table.

Role_1 and Role_2 are the names you use to identify Class_1 from the point of view of this association. A class may have multiple associations and it plays a role in each one of them.

MAX/Cardinality allows you to limit the number of objects of that class which participate in that association. For example, consider the association between uniqueKey and foreignKey. The max_cardinality of uniqueKey is 1, because a given foreignKey object can be associated with at most one uniqueKey object. MAX_CARDINALITY can be set to any positive integer, or the reserved word INFINITE.

ROLE_1_NAVIGABLE is used by Lineage/Impact analyzer. If set to TRUE, it means that the analyzer can traverse the association in either direction between Class_1 and Class_2. If the property is set to FALSE, it means that the analyzer can traverse the relationship from Class_1 to Class_2, but not the other way around.

DEPENDENCY_DEFINITION is a mandatory parameter of an association and must always be set to DATAFLOW.

To associate the UDO to a table, complete the following steps:

1. In the Project Explorer, expand the node User Defined Modules.
2. Right-click Java Applications and select New.
3. Specify a name for the application.
4. Right-click the node Java Class Files and select New.
5. Specify a name for the Java class.

Figure 8–3 shows a new user defined module created from the Project Explorer.
6. Right-click the object you just created, and select Open Editor to open the UDO Editor.

7. Click the Object and Association Tree tab and select CUSTOMERUPDATE. You can see the properties for the FCO, UD_JCLASS.

Figure 8–4 displays the editor for an FCO.

Figure 8–4 Editor for UD_JCLASS
8. Right-click **Methods** and select **Create**.  
    An SCO called JMETHOD_1 is created.

9. JMETHOD_1 contains two nodes: Java Method Lines, which is the child SCO, and TABLEUSED, which is the value specified for ROLE_1 when the association UD_XMETHOD2TABLE was created.  
    Figure 8–5 displays the nodes in an SCO.

![Figure 8–5 Nodes Within an SCO](image)

10. Right-click TABLEUSED and select **Reference**.  
    The Object Selector dialog box is displayed, and allows you to select the table to which you want to connect the UDO.  
    Figure 8–6 displays the Object selector dialog box.

![Figure 8–6 The Object Selector Dialog Box](image)

---

**Working with UDOs and UDPs**

In the graphical user interface, you can view UDOs and UPDs in the Project Explorer and in the **Repository Design Browser**. However, in the Project Explorer, you can also edit the UDOs and UDPs.
Repository Design Browser
The Repository Browser is a metadata management and reporting portal. This browser displays objects, object properties, object relationships, including UDOs and UDPs. The browser also displays lineage and impact analysis reports for all objects including UDOs.

If you define a UDP for a given object, then the browser lists the UDP name and values as Extended Property Name and Extended Property Value.

Figure 8–7 displays the sample properties sheet.

Figure 8–7 Sample Properties Sheet with User-Defined Properties

Propagating UDOs and UDPs to Other Workspaces
The primary method for propagating changes from one workspace to another is by using MDL. MDL enables you to export and import the metadata definition of the UDP and its contents.

Exporting UDOs and UDPs
You can export UDOs and UDPs as any other object.

In the MDL Control file, the option is `DEFINITIONFILE=filename` to export the metadata definition. For example:

```plaintext
## Sample Export file
USERID=UserName/Password@HostName:PortID:OracleServiceName
# DEFINITIONFILE=Drive:\DirectoryName\filename.mdd
FILE=Drive:\DirectoryName\filename.mdl
LOG=Drive:\DirectoryName\filename.log
```
Importing UDOs and UDPs

You can import UDPs from the command line only. During import, MDL updates the UDPs for all objects. In the MDL Control file, the option is DEFINITIONFILE=filename to import the metadata definition. For example:

```%
# Sample Import file
USERID=UserName/Password@HostName:PortID:OracleServiceName
#
DEFINITIONFILE=Drive:\DirectoryName\filename.mdd
FILE=Drive:\DirectoryName\filename.mdl
LOG=Drive:\DirectoryName\filename.log
%
```

You can import UDPs using one of the following search criteria:

- **Universal ID**: The metadata definition contains a Universal Object ID (UOID). The UOID uniquely identifies objects across workspaces. If you import the MDL file by UOID, then MDL looks up the metadata definition by UOID. If the metadata definition name in the source MDL file is different from the metadata definition in the workspace, then MDL renames it.

- **Physical Name**: MDL looks up the metadata definition by physical name.

Regardless of the import mode, MDL either adds the metadata definition if it does not exist in the workspace, or updates the metadata definition if it already exists. MDL does not delete metadata definitions in the workspace.

When updating the metadata definition, MDL only renames the object if the names are different (search criteria is by UOID), and updates the default value. MDL does not change the data type.

Creating New Icons for Workspace Objects

Icons are graphics that visually suggest the availability of a function or type of an object to end users. There are many types of pre-defined workspace objects, each with their own icon. You may want to change the icon associated with an existing object or instance of an object to something more recognizable. For example, you could visually highlight a particular table by altering its icon. Additionally, for UDOs, you may want to change the default icon to something representative of the object. You can create your own icons using a graphics editor or third-party software.

In the Design Center, you can associate icon sets with an instance of an object. Or, use the OMB Plus scripting language to associate icon sets at the object type level (and thus inherited by any instance of that object).

**Note:** You can assign new icons to most workspace objects with the exception of pre-defined objects like public transformations and public data rules and DEFAULT_CONFIGURATION, DEFAULT_CONTROL_CENTER, and OWB_REPOSITORY_LOCATION.

Every object has a set of icons of varying sizes to represent it throughout the various editors and toolbars. Each icon set includes a canvas icon, palette icon, and a tree icon as described in Table 8–1. When you define a new icon set, follow the sizing guidelines. If you specify a new icon with an incorrect size, it is automatically resized, which may distort your intended design.
Creating New Icons for Workspace Objects

To create a new icon set, complete the following steps:

1. Log in to the client as an administrator.
2. In the Global Explorer, right-click the Icon Sets node and select New.
3. The Create Icon Set dialog box is displayed. For details on the values to be entered on this page, see “Create Icon Set Dialog Box” on page 8-14.

Create Icon Set Dialog Box
The Create Icon Set dialog box enables you to specify values for the Icon Set. Enter the following values and click OK to define a new Icon Set:

- **Name**: The name of the Icon Set.
- **Group**: You can assign the Icon Set to any of the groups available in the list. This is useful when you are creating a large number of icon sets and want to organize them into different groups.
- **Description**: A description of the Icon Set.
- **File Name**: Navigate and select the image that you want to assign to the new Icon Set. You need to select an image for Canvas, Palette, and Tree Icon.

**Note**: You can use any image of your choice to represent the new icon.

The newly created icon set will be available under Icon Sets in the Global Explorer.

Assigning New Icon Sets to Workspace Objects
You can change the default icon setting for any object by editing its properties. In any of the Explorers in the Design Center, right-click an object and select Properties to assign a new icon set to an object. When you save your changes, the new icon set appears throughout the user interface.

To revert back to the original icon, select Use Product Default Icon.

For more information about icon sets, see “Creating New Icons for Workspace Objects” on page 8-13.

Table 8–1 Icon Sets

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canvas Icon</td>
<td>Represents instances of objects in the canvas of an editor. For example, it displays a table icon in the canvas of the Mapping Editor or in a Lineage Report. The correct size is 32 x 32 pixels in GIF or JPEG format.</td>
</tr>
<tr>
<td>Palette Icon</td>
<td>Represents types of objects in editor palettes. For example, it displays the table operator in the Mapping Editor operator palette. The correct size is 18 x 18 pixels in GIF or JPEG format.</td>
</tr>
<tr>
<td>Tree Icon</td>
<td>Represents types and instances of objects in navigation trees such as the Project Explorer in the Design Center. The correct size is 16 x 16 pixels in GIF or JPEG format.</td>
</tr>
</tbody>
</table>
Assigning a New Icon Set to an Object

Complete the following steps to assign a newly created icon set, CSV_ICONSET, to an instance of an object.

1. Right-click any instance of an object and select Properties to open Property Inspector.
2. On the General tab, click the field Use Product Default Icon. An Ellipsis button is displayed in this field as shown in Figure 8–8.

![Properties Inspector for EXPENSE_CATEGORIES_CSV](image)

3. Click the Ellipsis button to open the Icon Object dialog box.
4. Select CSV_ICONSET and click OK.
5. CSV_ICONSET is now assigned to the instance EXPENSE_CATEGORIES_CSV. Figure 8–9 shows the Project Explorer in which the icon set is assigned to the instance.

![Icon Set for EXPENSE_CATEGORIES_CSV](image)

Assigning New Icon Sets to Activities in Process Flow

You can assign new icon sets to the activities in a process flow. The Explorer panel of the Process Flow Editor contains the Select Icon button at the top. This button is enabled when you select an activity.

To assign a new icon set to an activity:

1. Select the activity and click the Select Icon button. The Select Icon Set dialog box displays.
2. Select an icon from the list and click Select.
The icon set is assigned to the activity.

Assigning New Icon Sets to Tasks in Expert Editor
You can assign new icon sets to the tasks in Expert Editor using the Select Icon button available on the Expert Editor. This is similar to assigning icon sets to activities in a process flow.

To assign a new icon set to a task:
1. Select the task and click the Select Icon button. The Select Icon Set dialog box displays.
2. Select an icon from the list and click Select.
The icon set is assigned to the task.

For information about experts, see Oracle Warehouse Builder API and Scripting Reference.
Managing Metadata Dependencies

The Metadata Dependency Manager enables you to detect and resolve the impact of
the changes made to the object definitions or the metadata in the workspace.

This chapter contains the following topics:

■ Introduction to the Metadata Dependency Manager
■ Generating an LIA Diagram
■ Modifying the Display of an LIA Diagram
■ Propagating Metadata Changes Throughout Warehouse Builder
■ Propagating Changes in the Dependency Manager

Introduction to the Metadata Dependency Manager

The Metadata Dependency Manager generates lineage and impact diagrams for any
data object. A lineage diagram traces the data flows for an object back to the sources
and displays all objects along those paths. An impact diagram identifies all the objects
that are derived from selected object.

This type of information can help you in many circumstances such as the following:

■ Comply with government regulations or other audits that require tracking of
  financial data.
■ Modify the system design because of changes in the source data.
■ Modify the system design because of new requirements for reporting and analysis.
■ Assess the impact of design changes in a pluggable mapping that is used
  throughout a project.

The Dependency Manager enables you to plan your project by previewing the impact
of the changes or future changes. If you are planning to introduce changes to your
source systems, you can use the Dependency Manager to gauge the impact of that
change on your warehouse design. Or, if the change has already been introduced, then
you can plan the time required to update your ETL design and rebuild your data
warehouse.

Figure 9–1 shows the Metadata Dependency Manager. Like other windows in
Warehouse Builder, the Dependency Manager has menus, toolbars, a navigator, a
property inspector, and a canvas. The canvas displays one or more diagrams.
Introduction to the Metadata Dependency Manager

Figure 9–1 Metadata Dependency Manager

Usage Scenario

The metadata from sources such as files, databases, and applications can change even after the design and implementation of a data integration system. A change in the source metadata implies a corresponding impact in your Warehouse Builder implementation. Warehouse Builder enables you to reimport the modified source definitions into the workspace. However, your original warehouse design may no longer remain valid with the reimported definitions, and you may need to make changes to the design and fix the inconsistencies.

You need to first find out how the warehouse design is effected by the changes in source and then determine all the design objects that are dependent upon the sources must synchronize the metadata so that all the effected design objects are updated to reflect the changes in the source. After this process is complete, you can redeploy the updated design to rebuild your data warehouse and synchronize the data.

In this scenario, a company retrieves all its statistical reports from a flat file named CUSTOMERS. Over a period of time, the file definition needs to be changed to generate reports on additional parameters. The statistical analysis database hosted on the Oracle Database has also recently migrated to the latest version of the Database.

The designers at this company first need to synchronize the modified metadata definitions for this file in the design workspace with all the design objects and mappings that are based on its data. The CUSTOMERS flat file is part of multiple mappings in the ETL design, and any changes to its definitions invalidate all them. Manually tracking all the areas impacted by the CUSTOMERS flat file and updating the definitions is a process prone to errors.

The Dependency Manager enables them to identify all of the objects that are effected by this change to the data source. They can be sure that each related design object is updated so that the ETL designs remain valid. After finishing this process, they can redeploy the effected tables and the updated ETL design objects to their data warehouse.
What are Lineage and Impact Analysis Diagrams?

Lineage and Impact Analysis (LIA) diagrams show the relationships among objects managed by Warehouse Builder. These relationships are constructed by mappings and structural relationships (for example, a primary key and foreign key relationship). The lineage diagram for a particular object shows its source objects, and the impact diagram shows its targets.

Lineage and impact are mirror images of each other. If Object A is part of the lineage diagram of Object B, then Object B is part of the impact diagram of Object A. When you read a diagram from left to right, you are seeing impact. When you read it from right to left, you are seeing lineage.

For example, you might have a mapping that extracts data from a file and loads it into a table by way of an external table. This is the relationship:

flat_file > external_table > table

Figure 9–2 shows a lineage diagram of an external table named ADDRESS_EXT_TABLE. ADDRESS_CSV is a flat file, and it is part of the lineage of ADDRESS_EXT_TABLE. Thus, any change to ADDRESS_CSV will impact ADDRESS_EXT_TABLE.

Figure 9–2 Lineage Analysis Diagram for ADDRESS_EXT_TABLE

Figure 9–3 shows an impact diagram of ADDRESS_EXT_TABLE, which includes the ADDRESS_TBL. Any change to ADDRESS_EXT_TABLE will impact ADDRESS_TBL. ADDRESS_EXT_TABLE is part of the lineage of ADDRESS_TBL.

Figure 9–3 Impact Analysis Diagram for ADDRESS_EXT_TABLE

You can see both the lineage and the impact of an object by clicking the plus signs (+) on either side of the object icon, as shown in Figure 9–4.
Generating an LIA Diagram

You can generate an LIA diagram from the Project Explorer in the Design Center or by first opening the Metadata Dependency Manager.

To generate a diagram from the Design Center:
1. Expand the Project Explorer until you see the object that you want to analyze.
2. Right-click the object and select Lineage or Impact.

The Metadata Dependency Manager will open with the diagram.

The Lineage and Impact commands are also available from the View menu.

To generate a diagram from the Metadata Dependency Manager:
1. From the Design Center Tools menu, select Metadata Dependency Manager.

The Metadata Dependency Manager is displayed.
2. Expand the DM Context Tree until you see the object that you want to analyze.
3. Right-click the object and select Show Lineage or Show Impact.

The diagram will be displayed on the canvas.

Modifying the Display of an LIA Diagram

Your initial selection of an object and a diagram type simply determine the initial starting point and the direction that the diagram branches from that object. You can modify an LIA diagram in the following ways:

- Drag-and-drop another object onto the diagram.
- Click the plus (+) and minus (-) signs next to an object icon to expand or collapse a branch of the diagram.
- Use the grouping tool to collapse a section of the diagram into a single icon, as described in "Using Groups in an LIA Diagram" on page 9-5.
- Double-click an object to display its attributes, as described in "Displaying an Object's Attributes" on page 9-5.
Using Groups in an LIA Diagram

Groups enable you to organize the objects in a complex diagram so that they are easier to locate and edit. By reducing the number of objects in a diagram, you can more easily focus on the objects currently of interest.

To create a group:
1. Select a group of objects by dragging and dropping a box around them.
2. Click the Group Selected Objects tool.
   The Group Selected Data Objects dialog box is displayed.
3. Enter a name for the group.
   The selected objects are collapsed into a single folder icon.

To display the individual objects in a group, double-click the folder icon. You can work on these objects in the same way as ungrouped objects.

To ungroup the objects, select the group and click the Ungroup Selected Object tool.

Displaying an Object’s Attributes

You can expand an object icon in a diagram so that you can examine its attributes. To expand an icon, double-click it. To reduce it to an icon, click the down arrow in the upper right corner.

To generate an LIA diagram for an attribute:
1. Generate an LIA diagram for an object.
2. Double-click the icons to display their attributes.
3. Right-click an attribute and select Show Lineage or Show Impact.
   The attributes along the lineage or impact path for the selected attribute are highlighted in a different color.

You may use this detailed information for auditing or when planning to propagate changes.

Figure 9–5 shows two expanded icons whose column attributes are connected by a mapping.

Figure 9–5 Expanded Icons in an LIA Diagram
Propagating Metadata Changes Throughout Warehouse Builder

The LIA diagrams identify all of the objects that may be invalidated by a change to one or more objects. With this knowledge, you can examine the effected objects and modify them as necessary.

You can either modify objects manually or instruct the Dependency Manager to propagate the changes for you.

To manually modify objects:
1. In the Dependency Manager, navigate to the first object to be changed. For example, navigate to a source table.
2. Right-click the object icon in a diagram and select Open Editor. Warehouse Builder opens the editing tool for the object. For example, if you selected a table, then Warehouse Builder open the Data Object Editor.
3. Make the necessary changes in the editing tool and then save your changes.
4. Repeat these steps for all objects identified in the LIA diagram as needing change.

In the case that only a few objects are effected by a change, then you may prefer to modify the object manually. However, if many objects are effected, you can reduce the possibility of human error by utilizing the propagate change functionality.

To instruct the Dependency Manager to propagate metadata changes for you:
1. Double-click the object icon in a diagram. For example, double-click on the icon for a source table.
2. Right-click the metadata that you want to change, and select Propagate Change. For example, right-click on a column in the table.
3. Change the attributes as described in "Propagating Changes in the Dependency Manager".
4. Click OK.

Propagating Changes in the Dependency Manager

In the Propagate Change dialog box, you specify metadata changes which Warehouse Builder propagates to all dependent objects, as indicated in the Lineage Impact Analysis diagram.

Begin by marking Propagate for each attribute you want to change. For example, if you want to change the data type and length for a column, ensure that there are two check marks under Propagate.

Go to New Value and type the desired values for each attribute.

About the Metadata Dependency Manager User Interface

The Metadata Dependency Manager is a graphical interface that represents the potential impact of a change in the definition of an object. It has the following components:

- Menu Bar
- Toolbars
- DM Tree
About the Metadata Dependency Manager User Interface

- Property Inspector
- Canvas
- Bird’s Eye View

Menu Bar
The Metadata Dependency Manager menu bar provides commands for performing various tasks. Some of these commands are also available on the toolbars.

Analysis
The Analysis menu contains the following commands:
- **Close**: Closes the Dependency Manager.
- **Export Diagram**: Exports the active diagram to the local file system as an SVG or JPEG file.
- **Print Options**: Provides Print Setup, Preview, and Print options for printing the diagram.

Edit
The Edit menu contains the following commands:
- **Open Editor**: Opens the Editor for the currently selected object.
- **Hide**: Removes the selected objects from the canvas. Use the Refresh command on the View menu to restore the hidden objects.
- **Select All**: Selects all objects displayed on the canvas.
- **Propagate Changes**: Displays the Propagate Changes dialog box for a selected attribute. Use it to change the value of an attribute and to propagate that change to all objects downstream. For example, you can select a column in a table object and change its metadata such as its name or data type.
- **Group Selected Objects**: Creates a group containing the selected objects on the canvas. A folder icon represents all objects in the group. Double-click the icon to display the individual objects in the group. Grouping enables you to reduce clutter on the canvas when there are many objects.
- **Ungroup Selected Objects**: Eliminates the selected group so that all objects are represented individually.
- **Group By Module**: Automatically groups all objects by module. A folder icon represents the module and all objects in the module. Double-click the icon to display the individual objects in the group.
- **Ungroup Modules**: Eliminates the module groups so that all objects are represented individually.

View
The View menu contains the following commands:
- **Toolbars**: Displays or hides the Graphic tools or the Edit tools.
- **Mode**: Sets the pointer for one of these actions:
  - **Select**: Selects one or more objects on the canvas.
  - **Pan**: Moves the entire diagram on the canvas.
About the Metadata Dependency Manager User Interface

- **Interactive Zoom**: Expands the diagram as you move the pointer down, or shrinks the diagram as you move the pointer up.
- **Navigate Edge**: Selects the next object in the flow.
- **Zoom**: Displays a list of percentages that expand or shrink the diagram.
- **Fit in Window**: Automatically chooses a size for the diagram so that it fits on the canvas.
- **Auto Layout**: Organizes the objects and displays the diagram at its default size.
- **Center**: Centers the diagram on the canvas.
- **Show Full Impact**: Generates the full impact diagram of the selected object.
- **Show Full Lineage**: Generates the full lineage diagram of the selected object.
- **Show Lineage**: Displays the next level of objects in the lineage diagram of the selected object.
- **Hide Lineage**: Hides the lineage of the selected object.
- **Show Impact**: Displays the next level of objects in the impact diagram of the selected object.
- **Hide Impact**: Hides the impact of the selected object.
- **Expand**: Expands the selected icon in the diagram. The expanded icon shows the details of the object, such as the columns in a table or the attributes in a dimension.
- **Expand All**: Expands all the object icons in the diagram.
- **Collapse**: Collapses the expanded icon for a selected object.
- **Collapse All**: Collapses all the expanded icons on the canvas.
- **Refresh**: Refreshes the Dependency Manager diagram to reflect recent changes in the workspace.

**Window**
The Metadata Dependency Manager Window menu contains commands for toggling between displaying and hiding the following windows:

- **Property Inspector**
- **Bird's Eye View**
- **Tree View**

**Toolbars**
The Metadata Dependency Manager provides two toolbars as shortcuts to frequently used commands:

- **Graphic toolbar**: Provides icons for commands on the View menu. Refer to "View" on page 9-7 for descriptions of these commands.
- **Edit toolbar**: Provides icons for commands on the Edit menu. Refer to "Edit" on page 9-7 for descriptions of these commands.

**Bird's Eye View**
Use the Bird's Eye View to quickly change the portion of the diagram currently displayed on the canvas. This view displays a miniature version of the diagram on the canvas, with a scrollable box that represents the dimensions of the canvas. Drag the box to the area of the diagram currently of interest to you.
DM Tree
Use the DM Tree to change the content of the canvas. The DM Tree has these tabs:

- **DM Context Tree**: Lists objects in the current project. Right-click an object, and use the menu to generate a new diagram or to add the object to the current diagram. You can also drag-and-drop an object onto the current diagram.

- **DM Graph Tree**: Lists the current diagrams. You can close a diagram on the canvas, then use this list to redisplay the diagram without regenerating it.

Property Inspector
Use the Property Inspector to view an object’s properties.

To change the current object in the Property Inspector, right-click the object on the canvas and select **Update Property Inspector**.

For a description of a property, select the property in the Inspector. The description appears at the bottom of the window.

Canvas
Use the canvas to display one or more lineage and impact diagrams. Each diagram is displayed on a separate tab.

You can use these techniques to create and manipulate diagrams on the canvas:

- To open a new diagram, right-click an object in the DM Context Tree and select **Show Lineage** or **Show Impact**.

- To close a diagram, click the X on the tab. You can redisplay the diagram from the DM Graph Tree until you close the Metadata Management window.

- To add an object to the canvas, drag-and-drop it from the DM Context Tree.

- To display more of the lineage of an object, click the plus sign (+) to the left of its icon. To hide the lineage, click the minus sign (-) to the left.

- To display more of the impact of an object, click the plus sign (+) to the right of its icon. To hide the impact, click the minus sign (-) to the right.

- To display the details of an object, double-click it. You can then select an individual property and edit it by choosing **Propagate Changes** from the Edit menu.

- To edit an object, right-click its icon and select **Open Editor**.
Warehouse Builder provides several solutions for copying and moving metadata for the purposes of backup, history management, and version management. You can either take snapshots of your metadata or use the Metadata Loader (MDL) utility.

**Note:** A third solution is to utilize Transportable Modules, which requires the licensing of the Warehouse Builder Enterprise ETL Option. Refer to Chapter 12, “Moving Large Volumes of Data with Transportable Modules” for more information.

Snapshots enable you to capture the metadata definitions of design objects created in the Project Explorer. You can capture all the design objects in a project or selectively choose objects in a project to include in a snapshot.

The Metadata Loader utility enables you to copy and move all types of metadata objects in a repository. With this utility, you can move metadata between Oracle Database repositories that reside on platforms with different operating systems.

With both solutions, you can export the metadata into a third-party version control tool such as Oracle Repository, ClearCase, or SourceSafe.

This chapter includes the following topics:

- Snapshots Versus the Metadata Loader
- About Snapshots on page 10-2
- Version and History Management with the Metadata Loader (MDL)
- Using Metadata Loader in the Design Center

### Snapshots Versus the Metadata Loader

Because snapshots and the Metadata Loader have overlapping capabilities, it may not be readily apparent when to use one versus the other. In general, snapshots address the needs a development team for tracking changes to an evolving design; the Metadata Loader facilitates the more traditional administrative tasks of backing up, restoring, and migrating repositories.

A key factor to understand is that snapshots and the Metadata Loader differ in the scope of objects they handle. Snapshots are limited to handling only the design objects users create in the Project Explorer. The Metadata Loader, however, handles everything in the repository including all design objects and information related to location and security.

Refer to Table 10–1 for a list of important differences between the two tools.

#### Table 10–1 Snapshots Versus the Metadata Loader

<table>
<thead>
<tr>
<th>Feature</th>
<th>Snapshots</th>
<th>Metadata Loader</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Captures metadata related to objects in the Project Explorer only. Captures metadata for mappings, process flows, and other design objects. Does not capture information about locations, security, and user defined objects.</td>
<td>Captures all metadata related to a repository including all design objects and information about locations, security, and user defined objects.</td>
</tr>
</tbody>
</table>
Using snapshots, you can backup and restore your metadata, maintain a history of metadata changes, and compare different versions of the metadata.

A snapshot captures all the metadata information about the selected objects and their relationships at a given point in time. While an object can only have one current definition in a workspace, it can have multiple snapshots that describe it at various points in time. Warehouse Builder supports the following types of snapshots:

- **Full snapshots** provide backup and restore functionality.
- **Signature snapshots** provide historical records for comparison.

Snapshots are stored in the Oracle Database, in contrast to Metadata Loader exports, which are stored as separate disk files. You can, however, export snapshots to disk files.

Snapshots can be useful to both warehouse managers and designers. Managers can use full snapshots to perform large-scale actions, such as deploying a warehouse or restoring a warehouse to a previous point in history. Designers can create full snapshots of a particular component under development so that, if necessary, they can restore the component to its previous state. They can also use signature snapshots to track the changes made to a particular component.

When used with other facilities, such as MDL metadata imports and impact analyses, snapshots can help you manage your metadata. For example, you can use snapshots to determine the impact an MDL metadata import will have on the current metadata workspace. With the knowledge gained by the comparison, you might import the metadata at a lower level of detail to avoid overwriting the definition of related metadata objects.
Snapshots are also used to support the recycle bin, providing the information needed to restore a deleted metadata object.

This chapter describes the metadata change management feature using the graphical user interface. For information about creating and managing snapshots using scripts, see Oracle Warehouse Builder API and Scripting Reference.

Creating Snapshots

When you take a snapshot, you capture the metadata of all or specific objects in your workspace at a given point in time. You can use a snapshot to detect and report changes in your metadata.

You can create snapshots of any objects that you can access from the Project Explorer. Note that a snapshot of a collection is not a snapshot of just the shortcuts in the collection but a snapshot of the actual objects.

Opening the Create Snapshot Wizard

To open the Create Snapshot wizard and create a snapshot:

1. In the Project Explorer, select all the components you want to include in the snapshot. You do not need to select parent or child objects, because the wizard does that for you.
   
   For example, if you select a collection that contains two cubes, then the snapshot includes both cubes and any gives you the option to include any dependent objects such as dimensions and source tables.

2. Right-click and select Snapshot, then New.
   
   or

   From the Design menu, select Snapshot, then New.

3. On the Name page, specify a name and the type of snapshot.
   
   You can optionally provide a description.

4. Click Next to open the Components page of the wizard.
   
   The page displays the components to be captured in the snapshot. If a component is a node level object, such as a module, then select the Cascade option to include the subcomponents.

5. On the Dependency page, specify the depth of dependency to include dependent objects in the snapshot.

6. Click Next to view the Finish page which provides the details of the snapshot.

7. Click Finish to create the snapshot.

Adding Components to a Snapshot

After you create a snapshot, you can add more components. Keep in mind, however, that when you add components, you are changing the historical record provided by the snapshot, so its creation date changes.

To update a snapshot, use the Add to Snapshot wizard.

Opening the Add to Snapshot Wizard

To open the Add to Snapshot wizard and add new components:
1. In the Design Center Project Explorer, select all the components you want to add to the snapshot. For example, you can select tables, mappings, and dimensions from an Oracle module.

2. From the Design menu, select **Snapshot**, then **Add to Existing**.
   
   **or**

   Right-click and select **Snapshot**, then **Add to Existing**.

   The Welcome page of the Add to Snapshot wizard is displayed. The Welcome page lists the steps of the wizard.

3. Click **Next**.

   The Snapshot page is displayed. From the list, select the snapshot to which you want to add the components.

4. Click **Next**.

   The Components page is displayed. The page displays the components to be added to the snapshot. If a component is a folder level object, such as a module, then select the Cascade option to include the subcomponents.

5. Click **Next**.

   The Finish page displays the details of the components to be added.

6. Click **Finish** to add the components to the snapshot.

**Managing Snapshots**

You can manage your snapshots from the Metadata Change Management window in the Design Center. To open this window, select **Change Manager** from the Tools menu.

The Metadata Change Management window contains a menu bar and a toolbar. You can start most tasks in several different ways, either by using the menus, clicking the tools, or right-clicking a snapshot or a component.

You can perform the following activities from the Metadata Change Management window:

- Managing Snapshot Access Privileges
- Comparing Snapshots
- Restoring Workspace Objects From Snapshots
- Exporting and Importing Snapshots
- Deleting Snapshots

Figure 10–1 shows the Metadata Change Management window.
Managing Snapshot Access Privileges

You can control access to snapshots just like any other object. By default, everyone has full access rights to the snapshots.

To change access privileges:

1. In the left section of the Metadata Change Management window, right-click the name of the snapshot.

2. Click Security from the menu.

   The Snapshot Privilege Management dialog box is displayed.

3. For each role and user, select the privileges you wish to grant and clear the privileges you wish to deny. Click Help for additional information.

Comparing Snapshots

You can compare two snapshots or a snapshot with a current workspace object. The results list all objects and identify which objects are identical, which objects appear only in one place, and which objects appear in both but are not the same.

If you take a snapshot of a parent object and a child object changes, a snapshot comparison will show the parent object as having changed. For example, if you create a Cascade snapshot of a project, all the modules in that project are its children. If any of the modules change, a comparison of the snapshot to the workspace will show that the project has changed.

The Metadata Change Manager uses Universal Object Identifiers (UOIDs) as the basis for all comparisons. When you delete an object and re-create it with the same metadata, the object has a different UOID although it may be identical to the deleted object in all other aspects. When you compare the re-created object to its original version, the results show that all the metadata has changed. Likewise, objects will not
match if they have been deleted and re-created during an import or Intelligence Object derivation.

Comparing Two Snapshots
Use one of the following methods:

1. From the Metadata Change Management window, select Compare from the Snapshot menu.

   *Or*

   Select the two snapshots you want to compare, right-click, and select Compare. The Compare Two Snapshots dialog box is displayed. Snapshots that were selected in the Metadata Change Management window are also selected in the Compare Two Snapshots dialog box.

2. If you have not selected two snapshots for comparison yet, select one from each list.

3. Click Compare.

   The Snapshot Comparison window displays the differences between the two objects. If there are none, then a message informs you that the objects are the same. For more information, click Help.

Comparing a Workspace Object with a Snapshot Component
To compare the current version of an object with a snapshot version:

1. Select the object you want to compare from the Design Center Project Explorer.

2. Right-click, select Snapshot, and then Compare.

   The Choose Snapshot dialog box is displayed with a list of all the snapshots containing versions of that object.

3. Select the snapshot you want to compare with the current workspace object.

4. Click OK.

   The Snapshot Comparison window displays the differences between the two objects. If there are none, then a message informs you that the objects are the same.

Converting a Full Snapshot to a Signature Snapshot
You can convert full snapshots to signature snapshots when they are no longer needed for backup. Conversion preserves the snapshot history and results in significant space savings in your workspace.

**Note:** Converting a full snapshot to a signature snapshot means that you can no longer use that snapshot to restore metadata objects.

To convert a full snapshot:

1. Open the Metadata Change Management window, as described in "Managing Snapshots" on page 10-4.

2. Select the snapshot that you want to convert.

3. From the Snapshot menu, select Convert to Signature.
4. On the Warning box, click Yes.

Restoring Workspace Objects From Snapshots

You can replace the current definition of an object in the workspace with the snapshot image of that object. You can only use full snapshots; you cannot restore objects from signature snapshots. You can restore all components or only selected components of the snapshot.

To restore objects from a snapshot:
1. Save any work that you have done in the current session.
2. Open the Metadata Change Management window, as described in “Managing Snapshots” on page 10-4.
3. Select the snapshot that contains the version of the objects you want to restore.
4. To restore all objects, right-click the snapshot and select Restore.
   or
   To restore selected components, select them on the Components tab, right-click, and select Restore.
   The Restore Snapshot dialog box is displayed.
5. Review the objects in the list and verify that the correct objects will be restored.
6. Select Cascade Up to restore a component whose parents no longer exist in the workspace. Warehouse Builder creates new parent objects for the component. Otherwise, the procedure will fail.
7. Click Restore.

Exporting and Importing Snapshots

You can export a full snapshot to an MDL file and use this file to re-create metadata objects either in the same database or a different one. You cannot export signature snapshots or individual components of a snapshot.

To export a snapshot:
1. Open the Metadata Change Management window and select the full snapshot you want to export.
2. From the Snapshot menu, select Export.
   The Metadata Export dialog box is displayed.
3. Click Help for information about completing the dialog box.

To import a snapshot:
1. Open the Design Center and select Import from the Design menu.
2. Select **Warehouse Builder Metadata** from the menu.
   The Metadata Import dialog box is displayed.

3. Click **Help** for information about completing the dialog box.
   The imported snapshot will be listed in the Metadata Change Management window.

**Deleting Snapshots**

You can delete snapshots or components within a snapshot from the Metadata Change Management window.

**To delete a snapshot:**
1. Open the Metadata Change Management window and select the snapshot you want to delete.
2. Right-click the snapshot and select **Delete**.
   
   *Or*
   
   From the Snapshot menu, select **Delete**.
   Warehouse Builder displays a delete confirmation box.

3. Click **Yes** to delete the selected snapshot from the workspace.

**To delete components from a snapshot:**
1. Open the Metadata Change Management window and select the snapshot from which you want to delete components.
2. On the Components tab, select the components, right-click, and select **Delete**.
   
   *Or*
   
   From the Snapshot menu, select **Delete**.
   Warehouse Builder displays a delete confirmation dialog box.

3. Click **Yes** to delete the selected component from the snapshot.

**Version and History Management with the Metadata Loader (MDL)**

Using the Metadata Loader (MDL) utility, you can import and export metadata from any object in the Project Explorer, Global Explorer, and Connection Explorer. You can then move exported files into a third-party version control tool such as Oracle Repository, ClearCase, or SourceSafe. You can enter annotations for your MDL export file to keep track of the information contained in the file.

The Metadata Loader (MDL) enables you to populate a new repository and transfer, update, or restore a backup of existing repository metadata. You can copy or move metadata objects between repositories, even if those repositories reside on platforms with different operating systems.

**Accessing the Metadata Loader**

You can access the Metadata Loader using either the graphical user interface described in this chapter or using the OMB Plus scripting language described in Oracle Warehouse Builder Scripting Reference.
While the graphical interface guides you through the most commonly performed export and import tasks, the OMB Plus scripting language enables you to perform more specialized export and import tasks and enables you to manage a control file.

**About Metadata Loader Control Files**

When you use the OMB Plus commands related to the Metadata Loader, a control file provides you with greater control over how objects are imported or exported. For example, by default, the Metadata Loader exports objects into a binary zip format. To override the default of exporting to a zip file, use `OMBEXPORT MDL_FILE` with the `CONTROL_FILE` option with a control file that contains the option `ZIPFILEFORMAT=N`.

For more information about using a control file with the Metadata Loader, refer to the appendix in Oracle Warehouse Builder Scripting Reference. For information about each command, look up the following commands also in the Oracle Warehouse Builder Scripting Reference.

- OMBIMPORT
- OMBEXPORT
- OMUIMPORT
- O MUIMPORT MDL_FILE
- OMBIMPORT MDL_FILE
- OMUEXPORT MDL_FILE
- OMBEXPORT MDL_FILE

**Using Metadata Loader in the Design Center**

You can use the Design Center to run the Metadata Loader utilities. The Design Center provides a graphical interface that guides you through the process of exporting and importing metadata.

**Exporting Metadata from the Design Center**

You can use Design Center to export objects from a workspace into an MDL file. This includes objects that are part of the Project Explorer, Connection Explorer, and Global Explorer. The information related to the exported objects, such as table columns and their constraints, data loading configuration parameters, and named attribute sets, are also exported.

You have two option for exporting metadata. You could either export the entire design or you could export selected objects in the workspace. If you want to export select objects, then those objects must be within the same explorer. For example, you cannot export a table from the Project Explorer and a public transformation from the Global Explorer at the same time. You can export them in two separate steps.

**To export metadata from a workspace using the Design Center:**

1. Ensure that you are the only user accessing the objects to be exported.

   To ensure that you are exporting the most up-to-date metadata, it is advisable to ask all other users to log out of the workspace. For more details, see "Multiple Session Concurrency and MDL" on page 10-23.
Also ensure you have the READ privilege on the objects to be exported. For
details, see “Access Privileges Required to Export Metadata” on page 10-10.

2. From the Design Center, select the object or objects you want to export.

You can select multiple objects by holding down the Ctrl key and selecting the
objects.

You can export individual objects such as tables or groups of objects. When you
export projects nodes, or modules, you also export the objects they contain. When
you export collections, you also export the objects they reference.

3. From the Design menu, select Export and then Warehouse Builder Metadata.

If you made changes to the repository metadata prior to running the export utility,
a warning dialog box is displayed. Click Save to save changes or Revert to revert
to the previously saved version.

If you have not made any changes to the repository metadata after last saving the
design, the Metadata Export Dialog Box is displayed.

4. Ensure the destination computer has sufficient disk storage.

If you lack sufficient disk space on the computer to which you export the
metadata, the export fails. Your destination computer must be able to contain the
entire metadata file. The export utility cannot save portions of the metadata file.

Access Privileges Required to Export Metadata
Before you attempt to export metadata, ensure you have READ privileges on any
object that you want to export. You also need to have READ privileges on folder
objects. If you do not have READ privileges on a folder object, such as projects or
modules, the folder object and all objects that it contains will not be exported. During
an export, the Metadata Export Utility skips objects for which you do not have READ
privileges. It logs information about the list of objects that have not been exported due
to lack of security privileges in the Metadata Loader Log File.

By default, READ privileges are provided on all the workspace objects to all registered
users. If you want to export security information such as users, roles, role assignments,
and object privileges, see “Export Advanced Options Dialog Box” on page 10-11.

Metadata Export Dialog Box
The Metadata Export dialog box displays the names and the types of objects being
exported. It also contains the following:

- **Annotations**: Use this field to enter any comments about the file that contains the
  exported objects.

- **File Name**: Displays a default path and file name for the export file. You can retain
  this default or specify a directory and file name. Type the name of the export file to
  create or click Browse to locate a directory or file. The file name extension
  commonly used is .mdl.

- **Log File**: Use this field to specify the file name and path for the log file that stores
  diagnostic and statistical information about the export. For more information
  about log files, see “Metadata Loader Log File” on page 10-23.

- **Export all object dependencies**: Select this option to export all the dependencies
  of the objects being exported. For example, when you export a table, the location
  to which the table is deployed is also exported.
Using Metadata Loader in the Design Center

Version and History Management

Advanced: Use the Advanced button to export additional metadata such as user-defined properties, security information, and additional languages. For more information about the advanced options, see "Export Advanced Options Dialog Box" on page 10-11.

Click Export to export the metadata for the selected objects. The Metadata Export Progress dialog box is displayed. For more information about the contents of this dialog box, see "Metadata Progress Dialog Box" on page 10-11.

Metadata Progress Dialog Box  The Metadata Progress dialog box displays a progress bar that indicates the progress of the metadata export, import, or upgrade activity. If the export or import is successful, a message indicating this is displayed just above the progress bar. An error in the process is displayed too.

To view detailed information about the metadata export or import, click Show Details. The message log is displayed. The message log contains the following information:

- Start time of the export or import
- Names and types of objects exported or imported
- Warning or error messages
- End time of the export or import
- Location of the export or import log file
- Total export or import time in hh:mm:ss or milliseconds

You can hide the message log by clicking Hide Details.

To view details about the exported or imported objects, click Show Statistics. The Metadata Export Results dialog box is displayed. For more information about this dialog box, see "About Metadata Loader Results" on page 10-24.

Once the export or import completes, the Close button is enabled. To exit the Metadata Export or Metadata Import Utility, click Close.

Export Advanced Options Dialog Box
Use the Export Advanced Options dialog box to export any of the following:

- Additional language metadata
- User-defined definitions
- Security information

This dialog box contains two sections: Languages and Administration.

Languages
The Base Language field displays the base language of the repository. Warehouse Builder exports data in the base language.
You can specify additional languages to export for the objects that contain translations for their business names and descriptions. The Available Languages list displays the list of languages that are installed in the repository. To export additional languages, select the language and click the arrows to move the language from the Available Languages list to the Selected Languages list. You can choose multiple languages at the same time by holding down the Ctrl or Shift key while making your selection.

---

**Note:** The Available Languages list will contain language entries only if you installed additional languages in the repository.

---

For example, the repository has the base language as American English and additional languages Spanish and French. While exporting metadata from the repository, you can select French as the additional language. The Metadata Export Utility then exports the base language of the object, American English, and the additional language French for objects that contain a French translation. Note that additional languages will be exported for an object only if they contain translations for the business names and descriptions.

**Administration**

You can export additional metadata if you have administrator privileges. The options you can choose to export additional metadata are as follows:

- **Export user-defined definition:** Select this option to export the definitions of user-defined objects and user-defined properties. This option is enabled only if you have created any user-defined objects or properties for the objects being exported.

- **Export security information:** Select this option to include security information such as object privileges or role assignments made to users. For more information about security, see *Oracle Warehouse Builder Installation and Administration Guide*.

After you specify the options on the Export Advanced Options dialog box, click OK to close this dialog box and return to the Metadata Export dialog box.

**Importing Metadata Using the Design Center**

You can use the Design Center to import metadata. The Metadata Import Utility also automatically upgrades metadata that was created using an earlier version of Warehouse Builder to the current version. For more information about upgrading metadata, see “Upgrading Metadata from Previous Versions” on page 10-19.

**Before Importing Metadata**

Before you attempt to import metadata, ensure you have the following:

- **Required access privileges:** To import metadata, the user performing the import must have the following privileges:
  - **EDIT** privilege on existing objects that are being replaced by the import.
  - **CREATE** privilege on existing folder objects under which new objects will be created by the import.

By default, the **FULL_CONTROL** privilege is assigned on all workspace objects to registered users. The Metadata Import Utility skips objects for which the user importing metadata does not have the required privileges. The list of objects that have not been imported due to security privileges are logged to the Metadata Loader Log File.
You can import security information such as users and roles as described in "Import Advanced Options Dialog Box" on page 10-16. When importing user metadata, if a corresponding database user does not exist for a user, the import will fail and an error message is written to the Metadata Loader Log File.

Because the Metadata Import Utility is altering the repository, the metadata objects must be locked prior to importing. For more details, see "Multiple Session Concurrency and MDL" on page 10-23.

- **A backup of your current workspace**: Consider taking a backup of your existing workspace (either in the form of an export or a metadata snapshot) before attempting a large or complex import.

- **Multiple Language Support base language compatibility**: The base language is the default language used in the repository and is set using the Repository Assistant during installation. You cannot alter this setting after installing the repository. For more information about setting the base language in a repository, see Oracle Warehouse Builder Installation and Administration Guide.

**To import objects from an export file using the Design Center:**

1. From the Design Center, select **Design, Import**, and then Warehouse Builder Metadata.

   If you had made changes to the repository metadata prior to running the import utility, a warning dialog box is displayed. Click **Save** to save changes or **Revert** to revert to the previously saved version.

   If you have not made any changes to the repository metadata after last saving the design, the Metadata Import dialog box is displayed.

**Metadata Import Dialog Box**

Use the Metadata Import dialog box to specify the information required to import metadata contained in an export file. Specify the following information on this dialog box:

- **File Name**
- **Log File**
- **Object Selection**
- **Import Option**
- **Match By**

Click **Advanced** to import metadata for additional languages, security information, or user-defined properties. For more information about advanced options, see "Import Advanced Options Dialog Box" on page 10-16.

Click **Show Summary** to view a summary of the export file contents. For more information about the export file contents, see "File Summary Dialog Box" on page 10-17.

After specifying the options on the Metadata Import dialog box, click **Import** to import the metadata from the MDL file. The Metadata Import Progress dialog box that indicates the progress of the import is displayed. For more information about this dialog box, see "Metadata Progress Dialog Box" on page 10-11.
**Note:** If the MDL file that you selected for import was created using an earlier version of Warehouse Builder, clicking **Show Summary**, **Advanced**, or **Import** displays the Metadata Upgrade dialog box. This dialog box enables you to automatically upgrade the selected MDL file to the current version of Warehouse Builder. For more information about this dialog box, see "Metadata Upgrade Dialog Box" on page 10-20.

**File Name**  
Type the name of the MDL file or click **Browse** to locate the MDL file you want to import.

**Log File**  
Type the name of the log file, along with the path, that will store diagnostic and statistical information about the import. You can also click **Browse** to locate the log file. For more information about log files, see "Metadata Loader Log File" on page 10-23.

**Object Selection**  
The Metadata Import Utility enables you to select the objects that you want to import from the MDL file. The Object Selection section contains the following options:

- **Import all objects from file**  
  Select this option to import all objects contained in the export file.

- **Import selected objects from file**  
  Select this option to import only some of the objects contained in the MDL file. Click **Select Objects** to select the objects that you want to import. The Import Object Selection dialog box is displayed. This dialog box contains two sections: Available and Selected. The Available section contains the primary objects such as projects, modules, tables, views, connections that are specified in the MDL file. Expand the nodes in this section to view the objects they contain. When you select a node, all the objects that it contains are included in the import. For example, if you select a module node, all the objects contained in the module are imported. Use the shuttle buttons to move the selected objects from the Available section to the Selected section.

  The MDL file being imported can also contain administrative objects. To import these administrative objects, the user performing the import must have administrative privileges. If the user performing the import does not have the required privileges:

  - If the MDL file contains some administrative objects, the Available section of the Import Object Selection Page dialog box does not displays these objects.
  - If the MDL file contains only administrative objects, the Import Utility displays an alert informing that the user does not have the required administrative privileges to perform the import.

**Import Option**  
Use the Import Option section to select the import mode. You can select one of the following options for the import mode:

- **Create new metadata only**  
  This option adds new objects to a workspace. It is referred to as the create mode.

- **Update metadata (replace existing objects and create new metadata)**
This option is referred to as the update mode. Selecting this option adds new objects to a workspace and replaces existing objects with those in the MDL file being imported.

- **Merge metadata (merge existing objects and create new metadata)**

  When you select this option, the MDL adds new objects and overwrites existing objects in the workspace only if they differ from those in the MDL file. This option is referred to as the merge mode. The merge mode does not delete existing objects.

  **Note:** You cannot import metadata using the Merge mode for mappings, pluggable mappings, and data auditors.

- **Replace existing objects only**

  Selecting this option replaces existing objects in your workspace but does not add new objects. When importing metadata objects, the Metadata Import Utility overwrites any existing metadata when you use this mode. This mode is called the replace mode.

  When you import metadata using the Update or the Replace modes, the import completely replaces the child objects of existing objects so that the final object is exactly the same as the source object. Any existing children of a repository object that are not replaced or added are deleted. This occurs regardless of whether a child object occurs in a mapping or is a foreign, primary, or unique key column in a table or view.

  For example, in the MDL export file, the CUST table contains three columns with the physical names: `last_name`, `first_name`, and `middle_init`. In the workspace, the same table already exists, and contains four columns with the physical names: `last_name`, `first_name`, `status`, and `license_ID`. During a replace operation, the columns `last_name` and `first_name` are replaced, column `middle_init` is added, and column `status` and `license_ID` are deleted. The final result is that the CUST table in the workspace contains the same metadata from the CUST table in the export file.

  **Tip:** Using the replace and update modes can result in lost data constraints, metadata physical property settings, data loading properties, and mapping attribute connections. If you choose to use replace or update modes, ensure that you can restore your workspace, from a backup, to that state it was in prior to importing metadata in replace or update mode.

**Match By**

When you use the metadata import utility, it first searches the workspace for metadata objects that exist in the workspace and compares them to those in the file you are importing. To compare metadata in the import file with the existing workspace metadata, it uses the matching criteria. How the comparison is made is determined by the import mode and by the search method you choose.

The Match By section provides the following options for matching criteria:

- **Universal Identifier:** Searches your workspace using the Universal Object Identifiers (UOIDS) of the objects you are importing. The Metadata Import Utility uses these UOIDS to determine whether an object needs to be created, replaced, or merged during the import operation. Use this method if you want to maintain
Using Metadata Loader in the Design Center

UOIDs across different workspaces even when object names in the target workspace have changed.

- **Names**: Searches your workspace using the names of the objects that you are importing. Physical names are exported to the export file. The physical name determines whether an object needs to be created, replaced, or merged during an import operation. Use this method when object names in the target schema change, and you want to create new UOIDs for those objects.

By default, the import utility searches by UOIDs.

---

**Note**: MDL import does not support merging existing mappings.

---

**Import Advanced Options Dialog Box**

Use the Import Advanced Options dialog box to import any of the following:

- Additional language metadata
- User-defined definitions
- Security information such as object privileges and role assignments to users

The Import Advanced Options dialog box contains the following sections: **Languages** and **Administration**.

**Languages**

The Base Language displays the base language of the repository. By default, data is imported in the base language.

You can specify additional languages to import. The Metadata Import Utility imports the translations of the object for business name and description. The Available Languages list displays the list of languages that are specified in the MDL file. For example, the MDL file contains the additional languages French, German, and Spanish. But your repository contains only Spanish and German as the additional languages. Then the Available Languages list displays only Spanish and German. Select the language you want to import and click the arrow to move the language to the Selected Languages list. You can choose multiple languages at the same time by holding down the Ctrl or Shift key while making your selection. For more information about importing metadata in additional languages, see "Import Different Base Languages" on page 10-18.

**Administration**

This option is available only if you have administrator privileges and the metadata exists in the MDL file being imported. This section enables you to import the following additional metadata:

- **User-defined definitions**: To import the definitions for the user-defined objects and the user-defined properties, select the Import User-defined Definitions option.

- **Security Grants**: Select Import security information to import security information such as object privileges and role assignments made to users.

If the MDL file contains any of these objects, then you can import this additional metadata.

When you import and MDL file into a new workspace, if you want to inherit the security information from the old workspace, you must import the security
information before you import other objects. To do this you need to be connected to the workspace as a user with administrator privileges.

After you make your selections on the Import Advanced Options dialog box, click OK to save your selections and return to the Metadata Import dialog box.

**Name Conflicts**
Name conflicts can occur in one of the following cases:
- A different object with the same name already exists in the target workspace.
- A different object with the same business name already exists in the target workspace.
- A different object with the same UOID already exists in the workspace.

When a name conflict occurs, the MDL reports an error and terminates the import.

**File Summary Dialog Box**
The File Summary dialog box contains a brief summary of the contents of the export file. The information on this page is divided into the following sections: **File**, **Administration**, and **Statistics**.

**File**
The File section contains the name of the data file, the creation timestamp, the name of the export user, the workspace connection information, the version of the Design Center used for the export, and annotations.

**Administration**
The Administration section contains information about the users and roles. It also lists the following details:
- Base language of the export file
- Additional languages in the export file
- Whether security information were included in the export file
- Whether user-defined definitions were included in the export file

**Statistics**
The Statistics section contains details about the types of objects contained in the export file and the number of objects of each type.

**Combining Import Modes and Matching Criteria**
Each search method used as matching criteria can be combined with an import mode in several different combinations. Each combination can offer different results in the import process. The mode that you select determines how the metadata import utility will search for metadata objects in the workspace prior to importing.

For example, if the search is by the name of a repository object in the export file, the Metadata Import Utility searches the workspace for the object's name. If an object with the corresponding name is not found, the resulting actions are based on the import mode you select.

Table 10–2 describes what happens in the available import modes for repository objects that do not match the object names.
Using Metadata Loader in the Design Center

Table 10–2  Import Mode without Matching Names

<table>
<thead>
<tr>
<th>Import Mode</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Mode</td>
<td>A new object is created.</td>
</tr>
<tr>
<td>Replace Mode</td>
<td>A warning message is written to the log file that the object cannot be replaced because it does not exist in the workspace. The object is skipped.</td>
</tr>
<tr>
<td>Update Mode</td>
<td>A new object is created.</td>
</tr>
<tr>
<td>Merge Mode</td>
<td>A new object is created.</td>
</tr>
</tbody>
</table>

Table 10–3 describes what happens in the available import modes for repository objects that match the object names.

Table 10–3  Import Mode with Matching Names

<table>
<thead>
<tr>
<th>Import Mode</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Mode</td>
<td>A message is written to the log file that the object already exists and the object is skipped.</td>
</tr>
<tr>
<td>Replace Mode</td>
<td>The object is replaced.</td>
</tr>
<tr>
<td>Update Mode</td>
<td>The object is replaced.</td>
</tr>
<tr>
<td>Merge Mode</td>
<td>The object is merged.</td>
</tr>
</tbody>
</table>

The MDL reads and processes the imported metadata and writes status and diagnostic information in the log file.

Import Different Base Languages

When you import metadata in multiple languages, the language settings in the target repository can be different from the language settings in the export file. For example, the target repository can have the base language as English and additional language as French and German. But the export file can have the base language as French and additional language as English and German. This section describes how the MDL handles these conditions.

Base Language of the MDL Import File Different From the Base Language of the Target Repository

When you import metadata, MDL compares the ISO identification of the base language in the import file with the ISO identification of the base language of the target repository. The ISO identification consists of the language ID followed by the locale, in the format language_locale. For example, en_US is American English and fr_FR is French.

If the base ISO identification languages are different, MDL displays a warning dialog box informing you that the base languages are different and warns you that Oracle recommends that you import metadata with the same character set and base language. You have the option to continue with the import. Click Yes to continue with the import. Click No to cancel the import.
If the base ISO identification languages are the same, but the locales are different, the Metadata Import Utility displays a warning dialog box asking if you want to continue with the import. For example, the export file contains English and the base language of the repository is American English. Click Yes to import metadata. Click No to cancel the import.

**Importing Supported Languages**

During an import, MDL checks if the additional languages in the import file exist in the target repository. If the import file contains additional languages that do not exist in the target repository, and you specify that these additional languages are to be imported, the Metadata Import utility writes a warning message in the MDL log file stating that the additional languages are not installed in the repository.

**Validation Rules Governing Import**

When you import a set of definitions from exported metadata, the import utility can update existing definitions in a project. However, certain metadata definitions require attention to ensure that they are updated. The following are examples of some of the errors you can see:

- **Mapping Definitions.** The Metadata Import Utility will bind imported mapping operators to their physical objects if the associated objects exist in the workspace. However, if the associated physical objects do not exist in the workspace, the imported mapping operators are unbound. The Metadata Import Utility writes a warning message in the log file stating that the mapping operators are not bound. You must then synchronize the new mapping operators with the physical objects they represent.

- **Foreign Key Definitions.** It is possible that a source MDL file can contain foreign key references to unique or primary keys that are not in the target workspace. If the referenced unique or primary keys for any foreign key appearing in the MDL file does not exist in the target workspace, the MDL Import Utility writes a warning message in the log file. This message will state that the workspace does not contain a referenced key for the foreign key.

**Upgrading Metadata from Previous Versions**

While importing metadata, the Metadata Import Utility automatically upgrades metadata created using previous versions to Oracle Warehouse Builder 11g Release 1 (11.1). You do not need to manually upgrade your metadata from a previous version of Warehouse Builder.

When you import an MDL file, the version used to create the file is detected. If the MDL file was created using an earlier version earlier, the Metadata Upgrade dialog box is displayed. This dialog box enables you to upgrade the MDL file to the current version. For more information about the contents of this dialog box, see "Metadata Upgrade Dialog Box" on page 10-20.

---

**WARNING:** Under certain circumstances, continuing to import metadata when the base languages are different could mean corruption of the metadata being imported.

It is recommended that you move metadata between repositories with the same character set and base languages.
If you import an .mdl file containing metadata for gateway modules, such as DB2 or Informix, from an older version of Warehouse Builder, the file may not import the metadata into the corresponding source module folders in a project. The imported files are stored under the Others node in the Project Explorer. You need to manually copy the metadata for the gateway modules into the correct source module folders.

The production versions of Warehouse Builder from which metadata is automatically upgraded to Oracle Warehouse Builder 11g Release 1 (11.1) are as follows:

- Oracle Warehouse Builder 2.0.4 and 2.0.5
- Oracle Warehouse Builder 2.1.1
- Oracle Warehouse Builder 3.0 and 3.1
- Oracle Warehouse Builder 9.0.2, 9.0.3, and 9.0.4
- Oracle Warehouse Builder 9.2
- Oracle Warehouse Builder 10g Release 1
- Oracle Warehouse Builder 10g Release 2

**Metadata Upgrade Dialog Box**

This dialog box is displayed automatically when it is determined that the MDL file being imported was created using a previous version. Use the **File Name** field to specify the name of the file that stores the upgraded MDL file. You can also click **Browse** to locate a directory or MDL file.

Click **Upgrade** to upgrade the MDL file to the current version. After the Upgrade completes, the Metadata Upgrade dialog box is closed. Click **Cancel** if you do not want to upgrade the MDL file.

**Changes to Workspace Objects After Upgrading to Oracle Warehouse Builder 11g Release 1**

When you upgrade from previous versions to the current version, the upgrade utility makes the following changes to objects in the workspace:

- **My Project:** The sample project that is prepackaged in Warehouse Builder is renamed from My Project to **MY_PROJECT** to comply with physical name requirements.
- **External Processes:** External processes are upgraded to external process activities in a process flow. If you defined an external process in a mapping in a previous release, MDL Upgrade Utility redefines the object as an external process in a process flow.
- **Business Areas:** Business areas are upgraded to Collections. If you defined a business area in a previous release, the MDL File Upgrade Utility prefixes the module name to the business area name and redefines it as a collection. For example, a business area named **ORDERS** in a module named **REGION1** is upgraded to a collection named **REGION1_ORDERS**.
- **External process mappings:** External process mappings will be migrated to process flows.
- **Dimension and Cube Mapping Operators:** The mapping operators for dimensions and cubes are converted to table operators. These table operators use the physical tables created by the MDL Upgrade Utility for dimensions and cubes.
- **Dimensions**: An associated dimension table is created with the same name as the dimension. The table contains the columns, constraints, and attribute sets defined in the Dimension Editor Table Properties of the dimension in the previous release.

- **Mapping Display Sets for Dimension Hierarchies**: Any mapping sets originally created based on the named attribute set for a dimension hierarchy are removed. This is because display sets for dimension hierarchies are no longer automatically created and maintained.

- **Dimension Attributes**: For each level attribute upgraded, a dimension attribute with the same name is created, if it does not already exist in the dimension.

- **Cubes**: An associated cube table is created with the same name as the cube. The cube table contains columns, constraints, and attribute sets defined in the Cube Editor Table Properties of the cube in the previous release.

- **Cube Dimension Reference**: If a foreign key in the cube does not reference a unique key in the lowest level of a dimension, the dimension reference is not imported. A warning is written to the import log file.

- **Intelligence Objects and Reports**: In the previous release, intelligence objects and reports were available only using OMB Plus scripting. These objects are not upgraded.

- **Locations and Runtime Repository Connections**: Locations and runtime repository connections are moved out of the projects that own them so that they can be shared across the entire workspace. Thus the statistics in the import log file will display an additional project for these objects.

- **Control Centers and Locations**: After an upgrade, there is no association between the locations and the control centers that they reference. You must review the control center details using the Edit Control Center dialog box and select the locations associated with this control center.

- **Advanced Queues**: An associated queue table is created based on the property AQ queue table name. The queue table created by the MDL File Upgrade Utility contains a column whose data type is the object type for that advanced queue.

- **Advanced Queue Operator in a Mapping**: Mapping Advanced Queue operators are changed to contain only one attribute called PAYLOAD. For Mapping Advanced Queue operators that are used as a source, a new Expand operator is added after the Mapping Advanced Queue operator. For Mapping Advanced Queue operators that are used as a target, a new Construct operator is added before the Mapping Advanced Queue operator.

- **Mapping Operator Names**: The MDL Upgrade Utility ensures that the physical names and business names of all mapping operators are unique.

- **MIV Objects**: MIV objects are not upgraded.

### Checking for Warnings and Error Messages

After upgrading the metadata, check the log file for warnings and errors.

- If you receive warnings during the upgrade, the upgrade utility completes and logs the warnings. If you receive errors, the upgrade utility terminates and logs the errors.

- If warnings and errors are shown after an upgrade, search for the words Warning and Error in the log file to determine the problem.

- If an unexpected error occurs and the upgrade terminates, the log file contains the details. Check your log file or contact Oracle Support.
Using Metadata Loader in the Design Center

Metadata Loader Utilities

The Metadata Loader consists of the following two utilities:

- **Metadata Export Utility**
  Use the Metadata Export Utility to export metadata from a workspace.

- **Metadata Import Utility**
  Use the Metadata Import Utility to import metadata into a workspace.

MDL uses its own format, and the Metadata Import Utility only reads files of MDL format (files created by the Metadata Export utility). The Metadata Loader file is a formatted ZIP file.

**Metadata Export Utility**

The Metadata Export Utility extracts metadata objects from a workspace and writes the information into a ZIP format file. This ZIP file has a .mdl extension and contains the following files:

- **Metadata Loader XML file**
  This file contains the objects extracted from the workspace and formatted in XML. It has the same name as the of the ZIP file, but with the extension .mdx.

- **Catalog**
  The catalog file is called mdlcatalog.xml and it contains internal information about the Metadata Loader XML file.

The Metadata Export Utility enables you to specify a file name and a path for the exported MDL file. For example, you export the repository metadata into a file called sales.mdl. When you unzip this MDL ZIP file, you obtain two files. The file sales.mdx contains the repository objects. The file mdlcatalog.xml contains internal information about the MDL XML file.

You can export an entire project, collections, public objects, locations, or any subset of objects. If you export a subset of objects, the MDL exports definitions for each object that you have selected and the parent objects to which the subset belongs. This enables the MDL to maintain the tree relationships for those objects during metadata import.

For example, if you export a single dimension, the export file contains definitions for:

- The dimension
- The module to which the dimension belongs
- The project to which the module belongs

If you are exporting a subset of objects, make sure you export all referenced objects and import them as well. You can export the objects referenced by a set of objects by selecting the **Export All Dependencies** option on the Metadata Export dialog box. For example, if you export a table DEPT and it contains a foreign key reference to the table EMP, you can choose to export EMP along with DEPT.

**Metadata Import Utility**

The Metadata Import Utility reads the metadata information from an exported MDL file and creates, replaces, or merges the metadata objects into a workspace. It imports information belonging to exported metadata objects such as table columns and their constraints, data loading configuration parameters, and named attribute sets. The Metadata Import Utility enables you to import repository objects even if the references for those objects cannot be satisfied.
You can use the Metadata Import Utility to import objects into a project or a collection. The Metadata Import Utility only reads files created by the metadata export utility.

If the MDL file being imported was created using an earlier product version, the Metadata Import Utility automatically upgrades it to the current version. For more information about the automatic upgrade of MDL files, see "Upgrading Metadata from Previous Versions" on page 10-19.

**Multiple Session Concurrency and MDL**

The repository allows multiple clients to access the same workspace concurrently. Warehouse Builder uses locks to allow only one client to change repository objects. While an object is locked, other clients can only view it as it existed after the last transaction instigated by any user is saved.

When replacing or merging objects, the MDL acquires locks on the primary objects that exist both in the repository and in the MDL file. Primary objects include, but are not limited to projects, modules, tables, dimensions, cubes, mappings, views, and flat files. Secondary objects, such as columns and mapping attributes, are not locked. If locks cannot be obtained because other users are locking the primary objects, then the import fails. Therefore, you must be able to hold locks for primary objects that you are importing.

---

**Tip:** To ensure a successful metadata import, you may need to be the sole client accessing the workspace.

---

The MDL saves changes made to the workspace after a successful metadata import (any import with no error messages, including imports with only information or warning messages). The MDL also executes a rollback after an unsuccessful import.

**Metadata Loader Log File**

Whenever you export or import repository metadata, the MDL writes diagnostic and statistical information to a log file. You can specify the location of the log file when you call the MDL.

The log file enables you to monitor and troubleshoot export and import activities in detail and contains the following information:

- Name of the data file
- Start time and end time of the export or import
- Time taken for the export or import in hours, minutes, and seconds (in hh:mm:ss format) or milliseconds
- Object types exported or imported
- Number of objects of each object type exported or imported
- The import log file also displays the total number of objects that have been added, replaced, skipped, and deleted.
- Status messages

Status messages provide information about the import or export process. They are of the following types:
Using Metadata Loader in the Design Center

- **Informational:** Provides information about the import or export, such as missing metadata objects, whether or not objects were imported, and any reasons why objects were not imported or exported.

- **Warning:** Cautions you about the import or export of an object but does not indicate a failed or terminated export or import. A warning notifies you of the possibility of unexpected results that could occur as a result of the export or import.

- **Error:** Indicates that the MDL export or import was terminated and did not complete successfully. The error message provides a brief description of the reason for the failure.

### About Metadata Loader Results

When you use the Metadata Loader Export or Import utilities, you can view the results of a successful export or import task. Use the Metadata Export Results dialog box or the Metadata Import Results dialog box to ensure that all of the objects were exported or imported. To view the results dialog box, click **Show Details** on the Metadata Export Progress dialog box or the Metadata Import Progress dialog box. This displays the Message Log. Click **Show Statistics** at the end of this log.

The results dialog box contains the following information:

- The name of the project exported or imported (if applicable).
- The number of objects of each type exported or imported.
- The number of objects of each object type skipped.

Details about the number of skipped objects is displayed only when you import metadata.
11 Managing Multiple Environments from Development to Production

This chapter discusses the strategies for managing multiple environments such as development, test, and production environments. This chapter includes the following sections:

- Strategies for Managing Multiple Environments
- Using Configurations to Manage Multiple Environments
- Multiple Environments Based on Multiple Design Environments
- Using Snapshots to Manage Multiple Environments

Strategies for Managing Multiple Environments

To methodically implement mission critical projects, organizations typically maintain separate environments for developing and testing prior to releasing a project into a production environment. Although you can replicate the same Warehouse Builder project across environments, it is often desirable to assign different physical properties for each environment. For example, when loading a target table in the production environment, you may want to save all error messages to a log file; however, when loading the target in the development environment, you may decide to not save the error messages.

Furthermore, although the same logical design may be initially replicated in each environment, the designs can evolve separately. In the development environment, the development team continually implements new features and makes changes to improve performance. In the production environment, the design evolves slowly and in a controlled manner based on the discovery and correction of bugs.

Therefore, to maintain separate environments for development, testing, and production, it becomes necessary to utilize the same logical design across different physical environments and also to manage any differences in designs across those environments.

In Warehouse Builder, you can manage multiple environments by implementing either of the following solutions:

- Multiple Environments Based on a Single Design Environment (requires the Warehouse Builder Enterprise ETL Option)
- Multiple Environments Based on Multiple Design Environments on page 11-7

Multiple Environments Based on a Single Design Environment

In this strategy, all design work resides within a single Warehouse Builder repository as depicted in Figure 11–1.
Strategies for Managing Multiple Environments

In the single design repository, you define a configuration and a collection for each runtime environment. You promote the design from one environment to the next by activating the appropriate configuration and then deploying the associated collection.

To promote a single design across multiple runtime environments, complete the following general steps:

1. Ensure that Warehouse Builder components are installed on the necessary computers.

   Any computer designated for design work requires the Warehouse Builder repository. Any computers designated as runtime environments require the Warehouse Builder control center service. Computers that simply host source data do not require Warehouse Builder components.

   Figure 11–1 depicts the design repository as hosted exclusively on its own Oracle Database. In practice, however, you could host the design repository and, for example, the development runtime on the same Oracle Database.

2. Create a configuration for each of the runtime environments, as described in "Using Configurations to Manage Multiple Environments" on page 11-3.

3. Instruct the development team to use Warehouse Builder with the DEV configuration active.

   The development team implements the logical design by creating the objects such as mappings and process flows. The objects they deploy are subsequently executed in the DEV environment.

4. To manage which objects are ready to be deployed to the test and production environments, create collections as described in the Oracle Warehouse Builder User’s Guide.

   A collection is a group of shortcuts to objects in the same project. For the purposes of this example, you create two collections named Deploy To Test and Deploy To Production.

---

**Figure 11–1  Multiple Environments Based on a Single Design Environment**

![Diagram showing single design environment and multiple runtime environments with deployment arrows between them.](Diagram)
When the development team determines that various design objects are ready for testing, they can add shortcuts to the Deploy To Test collection.

5. Instruct QA users to use Warehouse Builder with the TEST configuration active. QA users can now deploy the objects in the Deploy To Test collection. They can deploy each object singularly or deploy all objects in the collection at once. When a user selects a shortcut in a collection, they acquire a lock on the underlying object.

6. After deploying objects, QA users can execute the objects in Deploy To Test collection. The executed jobs effect the targets associated with TEST configuration only. The development and production environments are not effected.

7. The pattern continues with QA users adding shortcuts to the Deploy To Production collection. And production users ensure that the PROD configuration is active when they use Warehouse Builder.

Using Configurations to Manage Multiple Environments

The preferred method for managing multiple environments begins with creating a configuration for each environment. A configuration is a Warehouse Builder entity for containing the physical details of all objects that are required to deploy a data system.

Note: Use of the multiple configuration functionality requires the Warehouse Builder Enterprise ETL Option. If your organization has not licensed this option, you can follow the instructions in “Using Snapshots to Manage Multiple Environments” on page 11-8.

A common scenario is to have separate environments for development, test, and production. Although the same design objects exist in all environments, it may be desirable to apply different physical configuration settings to the objects.

As a example, Table 11–1 demonstrates how you may want to specify a table to have different configuration properties depending on the environment.

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Development Environment</th>
<th>Test Environment</th>
<th>Production Environment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tablespace</td>
<td>DEV</td>
<td>TEST</td>
<td>PROD</td>
</tr>
<tr>
<td>Parallel Access Mode</td>
<td>PARALLEL</td>
<td>NOPARALLEL</td>
<td>PARALLEL</td>
</tr>
<tr>
<td>Logging Mode</td>
<td>NOLOGGING</td>
<td>NOLOGGING</td>
<td>LOGGING</td>
</tr>
</tbody>
</table>

To implement this scenario, create a configuration for each environment and define the tablespace settings unique to each environment.

About Configurations

Upon installing Warehouse Builder or creating a new repository, expand the Configurations node in the Project Explorer. Notice that there is a single configuration, the DEFAULT_CONFIGURATION. Also observe that the configuration is listed at the
bottom left corner of the Design Center as *Active Configuration: DEFAULT_CONFIGURATION*.

This default configuration contains the physical properties of all the data objects in the Design Center which are necessary for deploying those objects. As a single entity, the DEFAULT_CONFIGURATION is merely a device that is required for the internal workings of Warehouse Builder. Namely, it enables deployment to the default control center, DEFAULT_CONTROL_CENTER.

When you define new configurations, however, then you can manage how the same logical object is defined across multiple environments. That is, you can define multiple physical properties and deployment details for a single table, view, or other such data object.

A common approach is to create three new configurations such as DEV, TEST, and PROD. Each configurations works in conjunction with a single control center and its locations, as depicted in Figure 11–2.

Figure 11–2  Relationship Between Configurations, Control Centers, and Locations

Configurations specify physical object properties that correspond to the environment to which the objects are deployed. A configuration can only be associated with one control center at a time. You can change the control center that is associated with a configuration.

A control center refers to a repository on a target machine and it manages a set of source and target locations. A control center can be associated with only one configuration at a time.

A location corresponds to the database, file, or application that Warehouse Builder sources data from or deploys data to. A location can be owned by only one control center. Each location can have one or more connectors that provide connections to other locations.

When you deploy objects, Warehouse Builder creates these objects in the target associated with the currently active configuration. In a given session, only a single configuration may be active at a time. If you switch to a different configuration as described in “Activating Configurations” on page 11-6, the name of the newly activated configuration is displayed along the bottom left corner of the Design Center.

**Benefits of Using Multiple Configurations**

Configuration property values belong to the configuration object and are preserved. You do not have to reset configuration values when you switch between
configurations. The configuration properties that you see in the Design Center are the settings associated with the active configuration. The status bar at the bottom of the Design Center displays the name of the active configuration.

For example, the configuration associated with the development environment is currently active. Therefore, any changes you make to configuration property values are made to the development environment. For the table `MY_TABLE`, you set the Tablespace configuration parameter to `DEV`. Next activate the configuration associated with the production environment. The configuration values displayed will be the same values that you set the last time the production configuration was active. The Tablespace configuration parameter for `MY_TABLE` is null. You set it to `PROD`. This change effects only the production configuration. Switching back to the development configuration will show the Tablespace configuration parameter for `MY_TABLE` remains as `DEV`. Every object instance in a project has unique configuration values. So, in this example, setting tablespace value for `MY_TABLE` has no effect on any other table. Each table instance must be individually configured.

Another advantage of multiple configurations is the ease with which it enables you to make changes to your existing environment. For example, you design objects, implement your development environment, deploy objects, and then move to the testing environment. You then need to change some objects in the development environment. To do this, you just activate the configuration associated with the development environment, make the changes to objects, regenerate scripts, and deploy objects. To return to the testing environment, you activate the testing configuration.

**Deploying a Design to Multiple Environments**

Creating multiple configurations enables you to deploy the same design to multiple environments. To deploy a design to a particular environment, you activate the configuration associated with the environment and deploy your design.

**To deploy a set of design objects to multiple environments:**

1. Create a configuration for each environment to which design objects are deployed as described in "Creating New Configurations" on page 11-6.
   For each configuration, ensure that you create a separate control center that points to the environment.

2. Set the configuration properties for design objects in each configuration, as described in "Setting Configuration Properties for a Configuration" on page 11-6.

3. Activate the configuration associated with the environment to which design objects must be deployed as described in "Activating Configurations" on page 11-6.

4. Resolve errors related to deployment or execution.
   You may encounter some errors even if you validated data objects before deploying them. These errors could be caused by configuration properties you set. Configuration property values represent physical property information that is not semantically checked before deployment. For example, the value you specify for the Tablespace Name property is not checked against the database at validation time. Such errors are encountered during deployment.

5. Deploy the design objects.

6. Repeat steps 2 to 5 for each environment to which design objects must be deployed.
Creating New Configurations

To create a new configuration:

1. In the Project Explorer, select a project and expand the navigation tree.
2. Select Configurations.
3. From the Design menu, select New.

   The Create Configuration wizard opens.
4. On the Name page, provide a name and an optional description.

   Click Set and save as my active configuration for this project to set the new configuration the active configuration. Any configuration parameters that you set for design objects are saved in this configuration. Also, any objects that you deploy are deployed to the control center associated with this new configuration.
5. On the Details page, select the control center that should be associated with this configuration.

   If you have not already created the control center, click New to create one and associate it with the configuration.
6. Click Finish to close the wizard and create the configuration.

   The new configuration appears in the Configurations folder.

Setting Configuration Properties for a Configuration

To set configuration properties for a particular configuration:

1. Ensure the desired configuration is active.

   The name of the active configuration is displayed along the lower left corner of the Design Center. To switch to another configuration, see "Activating Configurations" on page 11-6.
2. In the Project Explorer, right-click the object that you want to configure and select Configure.
3. Specify the configuration properties for the selected object.
4. Repeat steps 2 and 3 for all objects in the project for which you want to set configuration properties.

   You can now deploy the design objects. The Control Center that is presented in the Deployment Manager is the Control Center that is associated with the active configuration.

Activating Configurations

There can be only one active configuration at a time. Any objects that you deploy are deployed to the control center associated with the active configuration. To implement your design to a different environment, you must deploy objects to the control center associated with that environment. You can do this by activating the configuration associated with the control center.

To activate a configuration:

1. Right-click the configuration you want to activate and select Open Editor.
2. On the Name tab, select Set and save as my Active Configuration for this project.

To activate a configuration for the current session only:
1. In the Project Explorer, select a project and expand the navigation tree.
2. Expand the Configurations folder.
3. Select a configuration.
4. From the Design menu, select **Set As Active Configuration**.

   The selected configuration is set as the active configuration for the current session. If you exit Warehouse Builder and log in subsequently, then the changes are not saved.

Any changes that you make to the configuration parameters of objects are saved in the active configuration. If you switch to the previous configuration, then these parameters maintain their previous settings. The Control Center associated with this configuration is now active and stores all new information about validation, generation, and deployment of objects in the project.

**Multiple Environments Based on Multiple Design Environments**

In this strategy, there is a separate design environment for each runtime environment as depicted in Figure 11–3.

*Figure 11–3  Multiple Design Environments*

With this strategy, it becomes necessary to manage the differences between the three design environments. You can use either snapshots or the Metadata Loader to share design components between the design environments. Each time you introduce or change design metadata in an environment, you must subsequently deploy the design to the respective runtime environment.
Using Snapshots to Manage Multiple Environments

Scenario
After a period of development and testing, a company puts into production a data integration project. The production version of the project, however, is not static. It is subject to change due to the discovery and correction of bugs. Meanwhile, the development version also continues to evolve as the development team incrementally implements new features. This company now faces a challenge familiar to all companies: how to best manage changes in different versions of the system.

One version of this common scenario is depicted in Figure 11–4, where the development environment is consistently more advanced than the functionality in production, and QA is somewhere between the two extremes. Development changes are incrementally propagated to QA and subsequently to production. At the same time, production has its own cycle of changes, denoted in Figure 11–4 as the shadow environment labeled ‘Production1’, and used for controlled problem solving. ‘Production’ and ‘Production1’ are at the same stage of development, and serve to illustrate the errors that occur in Production, which are fixed and implemented directly in Production, but that must somehow be merged with Development. Other companies may have fewer or more differing environments, but the same maintenance challenges still apply.

Companies may need multiple environments, as illustrated in Figure 11–4, because they typically implement incremental changes to the system. However, some companies implement only whole projects in production. Figure 11–4 does not apply to these companies.

In this case study, a company finds a problem with a mapping in production. The first step is to compare the production version of the mapping with the development version of the mapping, as illustrated in Figure 11–5. If the mapping is identical in both environments, the solution is simple: make the changes in either environment and copy the mapping to override the older version. If the mapping in production differs from its development version, then the approach you take depends on the maturity of the project.
Typically, there are two phases that mark the lifecycle of a project: Initial Phase and Mature Phase. The two phases present different needs and call for two different version management methodologies, each of which has benefits and drawbacks.

**Initial Phase**

After implementation of a project in Production, the system is generally in its initial phase, depicted in Figure 11–6. The initial phase is marked by aggressive changes in the Development environment, coupled with errors sometimes found in Production. Because Production bugs are more likely in this mode, consider a management methodology that facilitates quick updates to each environment.

**Figure 11–6 Initial Phase: Changes in Production More Likely**

Companies often have two to five different environments. For the initial phase, this company keeps a separate definition of the metadata in each different environment (in this case, Development, QA, and Production). To propagate a change from Production, the company exports only those portions of the system that have changed and imports them into the Development definition.

**Case Study**

The company has recently implemented its a data integration project in production, and the system is still in its initial phase, where many additional features are yet to be tested and rolled out. The production system is fairly new, and therefore the occurrence of problems is higher in this phase.
The company decides to keep a separate design repository—or definition of the system design—for each environment, as depicted in Figure 11–7. In addition, they implement their processes into a separate runtime repository for each environment.

**Figure 11–7  Initial Phase: Separate Design Repositories**

In this example, an error occurs in a Production mapping. The company changes the mapping in Production, then exports its definition, and merges it into Development, as illustrated in Figure 11–8.

**Figure 11–8  Initial Phase: Propagate Changes from Production to Development**

To correct an error found in a Production mapping during the initial phase:

1. For backup, capture the definition of any mapping before modifying it.

   Create a full metadata snapshot of the mapping in the Production Design Repository. Do the same with the Development and QA versions of the mapping. Because you can restore objects from full snapshots only, a full snapshot is essential when you create a backup.

2. Correct the mapping in the Production design repository and deploy it to the Production target schema.
This results in a changed version of the mapping that must be propagated to other environments.

3. Use Metadata Export utility to export only the changed mapping from Production.

   From the Design menu, select Export and then Warehouse Builder Metadata. This displays the Metadata Export dialog box.

4. Use Metadata Import to import and merge the change to Development and QA.

   ■ From the Metadata Import dialog box Import Options, select Merge metadata.

   ■ From the Metadata Import dialog box Match By options, select the Universal Identifier option.

   Matching objects by Universal Identifier is important when maintaining multiple individually changing environments.

Merging the change into Development and QA can vary in complexity depending on the changed object. If the change in the mapping in this example consists of increasing the column width of a table, the merge is simple. A merge can be more complicated and time-consuming if, for example, join criteria are changed, and other dependencies exist.

**Mature Phase**

The second is the mature phase, depicted in Figure 11–9. The mature phase is marked by continued changes in the Development environment, but a decrease in changes required in Production.

*Figure 11–9  Mature Phase: Fewer Changes in Production*

For this mode, the company chooses a methodology that saves space and administration costs: it maintains only one active definition of the design, and this definition reflects the development state of the system. The company stores the design definitions of the QA and Production environments in backup, and extracts and restores changed portions of these systems when required.

**Case Study**

At this stage, the project has stabilized and is now in its mature phase. Some additional functionality is still being developed in the Development environment, but fixes originating in Production are rare.

Although they continue to implement their processes into a separate runtime repository for each environment, the company decides to keep only one design repository, as depicted in Figure 11–10.
Using Snapshots to Manage Multiple Environments

The one design repository reflects the Development environment, because it is the one active environment that regularly originates design changes. The design repositories from the QA and Production environments are stored as metadata snapshots inside the Development Design Repository. Snapshots are a backup mechanism that consumes minimal space, and still provides access to any objects that you need to restore. Because design changes rarely originate in Production or QA, storing those definitions in snapshots makes sense.

Although it is more rare during the mature phase, errors still occur in the Production environment. In this example, an error occurs in a Production mapping. The company changes the mapping in Production, then restores its definition from a snapshot in Development and makes the same change there, as illustrated in Figure 11–11.
To correct an error found in a Production mapping during the mature phase:

1. Compare the Production version of the mapping in your Production snapshot to the Development version of the same mapping in your Design Repository.
   - If the two differ, the company follows the rest of the steps in this procedure.
   - If the two are identical, correct the mapping as in Step 8, then deploy it to their Design and Production Runtime Repositories, and then update their Production snapshot with the changed mapping.

Consult the online help for instructions on comparing snapshots to objects, deploying, and on updating snapshots.

2. Back up the Development version of the mapping by creating a full metadata snapshot of it.

   The Development version of the mapping may differ from the Production version if developers have been working on a new iteration of that mapping. This step preserves their work. Creating a full snapshot is essential, because you can only restore from a full snapshot.

3. Restore the mapping in question from the Production snapshot.

   This mapping should be identical to the one running in Production.
Consult the online help for instructions on restoring objects from metadata snapshots.

4. Correct the mapping that you have restored from the Production snapshot.

5. Deploy the corrected mapping to the Production Runtime Repository.

6. Remove the existing definition of the mapping from the snapshot of the Production Design Repository and update the snapshot with the new version of the mapping.

7. Restore the mapping from the full snapshot you took as a backup in Step 2.
   This is the mapping from the Development Design Repository. Typically, this mapping has had other work done to it as part of development of new features. Optionally repeat this same step for QA.

8. Make the same correction to this Development version of the mapping that you made in Step 4 to the Production version of the mapping.

The cost of this methodology is that every change has to be made at least twice, in the Production and Development versions of the object. The company uses this methodology only because the mature phase does not require frequent changes originating in Production. The benefits of this approach are the minimal administration costs and reduced space requirements on the database.
Oracle Warehouse Builder enables you to build and publish an enterprise data warehouse in stages. You can improve the performance and manageability of the data warehouse.

Warehouse Builder mappings access remote data through database links. Processing overhead and network delays make this data access process slower than local data access by the mappings. You can use one of the following strategies to speed up data access:

- Create a transportable module to copy remote objects (tables, views, materialized views, and so on) from a source database into a target database. The mappings in the target data warehouse can then access data locally.
- Data can be partially processed in the source database and then the preprocessed data can be copied, using a transportable module, from source to target database for final loading into the data warehouse.

A transportable module functions like a shipping service that moves a package of objects from one site to another at the fastest possible speed.

**Note:** To utilize transportable modules, ensure that your organization has licensed the Warehouse Builder Enterprise ETL Option.

The following sections provide information about transportable modules:

- **About Transportable Modules** on page 12-1
- **Benefits of Using Transportable Modules** on page 12-4
- **Instructions for Using Transportable Modules** on page 12-5
- **Editing Transportable Modules** on page 12-18

### About Transportable Modules

Transportable modules enable you to rapidly copy a group of related database objects from one database to another.

Using the Design Center, you first create a transportable module, and specify the source database location and the target database location. Then, you select the database objects to be included in the transportable module. The metadata of the selected objects are imported from the source database into the transportable module. The metadata is stored in the workspace. To physically move the data and metadata from source into target, you need to configure and deploy the transportable module to the target location. During deployment, both data and metadata are extracted from the source database and created in the target database.

A combination of the following technologies enables the movement of data and metadata:

- Oracle Data Pump
- Transportable Tablespace
- DBMS_FILE_TRANSFER
About Transportable Modules

- Binary FTP
- Local file copy
- code generated and deployment

You can configure transportable modules to influence which technologies are used.

You can add the following source objects to transportable modules:

- Tablespaces
- Schemas
- Tables
- Views
- Sequences
- Materialized Views
- External Tables (including their data)
- PL/SQL Functions, Procedures, and Packages
- Object Types
- Varying Array Types (Varrays)
- Nested Table Types

In Figure 12–1, the traditional Extract, Transform, and Load (ETL) process extracts data from remote databases through multiple remote accesses using database links.

**Figure 12–1 Extraction Of Data From Remote Databases Through Multiple Remote Accesses Using Database Links**

During remote accesses using database links, significant performance degradation occurs due to serial queries and serial DMLs, and network latencies. The performance degradation will appear more if the same source tables are accessed multiple times.

The transportable modules technology works as shown in Figure 12–2. In this architecture, all the source objects needed by the mappings are bundled together and moved to the target during a deployment. The transportable modules deployment uses Oracle Data Pump, FTP, and Oracle transportable table space to achieve very high transportation performance. This transportation absorbs the cost of the network delays just once. After deployment, mappings access data locally, which can easily benefit from parallel queries and parallel DMLs. Repeated accesses to the same data increases the performance benefit of transportable modules.
Using transportable modules, data warehouse loadings become more manageable. The source database needs to be shut down only for a short period of time for the transportable module to complete the deployment. Users of the source database do not have to wait until the entire data is loaded into the data warehouse. For example, if you are using the transportable tablespace implementation, transportable modules can copy a tablespace of 20 GB in about five minutes, resulting in a down time of five minutes in the source database.

Data copied into the target database is a snapshot of the information present in the source database. This can be used to create a data versioning system. Advanced users can create streams on the transported tables for capturing real-time changes from the source. The transported tables can also be copied into larger tables as a partition.

In a multidepartmental enterprise environment, the target database may actually be an operational data store that is used for intermediate reporting and updating purposes. This target database could in turn, serve as a source to the next stage of data collection. You can use the transportable modules at multiple stages, along the path on which the data is moved before it is stored in the data warehouse.

Transportable modules can also be used for publishing data marts. A data mart is normally a portion of a larger data warehouse for single or departmental access. At times, creating a data mart amounts to copying what has already been collected and processed in the data warehouse. A transportable module can be created to perform this task, as shown in Figure 12–3. You can also use the same transportable module to deploy a data mart to multiple locations.
Because a transportable module deploys a snapshot of the source database objects, the deployment time can be used to track the version of the data marts.

**About Transportable Modules and Oracle Database Technology**

Transportable modules work by leveraging technology in Warehouse Builder plus technology in the Oracle Database. A transportable module replicates parts of a source database into a target database. The parts of the source database that can be replicated include tablespaces, tables, indexes, constraints, and other relational objects.

Depending on the database version, the Oracle Database replicates the tablespace. When you transport data between two releases of 8i databases or between two releases of 9i databases, the database calls the Oracle transportable tablespaces functionality. When you transport data between two Oracle 10g databases, the database calls the Oracle Data Pump functionality.

In the case of Oracle Database 11g and Oracle Data Pump, you can transport tables without transporting their tablespaces. For example, if your table is 100 KB and its tablespace size is 10MB, then you can deploy the table without deploying the entire tablespace. Only Oracle Data Pump provides the option to copy an entire schema. For Oracle 10g release database, you specify either data pump or transportable tablespaces during configuration as described in "Configuring a Transportable Module" on page 12-12.

**See Also:** For more information about transportable tablespace and Data Pump, see the Oracle Database 11g documentation.

**Benefits of Using Transportable Modules**

Before the introduction of transportable modules, the most scalable data transportation method relied on moving flat files containing raw data. This method required data to be unloaded or exported into files from the source database, and then these files were loaded or imported into the target database. The transportable modules method entirely bypasses the unload and reload steps and gives you access to the Oracle Database technologies Transportable Tablespaces and Data Pump.
High Performance Data Extraction
Transportable modules reduce the need for mappings to access data remotely. If you have large volumes of data on remote computers, then use transportable modules to quickly replicate the source onto the Oracle target database. Warehouse Builder mappings can then directly access a local copy of the data. In addition, because the source becomes part of the target, you can perform the ETL operations directly on the source data.

Distribute and Archive Data Marts
A central data warehouse handles ETL processing while dependent data marts are read-only. You can use transportable modules to copy from a read-only data mart to multiple departmental databases. In this way, you can use your central data warehouse to periodically publish new data marts and then replace old data marts by dropping the old tablespace and importing a new one. Because duplication and distribution takes relatively less time, you can publish and distribute a data mart for daily analytical or business operations.

Archive Sources
You can set the source tablespaces to read-only mode and then export them to a target. All the data files are copied, creating a consistent snapshot of the source database at a given time. This copy can then be archived. The archived data can be restored in the source and target databases.

Instructions for Using Transportable Modules

Before You Begin
Ensure that you can connect to source and target databases as a user with the necessary roles and privileges as described in Verifying the Requirements for Using Transportable Modules on page 12-6.

Ensure that your organization has licensed the Warehouse Builder Enterprise ETL Option.

To use transportable modules, refer to the following sections:

Note to Database Administrators: Step 1 of these instructions requires some powerful database roles and privileges. Step 3 requires knowledge of schema passwords. Depending on security considerations, you can allow developers to perform Step 3 or restrict it to database administrators only.

1. Specifying Locations for Transportable Modules on page 12-7
   Ensure to successfully test these connections before proceeding to the next step.
2. Creating a Transportable Module on page 12-8
3. Configuring a Transportable Module on page 12-12
4. Generating and Deploying a Transportable Module on page 12-15
5. Designing Mappings that Access Data through Transportable Modules on page 12-17
6. Editing Transportable Modules on page 12-18
Verifying the Requirements for Using Transportable Modules

When creating a Transportable Module source location, the source location user must possess specific roles and/or privileges depending on the version of the source database.

- If the source database is earlier than Oracle 10g, then the SYSDBA role is required for the source location user.
- If the source database is Oracle 10g, then the SYSDBA role is not required, but the following must be assigned to the source location user.
  - CONNECT role
  - EXP_FULL_DATABASE role
  - ALTER TABLESPACE privilege

When creating a Transportable Module target location, the target location user must possess specific roles and/or privileges depending on the version of the target database.

- If the target database is earlier than Oracle 10g, then the SYSDBA role is required for the target location user.
- If the target database is Oracle 10g, then the SYSDBA role is not required but the following must be assigned to the target location user.
  - CONNECT role with admin option
  - RESOURCE role with admin option
  - IMP_FULL_DATABASE role
  - ALTER TABLESPACE privilege
  - EXECUTE_CATALOG_ROLE with admin option
  - CREATE MATERIALIZED VIEW privilege with admin option
  - CREATE ANY DIRECTORY privilege

Note: Transportable Module source and target location users need to be assigned many powerful roles and privileges in order for the transportable modules to read objects from the source database and for creating objects in the target database. In a production environment, if necessary, the DBA may choose to create the transportable module source and target locations (using the Connection Explorer) for the data warehouse developers, and conceal the passwords.

The following is a SQL script for the DBA to assign source location users the required roles and privileges in the source database:

```
grant connect to <TM src location user>;
grant exp_full_database,alter tablespace to <TM src location user>;
```

The following is a SQL script for the DBA to assign target location users the required roles and privileges in the target database:

```
grant connect,resource to <TM tgt location user> with admin option;
grant imp_full_database,alter tablespace to <TM tgt location user>;
grant execute_catalog_role to <TM tgt location user> with admin option;
```
grant create materialized view to <TM tgt location user> with admin option;
grant create any directory to <TM tgt location user>;

Specifying Locations for Transportable Modules

Before you create a transportable module, first define its source and target locations in
the Connection Explorer. Each transportable module can have only one source and one
target location.

To specify a transportable module location:
1. In the Connection Explorer, expand the Locations node.
2. Expand the Databases node.
3. Right-click either the Transportable Modules Source Locations or Transportable
   Modules Target Locations node and then select New.

   Warehouse Builder displays a dialog box for specifying the connection information
   for the source or target location.
4. The instructions for defining source and target locations are the same except that
   you do not specify optional FTP connection details for targets. Follow the
   instructions in "Transportable Module Source Location Information" to specify the
   connection information and then test the connection.

Transportable Module Source Location Information

Warehouse Builder first uses this connection information to import metadata for the
transportable module from the source computer into the workspace. During
deployment, the connection information is used to move data from the source to the
target.

Name
A name for the location of the source or target database.

Description
An optional description for the location.

User Name/Password
Warehouse Builder uses the database user name and password to retrieve the
metadata of the source objects you want to include in the transportable module.
Warehouse Builder also uses this information during deployment to perform
transportable tablespace or data pump operations.

To access databases for use with transportable modules, you must ensure that the user
has the necessary database roles and privileges as described in Verifying the
Requirements for Using Transportable Modules on page 12-6.

Host
Host name of the computer on which the database is installed.

Port
Port number of the computer on which the database is installed.
Service
Service name of the computer on which the database is installed.

Version
Choose the Oracle Database release number from the list.

FTP User Name/Password (Optional)
Specify FTP account credentials if you intend to use Oracle Transportable Tablespace as the method for transporting data. FTP credentials are not required if you do not plan to configure the Transportable Tablespace method.

You can leave the FTP account credentials blank, if you configure to use the Transportable Tablespace, but both source and target databases are located in the same computer, or both source and target can access shared disk volumes. Without the FTP credentials, an attempt is made to perform a plain copy of the source files from the source directory to target directory.

Test Connection
Click **Test Connection** to validate the connection information. Warehouse Builder attempts to connect to the source database and, if applicable, to the FTP service on the source computer. A success message is displayed only after both credentials are validated.

Creating a Transportable Module

To create a transportable module:

1. From the Project Explorer, expand the **Databases** node.
2. Right-click the **Transportable Modules** node and select **New**.
   
   The Welcome page of the Create Transportable Module Wizard is displayed.
3. The wizard guides you through the following tasks:
   
   **Describing the Transportable Module**
   
   **Selecting the Source Location**
   
   **Selecting the Target Location**
   
   **Selecting Tablespaces and Schema Objects to Import**
   
   **Reviewing the Transportable Module Definitions**

Describing the Transportable Module
On the Name and Description page, type a name and optional description for the transportable module.

Selecting the Source Location
Although you can create a new source location from the wizard page, it is recommended that you define locations for transportable modules before starting the wizard as described in "Transportable Module Source Location Information" on page 12-7.

When you select an existing location, the wizard tests the connection and does not allow you to proceed until you specify a location with a valid connection.
Selecting the Target Location
Select a target location from the list. If no target locations are displayed, click New and define a target location as described in "Transportable Module Source Location Information" on page 12-7.

Selecting Tablespaces and Schema Objects to Import
Use the Define Contents page to select tablespaces and schema objects to include in the transportable module. On the left pane, Available Database Objects lists all source tablespaces, schemas, and available schema objects. On the right pane, Selected Database Objects displays the objects after you select and move the objects.

Expand the tablespaces to display the schemas in each tablespace and the objects in each schema. Non-tablespace schema objects such as views and sequences are also listed under their respective schema owners, even though these objects are not stored in the tablespace. To select multiple objects at the same time, hold down the Ctrl key while selecting them. You can include the following types of objects in transportable modules:

- Tables
- Views
- Materialized Views
- Sequences
- External Tables
- PL/SQL Functions, Procedures, and Packages
- Object Types, Varray Types, and Nested Tables Types

Select the tablespaces and schema objects from the Available Database Objects field and click the arrow buttons in the center to move the objects to the Selected Database Objects field.

Available Database Objects
You can view the number of data files and their total size by placing your mouse over a node. The wizard displays the information in a tooltip.

Figure 12–4 displays the wizard with the tooltip.
Finding Objects in the Available Database Object List: Double-click the flashlight icon to find source data objects by type or name. In the **Object** field, type a name or a letter by which to filter your search. From the **Type** list, indicate the object type you are searching. Check the required box to perform the search by name or by description.

For example, type 'T%' in the **Object** field, select tablespaces from the **Type** field, and click **Find Next**. The cursor on the Available Database Objects navigation tree selects the name of the first tablespace that starts with a 'T'. If that is not the tablespace you want to select, then click **Find Next** to find the next tablespace. During this searching process, the navigation tree expands all the schema names and displays all the tablespaces.

Filtering the Available Database Objects List: You can double-click a schema node or any of the nodes in the schema to type in a filter pattern. For example, if you type T% and click **OK**, the navigation tree displays only those objects that start with the letter T.

Figure 12–5 displays the Define Contents page with a schema selected.
Objects Not Available for Inclusion in Transportable Modules

If you select items that cannot be included in a transportable module, then a dialog box is displayed listing items that cannot be included and describing why.

Figure 12–6 displays the Import Filter dialog box.

Reviewing the Transportable Module Definitions

Review the summary information and click Finish to import the metadata of the selected tablespace and schema objects.

After the transportable module is created in your workspace, you can locate it on the Project Explorer under the Transportable Modules node. Expand the tree to display the imported definitions.
Warehouse Builder creates separate modules for separate schemas. The schema names on the Project Explorer mirror the schema names in your source database.

Because the objects contained in a transportable module mirror the source database, you cannot edit these objects using the user interface. If the source database changes, then you can reimport the objects. If you want to delete objects from the transportable module, then right-click the object and select **Delete**. This action deletes the object from the definition of the transportable module but does not effect the underlying source database.

### Configuring a Transportable Module

In the Project Explorer pane, right-click a transportable module and select **Configure** to configure it for deployment to the target database. You set configuration properties at the following levels:

- **Transportable Module Configuration Properties**
- **Schema Configuration Properties**
- **Target DataFile Configuration Properties**
- **Tablespace Configuration Properties**

For most used cases, you can accept the default settings for all the configuration properties with the exception of the **Password** setting. You must specify a password for each target schema. If the schema already exists in the target, then specify an existing password. If the schema does not already exist, then the schema can be created with the password you provide.

Depending on your company security polices, knowledge of schema passwords may be restricted to database administrators only. In that case, the database administrator must specify the password for each schema. Alternatively, developers can define new passwords for new schemas if the target has no existing schemas that match source schemas.

### Transportable Module Configuration Properties

Set the following run-time properties for the transportable module:

**Target OS Type**

Select the type of operating system for the target. For versions earlier than Oracle Database 11g, the type of operating system on the target computer must be the same as the source computer. For versions Oracle Database 11g or higher, you can deploy to any operating system from any operating system.

**Work Directory**

You should create a directory on the target computer dedicated to the deployment of transportable modules. This dedicated directory stores files generated at run time including temporary files, scripts, log files, and transportable tablespace data files. If you do not create a dedicated directory and type its full path as the **Work Directory**, then the generated files are saved under the runtime home directory.

**What to Deploy**

Warehouse Builder enables you to select whether you want to deploy only the tables in your transportable module or all the related catalog objects, such as views and sequences, as well. Select the **TABLES_ONLY** if you want to deploy only tables. Otherwise, select the **ALL_OBJECTS**.
Use the **TABLES ONLY** option to refresh the data in a transportable module. If you had previously deployed a transportable module with the **ALL_OBJECTS** option and want to replace only the tablespace from the same source, then redeploy the transportable module with the **TABLES ONLY** option. The deployment drops the existing tablespace in the target, inserts the new one, and then recompiles the previously deployed metadata.

Similarly, if you previously deployed the transportable module using Data Pump, then the redeployment will only modify the tables in the transportable module.

**Transport Tablespace**

By default, this setting is enabled and the tablespaces are transported. If you enable this setting, then also specify the settings under **Target DataFile Configuration Properties**.

If both the source and target databases are Oracle 10g or higher, then consider disabling this setting. For example, if your table is 100 KB and its tablespace size is 10 MB, then you can deploy the table without deploying the entire tablespace. When you disable **Transport Tablespace**, Oracle Data Pump is used to deploy the table and you can specify the **Table Exists Action** setting.

---

**Note:** If source or target location is not Oracle 10g, the Transport Tablespace option is selected by default. In that case, Transportable Tablespace is the only implementation method for data movement. If both source and target locations are Oracle 10g, then you can deselect Transport Tablespace and use Data Pump.

---

If Transport Tablespace is selected, then there are further restrictions, depending on the versions of the source and target locations, as described in **Table 12–1**. When planning for data replications, take these restrictions into consideration. In general, Oracle 10g, particularly Oracle 10g release 2, is the preferred target database.

**Table 12–1  Requirements for Replicating Data Between Database Versions**

<table>
<thead>
<tr>
<th>Source location</th>
<th>Target location</th>
</tr>
</thead>
<tbody>
<tr>
<td>10g</td>
<td>Targeting another Oracle 10g location requires that both databases must have the same character set and the same national character set. Targeting an Oracle 8i or 9i location is not possible.</td>
</tr>
<tr>
<td>9i</td>
<td>Targeting an Oracle 9i or 10g location requires that both databases must have the same character set, the same national character set, and both databases must be on the same operating system platform. Targeting an Oracle 8i or 9i location is not possible.</td>
</tr>
</tbody>
</table>
Schema Configuration Properties

Set the following schema properties for the transportable module:

**Target Schema Name**
This property enables you to change the name of the source schema when it is deployed to the target. Select the Default or click the Ellipsis button to type the new name for your schema in the target and click OK. For example, you can change SCOTT to SCOTT1.

**Password**
For existing schemas, type a valid password for the schema. For schemas to be created, Warehouse Builder creates the schema with the password you provide.

**Default Tablespace**
Specify the default tablespace to be used when creating the target schema. If you leave this setting blank, then the default specified by the target are used.

**Schema Exists Action**
Specify what action should be taken if the schema already exists in the target. The default value is Skip.

**Schema Does Not Exist Action**
Specify what action should be taken if the schema does not already exist in the target. The default value is Create.

**Table Exists Action**
When Transport Tablespace is disabled, use this property to specify what action should be taken if the table already exists in the target. The default value is Skip.

**Copy Source Schema**
When you use Oracle Data Pump by deselecting Transport Tablespace, you can select this option to copy the entire source schema into the target.

---

**Table 12–1 (Cont.) Requirements for Replicating Data Between Database Versions**

<table>
<thead>
<tr>
<th>Source location</th>
<th>Target location</th>
</tr>
</thead>
<tbody>
<tr>
<td>8i</td>
<td>Targeting an Oracle 8i, 9i, or 10g requires all of the following:</td>
</tr>
<tr>
<td></td>
<td>■ Both source and target databases must have the same character set.</td>
</tr>
<tr>
<td></td>
<td>■ Both source and target databases must have the same national character set.</td>
</tr>
<tr>
<td></td>
<td>■ Both source and target databases must be on the same operating system platform.</td>
</tr>
<tr>
<td></td>
<td>■ Both source and target databases must have the same block size.</td>
</tr>
<tr>
<td></td>
<td>■ Cannot change schema names during transporting tablespaces.</td>
</tr>
<tr>
<td></td>
<td>■ Cannot change tablespace names during transporting tablespaces.</td>
</tr>
</tbody>
</table>
Parallel
When you use Oracle Data Pump by deselecting Transport Tablespace, specify the maximum number of processes for the Oracle Database to use for carrying out the transfer of data.

Target DataFile Configuration Properties
You need to set the following data file properties for the transportable module:

Directory
Indicate the directory where you want the data file to be stored on your target computer. If you leave the directory unspecified, then the data file is stored in the Work Directory.

File Name
Specify the name of the data file to be created in the target computer. You can use this parameter to rename the data file. Accept the DEFAULT to persist the data file name from the source database or click the Ellipsis button to type a new name for the data file, and click OK.

Overwrite
If this parameter is selected, then the existing data file is overwritten. Otherwise, the deployment is terminated if an existing data file is found.

Tablespace Configuration Properties
When you enable Transport Tablespace, set the following tablespace properties for the transportable module:

Tablespace Name
If you are using a database prior to 10g, then the target tablespace name must be the same as your source tablespace name. For such cases, this field is read-only. If a tablespace with the same name already exists in your target database, then the run-time operation will first drop the existing tablespace and replace it with the new one.

If you are using Oracle Database 10g or higher, then you can change the target tablespace name.

Drop Existing Tablespace
If this setting is selected, the existing tablespace is dropped and recreated in the target. By default, this setting is not selected and prevents you from deleting the tablespace in the target in the event that the tablespace with the same name already exists. In this case, the deployment process stops with an error.

Generating and Deploying a Transportable Module
When you deploy a transportable module, the Control Center displays the transportable module as including all the tables while the other catalog objects such as views are displayed separately. When you select a deploy action for the transportable module, the Control Center sets the associated catalog objects to the same deploy action.

During deployment of a transportable module, there are two ways for users to monitor the deployment progress. The first way is by the use of the Job Details
window. The status line is instantly refreshed with the most up-to-date status. The message box immediately above the status line shows all the messages logged so far. Figure 12–7 displays the Job Details window.

Figure 12–7  The Job Details Window

Another way of observing the progress is by viewing the log file that the transportable module deployment process generates. The transportable module log file is created in the Work Directory that the user has configured. The name of the file is always <The TM Name>.log, for example TM1.log if the name of the transportable module is TM1. This file is a plain text file containing the same messages that you can see in the message box in the Job Details window. Example 12–1 shows the contents of a transportable module log file.

Currently, there are a total of 16 steps to view the log files. Some steps may be skipped depending on the user configurations, and some steps may contain error messages that transportable module considers ignorable, such as failures in creating referential constraints due to referenced tables not found errors. This log file contains important information. It must be carefully examined during and after the transportable module deployment completes.

Example 12–1  Log file containing important information

```plaintext
step1 begin: making connection to target db ...
step1 end: connected to target
Target ORACLE_HOME = /data/oracle/ora1010
step2 begin: making connection to source db...
step2 end: skipped.
step3 begin: making source tablespaces read only...
step3 end: skipped.
```
Designing Mappings that Access Data through Transportable Modules

After you successfully deploy a transportable module, you can use the objects in the transportable module in ETL designs. When you add source and target operators to a mapping, you can select objects from the transportable module folder.
Editing Transportable Modules

A transportable module is located under the transportable modules node within the Databases node on the Project Explorer.

You can edit a transportable module by right-clicking the name of the transportable module from the Project Explorer and selecting Open Editor. Warehouse Builder displays the Edit Transportable Module dialog box containing four tabs.

Name

From the Name tab, you can edit the name and description of the transportable module.

Source Location

Warehouse Builder uses this connection information to access the source computer and import the metadata into its workspace. Warehouse Builder also uses this information during run-time to move the tablespace data from the source to the target.

The Source Database tab is read-only. Once you have imported tablespace definitions from a source computer, you cannot change the location information.

Tablespaces

The Tablespaces tab displays the tablespaces to be transported and their size. This tab is read-only. You can also view the tablespace size for individual data files in a tablespace. For details, see “Viewing Tablespace Properties” on page 12-18.

Target Locations

Displays the available and selected target locations. You can move a location from Available Locations to Selected Locations, or configure a new location.

Viewing Tablespace Properties

You can view the properties of a tablespace by right-clicking the name of the tablespace from the Project Explorer and selecting Open Editor. Warehouse Builder opens the Edit Tablespace dialog box. This property sheet displays the size of individual data files in a tablespace. It has two tabs, Name and Source Datafiles.

Reimporting Metadata into a Transportable Module

If your source data has changed since you last created a transportable module, then you can reimport the metadata to update your workspace definitions. When you open the Reimport dialog box, the source location you specified while creating the transportable module is stored and the source objects are displayed.

To reimport transportable module definitions:

1. From the Project Explorer, right-click the Transportable Modules name and select Reimport.
   The Re-create Transportable Module dialog box is displayed.
2. From the Available Database Objects column, select the objects you want to reimport.
The database objects that have been previously imported into the workspace are listed in bold. You can also choose to import new definitions.

3. Use the arrow buttons to move the objects to the Selected Database Objects column, and click OK.

Warehouse Builder reimports existing definitions and creates new ones. The transportable module reflects the changes and updates after the reimport is completed.
Index

A

access
  multiple-user access, about, 7-13
  read-only mode, 7-14
  read/write mode, 7-13
accessing
  Metadata Loader, 12-3
active configuration, 8-3
Administrator role, 7-9
architecture, 1-1
assigning
  icons to objects, 9-15
  roles, 7-6
  roles to users, 7-10
  security roles, 7-9
  system privileges, 7-8
assigning
  logs file locations, 6-11
associating objects, 9-9

B

business areas
  upgrading to collections, 12-16

C

changing
  metadata, 11-1, 11-7
  passwords, 7-12
client computers, preparation of, 1-15
client-server implementation of Warehouse Builder, 1-4
collections
  upgrading from business areas, 12-16
components
  product, 1-1
  required for Warehouse Builder, 1-7
configurations
  about, 8-2
  activating, 8-3
  active configuration, 8-3
  creating, 8-3
  named configuration, 8-2
  setting properties, 8-5
configuring
  owbluent.bat file for memory, 6-5
connecting
  database, errors connecting to, 6-6
  Design Repository, error, 6-3
Control Center Manager, 1-2
Control Center Service, 1-3
creating
  database users, 7-5
  icon sets, 9-15
  icons for objects, 9-14
  user-defined objects, 9-6
  user-defined properties, 9-4

data libraries
  Name and Address, 6-5
database configuration parameters
  Design Repository database, 1-12
  runtime Repository database, 1-12
database roles, changing, 7-5
database users
  creating in Warehouse Builder, 7-5
  registering as Warehouse Builder users, 7-4
databases
  preparing before installing Warehouse Builder, 1-10
default object privileges, 7-6
defining
  security roles, 7-8
  user-defined objects, 9-6
deinstalling Warehouse Builder, 4-1
deleting
  Repository users, 4-2
  schema objects, 4-3
  user-defined properties, 9-5
  workspace owner, 4-2
deploying
  deployment errors, 6-12
  single design to multiple target systems, 8-5
Design Center, 1-2
Design Repository
  connecting error, 6-3
  enabled roles, 6-6
  upgrading, 3-9
design repository database configuration parameters, 1-12

diagrams
impact analysis, 10-3 to 10-5
lineage, 10-3 to 10-5
-Dlimit parameter
owbclient.bat file, memory threshold, 6-5

E
editing
user-defined objects, 9-12
user-defined properties, 9-12
editing user profiles, 7-6
encrypting passwords, 7-13
enforcing security, 7-11
error logs
interpreting error logs, 6-12
execution
errors, 6-12
expert editor, icon set, 9-17
exporting
design metadata, 3-10
metadata, for user grants, 12-8
metadata, for user-defined definitions, 12-8
metadata, in additional languages, 12-7
metadata, results, 12-5
metadata, using the Design Center, 12-5
snapshots, 11-7
user-defined objects, 9-13
user-defined properties, 9-13
external directories
copying to your new database instance, 3-6
external processes, upgrading to user-defined processes, 12-16
external tables
migrating, 3-6

F
files
log file locations, 6-11
flat files
copying to your new database instance, 3-18
Full Database Export/Import
upgrading Oracle Database, 3-3

gn_ext_dirs.sql script, 3-7
generation
errors, 6-12
GLOBAL_DBNAME parameter
Net Service Name, 6-7
groups
in LIA diagrams, 10-5

I
icon set
expert editor, 9-17

J
Java Virtual Machine (JVM)
verifying and reinstalling JVM, 6-11
JServer option
Runtime Assistant, for the, 6-4
L
launching Warehouse Builder components, 1-18
lineage diagrams, 10-3 to 10-5
Linux, installing Warehouse Builder on, 1-9
listener.ora file
GLOBAL_DBNAME parameter, 6-7
LoadJava error
Runtime Assistant, 6-4
log files
locations of log files, 6-11
Metadata Loader logs, 12-4
target schema import log, 3-7
logs
interpreting error logs, 6-12

M
materialized views, insufficient table space, 6-5
MAX_ENABLED_ROLES parameter, 6-6
MDL
exporting design metadata, 3-10
importing design metadata, 3-10
see Metadata Loader
memory
errors during batch operation, 6-5
metadata
access tool, 9-2
changing, 11-1, 11-7
dependencies, 10-1 to 10-10
full security strategy, 7-3
import and export errors, 6-12
import and export overview, 12-1
import modes, 12-10
importing, matching criteria, 12-11
matching criteria for import, 12-11
minimal security strategy, 7-3
multiuser security strategy, 7-3
security, 7-1 to 7-13
upgrading, 3-10, 12-15
metadata dependencies, diagrams of, 10-1
Metadata Dependency Manager, 10-1
Metadata Export
design metadata, 3-10
metadata export
required access privileges, 12-5
Metadata Export Utility, 12-2
Metadata Import
design metadata, 3-10
metadata import
required access privileges, 12-8
Metadata Import Utility, 12-2
Metadata Loader
about, 12-1
accessing, 12-3
log files, 12-4
metadata matching criteria, 12-11
multiple user access, 12-3
overview of import and export, 12-1
results, 12-5
uses, 12-3
using with OMB Plus, 12-18
using with the Design Center, 12-5
utilities, 12-2
metadata management, 12-1
metadata objects
applying security properties, 7-10
metadata snapshots, 11-1 to 11-7
migrating
external directories, 3-6
flat files, copying to your new database instance, 3-18
Partial Database Export/Import, 3-4
Target Schema, importing, 3-7
modes
read-only mode, 7-14
read/write mode, 7-13
multiple configurations
scenario, 8-4
multiple user access with MDL, 12-3
multiple user security strategy, 7-3
multiple-user access
about, 7-13
read-only mode, 7-14
read/write mode, 7-13
-mx parameter
owbclient.bat file, virtual memory, 6-5

N
Name and Address
regional libraries unavailable, 6-5
Name and Address server
errors, 6-12
names and addresses
installing third-party data, 5-2
Net Service Name
SERVICE_NAME parameter, 6-7

O
object privileges, default, 7-6
objects
assigning icon sets to, 9-15
associating, 9-9
associating with user-defined objects, 9-9
creating icons for, 9-14
exporting, 9-13
importing, 9-13
restoring from snapshots, 11-6
user-defined, 8-1 to ??, 9-1 to 9-17
OMB, 9-2
OMB Plus, 9-2
about, 9-2
scripts, 9-2
OMBDEFINE, 9-2
OMBDESCRIBE, 9-3
OMBDISPLAYCURRENTMODE, 9-4
OMBREDEFINE, 9-2
OMBSWITCHMODE, 9-4
Oracle Database
Full Database Export/Import, 3-3
JServer option, 6-4
migrating, 3-2
moving to a new database instance, 3-3
Partial Database Export/Import, 3-4
selective migration, 3-4
upgrading, 3-2
versions compatible with Warehouse Builder, 3-2
Oracle E-Business Suite
integrating with Warehouse Builder, 5-1
Oracle home and Warehouse Builder, 1-16
Oracle Import
Target Schema, migrating, 3-7
Oracle Metabase Plus, 9-2
Oracle Workflow
installing, 5-3
owbclient.bat file
-Dlimit memory threshold parameter, 6-5
-mx virtual memory parameter, 6-5

P
Partial Database Export/Import
upgrading Oracle Database, 3-4
passwords
changing, 7-12
encrypting, 7-13
in Warehouse Builder, 7-12
performance
-Dlimit parameter, 6-5
performing a stack trace, 6-10
pre-installation checklist, 1-17, 1-22
privileges
default, 7-6
object, 7-6
SYS user, checking privileges, 6-3
system, 7-8
process flows
icon set, 9-16
product
architecture, 1-1
components, 1-1
profiles, role, 7-10
propagating
user-defined objects, 9-13
user-defined properties, 9-13
properties
security, 7-10
user-defined, 9-3

R
RAC environments, installing Warehouse Builder on, 1-21
read-only mode, 7-14
read/write mode, 7-13
remote runtime environment implementation of Warehouse Builder, 1-5, 2-6
repositories
propagating from one to another, 9-13
split, 1-4
Repository Assistant
log file location, 6-11
Repository Browser
defined, 1-3
installing, 5-2
log file location, 6-11
Repository users, deleting, 4-2
required access privileges
metadata export, 12-5
metadata import, 12-8
requirements
JServer option, Oracle Database, 6-4
reusing
advanced queues, 3-18
roles
Administrator, 7-9
assigning, 7-6
assigning to users, 7-10
changing default, 7-5
Everyone, 7-9
profiles, 7-10
security, 7-8, 7-9
Runtime Assistant
LoadJava error, 6-4
Runtime Platform Service
log file location, 6-11
Runtime Repository
enabled roles, 6-6
runtime repository
upgrading, 3-12
runtime Repository database configuration parameters, 1-12

S
schema objects, deleting, 4-3
scripting
OMB Plus, 9-2
to define user-defined objects, 9-6
scripts
gen_ext_dirs.sql, 3-7
OMB Plus, 9-2
security
enforcement, 7-11
metadata, 7-1 to 7-13
multiple-user access, 7-13
read-only mode, 7-14
read/write mode, 7-13
roles, 7-8
security properties, 7-10
security roles, 7-9
security strategy
full, 7-3
minimal, 7-3
multiuser, 7-3
Select Icon button, 9-16
service names
errors resolving service name, 6-7
snapshots
access privileges, 11-4
adding components, 11-3
comparing, 11-5
converting type, 11-6
creating, 11-2
deleting, 11-7
exporting, importing, 11-7
metadata, 11-1 to 11-7
restoring, 11-6
uses, 11-1
split repositories, 1-4
SYS user
verifying privileges, 6-3
SYSDBA privileges
checking for SYS user, 6-3
system privileges, 7-8

T
tablespace, and materialized views, 6-5
tablespace, setting for Runtime Repository, 6-5
Target Schema
enabled roles, 6-6
importing when migrating, 3-7
target schema
about, 1-2
tnsnames.ora file
SERVICE_NAME parameter, 6-7
LIA See lineage and impact analysis
lineage and impact analysis See also impact analysis
lineage and impact analysis See also lineage
Oracle Metabase Plus See also OMB Plus
UDO See also user-defined object
UDP See also user-defined properties
user-defined properties See also UDP's
troubleshooting
installations, 6-1

U
UDP, 9-1
upgrading
business areas, 12-16
design metadata, 3-10
Design Repository, 3-9
external processes, 12-16
runtime repository, 3-12
upgrading Warehouse Builder, 3-1
user defined properties, 9-1
user profiles, editing, 7-6
user-defined objects, 8-1 to ?, 9-1 to 9-17
adding to repository, 9-5
associating with objects, 9-9
associating with other objects, 9-9
creating, 9-6
defining using scripts, 9-6
editing, 9-12
exporting, 9-13
importing, 9-13
propagating, 9-13
viewing, 9-12
user-defined processes, upgrading from external processes, 12-16
user-defined properties, 9-3
creating, 9-4
deleting, 9-5
editing, 9-12
exporting, 9-13
importing, 9-13
propagating, 9-13
viewing, 9-12
users
assigning roles to, 7-10
creating in Warehouse Builder, 7-5
registering as Warehouse Builder users, 7-4
users, multiple
security strategy, 7-3
utilities
Metadata Loader, 12-2

V
validation
errors, 6-12
validation rules for metadata import, 12-15
viewing
user-defined objects, 9-12
user-defined properties, 9-12
virtual memory
errors, 6-5

W
Warehouse Builder
client, log file location, 6-11
deinstalling, 4-1
frozen, 6-10
hanging, 6-10
implementation strategies, 1-3
installing, 1-5, 1-22
integrating with Oracle E-Business Suite, 5-1
passwords, 7-12
required components, 1-7
upgrading, 3-1
Warehouse Builder components, launching, 1-18
workspace owner
deleting, 4-2
workspaces
about, 1-3
defining users for, 2-1
managing, 2-2

A
αχχε σσ
μυλτι πλε-ψερ αχχεσσ αβου τ 8
ρεαδν ωρι τε μοδε 9
ρεαδν-ονλψυμοδε 9
αχχε σσ νγ
Μετάδοσα Λοιπέρ 8
Αδμη σποτορ ρολε 9
ορη τάξιμα 2
οσσα γνι νγ
1 χονστο βασετσ 14
ρολες 6
ρολεστο υσερς 10
σεχριτσ ρολες 9
στητη πρι τα λεγες 8
οσσας απι νγ βασετσ 9
B
βας νεσσαρεας
υπροδι νγ το χολλεχτι ονσ 20
χηρνγι νγ
μεταδοσα 2, 8
ποσσωριδα 13
χλι εντ χομπυτερσα, πρεπαρατι ον οφ 17
χλι εντ-σερτις 1 μπλεμεντατι ον οφ
Ωαρετρισε Βυιλδερ 5
χολλεχτι ονσ
υπροδι νγ φρομ βας νεσσαρες 20
χομπουνεντς
προδουτ 2
ρεθυρεδ φορ Ωαρετρισε Βυιλδερ 8
χονφα γυροτι ονσ
οβουτ 3
αντι τατι νγ 6
χρεατι νγ 6
αεττι νγ προπερτι εσ 6
χονφα γυρι νγ
οαβλι εντ βατ φι λε φορ μεμορτσ 10
τρανσπορταβλε μοδυλεσ 12
χοννεχτι νγ
δοπαβαε, ερρορσ χοννεχτι νγ το
10
Δεα γν Ρεποσ τωρψ ερρορ 8
Χοντρολ Χεντερ Μαναγερ 3
Χοντρολ Χεντερ Σερτι κε 4
χρεατι νγ
δοπαβαε υσερς 5
1 χον σετα 14
1 χονσφορ βασετσ 13
υσερ-δεφι νεδ βασετσ 6
υσερ-δεφι νεδ προπερτι εσ 4
Δ
δοτα λι βραρι εσ
Νομι ανδ Αδρες 9
δοταβαε χονφα γυροτι νπαραμετερσ
Δεα γν Ρεποσ τωρψ δοταβαε 13
ρυντι με Ρεποσ τωρψ δοταβαε 14
δοταβαε ρολεα χηρνγι νγ 5
δοταβαε υσερς
χρεατι νγ ιν Ωαρετρισε Βυιλδερ 5
ρεγι σερτι νγ οσ Ωαρετρισε Βυιλδερ 6
υσερς 4
δοταβαες
πρεπαρε νγ βεθορε 1 νσταλλι νγ
Ωαρετρισε Βυιλδερ 11
δεπλομεντ ερρορσ 1
α νγλε δεα γν το μυλτι πλε τορ
γετ οστεμ 5
ταβλεσι ντρανσπορταβλε μοδυλεσ 12
τρανσπορταβλε μοδυλε νγ 15
Δεα γν Χεντερ 3
Δεα γν Ρεποσ τωρψ
χοννεχτι νγ ερρορ 8
εναβλεδ ρολες 10
υπροδι νγ 9
δεα γν Ρεποσ τωρψ δοταβαε χονφα γν
ρατι ον παραμετερσ 13
δι ογρωμα
1 μπορτ ονολγα 3≤5
λι νεσαγε 3≤5
-Αλι μι τ παραμετερ
οαβλι εντ βατ φι λε, μεμορψ
τρεσερδο 10
Ε
edi ti

trovasortoble modulies 18
vser-defi ved ofefex 11
vser-defi ved proteries 11
edi ti vgr prophi les 5
evripti vgr pasaxowos 13
vphorchi vgr sevri 11
error logos
i nterpreti vgr error logos 1
exvuti on
error 1
ex pert edi topor, i xon set 16
ex porti vgr
dea vn metadota 10
metadota, for vser grapanos 12
metadota, for vser-defi ved defi ni vgr os 12
metadota, i v addi ti onol lon nuxesos 11
metadota, resvltos 24
metadota, sani vgr dea vgr Xen ter 9
synsartos 7
vser-defi ved ofefex 12
vser-defi ved proteries 12
ex terno, di rektoris
chopi vgr to vour vno dota boase i v
stoxhe 6
ex terno, proxeza, vgradi vgr to
vser-defi ved proxeza 20
ex terno, toblese
mi grapi vgr 6
Φ
filos filo
chopi vgr to vour vno dota boase i v
stoxhe 18

Φιλλ. Dota boase. Ex port. I mdort
vgradi vgr Oroxle Dota boase 3

Γ
gevex_d1 risbli sevri 6
generati vgr
trovasortoble modulies 15
generati vgr
erroros 1

ΓΛ ΟΒΑΛ _ΔΒNAME paromeper
Net Servi xe Noame 11

γρουπο
i v LIA di ogramos 5

I
i xon set
ex pert edi topor 16
proxeza floa 15
i xon seto
ovs gi vgr to ofefex 14
chrasi vgr 14
i xon chrasi vgr 13
i mporo aynaxis vdi ogramos 3/5
i plpemntot to vgr opotegies 4
i plpemnti vgr

Ωχερμυα Βυϊ λδερ 4
i mporo loyn oxe opio vno 18
i mporo modes 14
i mporo sevri vgr 15
i mporo vgr
dea vn metadota 10
log fi le, target sxema 7
metadota, combi vgr i v mporo modes
and match vgr chi teri a 17
metadota, for sevri tgrantes 16
metadota, for vser-defi ved defi ni vgr os 16
metadota, i mporo modes 14
metadota, i v addi ti onol lon
nuxesos 16
metadota, loyn oxe opio vno 18
metadota, match vgr chi teri a 15
metadota, resvltos 24
metadota, sani vgr dea vgr Xen ter 12
metadota, sioli doci vno rylies 19
synsartos 7
Target Sxema sev vn mati vgr 7
vser-defi ved ofefex 13
vser-defi ved proteries 13
i v t.ora fi le

MAΞ _ENABA Ε∆_POL ES paroume
ter 10
i vsoleto vgr Ωχερμυα Βυϊ λδερ
on Pax evdo ronmnet 22
i vsoleto vgr
erroros 1
trou blestropi vgr 1
i vsole vgr

Index-7
δαταβάσει νιστανχε 18

Παρτί αλ. Δαταβάσει Εξπορτ/1μπορτ 4
Ταργέτ Σχήμα, μιμορτινγ 7

μοδες

reen αρι τε μοδε 9
reen ονλυμοδε 9
μυλτι πλε χονφι γυρατι ονσ
σχενορι 3

μυλτι πλε υσρ ακχεσσα α τη ΜΑΛ 23
μυλτι πλε υσρ εχρι τιρσπρατεγψ
μυλτι πλε υσρ ακχεσσα

μιξ παραμετερ

οαβχλι εντ. βατ φι λε, σι ρτυ ωλ μεμ

ορψ 10

Ο

Ομε ανδ Αδρεσσ

εγιονολ. λυ βραφιες υνασαι λ
αβλε 9
Ομε ανδ Αδρεσσαερςερ

ερρορς 1

ομειο ανδ οδραρεσσα

ι νσταλλι νγ τη ρδ-παρτψ δατα 2

Νετ Σερμι χε Νομε

ΣΕΡς Ι ΧΕ_ΝΑΜΕ παραμετερ 11
Ο

οβφεχτ πρι σι λεγες δεφαλετ 6
οβφεχτ

οσχη γυι νγ 1 χον σετστο 14
οσχηχι ατι νγ 9
οσχηχι ατι νγ α τη υσρ-δεφι νεδ οβ
φεχτσ 9

χρεστι νγ 1 χονσφορ 13
eξ πορτι νγ 12
ι μπορτι νγ 13

πρι ντι λεγες 7

ΟΜΒ 1
ΟΜΒ Πλυσ 1

οβουτ 1
σχρπισ 2
ΟΜΒΔΕΦ ΝΕ 2
ΟΜΒΔΕΣΧΡ ΒΕ 3
ΟΜΒΔΛ ΣΠΑ ΑΠΧΡΡΕΝΤΜΟΔΕ 4
ΟΜΒΡΕΔΕΦ ΝΕ 2

ΟΜΒΣΩΩ ΤΧΙΜΟΔΕ 3
Οραχλε Δαταβάσει

φυλλ Δαταβάσει Εξπορτ/1μπορτ 3

ομειον 8
μυ γρατι νγ 2

ομω νγ το ανεωδαταβάσει νιστανχε 3

Παρτί αλ. Δαταβάσει Εξπορτ/1μπορτ 4
σαλτςι σε μ γρατι νγ 4
υπγροδι νγ 2

σερα ονσ χομπατι βλε α τη Ωαρε

πρυσα βυι λδερ 2
Οραχλέ Ε-βυια νεςα Συι τε

ιντεγρατι νγ α τη Ωαρε

πραμε βυι λδερ 1
Οραχλέ ημε ανδ Ωαρενυσα βυι λδερ 1
Οραχλέ ιμπορτ

Ταργέτ Σχήμα, μι γρατι νγ 7
Οραχλέ Μεταβασα Πλυσ 1
Οραχλέ Μορκφλω

ι νσταλλι νγ 3

οαβχλι εντ. βατ φι λε

-Αλι μι τ μεμορψ τιρεσρλδ παραμε
tερ 10

-μιξ σι ρτυ ωλ μεμορψπαραμετερ 10

π

Παρτί αλ. Δαταβάσει Εξπορτ/1μπορτ
υπγροδι νγ Οραχλε Δαταβάσει 4

πασσαρδας

χτραγγι νγ 13
eξχρυμρι νγ 13

ι ν Ωαρε

πραμε βυι λδερ 12

περφομανχε

-Αλι μι τ παραμετερ 10

περφομι νγ α σοκχ τροχη 15

πρε-τ νσταλλι ον χεχκλι στ 18, 24

πρι σι λεγες

dεφαλετ 6

οβφεχτ 6

υσρ, χεχκλι νγ πρι σι λεγες 7

συστεμ 8

tρανσπορταβλε μοδιλεσ 7

προχεσσ φλωσ

i χον σετ 15

προδυχτ


"ρυντί με Ρεπόσι τορψυδατοβασε χονφι γ
υροτι ον παραμετερσ 14
Σ
χεμα οβφεχτα δειτι νγ 3
χορι πτι νγ
ΟΜΒ Πλυσ 2
το δει νε υσε-δει νεδ οβφεχτσ 5
χορι πτσ
γεν_εκ τ_δι ροσσι 6
ΟΜΒ Πλυσ 2
σεχυρι τγν
επορχεμεντ 11
μεταδατα 1_13
μυλτι πλε_υσερ οχχεοσ 8
ρεωι ντι τε μοδε 9
ρεοδνυλμιοδε 9
ρολεσ8
σεχυρι τγν
σεχυρι τγνπροπερτι εσ 10
σεχυρι τγνρολεσ9
σεχυρι τγνπροπετεγψ
φυλλ 3
μο μολ 3
μυλτι υσερ 3
Σελεχτ1 χον θυτον 15
σερδι νε νομεσ
ερφορσσουλκι νγ σερτι νε νομε 11
σασπατση
οχχεοσ πρι τι λεγεσ5
αοδ νι χομπονεντσ 3
χομπαρι νγ 5
χονερτι νγ τιμε6
δειτι νγ
εξπορτι νγ, 1 μπορτι νγ 7
μεταδατα 2_8
ρεστορι νγ 7
υσεσ2
σαλι τ ρεποσι τορεσ5
ΣΨ υσερ
σερι φμ νγ πρι τι λεγεσ7
ΣΨΔΒΑ πρι τι λεγεσ
χοξω κι νγ φορ ΣΨΣ υσερ 7
σατεμ δρι τι λεγεσ 8
Σ
τομπολεσχε, ανδ ματερι αολι ζεδ τι εωσ9
tομπολεσχε
ματερι αολι ζεδ τι εωσ αολοχετι φορ 9

ορχη τετμπε 2
χομπονεντσ 2
προφι λεσ, ρολε 10
προπογει νγ
υσερ-δει νεδ οβφεχτσ 12
υσερ-δει νεδ προπερτι εσ 12
προπερτι εσ
σεχυρι τγν 10
υσερ-δει νεδ 3
Ρ
ΡΑΧ ενσι πομμεντα 1 νοσαλλι νγ
Λομπερματι Βυλδερ ον 22
ρεοδι ντι τε μοδε 9
ρεοδνυλμιοδε 9
ρεμοτε ρυντι με ενσι πομμεντ 1 μπλε μεντατι ον ολι Λομπερματι Βυλδερ 6
ρεποσι τορεσ
προπογει νγ φρομ ον το ωνοτερ 12
σαλι τ 5
Ρεποσι τορψι Βροσερ
δει νεδ 4
1 νοσαλλι νγ 2
Ρεποσι τορψυσερ δελετι νγ 1
ρεθυεπ ερι νεδ οχχεοσ πρι θι λεγεσ
μεταδατα εξ πορτ 10
μεταδατα ι μπορτ 12
ρεθυεπρεμεντσ
θΣερσερ οπτι ν, Οραχλι Δωτα βασε 8
ρενα νγ
ωδπανεδ θυευσ 17
ρολεσ
Αομ νι σρατορ 9
ααα γνι νγ 6
ααα γνι νγ το υσερσ 10
χηνγι νγ δεφυλτ 5
Επαρρονε 9
προφι λεσ 10
σεχυρι τν 8, 9
τρονπορτοβαβε μοδυλεσ 7
Ρυντι με Ααα σιαττ
Λορδοσια ερρορ 8
Ρυντι με Ρεποσι τορψ
ενοβλεδ ρολεσ 10
ρυντι με ρεποσι τορψ
υπγροδι νγ 11
σεττι νγ ϕορ Ρυντι με Ρεποσ τορψ9
Τοργετ Σχεμα
 εναβλεδ ρολεσ10
ι μπορτι νγ ανφ μι γρατι νγ 7
tοργετ σχεμα
 αβουτ 3
tνννμεσια φι λε
ΣΕΡς Ι ΧΕ_ΝΑΜε παρομετερ 11
Λ Ι Α Σεε λι νεεγε ανδι μπορτ αναλυα σ
λι νεεγε ανδι μπορτ αναλυα σ Σεε αλσο
ι μπορτ αναλυα σ
λι νεεγε ανδι μπορτ αναλυα σ Σεε αλσο
λι νεεγε
Οροφε Μετοβαε Πλυσ Σεε αλσο ΟΜΒ
Πλυσ
ΥΑΟ Σεε αλσο υσερ-δεφα νεδ οβφεχτ
ΥΑΠ Σεε αλσο υσερ-δεφα νεδ προπερ
τι εσ
υσερ-δεφα νεδ προπερτι εσ Σεε αλσο
ΥΑΠσ
Τρανσπορτάβλε Μοδύλεσσ
Αβουτ 4
Τρανσπορτάβλε μοδύλεσσ4
tρανσπορτάβλε μοδύλεσσ
χονφα γυρι νγ 12
dεφα νι τι ονυν 11
dεπλου νγ 15
edi τι νγ 18
gενερατι νγ 15
λοχατι ονσφορ 7
μοππι νγ 17
πρι τι λεγεσ7
ρολεσ7
tρουβλεσποτι νγ
ι νσπαλλατι ονυν 1
Υ
ΥΑΠ 1
υπυροδι νγ
βυσα νεσα ωρεσσ 20
δεν νι γν μετοδοτα α9
δεν γν Ρεποσ τορψ9
eξ τερνολ. προχεσσεσ20
ρυντι με ρεποσ τορψ11
υπυροδι νγ Ωορεπυμε Βυι λδερ 1
υσερ-δεφα νεδ προπερτι εσ1
υσερ προφα λεά εδι τι νγ 5
υσερ-δεφα νεδ οβφεχτσ
αιδι νγ το ρεποσ τορψ5
οσοξι ατι νγ αι ηηοβφεχτσ9
οσοξι ατι νγ αι ηηοπερ οβφεχτσ9
χρειτι νγ 6
δεφα νι νγ υα νγ χρι πτσ5
εδι τι νγ 11
eξ πορτι νγ 12
ι μπορτι νγ 13
προπαγατι νγ 12
αι εα νγ 11
υσερ-δεφα νεδ προχεσσεσ υπυροδι νγ
φρομ εξ τερνολ προχεσσεσ20
υσερ-δεφα νεδ προπερτι εσ3
χρειτι νγ 4
δελετι νγ 5
εδι τι νγ 11
eξ πορτι νγ 12
ι μπορτι νγ 13
προπαγατι νγ 12
αι εα νγ 11
υσερσ
οσσα γνι νγ ρολεστο10
χρειτι νγ 1 ι ν Ωορεπυμε Βυι λδερ5
ρεγι σερι νγ ασ Ωορεπυμε Βυι λδερ5
υσερσ4
υσερσ μυλτι πλε
σεχυρι τψ προτεγψ3
Υα νγ Τρανσπορτάβλε Μοδύλεσ5
υτι ι τι εσ
Μετοβατι Λ ονδερ 22
ζ
ταιλ δατι ον
ερροσ1
ταιλ δατι ον ρολεσ φορ μετοβατι ι μ
πορτ 19
αι εα νγ
υσερ-δεφα νεδ οβφεχτσ11
υσερ-δεφα νεδ προπερτι εσ11
αι ρτυνωλ μεμορψ
ερροσ10
Ω
Ωορεπυμε Βυι λδερ
δει νυσαλλι νγ 1
φροζεν 15
πνγι νγ 15
ι μπλεμεντα ον σπρατεγει εσ4
ι νυσαλλι νγ 6, 23
ιντεγρατί νγ ατηΟροχλε Ε-Βυσ
νεοσ Σωτε ινε 1
πασσωρδι 12
ρεθυι ρεδ χομπονεντσ 8
υπροδι νγ 1
Ωρεθροςε Βυι λδερ χομπονεντσ
λοννχη νγ 19
ωρκαπαρε οωερ
δελετι νγ 2
ωρκαπαρε
αβουτ 4
δεφι νι νγ υερσφορ 1
μανογι νγ 2