Oracle® Applications

System Administrator's Guide - Configuration

Release 12

Part No. B31453-04

December 2007

ORACLE®

Oracle Applications System Administrator's Guide - Configuration, Release 12

Part No. B31453-04

Primary Author: Mildred Wang, Robert Farrington

Contributing Author: Ahmed Alomari, George Buzsaki, Anne Carlson, Steve Carter, Steven Chan, Siu Chang, Jennifer Collins, Ivo Dujmovic, Elanchelvan Elango, Osama Elkady, Mark Fisher, Clara Jaeckel, Ramkarthik Kalyanasundaram, Takafumi Kamiya, Senthil Madhappan. Teresa Mak, Revathy Narasimhan, Sarita Nori, Mladena Novakovic, Muhannad Obeidat, Gursat Olgun, Richard Ou, Lisa Parekh, Jan Smith, Dana Spradley, Seth Stafford, Susan Stratton, Leslie Studdard, Suchithra Upadhyayula, Venkat Vengala, Mark Warren, Aaron Weisberg, Sara Woodhull

Contents

4 System Administration Setup Tasks

5 Introduction to Oracle Applications Manager

6 Defining Concurrent Programs and Requests

7 Defining Concurrent Managers

8 Setting Up and Starting Concurrent Managers

9 Printers

10 Oracle Applications Help

11 Applications DBA Duties

12 Query Optimization in Oracle Applications

16 Administering Process Navigation

17 Administering Globalization

A Oracle Application Server with Oracle E-Business Suite

B Loaders

C Functional Administrator and Functional Developer Tasks

D Oracle Self-Service Web Applications

E Timezone Support

Index

Send Us Your Comments

Oracle Applications System Administrator's Guide - Configuration, Release 12

Part No. B31453-04

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document. Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

Note: Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Applications Release Online Documentation CD available on Oracle MetaLink and www.oracle.com. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: appsdoc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at www.oracle.com.

Preface

Intended Audience

Welcome to Release 12 of the *Oracle Applications System Administrator's Guide - Configuration.*

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Computer desktop application usage and terminology.

If you have never used Oracle Applications, we suggest you attend one or more of the Oracle Applications training classes available through Oracle University.

See Related Information Sources on page xviii for more Oracle Applications product information.

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.

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.

Structure

Related Information Sources

This book is included on the Oracle Applications Documentation Library, which is supplied in the Release 12 Media Pack. You can download soft-copy documentation as PDF files from the Oracle Technology Network at http://otn.oracle.com/documentation, or you can purchase hard-copy documentation from the Oracle Store at

http://oraclestore.oracle.com. The Oracle Applications Documentation Library Release 12 contains the latest information, including any documents that have changed significantly between releases. If substantial changes to this book are necessary, a revised version will be made available on the "virtual" documentation library on Oracle *MetaLink*.

For a full list of documentation resources for Oracle Applications Release 12, see *Oracle Applications Documentation Resources, Release 12*, Oracle*MetaLink* Document 394692.1.

If this guide refers you to other Oracle Applications documentation, use only the Release 12 versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF).

- **Online Help** - Online help patches (HTML) are available on Oracle*MetaLink*.
- **PDF Documentation** - See the Oracle Applications Documentation Library for current PDF documentation for your product with each release. The Oracle Applications Documentation Library is also available on Oracle*MetaLink* and is updated frequently.
- **Oracle Electronic Technical Reference Manual -** The Oracle Electronic Technical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for each Oracle Applications product. This information helps you convert data from your existing applications and integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. The Oracle eTRM is available on Oracle *MetaLink*.

Related Guides

You should have the following related books on hand. Depending on the requirements of your particular installation, you may also need additional manuals or guides.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Concepts

This book is intended for all those planning to deploy Oracle E-Business Suite Release 12, or contemplating significant changes to a configuration. After describing the Oracle Applications architecture and technology stack, it focuses on strategic topics, giving a broad outline of the actions needed to achieve a particular goal, plus the installation and configuration choices that may be available.

Oracle Applications CRM System Administrator's Guide

This manual describes how to implement the CRM Technology Foundation (JTT) and use its System Administrator Console.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer forms so that they integrate with Oracle Applications.

Oracle Applications Flexfields Guide

This guide provides flexfields planning, setup, and reference information for the Oracle Applications implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This guide also provides information on creating custom reports on flexfields data.

Oracle Application Framework Personalization Guide

This guide covers the design-time and run-time aspects of personalizing applications built with Oracle Application Framework.

Oracle Applications Installation Guide: Using Rapid Install

This book is intended for use by anyone who is responsible for installing or upgrading Oracle Applications. It provides instructions for running Rapid Install either to carry out a fresh installation of Oracle Applications Release 12, or as part of an upgrade from Release 11*i* to Release 12. The book also describes the steps needed to install the technology stack components only, for the special situations where this is applicable.

Oracle Applications Supportability Guide

This manual contains information on Oracle Diagnostics and the Logging Framework for system administrators and custom developers.

Oracle Applications System Administrator's Guide Documentation Set

This documentation set provides planning and reference information for the Oracle Applications System Administrator. *Oracle Applications System Administrator's Guide - Configuration* contains information on system configuration steps, including defining concurrent programs and managers, enabling Oracle Applications Manager features, and setting up printers and online help. *Oracle Applications System Administrator's Guide - Maintenance* provides information for frequent tasks such as monitoring your system with Oracle Applications Manager, managing concurrent managers and reports, using diagnostic utilities, managing profile options, and using alerts. *Oracle Applications System Administrator's Guide - Security* describes User Management, data security, function security, auditing, and security configurations.

Oracle Applications User's Guide

This guide explains how to navigate, enter data, query, and run reports using the user interface (UI) of Oracle Applications. This guide also includes information on setting user profiles, as well as running and reviewing concurrent requests.

Oracle Integration Repository User's Guide

This guide covers the employment of Oracle Integration Repository in researching and deploying business interfaces to produce integrations between applications.

Oracle iSetup User Guide

This guide describes how to use Oracle iSetup to migrate data between different instances of the Oracle E-Business Suite and generate reports. It also includes configuration information, instance mapping, and seeded templates used for data migration.

Oracle Workflow Administrator's Guide

This guide explains how to complete the setup steps necessary for any product that includes workflow-enabled processes. It also describes how to manage workflow processes and business events using Oracle Applications Manager, how to monitor the progress of runtime workflow processes, and how to administer notifications sent to workflow users.

Oracle Workflow Developer's Guide

This guide explains how to define new workflow business processes and customize existing Oracle Applications-embedded workflow processes. It also describes how to define and customize business events and event subscriptions.

Oracle Workflow User's Guide

This guide describes how users can view and respond to workflow notifications and monitor the progress of their workflow processes.

Oracle Workflow API Reference

This guide describes the APIs provided for developers and administrators to access Oracle Workflow.

Oracle XML Publisher Administration and Developer's Guide

Oracle XML Publisher is a template-based reporting solution that merges XML data with templates in RTF or PDF format to produce a variety of outputs to meet a variety of business needs. Outputs include: PDF, HTML, Excel, RTF, and eText (for EDI and EFT transactions). Oracle XML Publisher can be used to generate reports based on existing E-Business Suite report data, or you can use Oracle XML Publisher's data extraction engine to build your own queries. Oracle XML Publisher also provides a robust set of APIs to manage delivery of your reports via e-mail, fax, secure FTP, printer, WebDav, and more. This guide describes how to set up and administer Oracle XML Publisher as well as how to use the Application Programming Interface to build custom solutions.

Integration Repository

The Oracle Integration Repository is a compilation of information about the service endpoints exposed by the Oracle E-Business Suite of applications. It provides a complete catalog of Oracle E-Business Suite's business service interfaces. The tool lets users easily discover and deploy the appropriate business service interface for

integration with any system, application, or business partner.

The Oracle Integration Repository is shipped as part of the E-Business Suite. As your instance is patched, the repository is automatically updated with content appropriate for the precise revisions of interfaces in your environment.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using an Oracle Applications form can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

1

Introduction

Introduction to This Manual

A system administrator is involved in setting up an Oracle Applications installation, controlling access, and ensuring smooth ongoing operation. The tasks involved in these functions are described in the *Oracle Applications System Administrator's Documentation Set*, in these three volumes.

- Configuration
- Security
- Maintenance

This *Oracle Applications System Administrator's Guide - Configuration* volume describes the tasks involved in setting up and configuring Oracle Applications. These tasks may be done once upon installation, or may also be done as needed, such as setting up a printer or customizing online help files.

Basic Configuration Tasks

This chapter describes some of the recommended manual setup steps for Oracle Application Object Library. This chapter can also be used as reference for maintaining an Oracle Applications installation. For complete instructions on setting up a new installation, see *Installing Oracle Applications.*

Oracle Applications Tablespace Model and the Tablespace Migration Utility

The new Oracle Applications Tablespace Model (OATM) has fewer, consolidated tablespaces (twelve, including three system tablespaces: temporary, system and undo segments). Locally managed tablespaces are also supported.

The Tablespace Migration Utility is a menu-based Perl program that enables you to estimate future space requirements for the tablespaces and to migrate the Applications

database to OATM.

System Administrator Setup Tasks

This chapter describes some common system administrator setup tasks, typically done through the Oracle Applications user interface.

Introduction to Oracle Applications Manager

This chapter introduces the Oracle Applications Manager framework. Oracle Applications Manager provides a powerful set of features for managing all aspects of an Oracle Applications system. It provides a comprehensive summary of the system including configuration changes, infrastructure usage, performance, required maintenance activities, potential security issues, and diagnostic test results. It also includes utilities to help you manage Oracle Workflow, patching, configuration utilities, and concurrent processing.

Defining Concurrent Programs and Reports

This chapter explains how to define concurrent programs and organize those programs into groups and sets. This chapter also explains how to modify concurrent program definitions, modify the behavior of parameters the programs refer to, and define incompatibility rules among different programs.

Setting Up Concurrent Processing and Concurrent Managers

Concurrent managers run processes in the background (concurrent processes) on a server machine. You must set up and start concurrent managers for each product group before you can use the Oracle Applications products.

Setting Up Printers

This chapter describes how to set up printers and printing in Oracle Applications.

Oracle Applications Online Help

The Oracle Applications online help files can be customized, as explained in this chapter.

Oracle Applications DBA Duties

This chapter explains Oracle Applications tasks that require a database administrator to perform explicitly, or assist in by performing prerequisite tasks.

Query Optimization in Oracle Applications

Oracle Applications uses the cost–based optimizer (CBO) in order to choose the most efficient execution plan for SQL statements. Using this approach, the optimizer determines the most optimal execution plan by costing available access paths and factoring information based on statistics for the schema objects accessed by the SQL statement.

Oracle Applications and Oracle Real Application Clusters

This chapter describes the steps required to install Oracle Applications in an environment that uses Oracle Real Application Clusters (Oracle RAC).

Document Sequences

This chapter explains how to assign unique numbers to documents created in Oracle Applications.

Administering Process Navigation

The Process Navigator provides users with diagrams of business processes as a whole as well as the individual steps in each process. The Process Navigator also provides direct access to the form associated with each step in a process.

Administering Globalization

This chapter describes some of the features of internationalization support in Oracle Applications, including language values for user sessions and multilingual external documents.

Developer Tools

This chapter provides information on the Forms Personalization and Work Directory features, which can help system administrators and developers create and debug custom forms.

Loaders

Oracle Applications provides loader programs to help you move Oracle Applications data between database and text file representations.

Functional Administrator and Functional Developer Tasks

Specific system administrator and application developer tasks using HTML-based pages are grouped under the Functional Administrator and Functional Developer

responsibilities.

Oracle Self-Service Web Applications

Some features of Oracle Self-Service Web Applications (OSSWA) are supported for backward compatibility and described in an appendix.

Oracle Application Server with Oracle Applications

Oracle Application Server offers the industry's fastest, most complete, and integrated J2EE-certified application server. Refer to this section to learn about configuring Oracle Applications with Oracle Application Server.

Other Volumes for System Administrators

Listed below are other volumes in the *Oracle Applications System Administrator's Documentation Set*. Please refer to the Preface for additional related guides.

Oracle Applications System Administrator's Guide - Security describes security concepts, setup tasks, and maintenance tasks done in the following areas:

- Oracle User Management
- Function Security in Oracle Application Object Library
- Data Security in Oracle Application Object Library
- User and Data Auditing
- Oracle Single Sign-On Integration (optional)

Oracle Applications System Administrator's Guide - Maintenance describes tasks you might perform on frequent basis, such as monitoring your system, reviewing concurrent requests, and setting profile options. The following areas are covered in this manual:

- Managing Concurrent Processing and Concurrent Programs
- Oracle Workflow Manager
- Monitoring Oracle Applications
- Diagnostics and Repair
- Patching and Maintenance
- User Profiles and Profile Options in Oracle Applications Object Library
- Using Predefined Alerts

2

Basic Configuration Tasks

Oracle Applications Setup Steps

Proper setup is required before a number of Applications features will operate correctly. The following steps are carried out automatically during or after installation:

1. Run Rapid Install
2. Test Web listener virtual directories
3. Test Oracle HTTP Server configuration
4. Create DBC files
5. Test Java Servlet setup
6. Set Web Server profile options

The above is not an inclusive list of the installation and post-installation tasks that may be needed on a particular system.

Rapid Install is the installation program. Refer to *Installing Oracle Applications: Using Rapid Install* for detailed information on running Rapid Install.

Configuring the Login Page for Oracle Applications

Oracle Applications uses a configurable login page, which can be tailored to suit the needs of different organizations.

Oracle Applications Login page

Users log in to Oracle Applications using a client web browser. From the Oracle Applications Login page, users access the E-Business Suite Home Page, which provides a single point of access to HTML-based applications, forms-based applications, and

Business Intelligence applications. Users access the Oracle Applications Login page from the following URL:

```
http://<server:port>/OA_HTML/AppsLogin
```

For example,

```
http://oraapps1.oracle.com:8000/OA_HTML/AppsLogin
```

From this URL, you will be redirected to the central login page, "AppsLocalLogin.jsp".

Central Login Page

The following features are displayed in the default login page: Username field, Password field, Login button, and the Language Picker (if more than one language is installed).

The following user interface features can be turned on or off through the Local Login Mask profile option:

- Hints for username/password
- * Register URL - this link allows the user to perform self-service registration in User Management
- * Forgot Password URL - allows the user to have a password reset
- Language Picker
- Corporate Policy message

* Oracle User Management must be installed for "Register URL" and "Forgot Password URL" to be enabled.

The ICX login page (http://server:port/OA_HTML/US/ICXINDEX.htm) redirects the user to the central login page, "AppsLocalLogin.jsp". If, in a previous release, you

customized the ICX login page previously with a custom logo, you should make a copy of the new ICX login page and replace the existing image with your custom image in the copied file. The location for the company logo is $OA_MEDIA/FNDSSOCORP.gif. Ensure that the image is appropriately size. Also, you should change the text of the message 'FND_ORACLE_LOGO' in Message Dictionary to the appropriate text. The following login URL is supported, but no new features are being added to it:

```
http://server:port/OA_HTML/jtflogin.jsp
```

If the Oracle Applications instance is Single Sign-On enabled, the servlet directs the user to the Single Sign-On login page.

Passwords

Note that in previous releases of Oracle Applications, user passwords were treated as case insensitive. Now, Oracle Applications user passwords can optionally be treated as case sensitive, depending on the mode you choose. Case-sensitivity in passwords is controlled by the profile option Password Case Option.

Language

The current browser language, if it exists in the applications database also, is used for the login page. The user can choose a different language on the login page (if the Language Picker is enabled) and refresh the page to have it appear in that language.

If the current browser language does not exist in the Oracle Applications installation, the language set in the site-level setting of the ICX_LANGUAGE profile option is used to render the login page.

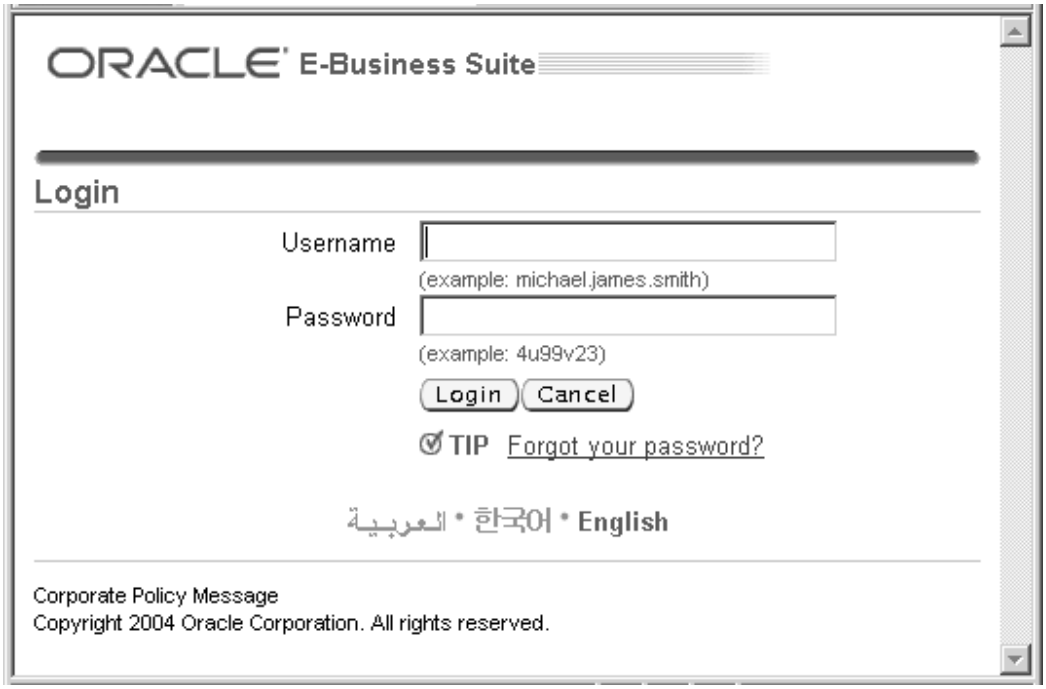
A user can override the value of the ICX_LANGUAGE profile option for a given session only.

Oracle Applications Manager

Oracle Applications Manager uses this central login page as well to authenticate users.

Customizing the Oracle Applications Login page

Customized Login Page



User interface features can be turned on or off using the profile option Local Login Mask (FND_SSO_LOCAL_LOGIN_MASK):

For the Login page to show one of more of these optional attributes, just add the numeric values of all desired attributes and set the value of the profile option to that value.

- Username Hint = 01
- Password Hint = 02
- Cancel Button = 04
- Forgot Password URL = 08
- Register URL = 16
- Language Picker = 32
- Corporate Policy Message = 64

For example to show the Password Hint and the Forgot Password URL only, set the Local Login Mask profile option to 10 (02+08). To show just the Language Picker, set the

value to 32, which is also the default value for the profile option.

The values for the Username Hint, Password Hint, Forgot Password URL text, Register URL text, and the Corporate Policy Message, as well as the Copyright text, are stored in Message Dictionary. You can update these messages using the Messages form or HTML page.

The table below lists the message names and their default values.

Login Page Message Names and Default Values

Message Name	Default value
FND_SSO_HINT_USERNAME	(example: michael.james.smith)
FND_SSO_HINT_PASSWORD	(example: 4u99v23)
FND_SSO_FORGOT_PASSWORD	Forgot your password?
FND_SSO_REGISTER_HERE	Register here
FND_SSO_SARBANES_OXLEY_TEXT	Corporate Policy Message
FND_SSO_COPYRIGHT_TEXT	Copyright (c) 2006, Oracle. All rights reserved.

Personalizing the Oracle E-Business Suite Home Page

The Oracle E-Business Home Page can be personalized to display the Worklist and to display the Applications Navigator in "Tree" or "Flat" mode. By default, the Applications Navigator is shown in Flat mode.

Refer to the *Oracle Application Framework Personalization Guide* for more information on personalizing Oracle Application Framework pages. Ensure that you have the relevant profile options set appropriately.

Follow these steps to use the Tree mode:

1. Log in to Oracle Applications with the system administrator responsibility.
2. From the Oracle E-Business Suite Home page, select **Personalize Page**.
3. Select **Apply** in the "Choose Personalization Context" page.
4. Under "Personalization Structure", expand the following nodes: 'Table Layout: (topTableLayoutContainer)' > 'Row Layout: (tableLayoutRow)' > 'Cell Format: (worklistResponsibilityLeftCell)' > 'Table Layout: Home Contenttable'.

5. Under the node 'Table Layout: Home Contenttable' > Row Layout: (responsibilityRow), click the Personalize icon.

6. Change the value for the 'Rendered' property for the column 'Function: Applications Home Page' to 'false'. Select **Apply.**

7. Click on the Personalize icon for 'Table Layout: Home Contenttable' > 'Row Layout: (appsNavTreeRow)'.

8. Change the value for the 'Rendered' property for the column 'Function: Applications Home Page' to 'true'. Select **Apply.**

9. Select **Return to Application.**

Administering Oracle HTTP Server

Oracle HTTP Server Powered by Apache provides the communication services within Oracle Application Server. This facilitates deployment of HTML-based applications within a multi-tiered computing environment.

Oracle HTTP Server

All incoming client requests to Oracle Internet Application Server (AS) are handled by the Communication Services component of AS. The Oracle HTTP Server, powered by Apache Web Server technology accepts and processes these requests. The Apache technology adopted by Oracle HTTP Server provides an extremely stable, scalable, and extensible platform on which to deploy web-based applications. The modular design of the Apache server allows for extension of the capabilities of the Oracle HTTP Server. In addition to the standard Apache modules (often referred to as modules, or simply mods), a number of Oracle specific modules are provided along with an extension to the functionality of several standard modules.

These modules include:

- **mod_ssl** - This module provides secure listener communications using an Oracle provided encryption mechanism using 128-bit *Secure Sockets Layer* (SSL). The mod_ossl module replaces mod_ssl. In contrast to the OpenSSL module, mod_ossl is based on the Oracle implementation of SSL, which supports SSL version 3 and uses the Oracle Wallet Manager for Certificate Management. The Apache HTTP Server SSL configuration file, *ssl.conf*, is located in $INST_TOP/ora/10.1.3/Apache/Apache/conf directory.

- **mod_oc4j** - This module routes all servlet requests to the Apache OC4J servlet engine embedded within Oracle HTTP Server. Servlets can be shared across multiple zones.

- **mod_perl** - This module forwards Perl requests to the Perl Interpreter. The Perl Interpreter is embedded within the Oracle HTTP Server, removing the necessity to spawn an external interpreter as well as providing a caching mechanism such that modules and scripts need only be loaded or compiled once. Oracle Applications does not currently utilize mod_perl.

  The Oracle HTTP Server is powered by a standard version of Apache. A number of books have been published describing the operation of the Apache server. To further add to your knowledge of the Apache server, you may wish to consult one of these.

  > **Note:** Refer to the Oracle HTTP Server documentation for a more detailed description of the operation and configuration of the Oracle HTTP Server. The information in this section is supplementary to that provided in the Oracle HTTP Server books.

Oracle Applications Installation Guide: Using Rapid Install should be consulted for additional information on directory structures and file locations referred to in this guide.

Apache Configuration Files

Apache is configured through directives contained in one or more configuration files. The directives necessary for operating Apache within the Oracle environment will be entered into the configuration files during the install process. It should not be necessary to modify these files unless the system is being reconfigured.

> **Warning:** An invalid directive entered into a configuration file will prevent Apache from starting. An incorrect definition provided to a directive may cause Apache to behave in an unintended fashion.

Location

The Apache configuration files are installed as part of the Oracle Applications Rapid Install process. On UNIX, for example, the files will be placed in $INST_TOP/ora/10.1.3/Apache/Apache/conf.

Secure Sockets Layer Configuration

Secure Sockets Layer (SSL) allows the Apache listener to encrypt HTML pages and transmit them on the network using the HTTPS protocol for secure transmissions.

SSL uses an encrypting method called public key cryptography, where the server provides the client with a public key for encrypting information. The server's private key is required to decrypt this information. The client uses the public key to encrypt and send information to the server, including its own key which identifies it to the

server.

In order for the Oracle HTTP Server, powered by Apache, to function in secure mode it is also necessary to utilize certificates which validate the server's identity. These certificates are used to ensure that the owner of a public key is who they say they are. Typically you will want to use a private key with an officially signed certificate, validated by a Certificate Authority (CA). The CA validates the company details, sets expiration dates on the certificates and may place policies on what information is contained within the certificate. A number of CAs exist, and include such authorities as Verisign, RSA, and GTE CyberTrust.

To obtain a CA approved certificate it is necessary to generate a certificate request, which includes details of the organization applying as well as the public key to be distributed, this is then sent to the CA, validated and returned. Authorities may also require proof of ownership of the company applying for the certificate, as well as proof of ownership of the domain name specified in the certificate request.

It is possible to generate a self-signed certificate that can be used to test SSL operation. Most browsers are configured to accept certificates from a number of recognized authorities. Receiving a certificate from an organization other than one of these will generate a warning, and the user will be prompted to accept or reject this certificate. A self-signed certificate will generate this warning.

Creating a self-signed certificate

The SSL module provides two utilities - Oracle Wallet Manager, and the Oracle Wallet Manager Command Line Interface (`orapki`) - that can be used to create a self-signed certificate for testing purposes.

> **Warning:** Self-signed certificates are inherently insecure, and should not be used in a environment where security is required.

In order to generate a self-signed certificate, perform the steps below using the Oracle Wallet Manager Command Line Interface:

Temporary SSL Environment Setup

Follow these steps for the temporary SSL environment setup.

1. **Source your environment**
    1. Log on to the application (middle) tier, as the OS user who owns the application tier files.
    2. Source your <APPL_TOP>/<SID_hostname>.env file to set your APPL_TOP variables.
    3. Navigate to the $INST_TOP/ora/10.1.3 and source the <SID_hostname>.env file

to set your 10.1.3 ORACLE_HOME variables. When working with wallets and certificates, you *must* use the 10.1.3 executables.

2. Create a Wallet

    1. Verify the location of the Web SSL directory:

        ```
        grep s_web_ssl_directory $CONTEXT_FILE
        ```

    2. Navigate to this directory, which will have Apache and opmn subdirectories

    3. Change to the Apache subdirectory

    4. Backup any existing wallets (demo certificates are included in Release 12)

    5. Create your new wallet:

        ```
        orapki wallet create -wallet . -auto_login -pwd <password>
        ```

        "-wallet ." (note the ".") tells `orapki` that you want to create the wallet in the current directory. You can also specify the full path to any directory where you wish to create a wallet.

        " -auto_login" tells `orapki` that you want to create cwallet.sso which is an obfuscated copy of ewallet.p12. With auto login enabled, processes submitted by the OS user who created the wallet will not need to supply the password to access the wallet.

    6. You should now see two wallet files in your directory: cwallet.sso and ewallet.p12.

3. Add your self-signed certificate to the wallet.

    1. Enter the following command all on one line, substituting the appropriate parameters for your instance:

        ```
        orapki wallet add -wallet . -dn
        "CN=server.us.oracle.com,OU=Unit,O=Org,L=Orlando,ST=Florida,C=US"
        -keysize 1024 -self_signed -validity 3650 -pwd welcome1
        ```

    2. Your wallet is now ready for use in environments where security is not required (as mentioned in the warning note above, self-certification is by definition not secure).

Configure Apache for SSL Connections using AutoConfig

Ensure that the following parameters are correctly set in your Applications context file:

Variable	Non-SSL Value	SSL Value
s_url_protocol	http	https
s_local_url_protocol	http	https
s_webentryurlprotocol	http	https
s_active_webport	Same as s_webport	Same as s_webssl_port
s_help_web_agent	URL constructed with http protocol and s_webport	URL constructed with http protocol and s_webport (same as non-SSL)
s_login_page	URL constructed with http protocol and s_webport	URL constructed with https protocol and s_webssl_port

Run AutoConfig, either through the Oracle Applications Manager interface, or by using the `adautocfg.sh` script in the application tier $ADMIN_SCRIPTS_HOME directory.

> **Note:** For more details, see Oracle*MetaLink* Note 387859.1, *Using AutoConfig to Manage System Configurations with Oracle E-Business Suite Release 12*.

Test Startup of Apache and Oracle Applications Sign-On

Restart the application tier Apache services using the `adapcctl.sh` script in the $ADMIN_SCRIPTS_HOME directory. Ensure that Apache can startup successfully, and that you successfully get an SSL connection to the default Apache banner screen using https:/<host.domain>:<SSL_port>, where <host.domain> is the fully qualified name of the machine running Apache, and <SSL_port> is the SSL port number defined in ssl.conf.

Once you have signed on to Oracle Applications, select a responsibility and process that will launch a Forms-based application. For example, System Administrator responsibility and the Define User process.

> **Note:** For further details of using SSL, see Oracle*MetaLink* Note 376700.1, *Enabling SSL with Oracle Application Server 10g and E-Business Suite Release 12*, and Note 376694.1, *Using the Oracle Wallet Manager Command Line Interface with Oracle E-Business Suite Release 12*.

AdminAppServer Utility

Important: The following details are provided for reference only. You should not need to carry out manual maintenance on dbc files, as they are managed by AutoConfig, which runs AdminAppServer automatically. See "Using AutoConfig to Manage System Configurations with Oracle E-Business Suite Release 12", Note 387859.1, on Oracle*MetaLink.*

Because Release 12 is deployed in a multi-tier configuration, the security model includes authentication of application servers to the database servers they access. When this layer of security is activated, the application server passes *server IDs* (similar to passwords) to the database server. If the database server recognizes the server ID, it grants access to the database. The server IDs are created using a Java script called AdminAppServer.

The application server security system is by default not activated; if it you must activate it after installation, if required. The application servers are not assigned server IDs and the database servers do not check for server IDs.

Using the AdminAppServer Utility

The Java script AdminAppServer is used to create .dbc files and to enable or disable application server security.

Prior to running AdminAppServer you must ensure that:

- JDBC classes are in the CLASSPATH and LD_LIBRARY_PATH
- $JAVA_TOP is in the classpath

For UNIX platforms, the script is run as:

```
java oracle.apps.fnd.security.AdminAppServer [parameters]
```

For Windows platforms, the script is run as:

```
jre -classpath %CLASSPATH% \
 oracle.apps.fnd.security.AdminAppServer [parameters]
```

The first parameter must be the connection string followed by the command string, for example:

```
apps/apps@dbname ADD
```

The following commands are supported:

- **ADD** - create a new .dbc file
- **UPDATE** - update an existing .dbc file

- **DELETE** - delete an existing .dbc file
- **STATUS** - check the serverID status for a database
- **AUTHENTICATION** - toggle authentication mode

Additional parameters depend on the operation. These include:

- **DBC** - The .dbc file to modified, or used to connect to the database. Used with UPDATE, DELETE, STATUS, AND AUTHENTICATION.
- **SECURE_PATH** - Used with ADD. This parameter specifies in which directory the .dbc file should be created, and defaults to the current directory if not provided. This parameter should always point to $FND_TOP/secure.
- **DB_HOST** - Required. The host machine of database.
- **DB_PORT** - Required. The port of database. The default is 1521.
- **DB_NAME** - For thin drivers. The database SID.
- **APPS_JDBC_DRIVER_TYPE** - THICK or THIN. This parameter must be set to THIN.
- **GUEST_USER_PWD** - Any valid applications user. This parameter defaults to the value of GUEST_USER_PWD profile if not provided. If passed with no arguments to an UPDATE call, it will refresh with the value from database.
- **GWYUID** - For thick drivers.
- **FNDNAM** - For thick drivers.
- **TWO_TASK** - For thick drivers. Name of database.
- **WALLET_PWD** - Used with the TCF Socket Server in SSL mode.
- **SERVER_ADDRESS** - Used with authentication.
- **SERVER_DESCRIPTION** - Used with authentication.
- **FND_MAX_JDBC_CONNECTIONS** - The maximum number of open connections in the JDBC connection cache. This number is dependent on the amount of memory available, number of processes specified in the init.ora file of the database and the per-processor file descriptor limit.
- **FND_IN_USE_CONNECTION_TIMEOUT** - The maximum number of seconds a connection can be in use. In order to avoid connections being locked up for too long, the connection cache uses this parameter to forcibly close connections that

have been locked for longer than this specified limit. If this parameter is unspecified, connections in use will not be cleaned up. This should be set to a number larger than the time taken to complete the largest transaction.

- **FND_UNUSED_CONNECTION_TIMEOUT** - The maximum number of seconds an unused connection can remain in the cache. The connection cache will close and remove from the cache any connection that has been idle for longer than this specified limit.

  Important: The following parameters are required: APPS_JDBC_DRIVER_TYPE (must be set to THIN), DB_HOST, and DB_PORT.

Administering .dbc Files

The .dbc file is contained on the web/applications server and holds information used by the database for authentication. The web/application server passes the information from the .dbc file, as well as login information, to the database server to authenticate the user. The authentication process is handled by the standard applications security feature.

The .dbc files required by the application server security system are not part of the delivered product and must be created after installation.

The Java utility AdminAppServer is used to create the .dbc files.

Prior to running AdminAppServer you must ensure that:

- JDBC classes are in the CLASSPATH and LD_LIBRARY_PATH
- $JAVA_TOP is in the classpath

Creating a .dbc file

Use the AdminAppServer utility to create a .dbc file for the application server to access the database server. In addition to creating the .dbc file this utility registers the application server with the database for the Applications Server Security feature.

To access additional database servers from the same application server, you must rerun the AdminAppServer utility for each additional database. You must run the AdminAppServer utility each time you create a .dbc file, and each .dbc file only allows access to one database.

To create a .dbc file for an application server:

1. You must set the username/password value for the GUEST_USER_PWD parameter. Create a valid username ("guest" for example) in Oracle Applications. Then use the username/password combination as the value for GUEST_USER_PWD. The syntax is illustrated in the following example:

```
GUEST_USER_PWD=guest/guest
```

Oracle recommends that you do not assign any responsibilities for this user.

2. From the command line, enter the appropriate command for your platform:

   For UNIX platforms:

   ```
   java oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \
    ADD DB_HOST=database_host_name DB_PORT=database_port \
    DB_NAME=database_sid \
    [env_name=env_value] SECURE_PATH=$FND_TOP/secure
   ```

   For Windows platforms:

   ```
   jre -classpath %CLASSPATH% \
    oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \
    ADD DB_HOST=database_host_name DB_PORT=database_port \
    DB_NAME=database_sid \
    [env_name=env_value] SECURE_PATH=$FND_TOP/secure
   ```

GWYUID, FNDNAM, and GUEST_USER_PWD will be defaulted if not provided explicitly. dbc files should be located in $FND_TOP/secure, so SECURE_PATH should always be set to that, or this should be run directly out of of the $FND_TOP/secure area.

Updating a .dbc file (or Server ID)

When updating the .dbc file you can change as many parameters as you want, including the server ID, but you must enter at least one. Settings that you do not update retain their value.

To update a .dbc file or server ID:

Enter from the command line:

UNIX

java oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \ UPDATE DBC=$FND_TOP/secure/file.dbc APPL_SERVER_ID \ [env_name=env_value]

Windows

jre -classpath %CLASSPATH% \ oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \ UPDATE DBC=$FND_TOP/secure/file.dbc APPL_SERVER_ID \ [env_name=env_value]

If APPL_SERVER_ID is not passed, AdminAppServer will attempt to synchronize the current server ID value in the .dbc file and the database, generating a new one if neither contains a value at all. Passing in APPL_SERVER_ID by itself will force a brand new application server ID to be created always, overwriting any existing one.

Deleting a .dbc file

To delete a .dbc file, enter on the command line:

UNIX

java oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \ DELETE

DBC=$FND_TOP/secure/file.dbc

Windows

jre -classpath %CLASSPATH% \ oracle.apps.fnd.security.AdminAppServer apps/apps@dbname \ DELETE DBC=$FND_TOP/secure/file.dbc

This deletes the .dbc file and disallows access to the indicated database if Server Security is active.

Troubleshooting

The following are possible problems you may encounter and suggested solutions.

- Database connection failed.

  Check to see if your JDBC environment is correct. See: AdminAppServer Utility, page 2-11.

- File I/O error while adding the server.

  Check to see if the path you supplied as SECURE_PATH exists and that you have write permissions on it.

- Unable to read environment file.

  A value for SECURE_PATH may not have been specified. If a value is not specified, the AdminAppServer utility assumes you are running from JAVA_TOP and looks for the file $JAVA_TOP/oracle/apps/env.html to find the value of FND_TOP. Retry the command specifying the value of SECURE_PATH.

- Database error: Unique constraint violated.

  There can be only one entry for each application server per database. If you do not specify the value for SERVER_ADDRESS, the AdminAppServer utility will default the IP address of the machine from which you are running the command. To resolve this issue, run the STATUS command of AdminAppServer to ensure you are not trying to create a duplicate entry. Delete the old entry if you want to replace it. Retry, supplying the correct value for SERVER_ADDRESS.

Administering Server Security

Oracle Applications is deployed in a multi-tier configuration with one database server and many possible middle-tier application servers. The application servers include Apache JSP/Servlet, Forms, Discoverer and also some client programs such as Application Desktop Integrator. Any program which makes a SQL*Net connection to the Oracle Applications database needs to be trusted at some level. Oracle Applications uses the Server Security feature to ensure that such SQL*Net connections are coming from trusted machines and/or trusted programs.

The Server Security feature of Application Object Library supports authentication of

application server machines and code modules in order to access the database. When Server Security is activated, application servers are required to supply server IDs (like passwords) and/or code IDs to access a database server. Server IDs identify the machine from which the connection is originating. Code IDs identify the module and patch level from which the connection is originating. Code IDs are included in applications code by development. The database server can be set to allow access only from specific machines and/or by code at a desired patch level.

The application server security feature is not initially activated. You should activate it by using the commands described in this section.

Application Server Security can be in one of three states:

- OFF - Server security is not checked. Any application server machine can access the database. Code IDs are also not checked. Use this option on test systems or if you have full control over the software on all machines which can physically access your database.

- ON - Some level of trust is required to access the database. Either the application server must be registered with the database or the code must pass a module and version ID known to be trusted. Use this option only if you wish to maintain compatibility with application servers that you cannot yet patch to the code level required for best security.

- SECURE - Full trust is required for access to the database. Only registered application server machines may connect to the database, and only trusted code modules may connect to the database. This is the default setting for Release 12.

Server Security Status

You can check the Server Security status for a particular database using the STATUS command in the AdminAppServer utility, and providing the dbc file corresponding to that database. The STATUS command will display whether server security is turned on and whether the server ID in the dbc file is currently valid.

To check the Server Security status for a database:

On UNIX:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
     STATUS DBC=<dbc file path>
```

On Windows:

```
jre -classpath %CLASSPATH% \
     oracle.apps.fnd.security.AdminAppServer apps/apps \
     STATUS DBC=<dbc file path>
```

> **Important:** Check the Server Security status of your databases before you activate server security and ensure that all desired application servers have been registered.

Adding, Updating, or Deleting Server IDs

Application servers can be registered as trusted machines with a database server. This works by generating a large random ID number and storing that in both the application server and the database. When the application server attempts to connect to the database it will supply its server ID and the database will verify that it matches a trusted machine. The server ID is stored as a parameter in the DBC file for each application server. It can also be defined as an environment variable. The AdminAppServer utility is used to generate server IDs and then register them with a database server.

To add a server ID

Server ID values are generated by the AdminAppServer utility, and therefore cannot be supplied on the command line. They will be added to the database automatically when the AdminAppServer is used to create a dbc file.

On UNIX:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
ADD [SECURE_PATH=$FND_TOP/secure] \
DB_HOST=<database host> \
DB_PORT=<database port> \
DB_NAME=<database sid>
```

On Windows:

```
jre -classpath %CLASSPATH \
oracle.apps.fnd.security.AdminAppServer apps/apps \
ADD [SECURE_PATH=$FND_TOP/secure] \
DB_HOST=<database host> \
DB_PORT=<database port> \
DB_NAME=<database sid>
```

To update a server ID

You can update an application server's server ID at any time. From the command line enter:

On UNIX:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
UPDATE DBC= <dbc file path> APPL_SERVER_ID
```

On Windows:

```
jre -classpath %CLASSPATH% \
oracle.apps.fnd.security.AdminAppServer apps/apps \
UPDATE DBC=
```

By providing the APPL_SERVER_ID argument, you will force a new server ID to be generated and added to your DBC file. If the APPL_SERVER_ID argument is not provided, AdminAppServer will take care of syncing up the server ID's of your dbc file and your database automatically, if required.

To delete a server ID

Server IDs can be deleted. This must be done using the AdminAppServer utility as follows:

On UNIX:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
DELETE DBC= <dbc file path> APPL_SERVER_ID
```

On Windows:

```
jre -classpath %CLASSPATH% \
oracle.apps.fnd.security.AdminAppServer apps/apps \
DELETE DBC=
```

Activating Server Security

You can turn the server security feature to OFF, ON, or SECURE mode using the AdminAppServer utility. When you turn off server security, you will not change or delete your server IDs. You can re-enable server security without having to recreate server IDs for existing registered application servers.

On UNIX:

To activate basic server security on UNIX, enter on the command line:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
    AUTHENTICATION ON DBC=<dbc file path>
```

To activate full server security (SECURE mode) on UNIX, enter:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
    AUTHENTICATION SECURE DBC=<dbc file path>
```

To deactivate server security on UNIX, enter:

```
java oracle.apps.fnd.security.AdminAppServer apps/apps \
    AUTHENTICATION OFF DBC=<dbc file path>
```

On Windows:

```
jre -classpath %CLASSPATH% \
 oracle.apps.fnd.security.AdminAppServer apps/apps \
 AUTHENTICATION ON DBC=<dbc file path>
```

To activate full server security (SECURE mode), from the command line, enter:

```
jre -classpath %CLASSPATH% \
    oracle.apps.fnd.security.AdminAppServer apps/apps \
    AUTHENTICATION SECURE DBC=<dbc file path>
```

To deactivate server security, from the command line, enter:

```
jre -classpath %CLASSPATH% \
    oracle.apps.fnd.security.AdminAppServer apps/apps \
    AUTHENTICATION OFF DBC=<dbc file path>
```

Restricting Access to Responsibilities Based on User's Web Server

Oracle Applications has the capability to restrict access to a responsibility based upon the Web server from which the user logs in. This capability is provided by tagging Web servers with a "server trust level."

The server trust level indicates the level of trust associated with the Web server. Currently, three trust levels are supported: 1) administrative, 2) normal, and 3) external. Web servers marked as "administrative" are typically those used exclusively by system administrators. These servers are considered secure and may contain highly sensitive information. Servers marked as "normal" are those used by employees within a company's firewall. Users logging in from normal servers have access to only a limited set of responsibilities. Lastly, servers marked as "external" are those used by customers or employees outside of a company's firewall. These servers have access to an even smaller set of responsibilities.

Setting the Server Trust Level for a Server

To assign a trust level to a Web server, set the Node Trust Level profile option. The Node Trust Level profile option uses the Server profile hierarchy type, meaning that the value of the profile depends on the particular middle-tier server accessing the profile. This profile option can be set to either 1, 2, or 3, with the following meanings.

- 1 - Administrative
- 2 - Normal
- 3 - External

To avoid having to set the Node Trust Level profile option for every Web server, you may wish to set it to a default level of trust at the site level, such as level 1. If no value is set for the Node Trust Level profile option for a Web server, the Web server is assumed to have a trust level of 1 (Administrative).

Restricting Access to a Responsibility

To restrict access to a responsibility, set the security-based Responsibility Trust Level (internal name APPL_SERVER_TRUST_LEVEL) profile option value for that responsibility to be the number 1, 2, or 3. Setting this profile value ensures that only Web servers with the same or greater privileged trust level may access that responsibility.

Like the Node Trust Level profile option, the default value for the Responsibility Trust Level is 1.

When fetching the list of valid responsibilities for a user, Oracle Applications checks to find only responsibilities with a Responsibility Trust Level value greater than or equal to the Web server's Node Trust Level. In this way, a responsibility with Responsibility

Trust Level set to 1 would only be available if the Web server has the Node Trust Level set to 1 as well. A responsibility with Responsibility Trust Level set to 2 would only be available if the Web server has Node Trust Level set to either 1 or 2.

Application Object Library AOL/J Setup Test Suite

The AOL/J Setup Test suite consists of Java Server Pages (JSPs) and can be used to diagnose AOL/J setup issues. These JSPs exercise various features of AOL/J and provide feedback on the results. The test suite is accessed from the URL:

```
http://<host_name>:<port_number>/OA_HTML/jsp/fnd/aoljtest.jsp
```

where host_name and port_number correspond to the host name and port number of your instance's Apache listener. The host name and port number values are normally found in the APPS_SERVLET_AGENT profile option.

When you access the test suite, you will be asked to provide login information for your instance:

- Applications Schema Name
- Applications Schema Password
- TWO_TASK
- Host Name
- Port Number

The following is a list of functions and tests you can run with your instance.

- Connection Test
  - Locate DBC File
  - Verify DBC Settings
  - AOL/J Connection Test
- Virtual Directory settings
- APPS_WEB_AGENT
  - Virtual Directory Settings
  - FND_WEB.PING
  - Custom Authentication
  - GFM (Generic File Manager)

- APPS_SERVLET_AGENT
  - Virtual Directory Settings
  - Servlet Ping
  - Jsp Ping
- APPS_FRAMEWORK_AGENT
  - Virtual Directory Settings
  - Servlet Ping
  - Jsp Ping
  - Cabo Setup Tests
  - X Server Accessibility
  - OA Framework System Info
  - Versions for Loaded Classes
- HELP_WEB_AGENT
- TCF
  - Test Connection
- Tool Launcher Profile Settings
  - ICX_FORMS_LAUNCHER
  - ICX_REPORT_LAUNCHER
  - ICX_DISCOVERER_LAUNCHER
- Application Login
  - Login Page
  - Show Responsibilities
  - Show Session Properties

Using Oracle Application Framework

Oracle Application Framework is the Oracle Applications development and deployment platform. It is a 100% Java and XML, middle-tier application framework and services for the rapid development and deployment of HTML based applications.

For information on Oracle Application Framework, see "Oracle Application Framework Documentation Resources, Release 12", Note 391554.1 on Oracle*MetaLink*.

AutoConfig and Oracle Applications Manager

Oracle Applications Manager (OAM) provides access to the AutoConfig configuration management tool via the AutoConfig tab in the Site Manager screen. Previous configuration settings can easily be compared with the current settings, allowing changed settings to be identified and rolled back as necessary. The configuration editor also helps you add custom variables to the context files, to allow AutoConfig and AutoPatch to preserve your system configuration customizations.

For more information on AutoConfig, see: "Using AutoConfig to Manage System Configurations with Oracle E-Business Suite Release 12", Note 387859.1, on Oracle *MetaLink.*

For more information on Oracle Applications Manager, see: Introduction to Oracle Applications Manager, page 5-1.

From the Oracle Applications Manager Site Map, AutoConfig is on the Administration tab under the System Configuration heading. Click on the AutoConfig link to go to the tool's home page.

From the AutoConfig home page, you can view details of a context file by clicking Show. For example, you can see Version, Path, Creation Date. Last Updated By, Status (for example, Write succeeded), and Comments.

You can select a context file and choose Edit Parameters. This is the only supported way to modify parameters that AutoConfig maintains: if you edit any context files manually, they will be overwritten the next time AutoConfig is run.

Under Edit Parameters, choosing the Local tab will open a screen that shows environment-related details, including local host machine name, virtual hostname (used to connect), local domain name, and platform type.

You can select a context file and click on Show History, to see details of any previous configurations.

The Custom tab under Edit Parameters allows you to add a new context variable (not maintained by AutoConfig) to the context file. This is the only supported way to add customizations.

Related Topics

Configuration Overview, page 5-8

3

Oracle Applications Tablespace Model and Migration Utility

Introduction to Oracle Applications Tablespace Model

The Oracle Applications Tablespace Model (OATM) uses twelve consolidated tablespaces (including three system tablespaces: temporary, system and undo segments) and provides support for locally managed tablespaces. OATM was introduced in Release 11i.10. In prior 11i releases of the E-Business Suite, each product was allocated two tablespaces, one for data and one for indexes. The Migration Utility is a menu-based PERL program and a series of sizing estimate reports that enables conversion of E-Business Suite applications schemas either in a single comprehensive migration or a phased, schema-by-schema migration. In general Oracle recommends performing a single comprehensive migration, however this requires a sufficient amount of down time and disk space. Oracle does not support partial migration of tablespaces. You must still migrate all schemas when performing a phased schema-by-schema migration.

With OATM, each database object is mapped to a tablespace based on its Input/Output characteristics, which include object size, life span, access methods and locking granularity. This model allows for easier maintenance, and reduced space usage for the E-Business Suite.

Another configuration supported with Locally Managed Tablespaces, User Extent Management, is only relevant when Dictionary Tablespaces have been migrated to Locally Managed Tablespaces. Because migration to OATM is performed at the object level and not at the tablespace level, User Extent Management is not relevant. The default recommended configuration in OATM is Uniform Extent Management.

Advantages of Migrating to OATM

Migrating database objects to OATM provides the following benefits:

- Fewer and more consolidated tablespaces
- Locally Managed Tablespaces
- Accounts for the I/O characteristics of an object
- Reclaims space after migration
- Real Application Cluster (RAC) Support

Fewer and More Consolidated Tablespaces

OATM contains twelve locally managed tablespaces for all products, including temporary tablespace, system tablespace, and undo segments. The previous tablespace model contained two tablespaces for each Oracle product resulting in hundreds of tablespaces.

Locally Managed Tablespaces

This model provides support for either Uniform or Auto-allocate extent management, available with Locally Managed Tablespaces. Locally Managed Tablespaces have benefits over Dictionary Tablespaces in the previous model and allow for the sizes of extents to be determined automatically by the system (Auto-allocate). Alternatively, all extents can have the same size (Uniform) in and override object storage options.

OATM implements Automatic segment-space management, a simpler and more efficient way of managing space within a segment. It completely eliminates any need to specify and tune the PCTUSED, FREELISTS, and FREELISTS GROUPS storage parameters for schema objects created in the tablespace. Automatic segment-space management delivers better space utilization than manual segment-space management, and is self tuning because it scales by increasing the number of users, as well as instances. For a Real Application Clusters environment, automatic segment-space management enables dynamic affinity of space to instances, which avoids the hard partitioning of space inherent with using free list groups.

Uniform Extent Size

The value for uniform extent size should be carefully selected based on system requirements. For production environments and large tablespaces like transaction tables, transaction indexes, interfaces, summaries, archives, and media, a uniform extent size of 1MB or 10MB (with caution) should be considered. Choosing an extent size that is too small can result in frequent extensions and performance degradation of the system.

The Release 12 Rapid Install production database is delivered out-of-the-box with locally managed tablespaces with uniform extent sizes of 128Kb. If this size is not the best match for the characteristics of your system, you can follow subsequent

re-migration steps to create new tablespaces with the desired uniform extent size and migrate objects to those new tablespaces. The OATM Migration Utility supports all possible configurations of locally managed tablespaces. You have the flexibility to override the default recommendation of uniform extent size with Auto-Allocate extent management as per your requirements. Uniform extent size is also configurable.

> **Note:** The Oracle database server does not as yet support altering the extent management type or uniform extent size after the locally managed tablespaces have been created. Therefore, if the originally used extent management type or uniform extent size needs to be modified, re-creation of the tablespaces and re-migration of objects is the only available option.

Re-migration steps

Use the customization option to change names of existing OATM tablespaces as listed in the following table:

Tablespace Type	Old Tablespace Name	New Tablespace Name
Transaction Tables	APPS_TS_TX_DATA	APPS_TS_TX_DATA_1MB
Transaction Indexes	APPS_TS_TX_IDX	APPS_TS_TX_IDX_1MB
Reference	APPS_TS_SEED	
Interface	APPS_TS_INTERFACE	APPS_TS_INTERFACE_1MB
Summary	APPS_TS_SUMMARY	APPS_TS_SUMMARY_1MB
Nologging	APPS_TS_NOLOGGING	
Advanced Queuing/AQ	APPS_TS_QUEUES	
Media	APPS_TS_MEDIA	APPS_TS_MEDIA_1MB
Archive	APPS_TS_ARCHIVE	APPS_TS_ARCHIVE_1MB

Please note that for the types of tablespaces for which you do not wish to modify the uniform extent size, you should not change the tablespace name and objects will remain in the respective tablespace with the originally selected extent size and no attempt will be made to migrate them. Sizes of those tablespaces should be ignored in the sizing report and 'alter tablespace' statements removed from the 'create tablespace' script

before the script is run.

From the main menu, select option 7. Run Customization Steps:

Customization

1. Register new tablespace - tablespace type
2. Change name of the existing tablespace
3. Register object classification
4. Change object classification

Please select "2. Change name of existing tablespace".

```
Enter tablespace type: TRANSACTION_TABLES
Enter new tablespace name: APPS_TS_TX_DATA_1MB
Tablespace name for TRANSACTION_TABLES changed to APPS_TS_TX_DATA_1MB.

Do you want to continue changing tablespace names?[Y]:

Enter tablespace type: TRANSACTION_INDEXES
Enter new tablespace name: APPS_TS_TX_IDX_1MB

...
```

The same steps should be performed for all tablespace types for which you want to change the uniform extent size. When the customizations are complete, please run the following steps as described in the chapter Oracle Applications Tablespace Migration Utility, page 3-9.

- Run the sizing process and create new tablespaces.
- Run migration command generation.
- Complete post-migration steps and drop old tablespaces that have no remaining segments.

I/O Characteristics of an Object

OATM takes into account the following object I/O characteristics of an object:

- size
- life span
- access methods
- locking granularity

Automatic Segment-space Management

Automatic segment-space management is a simpler and more efficient way of managing space within a segment. It completely eliminates any need to specify and tune the PCTUSED, FREELISTS, and FREELISTS GROUPS storage parameters for schema objects created in the tablespace. Automatic segment-space management delivers better space utilization than manual segment-space management, and is self tuning because it scales with the increasing number of users and instances. For a Real Application Cluster (RAC) environment, automatic segment-space management enables dynamic affinity of space to instances, which avoids the hard partitioning of space inherent with using free list groups.

Reclaims Space After Migration

The tablespace migration utility migrates objects from the existing dictionary-managed tablespaces to locally managed tablespaces with automatic segment management and either uniform or Auto-allocate extent management. As a result, space is better utilized and less wasted. Migration of table and index segments from one tablespace to another also reclaims unused space, especially for indexes that are fragmented when the index columns are inserted, updated or deleted frequently.

Real Application Cluster (RAC) Support

OATM facilitates Real Application Cluster (RAC) support because of its reduced number of tablespaces. RAC is an Oracle database feature that harnesses the processing power of multiple interconnected computers where all active instances can concurrently execute transactions against a shared database disk system. The new tablespace model is critical when implementing RAC on Linux, where currently there is a limitation of 255 raw devices.

Additional Benefits

OATM provides the following additional benefits:

- Facilitates administration and configuration ease
- Increases block-packing to reduce the overall number of buffer gets.

OATM Tablespaces

The advantages of OATM's product tablespaces are best understood in terms of the tablespace model that preceded it. This model contained two tablespaces for each Oracle Applications product. One tablespace was allocated for tables and one for indexes. In this model, the standard naming convention for tablespaces contained the product's Oracle schema name with a suffix of either "D" for "Data" tablespaces or "X"

for "Index" tablespaces. For example, the tablespaces APD and APX were the default tablespaces for Oracle Payables tables and indexes, respectively.

In contrast to the previous tablespace model, OATM contains nine default tablespaces for applications objects in addition to Undo, Temp and System database tablespaces. Indexes on transaction tables are held in a separate tablespace dedicated for transaction table indexes whereas all other indexes are held in the same tablespace as the parent/base table. All Oracle Applications product schemas now have a default tablespace set to point to the TRANSACTION_TABLES tablespace type for data objects and the TRANSACTION_INDEXES tablespace type for index objects.

The Oracle Applications Tablespace Model uses Locally Managed Tablespaces and supports either Uniform or Autoallocate extent management. Another configuration supported with Locally Managed Tablespaces - User Extent Management, is of relevance only in case of Dictionary Tablespaces that have been migrated over to Locally Managed Tablespaces. Because migration to OATM is performed at the object level and not at the tablespace level, User Extent Management is not relevant. The default recommended configuration in OATM is Uniform Extent Management. The migration utility recommends the default of 128k uniform extents which can be changed to suit the customer database. Tablespace types are listed in the following table:

Tablespace Types

Tablespace Type	Tablespace Name	Content
Transaction Tables	APPS_TS_TX_DATA	Tables that contain transactional data.
Transaction Indexes	APPS_TS_TX_IDX	Indexes on transaction tables.
Reference	APPS_TS_SEED	Reference and setup data and indexes.
Interface	APPS_TS_INTERFACE	Interface and temporary data and indexes.
Summary	APPS_TS_SUMMARY	Summary management objects, such as materialized views, fact tables, and other objects that record summary information.
Nologging	APPS_TS_NOLOGGING	Materialized views not used for summary management and temporary objects.

Tablespace Type	Tablespace Name	Content
Advanced Queuing/AQ	APPS_TS_QUEUES	Advanced Queuing and dependent tables and indexes.
Media	APPS_TS_MEDIA	Multimedia objects, such as text, video, sound, graphics, and spatial data.
Archive	APPS_TS_ARCHIVE	Tables that contain archived purge-related data.
Undo	UNDO	Automatic Undo Management (AUM) tablespace. UNDO segments are identical to ROLLBACK segments when AUM is enabled.
Temp	TEMP	Temporary tablespace for global temporary table, sorts, and hash joins.
System	SYSTEM	System tablespace used by the Oracle Database

Tablespace Classification

OATM relies on specific explicit and implicit classification rules that are determined based on storage considerations for the object type in question. The Oracle Tablespace Migration Utility migrates objects based on these rules. The following table contains rules for implicit classifications that are applied in OATM, based on object types. Objects that do not have an implicit classification rule or an explicit object classification are migrated to the default tablespaces of the schema in which they reside.

Explicit Classification Rules

Explicit object classifications are seeded by Oracle based on the I/O characteristics of the object.

Implicit Classification Rules

The following table contains implicit classification rules for the Oracle Applications Tablespace Migration Utility.

Implicit Classification Rules

Object Type	Tablespace_Type
AQ Tables	AQ
IOTs (Index Organized Tables)	Transaction_Tables
Materialized Views	Summary
Materialized View Logs	Summary
All other Indexes	Same Tablespace type as the table
Domain Indexes	Transaction_Indexes
Indexes on Transaction Tables	Transaction_Indexes

Customizations and Extensions

The Oracle Applications Tablespace Migration Utility is primarily designed to migrate tables, indexes, materialized views, materialized view logs and other database objects that are owned by standard Oracle Applications schemas from their existing tablespace model to OATM. Custom or third party schemas can also be migrated using the Oracle Tablespace Migration Utility, customer preferred methods, or a database management tool such as the Oracle Enterprise Manager (OEM). Custom objects in standard Oracle application product schemas are migrated by default.

The Oracle Tablespace Migration Utility enables the following customizations:

- Changing tablespace names
- Registering custom tablespace types
- Registering custom object-tablespace classifications
- Changing existing object classifications.

Migrating Custom or Third Party schemas

Login to the Forms-based version of Oracle Applications with the System Administrator Responsibility. Navigate to Security -> ORACLE -> Register and register the external schema(s) if they are not already registered. Set Privilege to "Enabled".

Preventing Migration of Specific Schemas

In some cases, you may not want to migrate some of your schemas such as non-Oracle schemas that are registered with Oracle Applications. To accomplish this, you must disable those schemas by accessing System Administrator responsibility -> Security -> ORACLE -> Register and then selecting either "External" or "Disabled" for the schema in question. Conversely, if you want to flag specific schemas for migration, you can enable them by accessing System Administrator responsibility -> Security -> ORACLE -> Register and then selecting "Enabled".

Introduction to the Oracle Applications Tablespace Migration Utility

The Tablespace Migration Utility is a menu-based PERL program that enables you to estimate future space requirement for the tablespaces and to migrate the Applications database to OATM. Log files are available for user viewing and are created in the working directory from which your run the PERL program. The Log file name and location are displayed once you choose the required option.

The Tablespace Migration Utility enables you to perform either a single, comprehensive migration of all schemas or a phased, schema-by-schema migration. To minimize downtime, Oracle recommends that you perform the single comprehensive migration of all schemas, however this requires a sufficient amount of down time and available disk space. If , you do not have sufficient down time or disk space to accomplish this, then you can run the phased schema-by-schema migration. Once you migrate an object from its existing tablespace to OATM, this process cannot be reversed. Oracle does not support the rollback of schemas migrated to OATM in a phased schema-by-schema migration. The only method for achieving this result is to recover the migrated schemas from a backup.

> **Note:** Once you initiate migration of one or more schemas to OATM it is not possible to perform additional migrations from a different PERL menu. You must wait until one migration is completed before beginning another.

Planning for Migration

Sizing Requirements

Whenever possible, Oracle recommends the following:

- perform a single comprehensive migration of all schemas instead of performing a phased schema-by-schema migration.
- perform test runs to determine the amount of down time required to perform a

comprehensive migration of all schemas

- secure twice as much disk space as your existing space to perform the a single comprehensive migration of all schemas, and to be operational using the new model.

Setting Up the Tablespace Migration Utility

Setting up the Tablespace Migration Utility

When you first install the Tablespace Migration Utility, it does the following:

- Copies the PERL menu script, fndtsmig.pl, to the FND_TOP/bin directory.
- Copies the SQL scripts for the Tablespace Migration utility to FND_TOP/patch/120/sql directory.
- Copies the PLS files for the Tablespace Migration utility to the FND_TOP/patch/120/sql directory and creates packages in the database.
- Compiles the Java files into the packages oracle.apps.fnd.tsmig, in FND_TOP/java/apps.zip file.
- Creates and seeds the following tables:
  - FND_TABLESPACES
  - FND_OBJECT_TABLESPACES
  - FND_TS_SIZING
  - FND_TS_MIG_CMDS
  - FND_TS_MIG_RULES
  - FND_TS_PROD_INSTS

Invoking the Tablespace Migration Utility Main Menu

Invoke the Tablespace Migration Utility main menu by performing the following:

1. Run the fndtsmig.pl PERL script:

   ```
   perl $FND_TOP/bin/fndtsmig.pl.
   ```

2. Provide the following information when prompted to access the Tablespace Migration Utility main menu:

- OATM configuration file
- APPL_TOP[]
- FND_TOP[/fnddev/fnd/12.0]
- Database Connect String [dummy]: atgtsqa
- Password for your 'SYSTEM' ORACLE Schema
- Password for your 'SYS' ORACLE Schema
- Oracle Application Object Library Schema name [APPS]
- Password for APPS
- APPLSYS Schema Name [APPLSYS]

Information can be provided in an interactive manner or by providing an OATM configuration file with all information already specified. An OATM configuration file has the following format with the following valid tags:

```
APPL_TOP                  - valid APPL_TOP value or $ENV$
FND_TOP                   - valid FND_TOP value or $ENV$
APPS_SCHEMA               - valid apps schema name
APPLSYS_SCHEMA            - valid applsys schema name
ALLOC_TYPE                - UNIFORM/AUTOALLOCATE
UES                        - valid integer (uniform extent size)
DBF_DIR                   - valid directory for generated database file
INDIVIDUAL_DATAFILE_SIZE  - maximum datafile size
NUM_WORKER                - integer (number of concurrent workers.)
MIGRATION_SCHEMA          - % or comma separated list of schemas
CONNECT_STRING            - database connect string
AUTO_START_MIGRATION      - should migration be started automatically
after
                            preparatory steps are completed
Y(default)/N
```

ENV is a reserved word for the OATM configuration file. If ENV is used as a value for a specific token, then that token's real value will be derived from the customer's environment dynamically during the runtime.

Example

```
<!-- OATM Migration Configuration File>
<OATM>
  <APPL_TOP> $ENV$ </APPL_TOP>
  <FND_TOP> $ENV$ </FND_TOP>
  <CONNECT_STRING> atgtsmqa </CONNECT_STRING>
  <APPS_SCHEMA> APPS </APPS_SCHEMA>
  <APPLSYS_SCHEMA> APPLSYS </APPLSYS_SCHEMA>
  <ALLOC_TYPE> U  </ALLOC_TYPE>
  <UES> 1024 </UES>
  <DBF_DIR> /slot05/oracle/atgtsmqadata/ </DBF_DIR>
  <INDIVIDUAL_DATAFILE_SIZE> 2000 </INDIVIDUAL_DATAFILE_SIZE>
  <NUM_WORKER> 8 </NUM_WORKER>
  <MIGRATION_SCHEMA>  %   </MIGRATION_SCHEMA>
  < AUTO_START_MIGRATION >  Y  </ AUTO_START_MIGRATION >
</OATM>
```

If an OATM configuration file is provided, values specified in the configuration file can still be overridden by values provided interactively. A summary of all provided information will be displayed and can be reviewed and corrected if needed.

> **Note:** Please note that the configuration file will not be modified accordingly by information entered interactively. Values provided interactively will have an effect only for the OATM session when provided. No new tags added to the configuration files will be recognized. All values specified in the OATM configuration file above are just an example, not Oracle recommended values.

Understanding the Tablespace Migration Utility Main Menu

The Tablespace Migration Utility main menu lists six required sequential steps and one optional step to migrate your database objects to OATM. These steps are categorized in three phases. In Phase 1, you perform the necessary preparation steps for migrating your objects to OATM. In Phase 2, you perform the necessary steps to migrate your objects to OATM and in Phase 3 you run the required post migration steps.

1. (Preparatory Step) Generate Migration Sizing Reports
2. (Preparatory Step) Create New Tablespaces
3. (Preparatory Step) Generate Migration Commands
4. (Migration Step) Execute Migration Commands
5. (Migration Step) Run Migration Status Reports
6. (Post Migration Step) Run Post Migration Steps
7. (Optional) Run Customization Steps

8. (Optional, batch mode) Run Migration in Batch Mode

Steps 1, 2, 3 can be executed while Oracle Applications are still available to users. Step 4 must be executed when Oracle Applications are not available to users and steps 5 and 6 must be completed before making Oracle Applications available to users again. If you choose to perform optional Step 7 you should do so before the other steps.

> **Caution:** Oracle highly recommends that you back up your database after performing Step 3, which is the final preparatory step. You should have a backup copy of your database before performing the subsequent migration steps.

Backing up the Database

Oracle highly recommends that you perform a backup of your database ***twice*** as follows;

1. **Copy of previous tablespace model**. Back up your database before performing any of the migration steps. Because Oracle does not support the rollback of migrated database objects, this is the only available method for restoring your previous tablespace model.

2. **Copy of database that has been prepared for migration**. Backup your database after performing Step 3: Generate Migration Commands. This enables you to migrate your database objects to OATM using your last good copy of a database that has been prepared for migration.

If you choose to use the menu option 8. Migration in Batch mode, you should perform the database backup before the migration and/or after the preparatory steps are completed, if you choose to run the preparatory steps separately.

Phase 1: Preparatory Steps

Step 1: Generate Migration Sizing Reports

Select Step 1: Generate Migration Sizing Reports to access a list of reports that help you to gauge the space requirements for the new tablespace model and that assist you in determining which migration approach best suits your requirements. The reports perform sizing estimation by executing a program that calculates the size of each object using package DBMS_SPACE.UNUSED_SPACE and by populating table FND_TS_SIZING. The sizing reports use the data in this table to display the required information. The Plan Migration menu contains the following options:

1. Calculate total space required by each new tablespace to migrate all Oracle Application product schemas

2. Calculate total space required by each new tablespace to migrate each Oracle Application product schema (relevant for a schema-by-schema migration)

3. Calculate total space required by each Oracle Applications schema, with details for each object

4. Display Sizing Exception report

Option 1: Calculate Total Space Required by Each New Tablespace to Migrate all Oracle Application Product Schemas

Choose option 1 to calculate the total space required by each new tablespace when performing a single comprehensive migration of all Oracle Applications product schemas, and to generate the report fndtrep1.txt. Before running the report, the program prompts you to specify whether the information in the sizing table is current or must be updated. Enter the following to calculate space requirements for performing a single comprehensive migration of all schemas:

- Whether the sizing information is current or must be gathered before running the report.

- If you selected "Y" for the previous option, provide the extent management type for the tablespaces, since the space requirements are dependent on this.

- If Uniform Extent Management, provide the uniform extent size for the tablespaces or choose the default value provided on the screen.

Example

```
Sizing Program was last run on 02-SEP-03
Do you want to run the Sizing program again before running the report
[N]:y
Enter the Extent Allocation type A(utoallocate) or U(niform Extent Size)
[U]: U
Enter Uniform Extent Size for the Tablespaces in KBytes[128]:
```

Option 2: Calculate Total Space Required by Each New Tablespace to Migrate Each Oracle Application Product Schema

Choose Option 2 to calculate total space required by each new tablespace when migrating individual Oracle Application product schemas one at a time, and to generate the report fndtrep2.txt. The program prompts your for the following information:

- Schema name, enter the percent sign (%) for all schemas,

- Whether the sizing information is current or must be gathered before running the report.

- If you selected Y for the previous option, provide the extent management type for the tablespaces, since the space requirements are dependent on this.

- If Uniform Extent Management, provide the uniform extent size for the tablespaces or choose the default value provided on the screen.

Example

```
Enter the Schema name: <HR>

Sizing Program was last run on 02-SEP-03
Do you want to run the Sizing program again before running the report
[N]:
```

Option 3: Calculate Total Space Required by Each Oracle Applications Schema with Details for Each Object

Choose option 3 to calculate total space required by each Oracle Applications schema with details for each object and to generate the report fndtrep4.txt. The program prompts your for the following information:

- Schema name, enter the percent sign (%) for all schemas,
- Whether the sizing information is current or must be gathered before running the report.
- If you selected Y for the previous option, provide the extent management type for the tablespaces, since the space requirements are dependent on this.
- If Uniform Extent Management, provide the uniform extent size for the tablespaces or choose the default value provided on the screen.

Example

```
Enter the Schema name: <HR>

Sizing Program was last run on 02-SEP-03
Do you want to run the Sizing program again before running the report
[N]:
```

Option 4: Display Sizing Exception Report

Choose option 4 to generate report fndtrep5.txt, listing all the objects for which sizing estimation generated an error. The program prompts you for the schema name. Enter the percent sign (%) for all schemas.

Example

```
Enter the Schema name: <%>
```

Step 2: Create New Tablespaces

Select Step 2: Create New Tablespaces to create the OATM tablespaces to which you will migrate your database objects. The Create New Tablespaces menu contains the following options:

1. Generate the Tablespace Creation Script
2. Create New Tablespaces

Option 1: Generate the Tablespace Creation Script

Option 1 prompts you for the extent allocation type, such as Autoallocate, Uniform

Extent, and Uniform Extent Size, which will be used for creating the new tablespaces. The utility prompts for the name of the directory in which the datafile will be created. For every tablespace created as part of OATM, the utility prompts you for information, such as number and size of the datafiles. The utility will append a sequence number to the tablespace name and a .dbf extension to generated the datafile name. For example, if you enter the datafile directory as "/u01/oradata" and for transaction data tablespace, APPS_TS_TX_DATA, you enter the number of datafiles as 2 and size as 2000M, the utility will create tablespace creation script with 2 datafiles for the tablespace named "/u01/oradata/APPS_TS_TX_DATA01.dbf" and "/u01/oradata/APPS_TS_TX_DATA02.dbf", each of size 2000M. To create datafiles of different sizes or in different locations, you must modify the generated script, crtts.sql.

If you do not have limited disk space, create all tablespaces with the estimated sizes listed in report #1. This will eliminate the need to extend the tablespaces as the migration proceeds. If you do not have enough disk space to create all the tablespaces with the total size, use the estimated values in report #2 <Schema Name> for reference. If your operating system has a limit on the size of a dbf file, ensure you enter a value less than this when prompted.

Example

```
Enter the Extent Allocation type A(utoallocate) or U(niform Extent
Size)[U]:
**************************************************************
The utility will append a sequence number to the tablespace name
and a .dbf extension to generate the datafile names.
Datafile size should not be greater than OS file size limit.
Please edit the generated script to change the file name/size
**************************************************************

Enter the absolute path for the datafiles directory: /u01/oradata
Enter the Number of Datafiles for Transaction data tablespace[1]: 2
Enter the Datafile Size for Transaction data tablespace (MB): 2000
Enter the Number of Datafiles for Transaction index tablespace[1]: 2
Enter the Datafile Size for Transaction index tablespace (MB): 2000
Enter the Number of Datafiles for Reference tablespace[1]: 1
Enter the Datafile Size for Reference tablespace (MB): 2000
Enter the Number of Datafiles for Interface tablespace[1]: 1
Enter the Datafile Size for Interface tablespace (MB): 1700
Enter the Number of Datafiles for Summary tablespace[1]: 2
Enter the Datafile Size for Summary tablespace (MB): 2000
Enter the Number of Datafiles for Nologging tablespace[1]:
Enter the Datafile Size for Nologging tablespace (MB): 60
Enter the Number of Datafiles for Archive tablespace[1]:
Enter the Datafile Size for Archive tablespace (MB): 1400
Enter the Number of Datafiles for Queue tablespace[1]:
Enter the Datafile Size for Queue tablespace (MB): 150
Enter the Number of Datafiles for Media tablespace[1]:
Enter the Datafile Size for Media tablespace (MB): 2000
```

Option 2: Create New Tablespaces

Select option 2 to create new tablespaces by executing the script crtts.sql that was generated in the previous step. This script does not check the operating system limitation for the maximum size of a file.

Step 3: Generate Migration Commands

Select Step 3: Generate Migration Commands to generate migration commands for your schemas. The Generate Migration Commands menu contains the following options:

> **Caution:** You should not generate migration commands if *migration is already in progress.* OATM utility will prevent the generation of migration commands while generation of the commands or migration process is already in progress. Oracle does not recommend manually updating table FND_TS_MIG_CMDS, especially while you are generating or executing migration commands.

1. Invalid Indexes Report.
2. Generate Migration Commands for all Schemas
3. Generate Migration Commands for a List of Schemas

Option 1: Invalid Indexes Report. Please correct/drop these before generating migration commands

Select Option 1 to generate a report listing all the indexes which are invalid in the Oracle Applications schemas that is stored in report fndinvld.txt. You must correct or drop all invalid indexes before generating migration commands for all schemas or for a given schema. This is especially relevant for context indexes. Invalid indexes on an object may cause errors during migration of the base table and invalid context indexes will not be moved.

Option 2: Generate Migration Commands for all Schemas

Select Option 2 to generate the commands for migrating the objects in all the schemas to the correct tablespace. The migration commands are stored in the table FND_TS_MIG_CMDS. You can check the generated log file fndgmcmd <timestamp>.log for errors during the generation process. A threshold object size is calculated based on the sizing data in FND_TS_SIZING to determine whether an object will be moved sequentially or in parallel. Migration commands for all objects with total blocks greater than or equal to threshold blocks are generated with the PARALLEL clause and execution mode as sequential. Migration commands for objects with total blocks less than threshold are generated with NOPARALLEL clause and execution mode as parallel. Partitioned objects are always executed sequentially regardless of their size.

Option 3: Generate Migration Commands for a List of Schemas

Select Option 3 to generate the commands for migrating the objects in a given list of comma separated schema names into the correct tablespace. The migration commands are stored in the table FND_TS_MIG_CMDS. You can check the generated log file fndgmcmd <timestamp>.log for errors during the generation process. A threshold object size is calculated based on the sizing data in FND_TS_SIZING to determine whether an

object will be moved sequentially or in parallel. Migration commands for all objects with total blocks greater than or equal to threshold blocks are generated with the PARALLEL clause and are executed sequentially. Migration commands for objects with total blocks less than threshold are generated with NOPARALLEL clause and are executed in parallel using multiple processes. Partitioned objects are always executed sequentially regardless of their size.

Example

```
Enter a comma separated list of schema names: HR,AP
```

> **Caution:** Oracle highly recommends that you back up your database after performing Step 3, which is the final preparatory step. You should have a backup copy of your database before performing the subsequent migration steps.

Phase 2: Migration Steps

Step 4: Execute Migration Commands

Select Step 4: Execute Migration Commands to execute migration commands for your schemas. The Execute Migration Commands menu contains the following options:

1. Execute Migration Commands for all Schemas
2. Execute Migration Commands for a List of Schemas
3. Migrate CTXSYS Schema

Return to the OATM menu is possible as soon as all migration processes (sequential, parallel, and java process for tables with LONG and LONG RAW columns) have started and you get the prompt – "Press Return key to continue...". Returning to the menu does not mean that migration has been completed. The migration processes are running in the background but you can return to the menu in order to monitor migration status/errors by running the migration progress report. Migration sessions are internally spawned using **nohup** and are immune to any hangup signals. Therefore VPN connection expiration, etc. should not pose any problems.

> **Caution:** You should not execute migration commands if *migration is already in progress.* The OATM utility will prevent generating migration commands as well as starting additional migration execution processes while migration process is in progress. Starting additional migration process can cause errors that can result in data corruption and will seriously impact migration process performance. Oracle does not recommend manually updating table FND_TS_MIG_CMDS, especially while you are generating or executing migration commands.

Option 1: Execute Migration Commands for all Schemas

Select Option 1 to migrate the objects in all the schemas to the correct tablespaces. You are prompted for the number of parallel processes as input. This option executes the generated commands from FND_TS_MIG_CMDS table which generates the following events in sequence:

1. All constraints, triggers, policies are disabled first and then the queues are stopped.

2. Java program oracle.apps.fnd.tsmig.TSMigration is executed to migrate all the tables with LONG and LONG RAW columns along with their indexes. A log file for the migration of tables with LONG is generated as fndmlong<timestamp>.log.

3. A sequential process is started that executes the script fndemseq.sql to move all the objects generated with execution mode as sequential. A log file for the sequential process is generated as fndemseq<timestamp>.log.

4. Multiple processes are started to execute the SQL script fndemcmd.sql to migrate the objects generated with execution mode as parallel. A log file is generated as fndemcmd<timestamp>.log.

Example

```
Are you sure you want to migrate all schemas[N]: y
Enter the maximum number of parallel processes[4]:10

Starting the Migration process for all schemas. Please wait...

Migration processes for tables with LONG and LONG RAW columns started.
Please monitor the log file
$APPL_TOP/admin/log/fndmlong20050120230037.log
for errors

Sequential migration process started. Please monitor the log file
$APPL_TOP/admin/log/fndemseq20050120230038.log for errors

Parallel migration processes started. Please monitor the log file
$APPL_TOP/admin/log/fndemcmd20050120230048.log for errors

Press Return key to continue...
```

Option 2: Execute Migration Commands for a List of Schemas

Select Option 2 to migrate the objects in a given list of schemas to the correct tablespaces. You are prompted for a list of comma separated schema names and then for the number of parallel process as input. This option executes the generated commands from FND_TS_MIG_CMDS table which generates the following events in sequence:

1. All constraints, triggers, policies are disabled first and then the queues are stopped.

2. Java program oracle.apps.fnd.tsmig.TSMigration is executed to migrate all the tables with LONG and LONG RAW columns along with their indexes. A log file for

the migration of tables with LONG is generated as fndmlong<timestamp>.log.

3. A sequential process is started that executes the script fndemseq.sql to move all the objects generated with execution mode as sequential. A log file for the sequential process is generated as fndemseq<timestamp>.log.

4. Multiple processes are started to execute the SQL script fndemcmd.sql to migrate the objects generated with execution mode as parallel. A log file is generated as fndemcmd<timestamp>.log.

Example

```
Enter a comma separated list of schema names: HR,AP

Enter the maximum number of parallel processes [4]: <10>
```

> **Note:** If your migration process terminates before it is completed, please check enqueue/dequeue status of queue - SYSTEM.TBLMIG_MESSAGEQUE by querying the following:
>
> ```
> select NAME, ENQUEUE_ENABLED, DEQUEUE_ENABLED
>          from dba_queues
>          where owner = 'SYSTEM'
>          and name = 'TBLMIG_MESSAGEQUE';
> ```

Option 3: Execute Migration Commands for CTXSYS Schema

The CTXSYS schema is not an APPS schema and in order to be included in the migration process, the CTXSYS schema has to be registered in the following manner: Login to the Forms-based version of Oracle Applications with the System Administrator Responsibility. Navigate to Security -> ORACLE -> Register and register the CTXSYS schema if it is not already registered. Then Set Privilege to "Enabled". CTXSYS schema objects are not classified by default and will be migrated to Transaction Tables and Transaction Index tablespaces, for CTXSYS tables and indexes respectively. Using Oracle Tablespace Migration Utility customizations steps, it is possible to classify CTXSYS objects to be migrated to the desired tablespace or tablespace type.

Select Option 3 to migrate CTXSYS schema objects to the correct tablespace. You are prompted for the number of parallel processes as input. This option executes the generated commands from FND_TS_MIG_CMDS table, which generates the following events in sequence:

1. All constraints, triggers, policies are disabled first and then the queues are stopped.

2. The Java program oracle.apps.fnd.tsmig.TSMigration is executed to migrate all the tables with LONG and LONG RAW columns along with their indexes. A log file for the migration of tables with LONG is generated as fndmlong.<timestamp>.log.

3. A sequential process is started that executes the script fndemseq.sql to move all the objects generated with execution mode as sequential. A log file for the sequential process is generated as fndemseq<timestamp>.log.

4. Multiple processes are started to execute the SQL script fndemcmd.sql to migrate the objects generated with execution mode as parallel. A log file is generated as fndemcmd<timestamp>.log.

Step 5: Run Migration Status Report

Select Step 5: Run Migration Status Report to run progress and error reports on the migration process. The Run Migration Status Report menu contains the following options:

1. Run Migration Status Report
2. Run Migration Error Report

Option 1: Run Migration Status Report

Select Option 1 to generate a report containing the number of successfully migrated objects, objects in error (if any) and the percentage of completion, per schema, or for all the schemas and a breakdown of the objects per object type. This option prompts you to enter the schema name and generates report fndtrep8.txt.

Example

```
Enter the Schema name[%]: <HR>
```

Option 2: Run Migration Error Report

Select Option 2 to generate a report that provides a list of objects that generated an error during the migration process that includes the error details. This option prompts you to enter the schema name and generates report fndtrep10.txt.

Example

```
Enter the Schema name[%]: <HR>
```

Phase 3: Post Migration Steps

Caution: When a standby database exists and/or you want all transactions to be recoverable on a database, tablespace, or object-wide level, it is recommended that you check that all objects have appropriate values for the logging attribute before and after the OATM migration process to ensure that all transactions are logged and can be recovered through media recovery.

Step 6: Run Post Migration Steps

Select Step 6: Run Post Migration Steps to determine which objects have not yet been migrated to OATM, to enable constraints, triggers, policies, and start queues, and to re-size old tablespaces. The Run Post Migration Steps menu contains the following

options:

1. Run Audit Reports
2. Enable the Constraints, Triggers and Policies, and Start Advanced Queues
3. Re-size Old Tablespaces
4. Generate script to drop empty tablespaces

> **Note:** After running post migration steps, you must perform a complete refresh of all materialized views. This is a required manual step that is not included in the Tablespace Migration Utility menu.

Option 1: Run Audit Reports

Select Option 1 to generate a report that provides a list of objects that have not been migrated to the correct tablespace. This option prompts you to enter the schema name and generates the report, fndtrep6.txt.

Example

```
Enter the Schema name[%]: <HR>
```

Option 2: Enable the Constraints, Triggers and Policies, and Start Advanced Queues

Select Option 2 to enable all the constraints, triggers, policies and start queues, and to generate the log file, fndenabl<timestamp>.log is generated. This option prompts you to enter the schema name.

Example

```
Enter the Schema name[%]: <HR>
```

Option 3: Re-size Old Tablespaces

Select Option 3 to reduce the size of the old tablespaces. This option queries the data dictionary for all data files of the previous tablespaces to determine the level at which they can be re-sized, and generates the resize commands in a script resizdb.sql. This script is then executed to resize the data files.

Option 4: Generate Script to Drop Empty Tablespaces

Select Option 4 to generate a script to drop empty tablespaces. This option queries the data dictionary for all the previous tablespaces to determine it they still contain segments. For all the old tablespaces with no remaining segments a drop tablespace ... including contents and datafiles statement is generated in a script fndtsdrp.sql. When executed, this script drops all empty pre-OATM tablespaces including data files.

> **Note:** Please ensure that there is no residual data in the tablespaces to be dropped prior to running the drop tablespace script to avoid

irrevocable loss of data.

Step 7: Run Customization Steps

Select Step 7: Run Customization Steps if you wish to customize tablespaces, tablespace types, and object classifications as required. If you choose run customization steps you should do so before performing any of the other steps for migrating your database objects to OATM. The Run Customization Steps menu contains the following options:

1. Register new tablespace - tablespace type
2. Change name of the existing tablespace
3. Register object classification
4. Change object classification

Option 1: Register new tablespace - tablespace type

Select Option 1 to register any custom tablespace types that are not available by default with OATM. If the tablespace type or tablespace name is already registered, message will be displayed stating that it already exists

Example

```
Enter the tablespace type: CUSTOM_TABLESPACE_TYPE
Enter the tablespace name: CUSTOM_TABLESPACE

Tablespace CUSTOM_TABLESPACE registered.

Do you want to continue registering tablespaces ?[Y]:
```

Selecting the default, "Y" prompts you to enter next tablespace type/name pair. Selecting "N" returns you to the previous menu.

Option 2: Change name of the existing tablespace

Select Option 2: Change name of the existing tablespace to update the tablespace name of any default OATM tablespaces or registered custom tablespaces.

Example

```
Enter the tablespace type: CUSTOM_TABLESPACE_TYPE
Enter the new tablespace name: CUSTOM_TBLSP

Tablespace name for CUSTOM_TABLESPACE_TYPE changed to CUSTOM_TBLSP.

Do you want to continue changing tablespace names?[Y]:
```

Selecting the default, "Y" prompts you to enter next existing/new tablespace pair. Selecting "N" returns you to the previous menu.

Option 3: Register object classification

Select Option 3: Register object classification to register new object-tablespace

classifications. This is relevant only for objects such as tables that require explicit classification. These can include custom tables residing in Oracle Application product schemas or those residing in custom schemas. If object classification for the object is already registered, a message will be displayed stating that classification already exists, and will prompt you to enter a new object name. Existing classification can be modified by selecting option 4 from the Run Customization Steps menu.

Example

```
Enter the application short name: FND
Enter the object name: FND_TABLES
Enter the tablespace type: CUSTOM_TABLESPACE_TYPE

Tablespace type CUSTOM_TABLESPACE_TYPE for object FND_TABLES registered.

Do you want to continue registering tablespace types for other
objects?[Y]:
```

Selecting the default, "Y" prompts you to register the next object classification and selecting "N" returns you to the previous menu.

Option 4: Change object classification

Select Option 4: Change object classification to change any existing object-tablespace classifications. If object classification for the object is not already registered, a message will be displayed stating that classification does not exist, and you will be prompted to enter a new object name. New object classification can be entered by selecting Option 3 from the Run Customization Steps menu.

Example

```
Enter the application short name: FND
Enter the object name: FND_LOBS
Enter the tablespace type: MEDIA

Tablespace type for object FND_LOBS changed to MEDIA.

Do you want to continue changing tablespace types for other objects?[Y]:
```

Selecting the default, "Y" prompts you to enter the next object classification modification and selecting "N" returns you to the previous menu.

Step 8: Batch Mode Execution of Migration

> **Note:** Please note that Step 8: Run Migration in Batch Mode is an alternative step to the earlier migration steps. If you have followed OATM menu steps 1-7, you don't need to perform step 8.

Select Step 8: Run Migration in Batch Mode to execute both preparatory steps and migration commands as a single flow. The OATM menu-based design provides you fine-grained control and maximum flexibility over how the objects will be migrated. If you need to use the OATM utility in a repeated fashion, Step 8: Run Migration in Batch Mode gives you a greater level of automated control.

If you select OATM Batch Mode, the OATM migration utility will first collect all the

required information and then perform all necessary checks (including checks for the existence of all required scripts, invalid indexes, database log mode, and so on). If errors are encountered during the check stage, you are prompted to review the logs and correct any issues. Any errors during background execution will be recorded in the status table and the process will be terminated. Once errors are corrected, you can proceed with batch-mode-migration by restarting the process.

Option 1: Run Migration in Batch Mode

Menu option 8, "Run Migration in Batch Mode" has two submenus, one for the invocation of the batch-mode-migration and the other for monitoring the progress of batch mode migration. In batch mode, OATM migration utility will perform the following: a check for invalid indexes, sizing, a check for the disk space usage, a check for relevant system parameters, creation of new tablespaces, generation of migration commands, and execution of migration commands. Based on the value of the AUTO_START_MIGRATION parameter, migration will be either started automatically after preparatory steps are completed (the default); or the process will wait for your input before proceeding with migration execution (in case you want to perform some checks, back up the database, and so on). Each step will be started automatically once the previous step is finished and will be running in the background.

Option 2: Run Migration Monitor

The overall migration status report covers all OATM process phases, that is, the creation of tablespaces, generation of migration commands, and execution of migration commands. For all migration steps, status details such as the parameters passed and the name of the log files are reported, as well as the migration progress based on the total number of the migration commands, number of successfully executed commands and number of commands that failed during execution.

4

System Administration Setup Tasks

Setting Up Oracle Applications System Administrator

This section contains an overview of each task you need to complete before you can use Oracle Applications products.

Setup Checklist

After you log on to Oracle System Administrator, complete the following steps to set up your Oracle Applications:

Setup Steps

Create Accounts for Implementors to Complete Setting Up

Create individual Oracle Applications accounts for users who will be completing the implementation of your Oracle Applications. Assign these users the full access responsibilities for the products they will be implementing.

> **Note:** Updates by the SYSADMIN user are treated as seed data when applying patches.

See: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*.

Create New Responsibilities (Optional)

A responsibility in Oracle Applications is a level of authority that determines how much of an application's functionality a user can use, what requests and concurrent programs the user can run, and which applications' data those requests and concurrent programs can access. Oracle Applications provides a set of predefined responsibilities that you can use. You can also define your own responsibilities if the ones provided do not meet your needs.

See: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*.

Create Additional Users

You should use the procedure outlined in Step 1 to create additional application users. When you define a new user, you assign one or more responsibilities and a password that the user changes after the initial logon. You can use the LOV in the Responsibility field to get a list of the standard responsibilities for each application you specify. You can assign multiple responsibilities to a user.

See: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*.

Set Up Oracle Applications Manager

Oracle Applications Manager (OAM) allows you to configure and maintain many components of the Oracle Applications system. For information on setting up OAM, see: Oracle Applications Manager Setup, page 5-2.

Set Up Your Printers

Read the Setting Up Your Printers, page 9-8 page to learn how to set up your printers. You must define any printer types used at your site that are not shipped with Oracle

Applications, then register each printer with its name as determined by your operating system.

For every custom printer type or specialized print style you define, use the Printer Drivers form to assign a printer driver to use with each print style used by a printer type.

If you need more information on how to find your printer operating system names, refer to the Printing section of *Installing Oracle Applications*.

For more information on setting up your printers, see: Printers and Printing, page 9-1.

Specify Your Site-level and Application-level Profile Options

Use the System Profile Values form (Profile > System) to set site-level and other profile optons..

Optionally set your Site Name profile option to your site name.

Many profile options are set by AutoConfig and their values can be reviewed in Oracle Applications Manager. For more information, see *Oracle Applications Maintenance Utilities*.

Define Your Concurrent Managers (Optional)

Concurrent Processing is a feature of Oracle Applications that lets you perform multiple tasks simultaneously. Oracle Applications Concurrent Processing lets you run long, data-dependent functions at the same time as your users perform online operations. Concurrent managers are components of concurrent processing that monitor and run your time-consuming tasks without tying up your computers.

Oracle Applications automatically installs one standard concurrent manager that can run every request. You may want to take advantage of the flexibility of concurrent managers to control throughput on your system.

You can define as many concurrent managers as you need. Keep in mind, however, that each concurrent manager consumes additional memory.

You can specialize each of your concurrent managers so that they run all requests, requests submitted by a particular user, requests submitted by a particular application, or other constraints, or any combination of these constraints.

If you are using Parallel Concurrent Processing in a cluster, massively parallel, or homogeneous networked environment, you should register your Nodes and then assign your concurrent managers to primary and secondary nodes. You can spread your concurrent managers, and therefore your concurrent processing, across all available nodes to fully utilize hardware resources.

Use the Define Concurrent Manager form to define new concurrent managers.

Define Request Sets (Optional)

A request set is a group of reports or programs which you submit with one request. To

define and maintain request sets, use the Request Sets form.

Users can also define their own request sets.

Set Up AuditTrail (Optional)

If you want to keep track of the changes made to your data by application users, you should set up AuditTrail for the relevant tables.

Defining AuditTrail for your site involves defining Audit Groups, which are groups of tables and columns for which you intend to track changes. You then define Audit Installations to instruct AuditTrail which ORACLE IDs you want to audit. Finally, you run the Audit Trail Update Tables Report, which allows your AuditTrail definitions to take effect.

Define Internationalization Options (Optional)

Optionally define settings for internationalization features.

Modify Language Prompts (Optional)

If you want to modify the field name displayed in the Translations window, you should change the Description value for the language you want to modify in the Languages window.

Modify Territory LOV Values (Optional)

If you want to modify the territory value displayed in LOVs, you should change the Description value for the territory you want to modify in the Territories window.

Specify Preferences for Oracle Workflow Notifications (Required)

The SYSADMIN user is the default recipient for some types of notifications in Oracle Applications, such as error notifications. You need to specify how you want to receive these notifications by defining the notification preference and e-mail address for the SYSADMIN user.

By default, the SYSADMIN user has a notification preference to receive e-mail notifications. To enable Oracle Workflow to send e-mail to this user, navigate to the Users window and assign SYSADMIN an e-mail address that is fully qualified with a valid domain. However, if you want to access notifications only through the Oracle Workflow Worklist Web page, then you should change the notification preference for SYSADMIN to "Do not send me mail" in the Preferences page. In this case you do not need to define an e-mail address.

5

Introduction to Oracle Applications Manager

Introduction to Oracle Applications Manager

Oracle Applications Manager (OAM) allows administrators to manage Oracle E-Business Suite systems from an HTML console. Utilities available from OAM include Oracle Workflow Manager, Patch Wizard, and Concurrent Processing monitoring tools.

With Oracle Applications Manager, system administrators can view information on general system activity including the statuses of the database, concurrent managers and other services, concurrent requests, and Oracle Workflow processes. OAM provides a summary of configuration changes, infrastructure usage, performance, required maintenance activities, potential security issues, status of business flows, and diagnostic test results. In addition, they can manage downtime and patching. System administrators can also start or stop services, and submit concurrent requests.

Using Oracle Workflow Manager, administrators can control Workflow system services, such background engines, the Notification Mailer, agent listeners, queue propagation, and purging obsolete Workflow data.

OAM utilities are generally available from two main screens: the Applications Dashboard and Site Map. See: Applications Dashboard, *Oracle Applications System Administrator's Guide - Maintenance*and The Site Map, page 5-4 for more information on these.

Refer to the following documentation for additional information:

- *Oracle Applications System Administrator's Guide - Maintenance*
- *Oracle Workflow Administrator's Guide*
- *Oracle Applications Maintenance Procedures*
- *Oracle Applications Maintenance Utilities*
- *Oracle Applications Patching Procedures*

The Service Fulfillment Manager may optionally be installed. For more information, see: http://www.oracle.com/appsnet/products/service/index.htm.

Function Security and Oracle Applications Manager

Oracle Applications Manager uses with Oracle Application Object Library's function security model. You can create custom responsibilities and menus to control access to specific OAM features. These features can thus be directly available from the E-Business Suite Home Page.

Oracle Applications Manager Setup

Oracle Applications Manager (OAM) allows you to customize how certain components are monitored and how metrics are collected.

Navigation: Setup (global icon)

The Dashboard Setup page displays a summary of data collection for metrics and services. For each metric, you can see whether collection is enabled and whether you have alerting enabled for the metric. If alerting is enabled, you see the condition which must be met for an alert to be raised. To update the setup for the data collection, use the Dashboard Setup Wizard.

MetaLink Credentials

You maintain your Oracle*MetaLink* username, password, and e-mail address from the Metalink Credentials page. These credentials will be used when querying Oracle *MetaLink* through Knowledge Base links. Also, you should ensure that the following profile options are set appropriately for your proxy server:

- Applications Proxy Bypass Domains
- Applications Proxy Port
- Applications Server-Side Proxy Host and Domain

Business Flows Setup

Enable or disable monitoring of business flows.

Click the Metrics Refresh link to schedule requests for the OAM: KBF Metrics Rollup Program to update the setup status of your business flows

Knowledge Base

The Knowledge Base provides a catalog of useful documents relevant to managing your system.

Concurrent Requests

From this page you can enable alerting for concurrent requests that have been running or pending for a long time. You can specify the thresholds for which request must reach before alerts are raised.

Specifically, you can enable the system to do the following:

- Raise a general alert for any long-running requests, for any concurrent program.
- Raise an alert for a long-running request for a specific program. Specify the concurrent program(s) you want to monitor.
- Raise a general alert for any long-pending requests, for any concurrent program.
- Raise an alert for a long-pending request for a specific program. Specify the concurrent program(s) you want to monitor.

Signon Audit Setup

From this page, select the Enable Auditing button to set the Sign-On: Audit Level profile option to 'FORM', which enables Forms monitoring. Selecting the Disable Auditing button sets this profile option to 'NONE'.

Use the Enable Alerting/Disable Alerting buttons to control whether an alert should be raised if the Sign-On: Audit Level profile option is set at a value other than 'FORM'.

Dashboard Setup Wizard

Select the Update button to configure how data for the following are collected.

Metrics

Specify how you want metrics to be collected for the following:

- Activity
- Configuration Changes (last 24 hours)
- System Alerts
- Web Components Status
- User-Initiated Alerts

Services

Specify which services you want to monitor. For a given service, you can specify if you

want to collect data for it and whether you want to be alerted if the service is in a specified status.

The Site Map

The Site Map lists the features and applications available in the Oracle Applications Manager. Features are grouped into the following categories: Administration, Monitoring, Maintenance, and Diagnostics and Repair.

Site Map - Administration

System Configuration

These features provide detailed information on the configuration of your system. You can update many of your configuration settings from these links also.

- Hosts - For each of your hosts, you can view its status and configuration. You can also bring it online or offline, or disable it.
- AutoConfig - View and update your AutoConfig settings here. For more information on this feature, see *Oracle Applications Maintenance Utilities*.
- License Manager - With the License Manager you can license additional products, country-specific functionalities and languages. You can also generate reports on the licenses for your installation. For more information on this feature, see *Oracle Applications Maintenance Utilities*.

Application Services

Use these links to see information on various types of application services.

- Generic Services
- Request Processing Managers
- Transaction Managers

Workflow

Use these links to navigate to the Oracle Workflow Manager.

- Home
- Work Item Metrics
- Agent Activity
- Background Engines

- Notification Mailer
- Service Components
- Purge

Concurrent Request

Use these links to submit a new concurrent request or to view details on existing requests.

- Submit New - this link launches the Oracle Application Framework Schedule Request page in a separate window.
- Pending
- Running
- Completed (Last Hour)

Service Fulfillment Manager

Use this link to access the Service Fulfillment Manager. Service Fulfillment Manager (SFM) provides a complete set of tools to automate step-by-step fulfillment activities and integrate business flows for any type of service across multi-vendor application systems.

Others

- Applications Manager Log

Site Map - Monitoring

The Monitoring section links you to features to help you monitor your Oracle E-Business Suite.

Availability

Use these links to navigate to pages on the availability of these components.

- Hosts
- Database
- Web components
- Internal Concurrent Manager
- Request Processing Managers

- Transaction Managers
- Forms
- Workflow
- Business Flows

Performance

Use these links to see information regarding performance of these components.

- SQL Activity
- Forms Sessions
- Forms Runtime Processes
- Concurrent Processing Reports
- Concurrent Processing Charts
- Workflow

Current Activity

Use these links to view activity information for their respective areas.

- System Alerts
- Database Sessions
- Invalid Objects
- Forms Runaway Processes
- Forms Sessions
- Forms Runtime Processes
- Application Services
- Activity Monitors
- Concurrent Requests
- Critical Activities
- Logs

Configuration Changes

From the Overview link you can navigate to the Configuration Overview page.

Usage

Use these reports to learn more about application usage and concurrent processing.

Custom Reporting Utilities

The SQL Extensions page enables you to run seeded and custom scripts.

Site Map - Maintenance

These features help you maintain your Oracle Applications installation.

Note that most of these features are described in detail in the *Oracle Applications Maintenance Procedures* and the *Oracle Applications Maintenance Utilities* guides.

Patches and Utilities

- Applied Patches
- File History
- Patch Wizard
- Timing Reports
- Manage Downtime Schedules

Cloning

- Clones Status
- Simple Clone
- Advanced Clone

Purging

- Setup
- Monitor

Site Map - Diagnostics and Repair

Use these features in diagnostics and troubleshooting.

Diagnostics

- Test Statistics - links to the Diagnostics tab of the OAM Dashboard.
- Launch Tests - launches Oracle Diagnostics in a separate window.

Troubleshooting Wizards

- Concurrent Manager Recovery
- Service Infrastructure
- GCS and Forms Monitoring
- CP Signature
- Dashboard Collection

Configuration Overview

This page contains configuration information for the Applications system's configuration.

Overview

This page provides information on the following:

- Database Configuration
- Operating Units
- Registered Oracle Schemas
- Registered Applications
- Base and Installed Languages
- Localization Modules

Database

The Database section lists the database instances for the system, with this information:

- Host Name
- Instance Name

- Version
- Instance Number

Click on **NLS Parameters** to see a list of these parameters and their values. Click on **Initialization Parameters** to see a listing of these parameters.

Concurrent Processing

This section lists general information related to concurrent processing for this Applications system and concurrent processing settings.

A list of concurrent processing servers with their host names and platforms is shown. Also, the following site-level values for the Internal Concurrent Manager (ICM) are shown:

- Concurrent:GSM Enabled - This profile option indicates whether the Generic Service Management (GSM) feature is enabled. The default value is Yes.
- Concurrent: Attach URL - If this profile option is set to Yes, a URL is attached to request completion notifications. When a user submits a request, and specifies people to be notified in the Defining Completion Options region, everyone specified is sent a notification when the request completes. If this profile option is set to Yes, a URL is appended to the notification that enables them to view the request results online.
- Concurrent: Sequential Requests - You can force your requests to run one at a time (sequentially) according to the requests' start dates and times, or allow them to run concurrently when their programs are compatible. A value of Yes prevents your requests from running concurrently. Requests run sequentially in the order they are submitted. A No value means your requests can run concurrently when their concurrent programs are compatible.
- Sleep Time - The duration of time in seconds that the ICM should wait before checking for new requests. PMON Cycle Time - The duration of time in seconds between "process monitor checks" (checks for failed workers). Queue Sizing Interval - The duration of time in seconds between "worker quantity checks" (checks for the number of active workers).

Click on **ICM Environment** to see the environment variables and their values.

Forms

This section shows the ICX: Forms Launcher profile option setting, which should be set to the base URL for launching Oracle Applications forms.

A List of Forms Servers is also shown, with the Host Name, Port, and Log File location for each Oracle Forms server.

Web

This section shows the following profile option settings:

- Applications Web Agent - Provides the base URL for the Apps Schema's WebServer DAD. Oracle Applications use the value of this profile option to construct URLs for 'WWW' type functions, Attachments, Export, and other features.

- Applications Servlet Agent - This profile option must be set to the URL base for the servlet execution engine on Apache. Oracle Applications uses the value of this profile option to construct URLs for JSP and SERVLET type functions.

  The syntax is: http://<hostname>:<port>/<servlet_zone>

- Application Framework Agent - This profile option must point to the Apache Server.

A List of Web Servers is also shown, with the Host Name, Port, and Log File location for each web server.

Other

This section shows the following profile option settings:

- TCF:HOST and TCF Port - These profile options identify the network location of the TCF Server. The TCF Server supports various parts of the Oracle Applications user interface by executing some of their associated server logic and providing access to the database.

- ICX: Report Launcher, ICX: Report Server, ICX: Report Link - These profile options are used by the Oracle Business Intelligence System (BIS) reports. For more information on these, see the Oracle Business Intelligence System Implementation Guide.

6

Defining Concurrent Programs and Requests

Overview of Concurrent Programs and Requests

A concurrent program is an executable file that runs simultaneously with other concurrent programs and with online operations, fully utilizing your hardware capacity. Typically, a concurrent program is a long-running, data-intensive task, such as posting a journal or generating a report.

Standard Request Submission (SRS) is an Oracle Applications feature that allows you to select and run your concurrent programs from a single, standard form (*Submit Request*) or window (*Schedule Request*). Requests to run concurrent programs are called concurrent requests.

There are two main ways to group concurrent programs. Request sets are defined to run several concurrent program in a single request. Request groups are used to control access to concurrent programs via responsibilities. Both request sets and request groups are discussed in later sections

Limiting Active Requests by User

As System Administrator you can limit the number of requests that may be active (status of Running) for an individual user. This ensures that a user cannot monopolize the request queue. For example, if a user with an Active Request Limit of 5 submits 20 requests, only 5 requests will be run at the same time. The remaining requests will be run when the number of active requests for the user drops below 5. Use the Profile Options window to set the *Concurrent: Active Request Limit* profile. To set a global limit for all users, set this option at the site level. You can then modify limits for individual users by setting this profile option at the User level.

Related Topics

Organizing Programs into Request Sets, page 6-5

System Administrator Request Set Privileges, page 6-18

Organizing Programs into Request Groups, page 6-22

Copying and Modifying Program Definitions, page 6-43

Running Reports and Programs, *Oracle Applications User's Guide*

Controlling Access to Concurrent Programs

You can restrict users' access to submit and view concurrent requests.

Controlling Access to Concurrent Programs with Request Security Groups

Note: This method is used in releases prior to Release 12.

A *request security group* is a collection of reports or concurrent programs. A System Administrator defines request security groups in order to control user access to reports and concurrent programs. Only a System Administrator can create a request security group.

- The reports and concurrent programs that may be selected by a user in Standard Request Submission belong to a *request security group*, which is a request group assigned to a responsibility.

  Important: The use of request security groups is for backward compatibility only.

- The reports and concurrent programs that may be selected from a customized SRS form or window belong to a request security group that uses a code.

  See: Customizing the Submit Request Window using Codes, page 6-26.

Controlling Access to Concurrent Programs using Role-Based Access Control (RBAC)

RBAC allows administrators to have more granular control in securing data related on concurrent programs and requests.

Submitting Requests

Administrators can assign individual programs/sets, all programs/sets in a request group, programs/sets belonging to one or more applications, and so on, either to the

user directly or to a role that can then be assigned to one or more users. For more information on RBAC, see: Overview of Access Control with Oracle User Management, *Oracle Applications System Administrator's Guide - Security*.

If applications are included in the request groups, all programs/requests sets that are created in these applications will also be automatically included. Please note that request submission applies to both programs and request sets.

The following types of "instance sets" can be used for assignment (but administrators can create new instance sets based on their needs):

- All programs in a particular request security group
- All request sets in a particular request security group

To enable this functionality, the following are seeded:

- Permission "Submit Request"
- Permission "View Request"
- Permission Set "Request Operations" containing the permissions "Submit Request" and "View Request"
- Object "Concurrent Programs"
- Object Instance Set "Programs that can be accessed"
- Object Instance Set "Request sets that can be accessed"

To grant access to a request security group to a role, follow these steps:

1. Define your role (User Management responsibility).
2. Define your request security group (System Administrator responsibility).
3. Define your grant (Functional Administrator responsibility).
    1. Enter a Name and Description (optional) for the grant.
    2. Enter the Security Context for the grant.
    3. Under Data Security, choose "Concurrent Programs" or "Request Sets" as the object. Click Next.
    4. For the Object Data Context, select "Instance Set" for the Data Context Type. Choose either "Programs that can be accessed" or Request Sets that can be accessed" as appropriate. Click Next.
    5. Review the Instance Set information. Under Instance Set Details, enter the

request group and its application. Specifically, enter the request group name as Parameter 1 and the application short name as Parameter 2.

6. Choose "Request Operations" as the permission set under "Set". Click Next.

7. Review the grant information and save your work.

Note that there are two seeded grants for all users to account for request group assignments that already exist for legacy responsibilities. These are:

- Programs - Grant Defaults
- Request Sets - Grant Defaults

Viewing Requests

You can control users' access to viewing requests with RBAC.

> **Note:** In previous releases, the Concurrent: Report Access Level profile was used to control privileges to report output files and log files generated by a concurrent program. This profile is no longer used.

Seeded "instance sets" allow administrators to grant:

- All requests submitted by a user
- All requests submitted by a user for a given application
- All requests belonging to a program submitted by a user
- All requests belonging to a request set submitted by a user (irrespective of the constituent programs' owning application)

to another user (or a group of users - via a role).

System administrators can create new "instance sets" based on their needs. They can grant access to requests (of a particular program/set) submitted by all users to a specific set of users. For example, say a given application's administrators group want to track all requests of a particular type or program submitted by business users. Then the following approach, to grant specific programs' requests to a group of users, can be used:

1. Create an instance set that selects all the requests belonging to the particular program irrespective of which user submitted it.

   For example,

```
&TABLE_ALIAS.request_id in
    ( select cr.request_id
      from fnd_concurrent_requests cr, fnd_concurrent_programs cp
      where cr.concurrent_program_id = cp.concurrent_program_id
      and cr.program_application_id = cp.application_id
      and cp.concurrent_program_name = &GRANT_ALIAS.PARAMETER1)
```

If you want to grant access to a set of programs instead of a single program, '&GRANT_ALIAS.PARAMETER1' can be replaced with a subquery that returns all the programs in a particular request group.

2. Create a grant to grant this instance set to (an existing) role, for example, "<Application> Administrator" role, and assign the program name to grant. Use the "Concurrent Requests" data object in creating this grant.

3. Ensure that the role is assigned to all users that should have access to these requests.

Organizing Programs into Request Sets

Reports and concurrent programs can be assembled into request sets.

Request sets define run and print options, and possibly, parameter values, for a collection of reports or concurrent program. End users, with the appropriate privileges, and System Administrators can define request sets. A System Administrator has request set privileges beyond those of an end user. Request sets can be run from Forms-based applications and HTML-based applications.

Request sets are a quick and convenient way to run several reports and concurrent programs with predefined print options and parameter values. Request sets group requests into stages that are submitted by the set. The order in which the stages are submitted is determined by the status of previous stages.

Request sets can also be used by a System Administrator to customize access to reports and concurrent programs. Using request sets, a System Administrator can:

- grant users of a responsibility the ability to run selected reports and concurrent programs that are outside their request security group.
- grant access to requests and other concurrent programs on a user-by-user basis.
- guarantee that reports in the set run with print options and parameter values that cannot be edited by end users.

Note: Multilingual requests cannot be run within request sets.

As System Administrator, you have privileges beyond those of your application users, including a privileged version of the Request Set window.

Defining Request Sets

You can run the same set of concurrent requests regularly by defining a request set, and then submitting the request set from the Submit Requests form.

As System Administrator, you can include *any* Standard Request Submission report or concurrent program in the request sets you define. When end users define a request set, they can only select from reports and programs that belong to their responsibility's request security group.

Use the Request Set form to create and edit request sets.

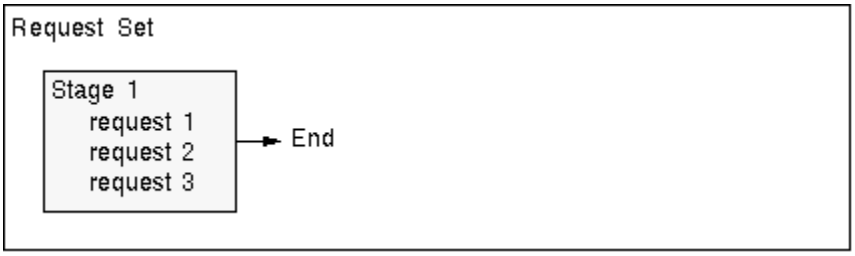
Request Set Stages

This section describes how request set stages are defined and organized.

Organizing Request Sets into Stages

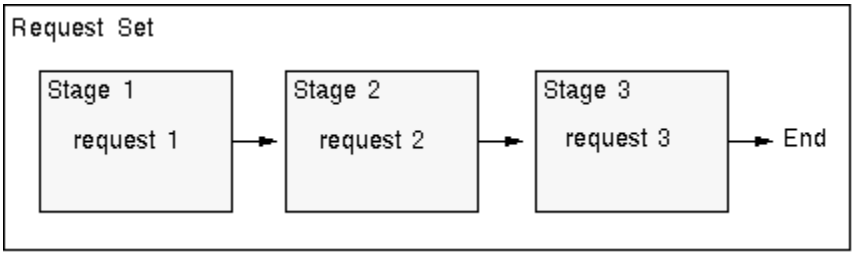
Request sets are divided into one or more "stages" which are linked to determine the sequence in which requests are run. Each stage consists of one or more requests that you want to run in parallel (at the same time in any order). For example, in the simplest request set structure, all requests are assigned to a single stage. This allows all of the requests to run in parallel.

Request Set with a Single Stage



To run requests in sequence, you assign requests to different stages, and then link the stages in the order you want the requests to run.

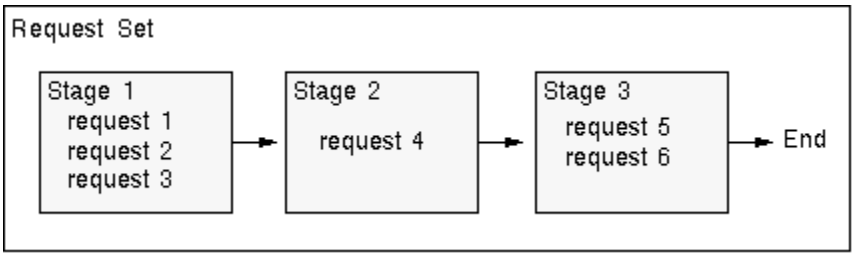
Request Set with a Sequence of Stages



The concurrent manager allows only one stage in a request set to run at a time. When one stage is complete, the following stage is submitted. A stage is not considered to be complete until all of the requests in the stage are complete.

One advantage of using stages is the ability to run several requests in parallel and then move sequentially to the next stage. This allows for a more versatile and efficient request set.

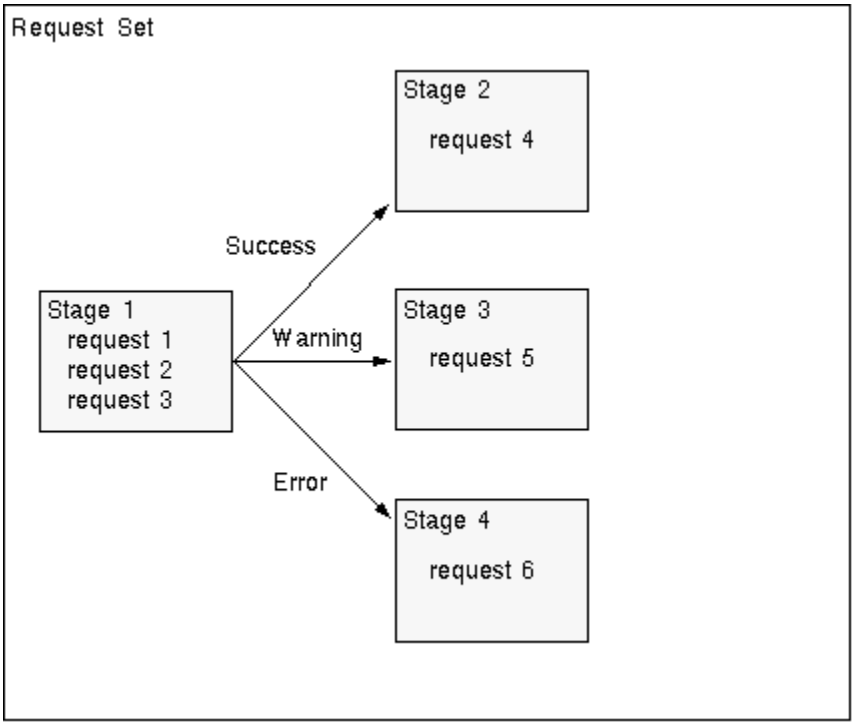
Request Set with Multiple Requests Running in Parallel within a Stage



Using Stage Status

Like request sets and concurrent requests, stages can complete with different statuses. Each stage can complete with a status of Success, Warning, or Error. You can use these completion statuses to structure your request set, by defining which stage will follow the current stage based on its completion status. For example: a request set always begins with Stage 1. If Stage 1 completes with the status Success, then the Success link is followed, and Stage 2 is submitted. After Stage 2 completes, the set ends. If Stage 1 completes with Warning, then the Warning link is followed, and Stage 3 is submitted. After Stage 3 completes, the set ends. If Stage 1 completes with Error, then the Error link is followed, and Stage 4 is submitted. After Stage 4 completes, the request set ends.

Request Set Using Stage Statuses

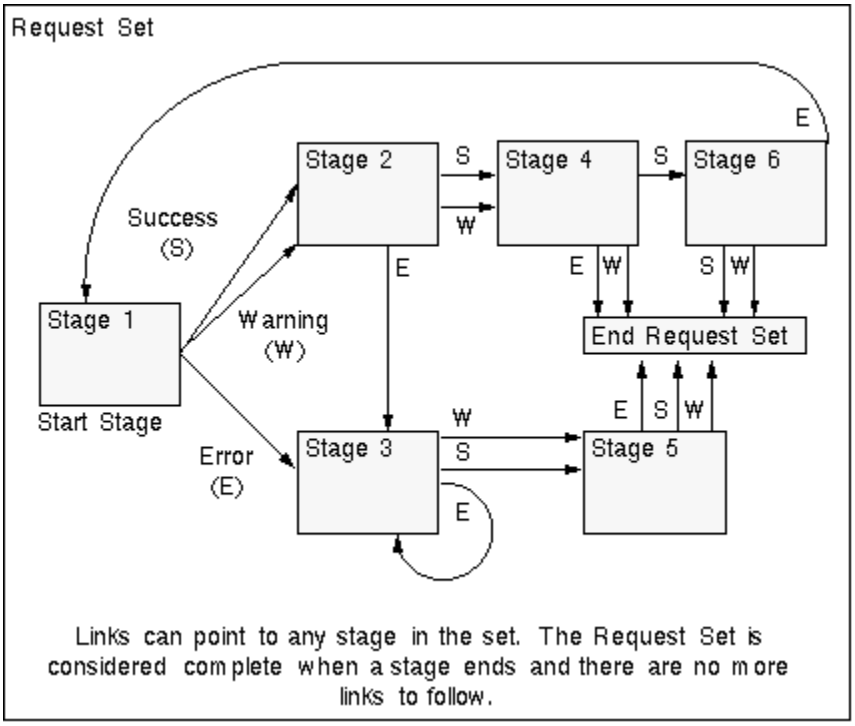


In this example, the stage status is determined using the Standard stage function. The Standard stage function uses the statuses of the requests within the stage to calculate the status for the stage. If all of the requests in a stage complete with a status of Success, then the status for the stage is Success. If one or more requests complete with a status of Error, then the status of the stage is Error. For a stage's status to be Warning, one or more of the requests must have a status of Warning, and no request may have a status of Error.

Linking of Stages

There are no restrictions on linking stages within a request set. Any stage may be linked to any other stage, including itself. Two or more links can point to the same stage. For example, Stage 1 can link to Stage 2 if the completion status of Stage 1 is Success or Warning, and link to Stage 3 if the status is Error.

Request Set with Multiple Links to Stages



You determine the end of a request set by not specifying a followup stage for each completion status. You can end a request set after any stage in the request set. When any stage completes with a status that does not link to another stage, the request set ends.

You can use the linking of stages to control your request set. In previous releases you had three options: run in parallel, run sequentially, and run sequentially but abort on Error. All of these are easy to recreate using the request set wizard. You can use the Request Set Wizard button in the Request Set window to start the wizard. The wizard takes your input and creates the request set as follows:

Run in Parallel — Creates one stage containing all of the requests you wish to run in parallel.

Run Sequentially — Creates a separate stage containing the request or requests for each step in the sequence and link in the appropriate order.

Run Sequentially but abort on Error — Sets up your sequence the same as it did for Run Sequentially, but when it links the stages, it does not enter a follow up stage as a link in the Error completion status field.

Stage Evaluation Function

The completion status of a stage is determined by a predefined function. The Oracle Applications Standard Stage Evaluation function uses the completion status of the requests it contains. Use this function to determine the status of a stage.

Request Set Completion Status

When a stage completes with a status for which there is no link defined, the request set ends. The completion status for the request set is determined by one of the following methods:

- Using the completion status of the last stage run in the request set. This method is used by default.
- The user can override the default behavior by defining a specific stage within the set to be "critical". If the request set has a single critical stage defined, and then runs this critical stage, then the completion status of the set will be the same as the completion status of the critical stage. This can be useful if the final stage of the set is a "clean up" stage and is not considered important to the overall status of the set.
- If a request set has more than one critical stages defined, the "worst" completion status of all the critical stages is considered the completion status of the set.

Printing Request Sets

On a report-by-report basis, you can select a different printer for each report in a request set. When you define a request set, print options, such as the printer a report is sent to, are saved so you do not have to specify them again when you run the request set.

> **Important:** If a printer is defined for a concurrent program using the Concurrent Programs form, then that value cannot be updated, either by a user profile option setting, a request set definition, or when running the program or request set.

> **Note:** Defining a printer for a request set concurrent program (e.g., Request Set Payables Aging Reports) in the Concurrent Programs form has no effect; the printer definition is not referred to.

Holding Request Sets

In some circumstances, such as when a request set has a large number of stages and takes a long time to execute, administrators may want to yield a request set to higher priority requests. By utilizing the *Hold Request Set* feature, users can place a running

request set on hold and effectively control the execution of request set stages.

The **Hold** and **Remove Hold** buttons are available on the OAM View Running Requests page. To hold a request set, simply select the request set and click the **Hold** button. Click **Remove Hold** when you want the request set to continue executing.

Request Sets as Concurrent Programs

When you define a request set or a stage within a request set that allows incompatibilities, a concurrent program is created to run the requests in your request set according to the instructions you enter.

All concurrent programs that run request sets are titled *Request Set <name of request set>*, and programs that run request set stages are titled *Request Set Stage <name of request set stage>*. In the Concurrent Programs form, to query request set or request set stage concurrent programs on the basis of a program's name, enter the following in the Name field:

- "Request Set" or "Request Set Stage" before the name of the concurrent program
- "Request Set %" or "Request Set Stage %" to perform a query on all request set programs

Request set and request set stage concurrent programs create log files documenting the execution of the request set or stage. Each report or concurrent program within a request set or stage also creates its own log file.

When you run a request set that allows incompatibilities, you submit a request to run the concurrent program that defines the request set. The request set concurrent program submits a request set stage concurrent program. The request set stage concurrent program submits the requests for the individual programs and reports within the stage. A request to run the request set concurrent program or the request set stage concurrent program is a *Parent* request, while the requests to run the programs and reports are *Child* requests.

You can review the status of a request set and the programs it contains using the Concurrent Requests form. The following table displays information pertaining to request sets in the Running phase.

Status	Description
Paused	Parent request pauses for all its Child requests to complete. For example, a request set stage pauses for all reports in the stage to complete.
Resuming	All requests submitted by the same Parent request have completed running. The Parent request resumes running.

Modifying Request Sets

A request set can only be modified by its owner or by a System Administrator. To make modifications, query the request set you want to modify in the Request Set window.

> **Note:** If you wish to retain modifications to request sets provided by your Oracle application during upgrades, you must rename or recreate the request set using a different name before you upgrade. If you modify a predefined request set without changing the name, your modifications are overwritten when you upgrade your Oracle Applications.

Creating Request Sets

Follow this procedure to create a request set:

1. Navigate to the Request Set window.
2. Enter a Name for your request set.
3. Enter a Set Code for your request set. This name is used internally to reference your request set.
4. Enter the Application with which you want to associate your request set.
5. Enter a Description of your request set if you like.
6. The Owner field defaults to your username and can only be changed by your system administrator.
7. Enter the Active Dates From and To fields to define an effective period when you and others can run the request set. If the current date is outside the range you define, the request set will not be available in the Submit Requests window.
8. Check the Print Together check box to send all your requests to the printer together when they complete, or uncheck the check box to send each request one at a time to the printer as it completes.
9. Check the Allow Incompatibility check box to allow your system administrator to specify programs that this request is incompatible with (may not run with). Leave Allow Incompatibility unchecked to specify that this request set may run with all other concurrent requests or request sets.
10. Choose **Define Stages** or **Link Stages** if you have finished defining your stages.

Defining Stages

Follow this procedure to define stages:

1. The value for the Display Sequence is defaulted in sequence as you enter your stages. You may change the display order of the stages by modifying this field.

2. Enter a Name for the stage.

3. Enter a Description of your stage if you like.

4. Enter a Stage Code for the stage. This code is used internally to reference the stage.

5. In the Function field of the Function region, use the List of Values to select a function. The default value for this field is the Standard Stage Evaluation function. This function bases its completion status on the normal completion status of the requests it contains. Other functions may be provided by your Oracle product. For a description of these functions, refer to the user's guide for that product.

6. Use the "The Return Value of this Stage Affects the Set Outcome" check box if you want to ensure that the request set's completion status is equal to the completion status of this stage.

   Note: If you choose this check box for more than one stage, the completion status of the request set will equal the completion status of the last of these stages to run within the set.

7. Use the Allow Incompatibility check box to allow your system administrator to specify programs that this stage is incompatible with (may not run with). Leave Allow Incompatibility unchecked to specify that this stage of the request set may run with all other concurrent requests or request sets.

8. Choose **Requests**.

Stage Requests Window

In the Stage Requests window you define which requests you want to include in the stage.

1. Select the report or program you want to include in your request set. A description of the request you choose and its associated application appears in the Description and Application fields.

   The list of requests you can choose includes the requests that your responsibility's request group lets you access from the Submit Requests form.

2. Use the Allow Stage Function to Use This Program's Results check box to indicate

which programs or reports should be included.

3. The Print Options region reflects the options for the current request. Specify the number of Copies of output to print, the Style to print, the Printer to print to, and whether to save the output to an operating system file.

   Standard Request Submission saves these options so you do not have to specify them again when you run the request set. If you do not wish to specify these options for each request when you define the set, Standard Request Submission uses the values of your personal profile options as the default when you submit the request set. See: Concurrent Processing User Profile Settings, *Oracle Applications System Administrator's Guide - Maintenance*.

   > **Note:** Some requests may have a required Style or Printer that you cannot change.

4. When you are done with the Print Options, choose Parameters to display the Request Parameters window.

Request Parameters Window

The Request Parameters window lets you customize the parameter values of a specific request in a request set. The fields at the top of the Request Parameters window list general information about the current request set and the request for which you can customize the parameter values. The multi-row portion of the window lists the parameters for that request.

1. The Sequence field displays the order in which each request parameter appears when you run the request in the Submit Requests window (lower numbers appear before higher numbers). Only your system administrator can change a parameter's order.

2. The Prompt field is a display-only field that shows the request parameter's prompt.

3. Check the Display check box to specify that you can see a request parameter at submission time, or uncheck the check box to specify that a parameter should not be displayed at submission time.

4. Check the Modify check box to specify that you can insert or change the value for a request parameter at submission time, or uncheck the check box to specify that a parameter cannot be changed at submission time.

5. Use the Shared Parameter field to set a default value for a parameter that occurs in more than one report or program of a request set. Once you enter the same parameter label in the Shared Parameter field for each occurrence of the same parameter, the value that you assign to the first occurrence of the parameter

becomes the default value for all subsequent occurrences of the parameter. The shared parameter label simply enables you to set an initial default value for all occurrences of the same parameter so you can avoid typing the same value all over again for every occurrence of the parameter.

For example, suppose you define a request set that includes three reports, and all reports include a parameter called "Set of Books". You want the "Set of Books" parameter to default to the same value in all reports. To accomplish this, enter a label called "Book" in the Shared Parameter field for the first occurrence of this parameter. You can also assign a value in the Default Value field of this parameter now, or wait until you run the request set to assign a default value when the parameter first appears. Enter the label "Book" in the Shared Parameter field of all other occurrences of the "Set of Books" parameter in your request set. When you submit this request set from the Submit Requests window, every parameter that you label "Book" defaults to the value you assign to the first occurrence of the "Set of Books" parameter.

> **Important:** Note that if you later change the value of a parameter that contains a shared parameter label, you change only the value for that instance of the parameter, and not the value for all other occurrences of that labelled parameter.
>
> We recommend that if you make a parameter with a shared parameter label modifiable, that you also display the parameter so you can always see what the parameter's current value is. This helps reinforce the understanding that a later value change to one labelled parameter cannot propagate a value change to all other similarly labelled parameters.

6. Optionally enter a Default Type and Value for the parameter.

7. Save your work.

8. Go back to the Stage Requests window and repeat Steps 9 through 11 to add more requests to the request set stage.

   You can select a request more than once if you want to run the same request with different default parameter values.

9. To start a new stage, return to the Stage window and choose New Record from the File Menu.

Linking Stages

Follow this procedure to link stages:

1. Enter the Start Stage. The stage you enter here is the first stage submitted for the

request set.

2. Enter the stages you want to run following the first stage in the Success, Warning, and Error columns. To ensure that a particular stage follows the preceeding stage regardless of the completion status, enter the desired stage in all three columns. To stop the request set if a stage ends in Error, leave the Error column blank. Any time you do not specifically indicate which stage should follow for a completion status, the request set will exit on that completion status.

   In the example shown in the table below, the request set will always exit if any stage returns a completion status of error. In addition, stages C and D will terminate the request set regardless of their completion status. If Stage A returns a status other than Error, Stage B will be submitted. Finally, when Stage B completes with a status of Success, it is followed by Stage C, or if the status is Warning, Stage D will follow.

3. Choose **Done.**

The following table shows an example of linking stages as in step 2 above:

Display Sequence	Name	Success	Warning	Error
1	Stage-A	Stage-B	Stage-B	
2	Stage-B	Stage-C	Stage-D	
3	Stage-C			
4	Stage-D			

Restarting Request Sets

If a request set completes with a status of *Error*, the **Restart** button, on the Oracle Applications Manager - View Completed Requests page is enabled. The system also automatically captures, records, and saves the information of the first stage that fails so that when the user clicks on the **Restart** button the request set can restart from that point.

Once the stage has been identified, the request set program submits the stage program in resubmit mode. In this mode, the program looks at the same stage from the previous run and determines which programs need to be rerun, (only those that ended in error), and runs those programs. If this stage completes successfully or has a *Warning* status, the system proceeds to the next stage using the normal mechanism of restarting the request set program.

Note: Users may restart a request set multiple times. The logs for each stage and individual programs are maintained independent of the number of runs as each stage and program submission generates a new request. However, the logs and associated files for a request set are rewritten each time the set is restarted.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Request Sets and Owners

There are significant differences between end user and System Administrator privileges when defining or editing request sets.

End users own the request sets they create

An end user can create a request set by selecting reports, other request sets, or concurrent programs that are part of the report security group assigned to his or her responsibility.

When an end user creates a request set, the user automatically becomes the "owner" of the request set. Ownership is identified by the person's application username.

End users use the Request Set form to create a new request set, or to query and update any request sets they own. End users can only edit request sets they own.

We sometimes refer to a request set that an end user owns as a *private* request set. Private request sets are not automatically added to a request security group. That is, other users cannot access your private request sets using the Submit Requests window unless the System Administrator assigns that request set to a request security group.

Request sets owned by an end user are always available to that user, regardless of what responsibility the user is operating under. However, a standard submission form customized to display only reports in a request group using a code does not display private request sets.

When a user signs on to Oracle Applications, the user can run requests, request sets, and concurrent programs included in:

- their responsibility's request security group
- any request sets they own.

End User Benefits from Private Request Sets

Private request sets offer two main benefits to end users:

1. The request sets that users own are always available to them, regardless of which

responsibility they are working under.

2. Users can create as many request sets as they want without adding request set choices to the list of standard submission concurrent programs that other users must select from.

System Administrator Request Set Privileges

As System Administrator, you can:

- create request sets that include *any* reports or concurrent program.
- query and edit *all* request sets using the Request Set form.
- permit and define incompatibility rules for individual request sets. See: Request Set Incompatibilities, page 6-19.

After you define a request set, you can assign a user to be its owner if you want the user to be able to run or edit this request set from any responsibility. Request sets without an owner cannot be edited or updated by any end users. In this way, you can guarantee print options and report parameters for a request set. You can also later edit the request set to remove or change its ownership properties.

Other users can also run a request set if you, as System Administrator, assign the request set to their responsibility's request security group. If you do not assign a request set to a request security group, then only the owner can run the request set. In this way, you can grant access to reports and concurrent programs on a user-by-user basis.

Request Security Groups, Request Sets, and Reports

As System Administrator you can add any request set, including private request sets, to a request security group. This allows you to provide members of a responsibility access to reports and programs outside their request security group.

Request set editing and report viewing privileges are different for reports that belong to a user's request security group than they are for reports that are not in the user's request security group.

- cannot edit the request set.
- cannot run an individual report by itself, but can only run the entire request set.
- can add any other requests in their request security group to the request set.
- can delete any request from the request set, regardless of whether that report is in their request security group.
- can update print options or parameters for an individual report in the request set, if the report is in their request security group.

- cannot run an individual report by itself, but can only run the entire request set.

System Administrator Benefits from Request Sets

Request sets offer three main benefits to System Administrators:

1. Request sets offer a means of controlling access to concurrent programs on a user-by-user basis.

   By defining a request set, assigning it an owner, and then not assigning the request set to any request security group, the reports and programs in the request set are only available to the owner.

2. By leaving the Owner field blank, System Administrators can create request sets whose individual programs and parameters cannot be edited or updated by end users.

   Only a System Administrator can edit a request set that has no owner.

3. System Administrators can provide members of a responsibility access to reports and programs outside their request security group.

   By defining a request set that contains reports or programs not in a request security group, and assigning that request set to the request security group, users can be granted run, but not edit privileges for selected reports or programs.

Request Set Incompatibilities

A request set is actually a concurrent program that submits requests to run each program in the request set. You can allow incompatibility rules to govern your request set so that the request set does not run at the same time as other reports or concurrent programs. You can also apply these rules to the stages that make up the request set.

Use the Concurrent Programs form to query the request set concurrent program and list those programs, and/or stages you want to define as incompatible with your request set. See: Concurrent Programs, page 6-63.

All concurrent programs that run request sets are titled *Request Set <name of request set>*. In the Concurrent Programs form, if you query a request set concurrent program on the basis of the program's name, you must enter in the Name field the words:

"Request Set" before the name of a concurrent program

"Request Set %" to perform a query on all request set programs

When you list a program as incompatible with your request set, the program will not run simultaneously within the same conflict domain as the request set or any of the reports within the set. See: Defining Program Incompatibility Rules, page 6-27.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Defining Program Incompatibility Rules, page 6-27

Concurrent Programs, page 6-63

Sharing Parameters in a Request Set

Parameters, also referred to as *arguments,* are values that define aspects of a program's execution. You can share a parameter and its entered value among some or all of the requests in your request set.

You identify a parameter as shared by giving it a label. Then, for each concurrent program in your request set, you can assign the same label to a parameter for that program. Among the programs in your request set, the parameters for each program share or accept a common value.

The *first time* you enter a value for any of the shared parameters, that value becomes the shared parameter's value. This is useful, because you only have to enter a value once, rather than for each program in the request set.

Behavior of Shared Parameters

Selecting a value for a shared parameter provides a default for subsequent occurrences of the parameter. Changing a shared parameter's value provides a new default for subsequent occurrences of the parameter, but does not affect prior requests in the set.

Once all the shared parameters contain values, changing the value for a shared parameter has no effect on the other shared parameters.

> **Important:** Do not hide shared parameters. Do not set shared parameters to Display = No (which prevents modifying the value) or Modify = No. This prevents updates to shared parameters, which are not propagated to other reports in the set, from generating unwanted inconsistencies.

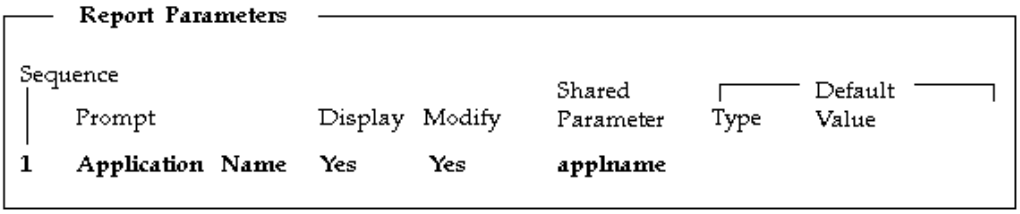
Example - Shared Parameter Value

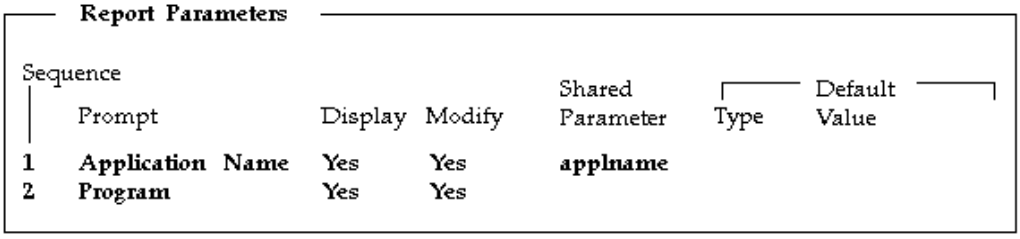
We've created a request set containing two reports, a *Concurrent Programs Report* and the *Concurrent Program Details Report*. The two reports and their parameters are listed in the table below:

REPORT	PARAMETERS
Concurrent Programs Report	Application Name
Concurrent Program Detail Report	Application Name, Program

We identify the parameter Application Name as a parameter shared between the two reports. We want to enter a value only once, that is, when the Report Parameters window appears for the first report in the set, requiring us to enter Application Name.

To identify a shared parameter, we give it a name, in this example, *applname*, and enter it as a Shared Parameter for each report.





Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Control the Behavior of Request Parameters, page 6-47

Concurrent Programs, page 6-63

Request Sets Report

This report documents request set definitions, including the set's owner, program incompatibilities, as well as printer and print style information. Use this report when defining or editing request set definitions.

Report Parameters

None.

Report Headings

The report headings provide you with general information about the contents of the report.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Organizing Programs into Request Sets, page 6-5

Concurrent Programs Report, page 6-56

Organizing Programs into Request Groups

This essay explains how you can organize your applications programs and reports into request groups. It presents the following topics:

- Request Security Groups
- Using Codes with Request Groups
- Customizing the Submit Requests Window using Codes
- Report Group Responsibilities report

Defining a Request Group

When defining a request group, you can include:

- all the reports and concurrent programs owned by an application
- individual reports and concurrent programs
- request sets, which are collections of reports and concurrent programs that may be selected from an application user's request security groups
- request set stage functions, which are used to calculate the status of stages within a request set.

Two Types of Request Groups

A request group is used by Oracle Applications at two different levels:

1. Responsibility level

   When a request group is assigned to a **responsibility,** it is referred to as a request security group, and it defines the reports, request sets, and concurrent programs

that a user, operating under that responsibility, can select from the Submit Requests Window.

2. Form level

   When a request group is assigned a **code**, that code can be passed as a parameter to the Submit Requests Window. The code helps define the function that calls the Submit Requests Window.

   The list of values for that unique Submit Requests Window lists the reports, request sets, and concurrent programs in the request group.

Request Security Groups

When a request group is assigned to a responsibility, the request group is referred to as a *request security group*. Any user signed on under a responsibility can run the reports and concurrent programs contained in their responsibility's request security group.

The Submit Requests standard submission form displays a list of all the reports and programs in the current responsibility's request security group.

Related Topics

Using Codes with Request Groups, page 6-23

Customizing the Submit Requests Window using Codes, page 6-26

Report Group Responsibilities Report, page 6-27

Request Groups, page 6-58

Using Codes with Request Groups

Normally, when a menu calls the standard request submission form, that form can list the reports and concurrent programs contained in the report security group for the current responsibility.

Alternatively, you can assign a code to a request group so that a customized standard submission form only displays a list of concurrent programs contained in that particular request group. A request group code is simply an argument that is passed from a menu to a customized standard submission form. To summarize:

- Request group codes provide a form-based method of controlling user access to concurrent programs and reports.
- A code can be assigned to a request group.
- You can use the code as an argument passed from a menu to the standard submission form.

- When a menu that calls the standard submission form uses the code, that form lists only those programs in the request group identified by the code.

Related Topics

Organizing Programs into Request Groups, page 6-22

Customizing the Submit Requests Window, page 6-24

Customizing the Submit Requests Window using Codes, page 6-26

Request Groups, page 6-58

Customizing the Submit Requests Window

You can customize the Submit Request window in several ways.

Rename the Window Title

You can change the title to reflect the requests available in the window. See: Customizing the Submit Requests Window using Codes, page 6-26.

Restrict Requests Available to A Request Group

You can restrict the reports and programs available to those in a specified request group. See: Customizing the Submit Requests Window using Codes, page 6-26.

Restrict Requests to a Single Request

You can call Submit Requests form for a single request submission by passing the program/set name as parameters

The parameters window pops up on navigation to the form when called with a program/report_set name. The form exits after the user acknowledges the displayed request ID for the submitted request.

Restrict Requests To A List of Requests

You can call Submit Requests form to submit one or more requests for a single program/set by passing the program/set name as parameters

The parameters window pops up on navigation to the form and the user can submit one or more requests for the program that was passed as a parameter. Requests for other programs cannnot be submitted in this case.

Pass Parameters Used in Value Set Parameters

You can pass additional parameters to the Submit Requests form that can be referenced in the value sets to validate the request parameters.

Pass Manufacturing "ORG" Parameters

You can pass 5 ORG related parameters and refer to them in the value set. Alternatively, you can bring up a ORG LOV on navigation to the Submit Requests form that populates the ORG parameters which can be referenced in the value sets.

Complete List of All Submit Request Paramters

Below is the comprehensive list of parameters supported by the "Run Requests"/SRS form and additional information about their usage.

- REQUEST_GROUP_CODE
- REQUEST_GROUP_APPL_SHORT_NAME (used with REQUEST_GROUP_CODE)
- CONCURRENT_PROGRAM_NAME
- PROGRAM_APPL_SHORT_NAME (used with CONCURRENT_PROGRAM_NAME)
- REQUEST_SET_NAME
- SET_APPL_SHORT_NAME (used with REQUEST_SET_NAME)
- SUBMIT_ONCE (default 'N').

  SUBMIT_ONCE can be set to either Y or N (N is the default).

  SUBMIT_ONCE is used in conjunction with CONCURRENT_PROGRAM_NAME or REQUEST_SET_NAME.

  If SUBMIT_ONCE is set to Y, then the form will exit after the Submit button is clicked.
- TITLE
- LOOKUP (default 'N')
- USE_ORG, ORG_ID, ORG_NAME, ORG_CODE, CHART_OF_ACCOUNTS_ID (five parameters)

  If USE_ORG is set to 'Y' (default is 'N') then the Submit Requests form checks to see if the other ORG parameters are set. If the parameters are not set, then it attempts to populate the parameters from the globals (GLOBAL.FND_ORG_ID, GLOBAL.FND_ORG_NAME, etc.). If the globals have not yet been set, the an ORG LOV shows, and both the parameters and the globals are populated from the LOV.

  Values sets should always reference the parameters, not the globals.
- CHAR1, CHAR2, CHAR3, CHAR4, CHAR5

- DATE1, DATE2, DATE3, DATE4, DATE5
- NUMBER1, NUMBER2, NUMBER3, NUMBER4, NUMBER5

  In your value sets, refer to these parameters as:

  ```
  :PARAMETER.CHAR1, :PARAMETER.DATE1, :PARAMETER.NUMBER1  etc.
  ```

Customizing the Submit Requests Window using Codes

You can give the Submit Requests Window a different title, and define the form so that it allows users to select only those reports or concurrent programs belonging to a request group that you have assigned a code to. To do this, you register a form function that references the Submit Requests Window, and you pass certain arguments to the function. Then you construct your menu to include this form function. For more information, see: the *Oracle Applications System Administrator's Guide: Security*.

Using a Request Group Code as an argument

The following table describes the parameters passed to associate a request group with the Submit Requests Window and to customize the title of that form. Text is entered in the Parameters field of the Form Functions form.

Parameter Syntax followed by Example	Explanation
REQUEST_GROUP_CODE ="*Request Group Code*" REQUEST_GROUP_CODE = "OE_CONC_PROGRAMS"	This parameter passes the request group's code. (Required)
REQUEST_GROUP_APPL_SHORT_NAME = "*Application short name*" REQUEST_GROUP_APPL_SHORT_NAME = "OE"	This parameter identifies the short name for the application associated with the request group. (Required)
TITLE ="*Application_short_name:Message_Name* " TITLE = "FND:SRS_NEWTITLE"	This parameter identifies a message whose contents define the title, as well as the application short name of that message. (Optional)
LOOKUP = "Y|N" LOOKUP = "Y"	This parameter indicates whether the TITLE parameter is a message name or a hardcoded string. The default value is "Y", which indicates that TITLE is a message name. (Optional)

Related Topics

Customizing the Submit Requests Window, page 6-24

Organizing Programs into Request Groups, page 6-22

Using Codes with Request Groups, page 6-23

Report Group Responsibilities Report, page 6-27

Request Groups, page 6-58

Report Group Responsibilities Report

This report lists those responsibilities which have access to a report or a request set. Use this report when granting access privileges to reports and request sets, either by assigning reports and request sets to request security groups, or when assigning owners to a request set.

Report Parameters

Application Name

Choose the application name associated with the report or request set.

Report Name/Request Set Name

Either choose the name of a report or request set.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Organizing Programs into Request Groups, page 6-22

Request Groups, page 6-58

Defining Program Incompatibility Rules

This essay explains how you can define incompatibility rules for your concurrent programs and reports.

Incompatible and Run Alone Programs

When a concurrent program is incompatible with another program, the two programs cannot access or update the same data simultaneously.

When you define a concurrent program, you can list those programs you want it to be incompatible with. You can also list the program as incompatible with itself, which

means that two instances of the program cannot run simultaneously.

You can also make a program incompatible with *all other* concurrent programs by defining the program to be run-alone.

There are two types of program incompatibilities, "Global" incompatibilities, and "Domain-specific" incompatibilities.

You can define a concurrent program to be globally incompatible with another program -- that is, the two programs cannot be run simultaneously at all; or you can define a concurrent program to be incompatible with another program in a Conflict Domain. Conflict domains are abstract representations of groups of data. They can correspond to other group identifiers, such as sets of books, or they can be arbitrary.

You define a concurrent program to be run-alone or to be incompatible with specific concurrent programs by editing the concurrent program's definition using the Concurrent Programs window. See: Concurrent Programs, page 6-63.

Program incompatibility and run-alone program definitions are enforced by the Conflict Resolution Manager (CRM).

> **Note:** The concept of "Global" incompatibilities was introduced with Patch 2364876.
>
> With this patch, all pre-existing incompatibilities are converted to the type Global, unless both of the programs have a conflict domain parameter registered. This may mean that if you have been using the Concurrent:Conflicts Domain profile option for your custom programs, you may need to switch the incompatibility type to "Domain-specific" to keep the expected behavior.
>
> Also, the two user-level constraints, set by the Concurrent:Active Request Limit and Concurrent:Sequential Requests profile options, are now enforced globally across all conflict domains.

Request Sets - Incompatibilities Allowed

When you define a request set or request set stage that allows incompatabilities, you create a concurrent program that runs the reports in your request set or stage according to the instructions you entered. Using the Concurrent Programs window, when you list programs as incompatible with a request set, those programs are prevented from starting until all the reports in the set or stage have completed running.

To define incompatibility rules for a request set and request set stage:

- For a request set check the Allow Incompatibility check box on the Request Set window.
- For a request set stage check the Allow Incompatibility check box on the Stages window.

- Navigate to the Incompatible Programs block in the Concurrent Programs form and list those programs that your request set or stage is incompatible with.

All concurrent programs that run request sets are titled *Request Set <name of request set>* while all concurrent programs that run request set stages are titled *Request Set Stage <name of stage>-Request Set <name of request set>*. In the Concurrent Programs form, if you query a request set or stage concurrent program on the basis of the program's name, you must enter in the Name field the words:

- "Request Set" or "Request Set Stage" before the name of a concurrent program
- "Request Set %" to perform a query on all request set and stage programs

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Request Set Incompatibilities, page 6-19

Modifying an Incompatible Programs list, page 6-46

Data Groups, page 6-77

Concurrent Programs, page 6-63

Concurrent Conflict Domains

If two programs are defined as incompatible with one another, the data these programs cannot access simultaneously must also be identified.

In other words, to prevent two programs from concurrently accessing or updating the same data, you have to know *where*, in terms of data, they are incompatible. A Conflict Domain identifies the data where two incompatible programs cannot run simultaneously.

Conflict Domains

In Oracle Applications, data is stored in database tables that belong to a particular application. Each table may also contain information used to determine what conditions need to be met to access the individual records. These conditions may consist of one or more of the following data groupings:

- SOB - based on the profile option GL_SET_OF_BOOKS
- Multiple installations (referred to as MSOB)
- Multiple Operating units (determined by profile option MO_OPERATING_UNIT) (referred as MULTIORG).
- Multiple Orgs (determined by profile option INV_ORGANIZATION_ID, Used by

Manufacturing Applications)

- HR may use business group as a conflict resolution domain
- FA may use FA book
- etc...

A conflict domain is an abstract representation of the groupings used to partition your data. There is no limit to the number of domains that can be defined, but excessive domains may hurt performance.

All programs are assigned a conflict domain when they are submitted. If a domain is defined as part of a parameter the concurrent manager will use it to resolve incompatibilities. If the domain is not defined by a parameter the concurrent manager uses the value defined for the profile option Concurrent:Conflicts Domain. Lastly, if the domain is not provided by a program parameter and the Concurrent:Conflicts Domain profile option has not been defined the 'Standard' domain is used. The Standard domain is the default for all requests.

All programs use the Standard conflict domain unless a value is defined for the profile option Concurrent:Conflicts Domain or a conflict domain is defined through a program parameter.

Each request submitted uses parameters which identify the records that it will access. For programs that are defined with incompatability rules an additional parameter (conflict domain parameter) is used. The conflict domain may be set automatically based on such variables as a login ID, set of books, or the organization the user is working in. The conflict domain parameter may in some cases be selected in the parameters field of the Submit Requests form. Once the parameter is determined the Conflict Resolution Manager (CRM) uses the domain to ensure that incompatible programs do not run simultaneously in the same domain.

Enforcing Incompatibility Rules

Concurrent managers read requests to start concurrent programs running. The Conflict Resolution Manager checks concurrent program definitions for incompatibility rules.

If a program is identified as Run Alone, then the Conflict Resolution Manager prevents the concurrent managers from starting other programs in the same conflict domain.

When a program lists other programs as being incompatible with it, the Conflict Resolution Manager prevents the program from starting until any incompatible programs in the same domain have completed running.

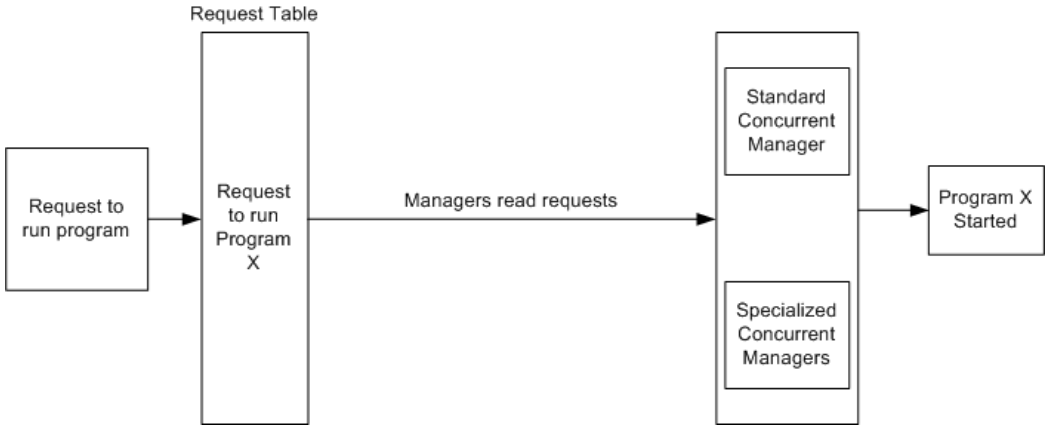
The figures below illustrate the role of the Conflict Resolution Manager when enforcing program incompatibility rules.

In a simple example without incompatibilities, a user submits a request to run a program. This request is then added to the request table which contains a list of

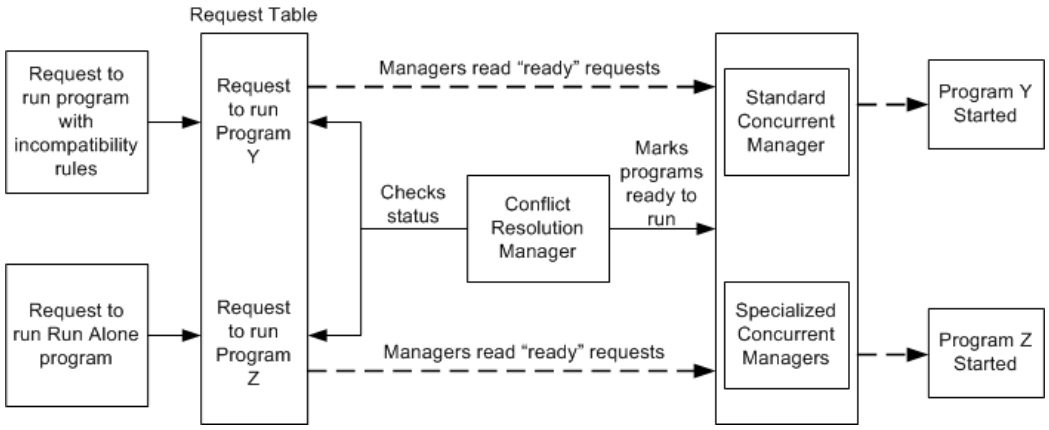
requests. Managers then read requests from this table and start the associated concurrent programs.

A more complex example users may have submitted one request with incompatibility rules and another request to run a program that must be run alone. In this case these requests are added to the request table, but the Conflict Resolution Manager then checks the statuses of the requests in the table and marks which requests are ready to be run. The concurrent managers then read only the "ready" requests and start their concurrent programs.

Simple Run Program Request



Complex Run Program Request



Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Copying and Modifying Program Definitions, page 6-43

Modifying an Incompatible Programs list, page 6-46

Concurrent Programs, page 6-63

Custom Concurrent Programs

This section provides information for system administrators on custom concurrent programs. It explains certain procedures and conventions for creating customized concurrent programs:

- Log and Output Filenames
- Oracle Tool Concurrent Programs
- Pro*C Concurrent Programs
- Submitting Concurrent Requests (CONCSUB)

For information on creating custom concurrent programs, see the *Oracle Applications Developer's Guide*.

For information on setting up the development environment , see the *Oracle Applications Concepts Guide*.

Log and Output Filenames

Log and output files must have specific names and locations for users to review the files online.

If you use the Oracle Application Object Library routine fdpwrt() to write to files, the concurrent managers automatically name the files according to the operating system's naming conventions. This method of writing to files is completely portable. You do not have to rewrite your programs to name your log and output files differently if you port your application to another platform.

Standard Names

Standard names for log and output files are listed in the following table:

File Type	Location	Filename
Log	*Default:* $<PROD>_TOP/$APPLLOG with Common Directory: $APPLCSF/$APPLLOG	l<request ID>.req

File Type	Location	Filename
Output	*Default:* $<PROD>_TOP/$APPLOUT with Common Directory: $APPLCSF/$APPLOUT	*Default:* <USERNAME>.<request ID> or O<request ID>.out *or* user.out based on value of APPCPNAM

The variable parameters shown in this table have the following values:

- <PROD>_TOP - The application's top environment variable.
- <Request ID> - The number that identifies the concurrent request.
- <USERNAME> - Up to eight characters (uppercase) of the application username of the person who requested the concurrent process.

Oracle Tool Concurrent Programs

If you write concurrent programs in PL/SQL, SQL*Plus, or Oracle Reports, name the program exactly as you identified it in the Execution File field of the Concurrent Program Executable window, plus an extension if necessary.

The following table lists the file extensions used for these programs and the directories where the programs should reside. (This does not apply to PL/SQL stored procedures, which are stored in the database.) The directories are under your custom application's TOP directory, $<PROD>_TOP.

If you use shared PL/SQL libraries with your Oracle Reports programs, and you want to include the libraries you write for your custom application, place the libraries in the $APPLPLS directory under your custom application's TOP directory.

Tool	Extension	Directory	Comments
SQL*Plus and PL/SQL	.sql	$APPLSQL	The program name is case sensitive and must exactly match the Execution file you defined with Oracle Application Object Library.

Tool	Extension	Directory	Comments
Oracle Reports	.rdf	$APPLREP	Oracle Application Object Library looks for the .rdf file first. It uses the .rex file if it does not find the .rdf file. The program name is case sensitive and must exactly match the execution file name you defined with Oracle Application Object Library.
SQL*Loader	.ctl	$APPLBIN	

Pro*C Concurrent Programs

When you write a concurrent program in Pro*C, copy the skeleton programs EXMAIN.c and EXPROG.c from the directory $FND_TOP/$APPLUSR. Rename the files and globally replace SUBROUTINE_NAME with the name of your subroutine.

EXMAIN.c is the skeleton used for your spawned programs. EXPROG.c is the skeleton used for your program's logic. This module can be used to create a spawned or an immediate program. For immediate programs, you must include your copy of EXPROG.c in a program library. See below for information on building a program library.

You can use programs written with these skeleton programs as spawned or immediate concurrent programs. Spawned programs run as a separate process while immediate programs run linked in with a concurrent manager.

> **Important:** Oracle provides information on immediate concurrent programs for backwards compatibility only. We strongly recommend that you do not create any new immediate concurrent programs. You should define your new Pro*C concurrent program executables as spawned.

Naming Your Executable File

Name your program's executable file exactly as you identified it in the Execution File field of the Concurrent Program Executable window. Put your executable file in the $APPLBIN directory under your application's TOP directory.

Building Your Program Library

Register a new program library with the Register Concurrent Program Library form and register all the programs you want to include in this library. Then enter "Yes" in the Rebuild field and commit. This creates a request to build a new catalog file called <Library Name>.c under $<PROD>_TOP/$APPLLIB$. You should compile the <Library Name>.c file after the request completes.

Sample program libraries such as prgcat.c and prglib.c are located under $FND_TOP/$APPLUSR.

> **Tip:** For ease of maintenance, define your concurrent program executables as spawned.

Compiling C and Pro*C Programs

Your environment for compiling custom code depends on the file $FND_TOP/usrxit/devenv. If you change this file, you should reread it by logging in again so that the changes take effect.

You compile your C or Pro*C programs into object modules using $FND_TOP/usrxit/Makefile. You then link your programs using adrelink. We do not support both compiling and linking executables using a single makefile or utility.

To compile the C program example.c, use the following syntax. In all the examples, you should run the commands from the directory in which your files are located.

```
$ make -f $FND_TOP/usrxit/Makefile example.o
```

To compile the Pro*C program proexamp.pc, use the following syntax:

```
$ make -f $FND_TOP/usrxit/Makefile proexamp.o
```

To compile the four C and Pro*C programs a.c, b.c, c.pc, d.pc (all of which are in the current directory), use the following syntax:

```
$ make -f $FND_TOP/usrxit/Makefile a.o b.o c.o d.o
```

Linking Spawned Concurrent Programs as Stand-alone Programs

If you want your spawned concurrent program to run as a stand-alone program, perform the following steps before compiling your stand-alone executable.

For custom concurrent programs you define under your custom application (as recommended), you should copy the sample.mk file from $FND_TOP/usrxit to your $<PROD>_TOP/$APPLLIB directory. Modify your copy according to the instructions contained in the file. This is the file adrelink uses to link your stand-alone executables.

Then enter the following commands.

```
$ . $FND_TOP/fndenv
```

Move to the directory in which your source files are kept.

```
$ cd <source_directory>
$ make -f $FND_TOP/$APPLLIB/Makefile <source file>.o
```

Here, <source file> is the name of the file containing your program and <directory> is the directory where the source file is located.

You can then link your stand-alone executable and place the executable in the $APPLBIN directory under the TOP directory for your custom application:

```
$ adrelink force=y "<appl_short_name> <program name>"
```

In this relink command, <appl_short_name> is the application short name of the application your program belongs to, and <program name> is the program name.

Linking your Immediate Concurrent Program

To create a program library, you link your compiled library catalog with your program object files using an Oracle Application Object Library link procedure.

> **Important:** Oracle provides information on immediate concurrent programs for backwards compatibility only. We strongly recommend that you do not create any new immediate concurrent programs. You should define your new Pro*C concurrent program executables as spawned.

Make sure the environment variable $LUSRLIB includes the modules that define the functions for the immediate concurrent programs and your program library. Set the $LUSRPRG variable to include the object modules of your library catalog. The file devenv in the directory $FND_TOP/$APPLUSR defines the variables $LUSRLIB and $LUSRPRG. The file fndenv executes devenv.

The files devenv and fndenv are UNIX shell scripts that set up the necessary environment variables.

We recommend that you make a copy of the working program library before linking your new immediate concurrent program library in case your new program library does not function as expected. To link your program library, execute this command from the operating system:

```
$ adrelink force=y "fnd UFNDLIBR"
```

This creates your new program library as UFNDLIBR. You can rename it, but the name of your new program library must be eight characters or less.

Testing Pro*C Concurrent Programs

You can use the following method to test your program. You must pass each argument needed by your program. To pass parameters, enter the following at the operating system prompt:

```
$ <program name> <ORACLE username>/<ORACLE password> 0 Y \
[<parameter 1> <parameter 2>... ]
```

Instead of the Oracle username and password, you can use an Oracle Applications username and password, if the corresponding user has the System Administrator responsibility.

The program name must be uppercase and the same name that you entered in the Execution File field of the Concurrent Program Executable window. The 0 and Y arguments are required.

If any of your program-specific parameters includes spaces, enclose that parameter in double quotes. If a parameter contains a literal double quote, precede that mark with a backslash [\].

Host Language Concurrent Programs

Name your program <name>.prog, where <name> is the value you enter in the Execution File field of the Concurrent Executable window. Then make a symbolic link using your execution file name (without an extension) to fndcpesr, which is located in the $FND_TOP/$APPLBIN directory. Put your executable file and the linked file in the $APPLBIN directory under your application's TOP directory.

For example, name your custom shell script CUSTOM.prog. Create a symbolic link to fndcpesr named CUSTOM. Place both files in your $APPLBIN directory. Create your concurrent program executable using the execution file CUSTOM.

Host Program Parameters

The concurrent manager running your program puts your program name in $0, the four arguments orauser/pwd, userid, username, and request_id in $1 to $4, and your program specific parameters in $5 and beyond. Each of these arguments can be at most 50 characters.

For example, if you pass two parameters into your program, you use $5 to refer to the first parameter and $6 to refer to the second parameter.

Protecting Your Oracle User Password

In some cases, there are security concerns with passing your Oracle username and password directly to your HOST program. If you do not want the concurrent manager to pass your username/password to your program, you can have the manager pass it as an environment variable instead. Or you can pass an Oracle Applications username/password for a user with the System Administrator responsibility. Alternatively, you can not pass it at all.

First, define your concurrent program executable as a HOST program in the Concurrent Program Executable form.

To have the username/password passed as an environment variable, enter the term 'ENCRYPT' in the Execution Options field of the Concurrent Programs window when defining a concurrent program using this executable. 'ENCRYPT' signals the concurrent manager to pass the username/password in the environment variable fcp_login. The

argument $1 is left blank.

If you do not want the username/password passed to the program at all, enter 'SECURE' in the Execution Options field. The concurrent manager will not pass the username/password to the program.

Success Codes

By default, a shell script returns success (status code 0). If your script traps an error, use the UNIX exit command "exit 1" to return failure (status code 1) to the concurrent manager running the program.

Log and Out Files

Use names in FCP_LOG and FCP_OUT. This way log and output/report files can be viewed online.

Testing Your Program

You should test using the <name>.prog file to make sure your script behaves correctly.

Submitting Concurrent Requests (CONCSUB)

You can test your concurrent program by submitting the program using the CONCSUB utility from the operating system.

Syntax

You can submit a concurrent request to run any concurrent program by running the CONCSUB program with the following syntax:

```
$ CONCSUB <APPS username>/<APPS password> \
<responsibility application short name> \
<responsibility name> \
<username> \
[WAIT=N|Y|<n seconds>] \
CONCURRENT \
<program application short name> \
<program name> \
[PROGRAM_NAME="<description>"] \
[REPEAT_TIME=<resubmission time>] \
[REPEAT_INTERVAL= <number>] \
[REPEAT_INTERVAL_UNIT=< resubmission unit>] \
[REPEAT_INTERVAL_TYPE=< resubmission type>] \
[REPEAT_END=<resubmission end date and time>] \
[NLS_LANGUAGE=<language of the request>] \
[NLS_TERRITORY=<territory of the request>] \
[START=<date>] \
[IMPLICIT=< type of concurrent request> \
[<parameter 1> ... <parameter n>]
```

For parameters that follow the CONCURRENT parameter and include spaces, enclose the parameter argument in double quotes, then again in single quotes. Oracle Application Object Library requires this syntax because it parses the argument string

twice. For example, to pass this argument to a program:

```
This is an example
```

pass this argument through CONCSUB:

```
'"This is an example"'
```

Example

Here is an example of the command to run CONCSUB:

```
$ CONCSUB APPS/APPS \
SYSADMIN \
"System Administrator" \
SYSADMIN \
WAIT=N \
CONCURRENT \
FND \
FNDFMRTC \
PROGRAM_NAME='"Register Custom Tables Weekly"' \
REPEAT_INTERVAL=7 \
REPEAT_INTERVAL_UNIT="DAYS" \
REPEAT_INTERVAL_TYPE="START" \
START='"08-JUN-96 23:55:00"'
CGL
APPLSYS
ALL
CGL
```

Parameters

The following entries explain the required and optional parameters for submitting a concurrent program with CONCSUB. Default values are listed to the right.

username/ password
Required. The ORACLE username and password that provides access to the data that your program uses. Alternatively, an Oracle Applications username and password for a user with the System Administrator responsibility.

responsibility application short name
Required. The application short name of the responsibility whose concurrent processing options you want to use.

responsibility name
Required. The name of your responsibility. If the name of your responsibility includes spaces, enclose that name in double quotes.

username
Required. The uppercase username of the application user whose concurrent processing options you want to use.

WAIT
Optional. A flag that indicates whether to wait for the submitted request to complete. If you leave this parameter out, the default value of N makes CONCSUB return you to

the operating system prompt without waiting for your request to complete.

Set WAIT=Y to have CONCSUB check the request status every 60 seconds and return you to the operating system prompt when your request is completed. You can also enter an integer value for a number of seconds, as in WAIT=30, for CONCSUB to check for request completion every <number> seconds.

> **Important:** Using WAIT=Y or WAIT=<number> requires that your request completes before CONCSUB returns you to the operating system. If the concurrent manager is down, your CONCSUB process waits indefinitely until the concurrent manager is started and the request completes.

CONCURRENT
Required. A flag that separates the program-specific parameters from the operating system parameters.

program application short name
Required. The application short name of your concurrent program.

program name
Required. The uppercase name of your program. It must be the short name that you enter in the Concurrent Programs window when defining a concurrent program.

PROGRAM_ NAME
Optional. A descriptive name for your program. The program field on the View Requests form displays this as the user-friendly program name. The concurrent program short name passed to CONCSUB is often hard for end users to understand, so the PROGRAM_NAME parameter allows you to pass a more easily remembered name for your concurrent program. If you do not specify a PROGRAM_NAME, the View Requests form displays the user-friendly program name specified in the Concurrent Programs window.

You may also use the PROGRAM_NAME parameter to indicate the batch that your request processes for programs that process a set of data, where there could be several requests for a given program that are active at the same time.

REPEAT_TIME

Optional. The time of day to resubmit the request. The format for the time is HH24:MI or HH24:MI:SS. For example, REPEAT_TIME=14:30 resubmits your request daily at 2:30 p.m.

> **Important:** Do not use REPEAT_TIME with other resubmission parameters except for the optional parameters REPEAT_END and START.

REPEAT_ INTERVAL

Optional. The interval between resubmission (a positive integer or real number). Use this parameter along with REPEAT_INTERVAL_UNIT to specify the time between resubmissions.

REPEAT_ INTERVAL_ UNIT

Optional. The unit of time used for the interval between resubmissions. The available units are MINUTES, HOURS, DAYS or MONTHS. Use this parameter along with REPEAT_INTERVAL to specify the time between resubmissions. For example, setting REPEAT_INTERVAL=12 and REPEAT_INTERVAL_UNIT=HOURS resubmits your request every twelve hours. The default value is DAYS.

> **Important:** Do not use REPEAT_INTERVAL and REPEAT_INTERVAL_UNIT with REPEAT_TIME.

REPEAT_ INTERVAL_ TYPE

Optional. Whether to time the resubmission interval from the requested start time of the request or from its completion. Set this parameter either to START or END. The default value is START.

> **Important:** Use REPEAT_INTERVAL_TYPE only if you use REPEAT_INTERVAL.

REPEAT_END

Optional. The date and time to stop resubmitting the concurrent request. Use one of the following for the format of the end date:

'"DD-MON-RR HH24:MI:SS"' (as in '"07-APR-02 18:32:05"')

or

	'"DD-MON-RRRR HH24:MI:SS"' (as in '"07-APR-2002 18:32:05"') Note that because this date format includes a space, you must enclose the date in double quotation marks and single quotation marks. You can also specify just the date: 'DD-MON-RR' or 'DD-MON-RRRR'
NLS_LANGUAGE	Optional. The NLS language for the request.
NLS_TERRITORY	Optional. The NLS territory for the request.
START	Optional. A start date and time for your program in this format: '"DD-MON-RR HH24:MI:SS"' (as in '"07-APR-02 18:32:05"') Because this date format includes a space, you must enclose the date in double quotation marks and single quotation marks. If you do not specify a start time, your program submits immediately and is processed by the next available concurrent manager. The default value is the current time.
IMPLICIT	Optional. Whether to show this concurrent request on the View Requests form. Specify NO, YES, ERROR or WARNING. The value IMPLICIT=NO allows the request to appear on the View Request form. The default value is NO. The value IMPLICIT=YES means that only the System Administrator's privileged View Concurrent Requests form displays this request. Use this value if the request is not interesting to the user. Specify IMPLICIT=ERROR or IMPLICIT=WARNING, respectively, if you want the request to appear only if it fails or completes with warnings.
REPEAT_DAYS	Optional. The number of days after which to repeat the concurrent request, calculated from the last requested start date. The number can be a positive integer or real number. For example, REPEAT_DAYS=1.5 resubmits your request every 36 hours. **Important:** Do not use REPEAT_DAYS

with other resubmission parameters except for the optional parameters REPEAT_END and START.

Tip: REPEAT_DAYS will become obsolete in a future release. You may therefore want to use REPEAT_INTERVAL, REPEAT_INTERVAL_TYPE and REPEAT_INTERVAL_UNIT instead of REPEAT_DAYS.

parameter 1 ... parameter *n* Optional. Your program-specific parameters. If a parameter includes spaces, enclose that parameter in double quotes, then in single quotes. If a parameter contains a double quotation mark as part of the argument, precede that mark with a backslash [\].

Copying and Modifying Program Definitions

These sections explain how you can copy and modify concurrent program definitions.

Warning: Do not overwrite program definitions for existing concurrent programs. Copy the program, rename it, then make any desired modifications to the new program.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Copying and Renaming a concurrent program, page 6-44

Defining Program Incompatibility Rules, page 6-27

Alter Program Priority, page 6-45

Modifying an Incompatible Programs list, page 6-46

Concurrent Program Parameters, page 6-46

Example of modifying a program's parameters, page 6-53

Concurrent Programs, page 6-63

Warnings for Modifying Program Definitions, page 6-51

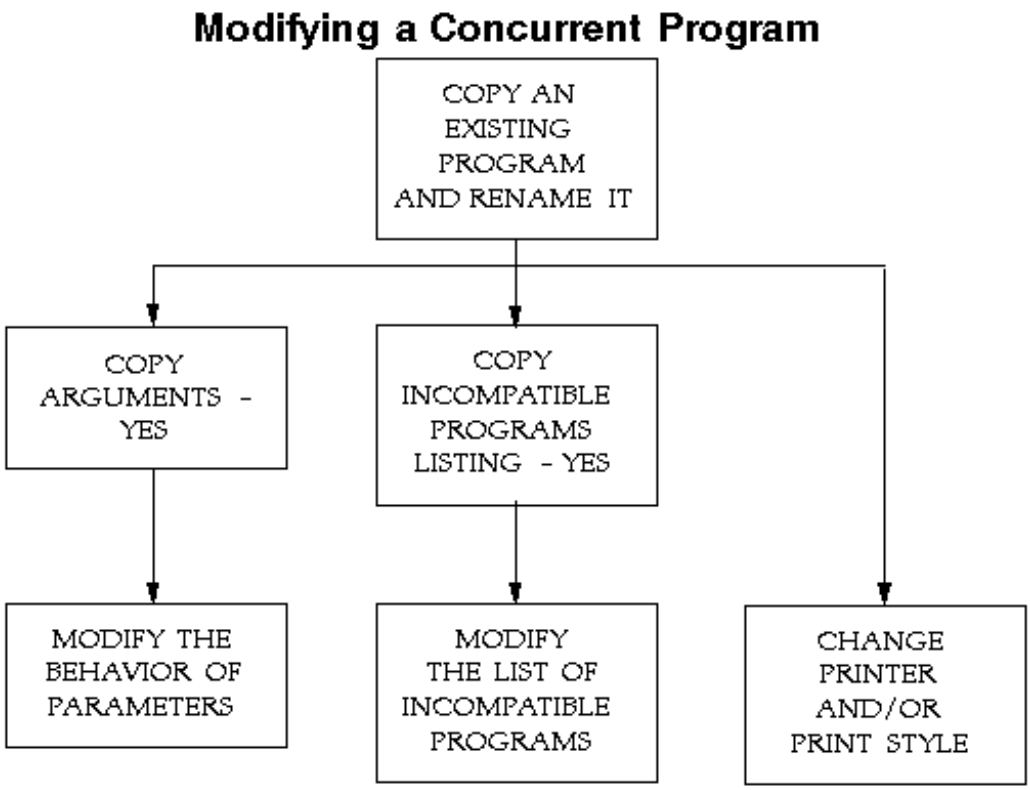
Copying and Renaming a concurrent program

You can copy your concurrent programs and modify them to create new programs with definitions that meet your needs. You can modify how a concurrent program operates by changing the program's definition of:

- incompatible programs
- parameters (arguments)
    - parameter value sets
- printer, print style, etc.

Rather than overwrite a concurrent program's definition, you should customize a program by copying and renaming an existing program, then modifying the new program to suit your needs. The figure below illustrates the basic steps in copying and modifying a new concurrent program.

As the figure illustrates, you can copy parameters, and then modify the behavior of the parameters. Or you can copy the list of incompatible programs, and then modify the list. Finally , you can change the associated printer and/or print style.



Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Copying and Modifying Program Definitions, page 6-43

Example of modifying a program's parameters, page 6-53

Concurrent Programs, page 6-63

Alter Program Priority

You may wish to control the priority of some requests on a program level rather than at the user level.

Setting the priority for a program allows any request to run that concurrent program to use your selected priority rather than the priority of the user submitting the request.

For example, a user can submit a variety of requests at the standard priority determined by the value of the user profile Concurrent:Priority. However, when the user submits a request for a particular concurrent program, you want that request to have a higher priority.

You assign that program a priority of 10. When the user requests that program to run, it receives the higher priority defined on the Concurrent Program window rather than the

user's standard priority and is processed ahead of other requests. When the users requests other concurrent programs that do not have a specified priority, those requests use the user's Concurrent:Priority profile value.

Related Topics

Copying and Modifying Program Definitions, page 6-43

Concurrent Programs, page 6-63

Modifying an Incompatible Programs List

A concurrent program's definition may include a list of incompatible programs. When a program is listed as incompatible with another program, the two programs cannot run simultaneously in the same conflict domain. See: Defining Program Incompatibility Rules, page 6-27.

You can view which programs are incompatible with a concurrent program from the Incompatible Programs block on the Concurrent Programs window. The programs listed cannot run simultaneously within the same conflict domain as the concurrent program whose definition you are viewing.

To modify the list of incompatible programs you can either:

Add new programs to the list.

The *Scope* field refers to whether you want the program by itself to be incompatible, or whether you want the program and all child requests, that is, concurrent programs started by the program as part of a request set, to be incompatible.

- Delete programs from the list.

> **Important:** To immediately effect any changes you make in the Incompatible Programs zone, you must navigate to the Administer Concurrent Managers window and choose *Verify* for the Internal Concurrent Manager.

Related Topics

Copying and Modifying Program Definitions, page 6-43

Concurrent Programs, page 6-63

Administer Concurrent Managers, page 7-49

Concurrent Program Parameters

Parameters, also referred to as *arguments*, are assigned to standard submission concurrent programs. To define a program as standard submission, set the value of the Standard Submission field in the Concurrent Programs form to Yes.

Important: All the mechanisms for parameter defaulting (including references to values of other parameters, user profiles, etc.) are evaluated only at submission time.

There are two aspects to a parameter associated with a concurrent program: its value set and its behavior.

Parameter value set The valid values the parameter can accept. The set of valid values is referred to as a *value set*.

Parameter behavior How the parameter behaves within an application. For example, whether:

- an entry value for the parameter is required in order for the program to work
- the parameter is displayed to the end user
- a default value is automatically provided for the parameter

If you wish to define or modify a value set, you must first carefully plan your value set's purpose and implementation.

Using the Concurrent Programs form, you can see a concurrent program's parameters by choosing *Parameters*. Each parameter has a value set that defines what values are permissible for the parameter. To see the name of a parameter's value set, look at the Value Set field in the Argument Details block.

Related Topics

Copying and Modifying Program Definitions, page 6-43

Control the Behavior of Request Parameters, page 6-47

Example of modifying a program's parameters, page 6-53

Concurrent Programs, page 6-63

Control the Behavior of Request Parameters

The behavior of parameters in programs running individually may differ from when those programs are run as part of a request set.

You define how a program's parameters behave when you define the program using the Concurrent Programs form.

Using the Request Set form, you can also define how a program's parameters behave when the program is run as part of a request. In addition, you can define parameters in different programs in a request set to all share the same value by labeling them as

Shared Parameters. See: Sharing Parameters in a Request Set, page 6-20.

> **Warning:** Modifying a concurrent program's definition by adding new or deleting existing parameters, or changing a parameter's value set can prevent the program from running. See: Warnings for Modifying Program Definitions, page 6-51.

Not Displaying Parameters

Using the Concurrent Programs form or the Request Set form, you can set a parameter so it does not display to an end user. Because parameters that do not display cannot be modified, setting a parameter to not display:

- is a good security measure, guaranteeing a desired default value is used
- means you should enter a valid default type and value at either the program's definition, or if the program is part of a request set, at the request set's definition.

> **Warning:** Set defaults for required parameters before setting the Display field to No. Otherwise the Submit Requests form returns an error when attempting to submit the program.

If you define a parameter to not display, then the parameter does not appear when the program is run using the Submit Requests form, nor does it appear in the Request Set form.

If you define a parameter to not display, using the Request Set form, then the parameter does not appear on the Submit Requests form when the program is run as part of a request set.

Viewing displayed parameters after a request is submitted

After a request is submitted to run a concurrent program, the program's parameters may be displayed in the Details block of the Concurrent Requests form.

When a parameter is set to not display, it does not appear in the Details block of the Concurrent Requests form.

These displayed parameter values exactly match the values that the concurrent manager passes to the concurrent program, and may or may not correspond to the displayed value that the user chose.

For example, in the Submit Requests form, the user may choose "Oracle General Ledger" as a parameter, but the corresponding application ID displays in the Concurrent Requests form.

> **Tip:** If your users encounter errors when running a program, you can

look at the exact values that the concurrent program uses to help you diagnose the problem.

Setting Default Values for Parameters

Parameter default values can be changed by users when they submit a program or request set to run.

You can set a default value for a parameter using the:

- Default Type and Default Value fields in the Concurrent Programs form. These values cannot be changed on the Request Set form.
- Default Type and Default Value fields in the Request Set form.

  This default definition applies only when the program is run as part of a request set.
- Shared Parameter and Default Value fields in the Request Set form

  This default definition applies only when the program is run as part of a request set. All parameters labeled with the *same shared parameter label* default to the value you set in the Default Value field.

Entering erroneous default values

If the Default Type or Default Value for a parameter is incorrect, when the program is being set to run using the Submit Requests form, a window displays along with an error message.

If the parameter is not displayed, you receive an error message. You cannot update a field that is not displayed.

> **Warning:** Be careful when entering the default type and default value, because these values are not validated with the value sets for your parameters. If you enter incorrect values, they do not appear as defaults when you run this request set using the Submit Requests form.

Preventing modification of parameter values in a Request Set

If a parameter is displayed in the Request Set form and there is no default value provided by the program's definition, you can define a default value or have the parameter inherit a shared value, and then prevent end users from modifying that value.

Set the Modify field in the Request Set form to No if you want to show the value for a parameter but not allow changing it when the request set is run using a Standard Submission form. You can set a value for the parameter using a default value or a

shared parameter.

If the Display field is set to No, the Modify field automatically defaults to No, and you cannot update it.

> **Caution:** Set defaults for required parameters before turning Modify to No. Otherwise the Submit Requests form returns an error when attempting to submit this report.

Changing responsibility to see changes take effect

Modifying parameter behavior, for example, changing whether a parameter is displayed to the end user, takes effect immediately after you commit your change. However, some changes do not appear to you unless you change responsibility or select your current responsibility again.

Behavior of Program Parameters

The following table describes how a parameter's details affect its behavior in the Concurrent Programs form and the Run Requests form.

Parameter Details	Concurrent Programs form	Run Requests form
Required	Yes	Parameter requires a value (entered by user or a default).
Display	Yes	Parameter is displayed.
Display	No	Parameter is not displayed, and cannot be modified.
Default Type & Value	Yes - Default Type and Value entered.	A default value displays, and can be changed by the user.
Default Type and Value	No default entered.	No default value is displayed.

The following table describes how a parameter's details affect its behavior in the Request Sets form and Run Requests form.

Parameter Details	Concurrent Programs form	Request Set form	Run Requests form
Required	Yes	Parameter does not require a value.	Parameter requires a value.
Display	Yes	Parameter is displayed. - Display set to Yes.	Parameter is displayed.
Display	Yes	Parameter is displayed. - Display set to No.	Parameter is not displayed.
Display	No	Parameter not displayed.	Parameter not displayed.
Modify	n/a	Yes	Value can be modified.
Modify	n/a	No	Value cannot be modified.
Default Type & Value	Yes - Default Type and Value entered.	Default Type and Value cannot be modified.	Default values can be changed by the user.
Default Type & Value	No default entered.	Yes - a Default Type and Value can be entered.	Default values can be changed by the user.
Default Type & Value	No default entered.	No - Default Type and Value are not entered.	No default value is displayed.

Warnings for Modifying Program Definitions

The following table lists warnings for modifying program definitions:

Action	Form Used	Warning
Changing the number of columns or rows in a report program.	Concurrent Programs - Report Information region.	Some report programs are written to produce a precise output. Changing the output definition could prevent the program from running, or produce poor results.
Setting print style to Dynamic.	Concurrent Programs - Report Information region - Style field.	Dynamic print style informs the program to generate its output based on output dimensions that may vary. Special coding within a program is required to support the Dynamic print style.
Changing the number of parameters in a program definition.	Concurrent Programs - Parameters window.	Programs are defined to expect x number of parameters. If you add a new parameter (x + 1), the program will ignore it. Deleting a parameter can cause a program not to run.
Changing Value Sets.	Concurrent Programs - Argument Details region - Value Set field.	Programs expect values of a certain type and length. Programs may not operate if value set is changed.
Changing tokens.	Concurrent Programs - Argument Details region - Token field.	Programs expect values of a certain type and length. Program may not operate if expected token is not received.
Defining a concurrent executable or program's execution method as Immediate.	Concurrent Program Executables - Execution Method field. Concurrent Programs - Executable Information region - Method field.	Concurrent programs whose execution method is Immediate must be registered with the program library FNDLIBR. Application developers can register programs with program libraries, System Administrators cannot.

Related Topics

Copying and Modifying Program Definitions, page 6-43

Concurrent Program Parameters, page 6-46

Example of modifying a program's parameters, page 6-53

Concurrent Program Details Report, page 6-56

Example of modifying a program's parameters

Consider the following example of when and how to modify a concurrent program's parameters.

If one user submits a large number of concurrent requests on a daily basis, for example, an Oracle Bill of Materials or Oracle Purchasing supervisor, you can create a streamlined purge program that *only* purges that user's concurrent processing records.

You can run this program as System Administrator and have it automatically resubmitted on a specific time interval.

You could also create a request set containing this one program and define the user as the owner of the request set. Then, if you do not assign the request set to any report security group, only the user (owner) can run the program. This way, the user can be responsible for purging their own records.

The System Administrator's Purge Concurrent Request and/or Manager Data program contains twelve parameters. You can copy, rename, and modify the program so it displays only three parameters, with only one parameter requiring user entry. See: Purge Concurrent Request and/or Manager Data, *Oracle Applications System Administrator's Guide - Maintenance*.

The table below summarizes the steps to follow in our example.

Form Used	Task
Concurrent Programs (Concurrent Programs Define)	Query the Application Object Library program named "Purge Concurrent Request and/or Manager Data" and press Copy. Select both Copy Arguments and Copy Incompatible Programs. Enter a new name for the program you are going to copy, for example, enter JSMITH PURGE.
Concurrent Programs	To modify the JSMITH PURGE program's parameters, select the Parameters button.

Form Used	Task
Concurrent Programs, Parameter Window	Modify the following seven parameters so they do not display (user JSMITH cannot see nor change the program's default values). - Oracle ID - Program Application - Program - Manager Application - Manager - Responsibility Application - Responsibility Modify the following three parameters so they do not display (user JSMITH cannot see nor change the default values you set). Set the parameters to the following (Type=Constant) defaults: - Entity = Request - Mode = Age - User Name = JSMITH Leave the following two parameters unchanged so they display. Mode Value will require JSMITH to enter a value, and Report is set to a default value of "Yes". - Mode Value - Report
Request Set (Reports Set)	Create a request set with one program in it, the JSMITH PURGE program. Enter JSMITH in the Owner field. If this request set is not assigned to any report security group, only JSMITH will be able to run the JSMITH PURGE program.
Standard Request Submission program form. For example, the Run Reports form (Reports Run)	When first submitting the JSMITH PURGE program to run, navigate to the Resubmission Options region and enter, for example, "5" and "Days" in the Interval field.

Related Topics

Copying and Modifying Program Definitions, page 6-43

Concurrent Program Parameters, page 6-46

Control the Behavior of Request Parameters, page 6-47

Concurrent Program Details Report, page 6-56

Conflict Domains

A conflict domain is a set of related data stored in one or more ORACLE schemas and linked by grants and synonyms. Do not confuse logical databases with your ORACLE database. The ORACLE database contains all your Oracle Applications data, with each application's data usually residing in one ORACLE schema. You can think of a logical

database as a line drawn around a set of related data for which you wish to define concurrent program incompatibilities. In other words, logical databases determine which concurrent programs cannot run at the same time.

Logical Databases and Program Incompatibilities

When an ORACLE schema is identified as belonging to a logical database, concurrent program incompatibility rules are enforced when concurrent programs connect to the ORACLE schema.

By checking for incompatibilities between programs running concurrently, accessing the same data, Oracle Applications ensures that data retrieved by one program is not incorrect or adversely affected when retrieved by another program.

Example - Program Incompatibilities

An example of a concurrent program that is incompatible with other concurrent programs is Oracle General Ledger's Posting program, used to post journal entries.

If the Posting program's incompatibility with other Oracle Applications concurrent programs were not enforced, other financial reports running simultaneously with the Posting program could contain incorrect account balance information. Logical databases ensure that this does not happen.

Defining Logical Databases

The installation process automatically defines logical databases and assigns ORACLE schemas to them.

A *Standard* logical database can be assigned to every Oracle Applications product so that every concurrent program, if incompatible with any other program, does not run concurrently with that program, regardless of which ORACLE schema those two programs connect to. Assigning every ORACLE schema to the same (e.g., Standard) logical database is a fail-safe method of enforcing program incompatibility rules.

You must define new logical databases only if you build a custom application whose data do not interact with data found in existing logical databases.

As a general rule, you should define a logical database for each custom application, and assign that application's ORACLE schema(s) to the corresponding logical database.

However, if a custom application's data interacts with another application's data, you should assign the two applications' ORACLE schemas to the same logical database.

Registering your custom application's tables ensures that the table names appear as QuickPick values in the Define Alerts form.

Concurrent Program Details Report

This report documents concurrent program definitions, including executable file information, execution method, incompatible program listings, and program parameters. If a concurrent program generates a report, column and row information, as well as print output and print style, are also documented.

Use this report when considering concurrent program modifications, such as modifying program incompatibility rules.

Report Parameters

Caution: If you do not enter any parameters, the report returns values for *all* concurrent programs, and may be very lengthy.

Application Name

Choose the application name associated with the concurrent program whose program definition details you wish to report on.

Choose only an application name, without a program name, if you wish to run a program definition details report on all concurrent programs associated with an application.

Program

Choose the name of a concurrent program whose program definition details you wish to report on. You must enter a value for Application Name before entering a value for Program.

Report Headings

The report headings display the specified report parameters and provide you with general information about the contents of the report.

Concurrent Programs Report, page 6-56

Concurrent Programs Report

This report shows which concurrent programs are currently enabled nand which programs are disabled.

Use this report to record the execution method, argument method, run alone status, standard submission status, request type, and print style information associated with your concurrent programs.

Report Parameters

Application Name

Choose the application name associated with the concurrent programs whose program information you wish to report on.

If you do not enter an application name, the report will return values for *all* concurrent programs.

Report Headings

The report headings display the specified report parameters and provide you with general information about the contents of the report.

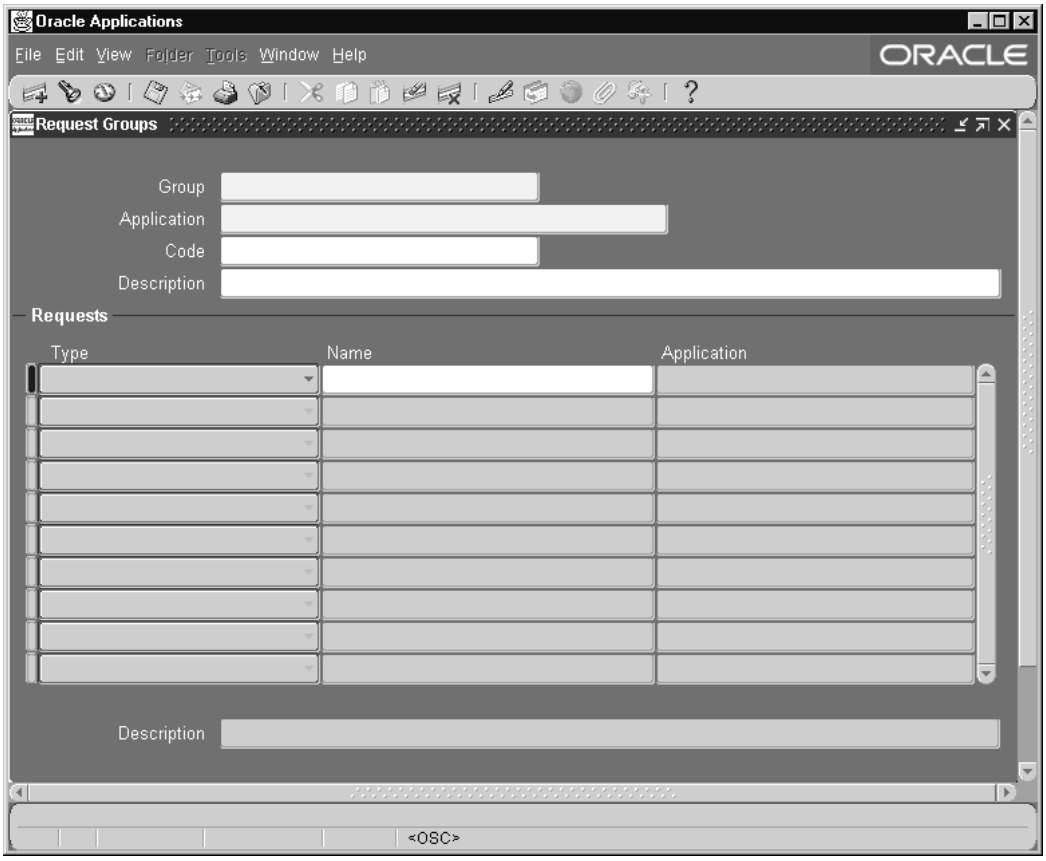
Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Concurrent Program Details Report, page 6-56

Concurrent Programs, page 6-63

Request Groups Window



Use this window to define a request group. A request *security* group is the collection of requests, request sets, and concurrent programs that a user, operating under a given responsibility, can select from the Submit Requests window.

System Administrators:

- Assign a request security group to a responsibility when defining that responsibility. A responsibility without a request security group cannot run any requests using the Submit Requests window.

- Can add *any* request set to a request security group. Adding a private request set to a request security group allows other users to run that request set using the Submit Requests window.

Users:

- Can create their own private request sets using the Request Sets window. In a private request set, users can include only the requests you assign to their request security group.

- Cannot update another user's private request set using the Request Sets window.
- Cannot delete a private request set if it is assigned to a request security group.

Request Groups Block

Group

Use the request group's name to assign the request group to a responsibility on the Responsibilities window. An application name and request group name uniquely identify a request group.

Application

Select the name of the application you wish to associate with your request group. An application name and a request security group name uniquely identify a request security group. This application name does not prevent you from assigning requests and request sets from other applications to this request group.

Code

Assign a code to this request group. Some products use the request group code as a parameter that identifies the requests a customized standard submission form can select. See: Customizing the Submit Requests Window using Codes, page 6-26.

Requests Block

Specify the requests and request sets in the request group.

Type

Choose program or set to add one item, or choose application to include all requests in an application.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

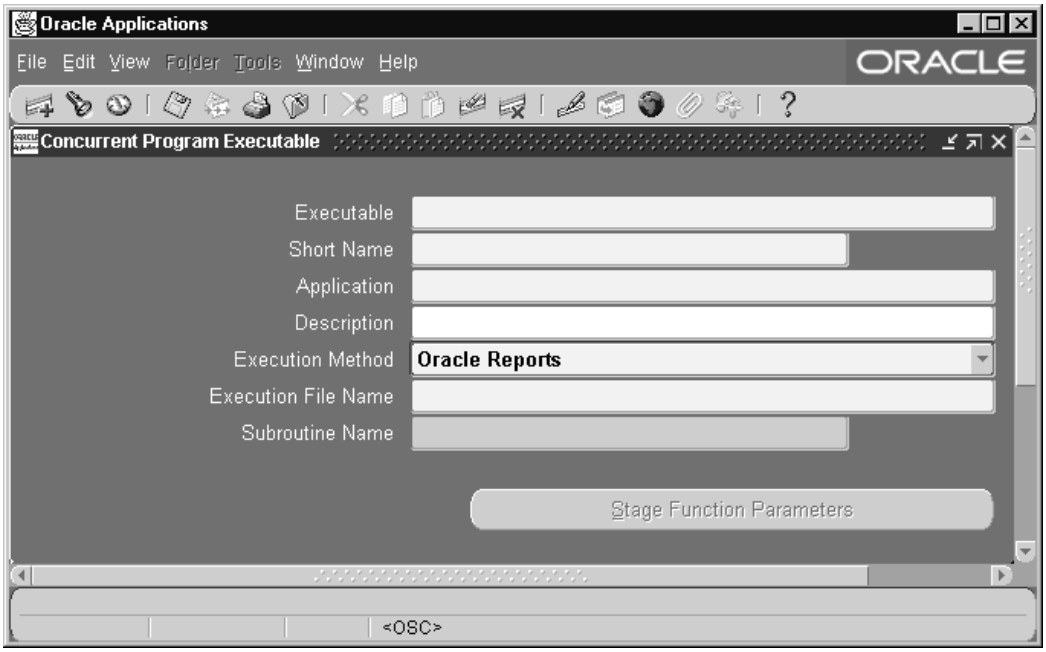
Organizing Programs into Request Groups, page 6-22

Using Codes with Request Groups, page 6-23

Customizing the Submit Requests Window using Codes, page 6-26

Report Group Responsibilities Report, page 6-27

Concurrent Program Executable Window



Define a concurrent program executable for each executable source file you want to use with concurrent programs. The concurrent program executable links your source file logic with the concurrent requests you and your users submit to the concurrent manager.

> **Important:** You cannot add new immediate programs to a concurrent manager program library. We recommend that you use spawned concurrent programs instead.

Concurrent Program Executable Block

The combination of application name plus program name uniquely identifies your concurrent program executable.

See: Concurrent Programs Window, page 6-63

Executable

Enter a name for your concurrent program executable. In the Concurrent Programs window, you assign this name to a concurrent program to associate your concurrent program with your executable logic.

Short Name

Enter a short name for your concurrent program executable.

Application

The concurrent managers use the application to determine in which directory structure to look for your execution file.

Execution Method

The execution method cannot be changed once the concurrent program executable has been assigned to one or more concurrent programs in the Concurrent Programs window.

The possible execution methods are:

Host	The execution file is a host script.
Oracle Reports	The execution file is an Oracle Reports file.
PL/SQL Stored Procedure	The execution file is a PL/SQL stored procedure.
Java Stored Procedure	The execution file is a Java stored procedure.
Java Concurrent Program	The execution file is a program written in Java.
Multi Language Function	The execution file is a function (MLS function) that supports running concurrent programs in multiple languages.
SQL*Loader	The execution file is a SQL script.
SQL*Plus	The execution file is a SQL*Plus script.
Spawned	The execution file is a C or Pro*C program.
Immediate	The execution file is a program written to run as a subroutine of the concurrent manager. We recommend against defining new immediate concurrent programs, and suggest you use either a PL/SQL Stored Procedure or a Spawned C Program instead.
Request Set Stage Function	PL/SQL Stored Function that can be uesd to calculate the completion statuses of request set stages.

Execution File Name

Enter the operating system name of your execution file. Some operating systems are

case sensitive, so the name entered here should match the file name exactly.

Do not include spaces or periods (.) in the execution file name, unless the execution method is PL/SQL stored procedure or Request Set Stage Function.

The maximum size of an execution file name is 60 characters.

Subroutine Name

Enter the name of your C or Pro*C program subroutine here. Do not use spaces or periods (.) in this field.

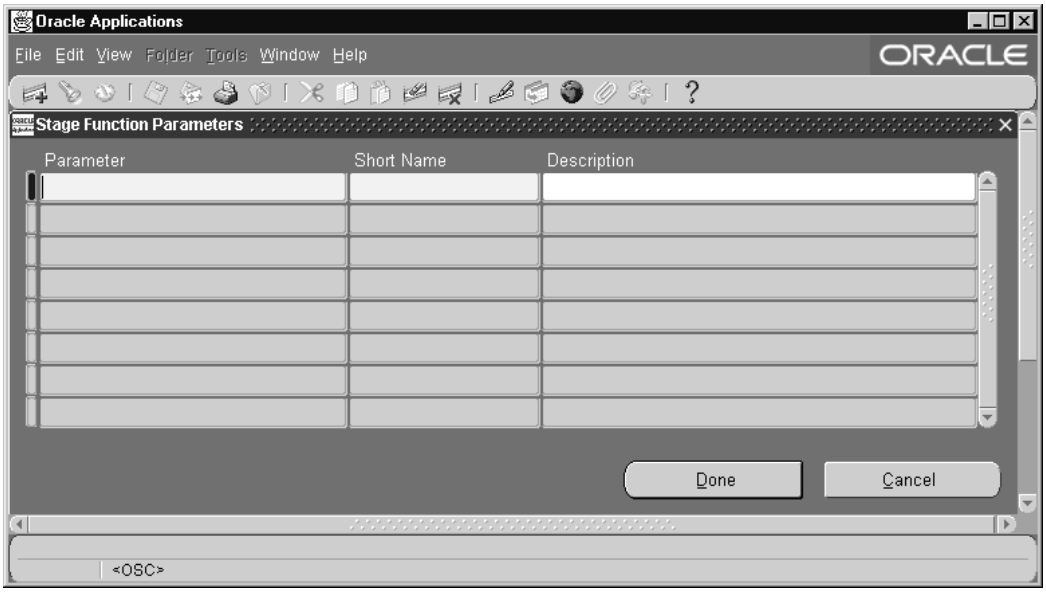
Only immediate programs or spawned programs using the Unified C API use the subroutine field.

We recommend against defining new immediate concurrent programs, and suggest you use either a PL/SQL Stored Procedure or a Spawned C Program instead.

Stage Function Parameters

The Stage Function Parameters button opens a window that allows you to enter parameters for the Request Set Stage Function. This button is only enabled when you select Request Set Stage Function as your Execution Method.

Stage Function Parameters Window



List the Parameters that your custom Stage Function uses.

Parameter

Enter a name for the Parameter. This name will be displayed in the Stage Functions Parameter window of the Request Set form.

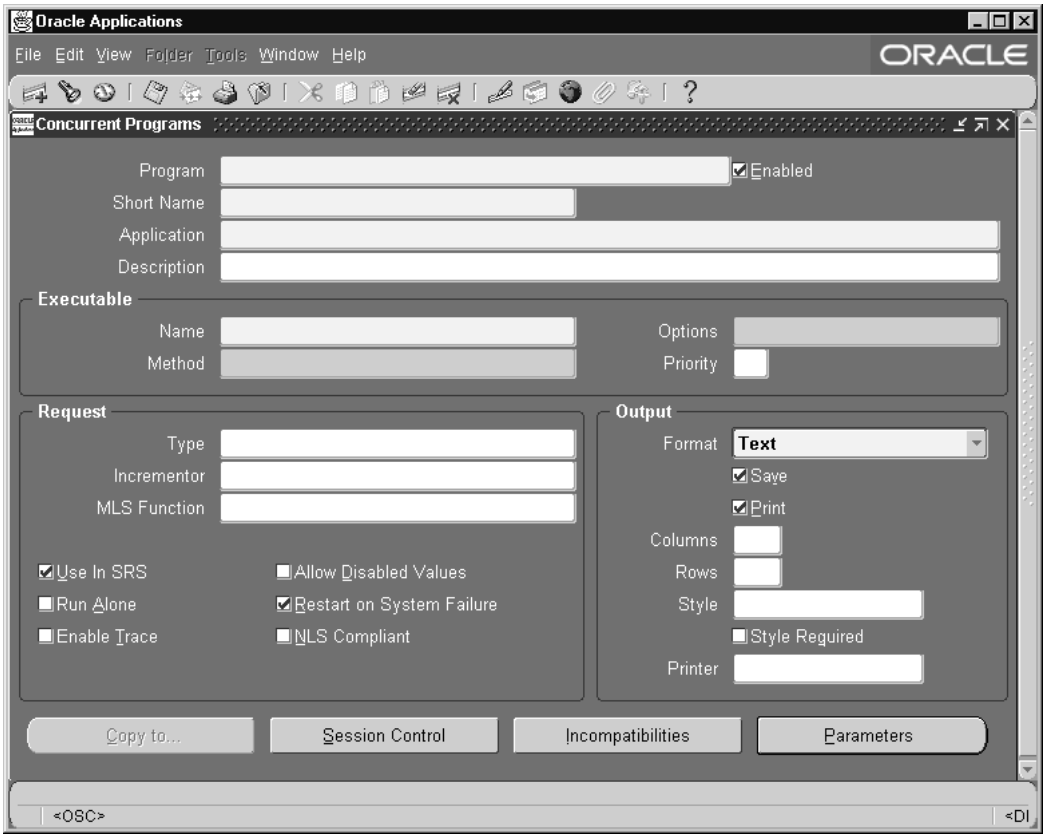
Short Name

Enter a short name that will be used by the function to reference the parameter.

Related Topics

Concurrent Programs, page 6-63

Concurrent Programs Window



Use this window to define and modify your concurrent programs.

Prerequisites

Build the execution file for your concurrent program.

Use the Concurrent Program Executables window to define a concurrent program executable for your operating system program.

Concurrent Programs Block

The combination of application name plus program name uniquely identifies your

concurrent program.

Program

You see this longer, more descriptive name when you view your requests in the Requests window. If this concurrent program runs through Standard Request Submission, you see this name in the Submit Requests window when you run this program.

Short Name

Enter a brief name that Oracle Applications can use to associate your concurrent program with a concurrent program executable.

Application

The program's application determines what ORACLE username your program runs in and where to place the log and output files.

Enabled

Indicate whether users should be able to submit requests to run this program and the concurrent managers should be able to run your program.

Disabled programs do not show up in users' lists, and do not appear in any concurrent manager queues. You cannot delete a concurrent program because its information helps to provide an audit trail.

(Executable) Executable: Name

Select the concurrent program executable that can run your program. You define the executable using the Concurrent Program Executables window. You can define multiple concurrent programs using the same concurrent program executable. See: Concurrent Program Executables, page 6-60.

(Executable) Executable: Options

Some execution methods, such as Oracle Reports, support additional execution options or parameters. You can enter such options in this field. The syntax varies depending on the execution method.

If you define a concurrent program with the bitmapped version of Oracle Reports, you can control the orientation of the bitmapped report by passing the ORIENTATION parameter or token. For example, to generate a report with landscape orientation, specify the following option in the Options field:

```
ORIENTATION=LANDSCAPE
```

Do not put spaces before or after the execution options values. The parameters should be separated by only a *single* space. You can also specify an orientation of PORTRAIT.

You can control the dimensions of the generated output with the PAGESIZE parameter. A specified <*width*>x<*height*> in the Options field overrides the values specified in the report definition. For example:

```
ORIENTATION=LANDSCAPE PAGESIZE=8x11.5
```

The units for your width and height are determined by your Oracle Reports definition. You set the units in your Oracle Reports menu under Report => Global Properties => Unit of Measurement.

If the page size you specify with the PAGESIZE parameter is smaller than what the report was designed for, your report fails with a "REP-1212" error.

(Executable) Executable: Method

The execution method your concurrent program uses appears here.

Valid values are:

Spawned Your concurrent program is a stand-alone program in C or Pro*C.

Host Your concurrent program is written in a script for your operating system.

Immediate Your concurrent program is a subroutine written in C or Pro*C. Immediate programs are linked in with your concurrent manage and must be included in the manager's program library.

Oracle Reports Your concurrent program is an Oracle Reports script.

PL/SQL Stored Procedure Your concurrent program is a stored procedure written in PL/SQL.

Java Stored Procedure Your concurrent program is a Java stored procedure.

Java Concurrent Program Your concurrent program is a program written in Java.

Multi Language Function A multi-language support function (MLS function) is a function that supports running concurrent programs in multiple languages. You should not choose a multi-language function in the Executable: Name field. If you have an MLS function for your program (in addition to an appropriate concurrent program executable), you specify it in the MLS Function field.

SQL*Loader Your concurrent program is a SQL*Loader program.

SQL*Plus Your concurrent program is a SQL*Plus or PL/SQL script.

Request Set Stage Function PL/SQL Stored Function that can be used to calculate the completion statuses of request set stages.

You can switch between Spawned and Immediate, overriding the execution method defined in the Concurrent Program Executable window, only if either method appears when the executable is selected and both an execution file name and subroutine name have already been specified in the Concurrent Program Executable window. See: Concurrent Program Executables, page 6-60.

(Executable) Priority

You can assign this program its own priority. The concurrent managers process requests for this program at the priority you assign here.

If you do not assign a priority, the user's profile option Concurrent:Priority sets the request's priority at submission time.

(Request) Type

If you want to associate your program with a predefined request type, enter the name of the request type here. The request type can limit which concurrent managers can run your concurrent program.

(Request) Incrementor

For use by Oracle Applications internal developers only. The incrementor function is shown here.

(Request) MLS Function

The MLS function, if any, used by the program.

The Multilingual Concurrent Request feature allows a user to submit a request once to be run multiple times, each time in a different language. If this program utilizes this feature the MLS function determines which installed languages are needed for the request.

See:

Oracle Applications Developer's Guide

(Request) Use in SRS

Check this box to indicate that users can submit a request to run this program from a Standard Request Submission window.

If you check this box, you must register your program parameters, if any, in the Parameters window accessed from the button at the bottom of this window.

(Request) Allow Disabled Values

If you check the Use in SRS box, you can also check this box to allow a user to enter disabled or outdated values as parameter values.

Many value sets use special table columns that indicate whether a particular value is enabled (using ENABLED_FLAG, START_DATE_ACTIVE, and END_DATE_ACTIVE columns). These value sets normally allow you to query disabled or outdated values but not enter them in new data. For Standard Request Submission, this means that a user would not normally be allowed to enter disabled values as report parameter values when submitting a report, even if the report is a query-only type report.

(Request) Run Alone

Indicate whether your program should run alone relative to all other programs in the same logical database. If the execution of your program interferes with the execution of all other programs in the same logical database (in other words, if your program is incompatible with all programs in its logical database, including itself), it should run alone.

You can enter any specific incompatible programs in the Incompatible Programs windows.

(Request) Enable Trace

Turns on SQL tracing when program runs.

(Request) Restart on System Failure

Use this option to indicate that this concurrent program should automatically be restarted when the concurrent manager is restored after a system failure.

(Request) NLS Compliant

This box is checked if the program allows for a user to submit a request of this program that will reflect a language and territory that are different from the language and territory that the users are operating in.

For example, users can enter orders in English in the United Kingdom, using the date and number formats appropriate in the United Kingdom, then generate invoices in German using the date and number formats appropriate to their German customers.

If this box is left blank then a user can associate any installed language with the request, but the territory will default to the territory of the concurrent manager environment.

Note that this option should be set only by the developer of the program. The program must be written as NLS Compliant to utilize this feature. See: the *Oracle Applications Developer's Guide.*

Note that this option should be set only by the developer of the program. The program must be written as NLS Compliant to utilize this feature.

(Output) Format

Select the output format from the following:

- HTML
- PCL (HP's Printer Control Language)
- PDF
- PS (Post Script)
- Text

> **Important:** If you choose HTML or PDF as the output type with Oracle Report programs, you must use an appropriate printer driver that handles HTML or PDF files.

(Output) Save

Indicate whether to automatically save the output from this program to an operating system file when it is run. This value becomes the default for all requests submitted for this program. The output of programs with Save set to No is deleted after printing.

If this is a Standard Request Submission program, users can override this value from the Submit Requests window.

(Output) Print

If you enter No, your concurrent program's output is never sent to the printer.

(Output) Columns / Rows

Enter the minimum column and row length for this program's report output. Oracle Applications uses this information to determine which print styles can accommodate your report.

(Output) Style

The print style you select depends on your system and printer setup. Print styles include:

- 132 columns and 66 lines (Landscape)
- 180 columns and 66 lines (Landwide)
- 80 columns and 66 lines (Portrait)

- 132 columns and 62 lines (A4)

Your list is limited to those styles that meet your program's columns and row length requirements.

(Output) Style Required

If your program requires a specific print style (for example, a checkwriting report), use this check box to enforce that print style.

(Output) Printer

If you want to restrict your program's output to a single printer, enter the name of the printer to which you want to send your output. If your program has minimum or maximum columns or rows defined, your list of values is limited to those printers that can support your program's requirements.

Users cannot override your choice of printer from the Submit Requests or Requests windows.

Concurrent Programs Buttons

Copy to... Choose this button to create another concurrent program using the same executable, request and report information. You can elect to copy the incompatibility and parameter details as well.

Session Control Choose this window to specify options for the database session of the concurrent program when it is executed.

Incompatibilities Choose this button to open the Incompatible Programs window.

Parameters Choose this button to open the Concurrent Program Parameters window.

Copy to Window

Create another concurrent program using the same executable, request and report information as the current program. You can optionally copy the incompatibility and parameter details information as well.

Related Topics

See: Incompatible Programs Window, page 6-71

Concurrent Program Parameters Window, page 6-73

Concurrent Program Executables, page 6-60

Session Control Window

Use this window to specify options for the database session of the concurrent program when it is executed.

Consumer Group

Optionally specify the resource consumer group for the concurrent program.

See: Resource Consumer Groups in Oracle Applications, page 11-3.

Rollback Segment

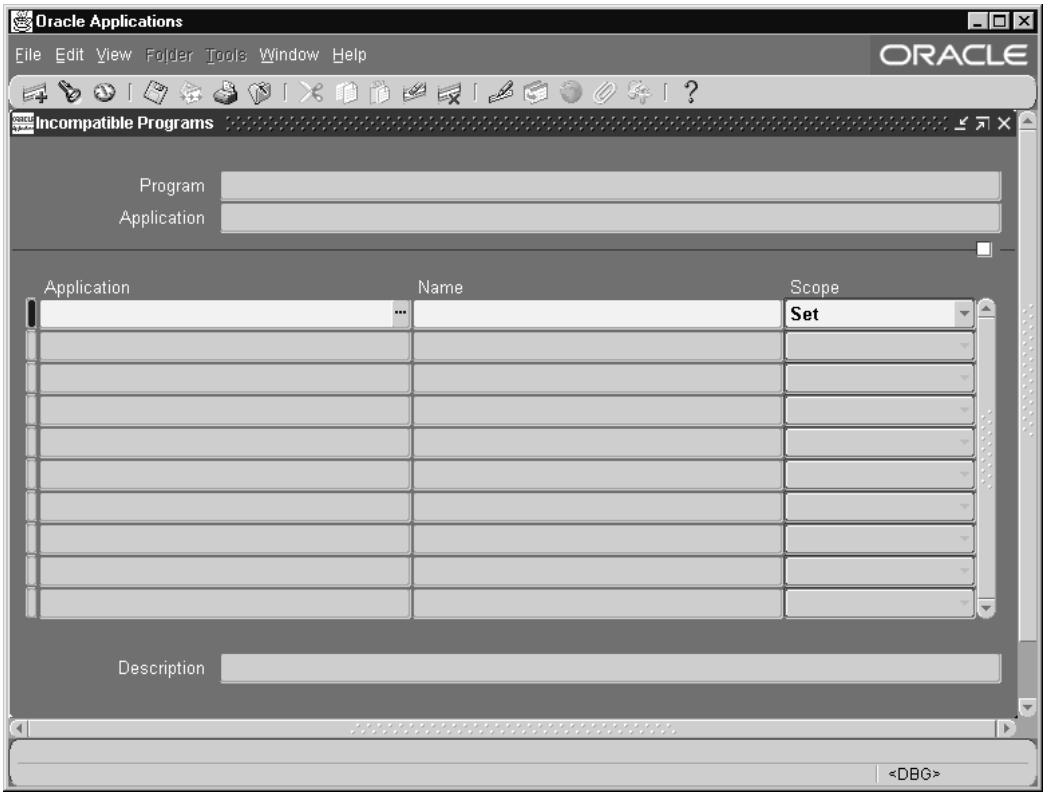
Optionally specify a rollback segment to be used with the concurrent program. This rollback segment will be used instead of the default and will be used up until the first commit.

> **Important:** If you specify a rollback segment here, your concurrent program must use the APIs FND_CONCURRENT.AF_COMMIT and FND_CONCURRENT.AF_ROLLBACK to use the specified rollback segment. See: the *Oracle Applications Developer's Guide*.

Optimizer Mode

Optionally specify an optimizer mode. You can choose ALL_ROWS, FIRST_ROWS, Rules, or Choose. You would specify an optimizer mode only for a custom program that may not perform well with the default cost-based optimizer (CBO) and needs tuning. You can use a different optimizer mode until your program is tuned for CBO.

Incompatible Programs Window



Identify programs that should not run simultaneously with your concurrent program because they might interfere with its execution. You can specify your program as being incompatible with itself.

Application

Although the default for this field is the application of your concurrent program, you can enter any valid application name.

Name

The program name and application you specify must uniquely identify a concurrent program.

Your list displays the user-friendly name of the program, the short name, and the description of the program.

Scope

Enter Set or Program Only to specify whether your concurrent program is incompatible with this program and all its child requests (Set) or only with this program (Program Only).

Type

Enter Domain or Global. If you choose Domain, the incompatibility is resolved at a domain-specific level. If you choose Global, then this concurrent program will be considered globally incompatible with your concurrent program, regardless of which domain it is running in.

Related Topics

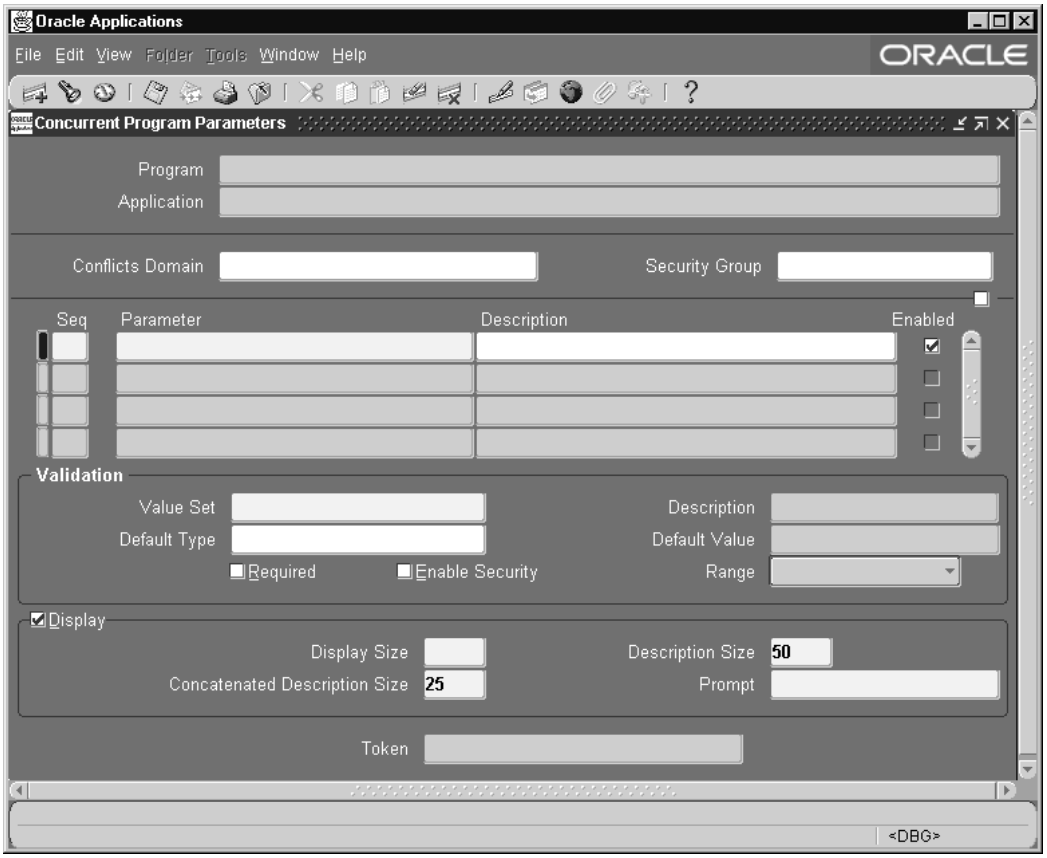
Concurrent Programs Window, page 6-63

Concurrent Program Parameters Window, page 6-73

Defining Program Incompatibility Rules, page 6-27

Modifying an Incompatible Programs List, page 6-46

Concurrent Program Parameters Window



Enter and update the program parameters that you wish to pass to the program executable. Program parameters defined here should match the variables in your execution file.

Conflicts Domain Parameter

Enter the parameter which will hold the value of the conflict domain of the program. For information on conflict domain parameters, see Concurrent Conflict Domains, page 6-29.

Enter the parameter which will hold the value of the conflict domain of the program.

Security Group

This field is for HRMS security only. See: *Customizing, Reporting, and System Administration in Oracle HRMS*.

Sequence

Choose the sequence numbers that specify the order in which your program receives

parameter values from the concurrent manager.

Enabled

Disabled parameters do not display at request submission time and are not passed to your execution file.

Argument Detail

You specify information about your parameter almost exactly as you define a flexfield segment.

(Validation Information) Value Set

Enter the name of the value set you want your parameter to use for validation. You can only select from independent, table, and non-validated value sets.

The maximum size of your value set is 240 characters.

> **Important:** If you are using a value set of dates, this value set should have a format type of either Standard Date or Standard DateTime if you are using the Multilingual Request feature.

(Validation Information) Default Type

If you want to set a default value for this parameter, identify the type of value you need.

Valid types include:

Constant The default value can be any literal value.

Profile The default value is the current value in the user profile option defined in the Default Value field. Use the profile option name, not the end-user name. You do not need to include $PROFILE$.

SQL Statement The default value is determined by the SQL statement you defined in the Default Value field.

Segment The default value is the value entered in a prior segment of the same parameter window.

(Validation Information) Default Value

You can enter a default value for the parameter. This default value for your parameter automatically appears when you enter your parameter window. You determine whether the default value is a constant or a context-dependent value by choosing the default type.

Your default value should be a valid value for your value set. Otherwise you see an error message when you enter your parameter window on the Run Request window and your default value does not appear.

Valid values for each default type include:

Constant Enter any literal value for the default value.

Profile The default value is the current value of the user profile option you specify here. Enter the profile option name, not the end-user name.

Segment The default value is the value entered in a prior segment of the same flexfield window. Enter the name of the segment whose value you want to copy.

SQL Statement The default value is determined by the SQL statement you enter here. Your SQL statement must return exactly one row and one column in all cases.

(Validation Information) Required

If the program executable file requires an argument, you should require it for your concurrent program.

(Validation Information) Enable Security

If the value set for this parameter does not allow security rules, then this field is display only. Otherwise you can elect to apply any security rules defined for this value set to affect your parameter list.

(Validation Information) Range

Choose either Low or High if you want to validate your parameter value against the value of another parameter in this structure. Parameters with a range of Low must appear before parameters with a range of High (the low parameter must have a lower number than the high parameter). For example, if you plan two parameters named "Start Date" and "End Date," you may want to force users to enter an end date later than the start date. You could assign "Start Date" a range of Low and "End Date" a range of High. In this example, the parameter you name "Start Date" must appear before the parameter you name "End Date."

If you choose Low for one parameter, you must also choose High for another parameter in that structure (and vice versa). Otherwise you cannot commit your changes.

(Window Information) Display

Indicate whether to display this parameter in the Parameters window when a user submits a request to run the program from the Submit Requests window.

You should provide a default type and value for any non-displayed parameter.

(Window Information) Display Size

Enter the field length in characters for this parameter. The user sees and fills in the field in the Parameters window of the Submit Requests window.

You should ensure that the total of the value set maximum sizes (not the display sizes) for all of your parameters, plus the number of separators you need (number of parameters minus one), does not add up to more than 240. If your program values' concatenated length exceeds 240, you may experience truncation of your data in some forms.

(Window Information) Description Size

Enter the display length in characters for the parameter value description. Your window may show fewer characters of your description than you specify here if there is not enough room (determined by the sum of your longest prompt plus your display size for this parameter plus seven). However, your window does not display more characters of the description than you specify here.

(Window Information) Prompt

A user sees the prompt instead of the parameter name in the Parameters window of the Submit Requests window.

(Window Information) Concatenated Description Size

Enter the display length in characters for the parameter value description. The user sees the parameter value in the Parameter Description field of the Submit Requests and View Requests forms. The Parameter Description field concatenates all the parameter values for the concurrent program.

> **Tip:** We recommend that you set the Concatenated Description Size for each of your parameters so that the total Concatenated Description Size for your program is 80 or less, since most video screens are 80 characters wide.

(Window Information) Token

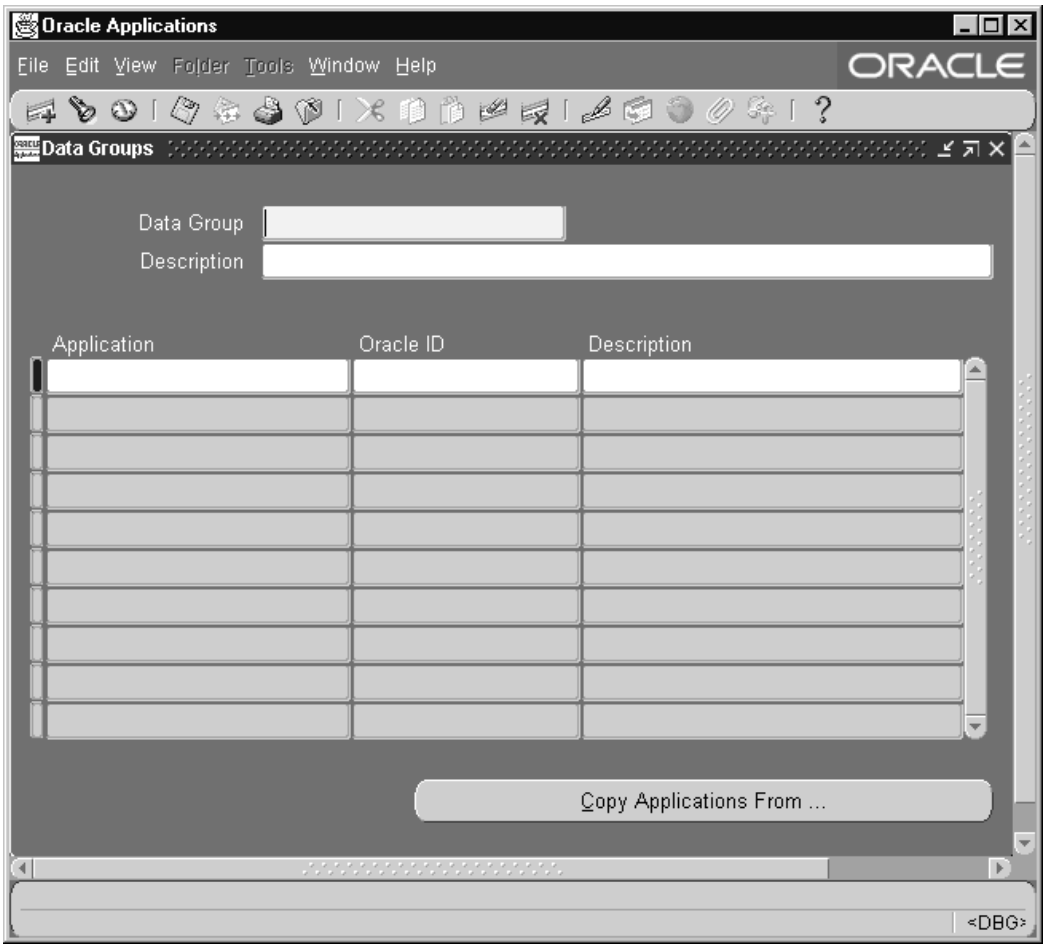
For a parameter in an Oracle Reports program, the keyword or parameter appears here. The value is case insensitive. For other types of programs, you can skip this field.

Related Topics

Concurrent Programs, page 6-63

Incompatible Programs Window, page 6-71

Data Groups Window



Note: Data groups are no longer supported. This section is provided for reference only.

Use this window to define data groups. A data group is a list of Oracle Applications and the ORACLE usernames assigned to each application.

- If a custom application is developed with Oracle Application Object Library, it may be assigned an ORACLE username, registered with Oracle Applications, and included in a data group.

An ORACLE username allows access to an application's tables in an ORACLE database. Alldata groups automatically include an entry for Application Object Library.

- A concurrent manager running reports or programs under Oracle Applications refers to a data group to identify the ORACLE username it uses to access an

application's tables in the database.

- Transaction managers running synchrous programs can only run programs submitted from responsibilities assigned the same data group as the transaction manager. If you create custom data groups, you should create new transaction managers for the applications that use transaction managers. Consult your product documenation to determine if your application uses transaction managers.

Each responsibility within Oracle Applications is assigned a data group.

During installation or upgrading of Oracle Applications, a standard data group is defined, pairing each installed application with an ORACLE username (note: a standard data group is defined for each set of books). You cannot change or delete the predefined values for *Application* or *ORACLE username* in a Standard data group. However, you may:

- Modify the Tool ORACLE username and description associated with an Application-ORACLE username pair.

- Add new Application-ORACLE username pairs to the group.

Data Groups Block

Create a new data group, or modify an existing data group.

You cannot change or delete the predefined values for *Application* or *ORACLE username* in a Standard data group. However, you may modify the Tool ORACLE username and description, or add new Application-ORACLE username pairs to a Standard group.

Data Group

A data group is uniquely identified by its name. You cannot create a data group with a name already in use.

Once saved, data group names cannot be edited.

Application-ORACLE ID Pairs Block

Pair applications with ORACLE usernames.

When you copy a data group, each application, its assigned ORACLE username, and, if present, its Tool ORACLE username and description, appear in this zone automatically. All data groups automatically include an entry for Application Object Library.

Application

Within each data group, an application can be listed only one time.

Oracle ID

Select the ORACLE ID you want to assign to an application. An application uses an ORACLE ID to access tables in the database. Each ORACLE ID allows access to a predefined set of tables in the database.

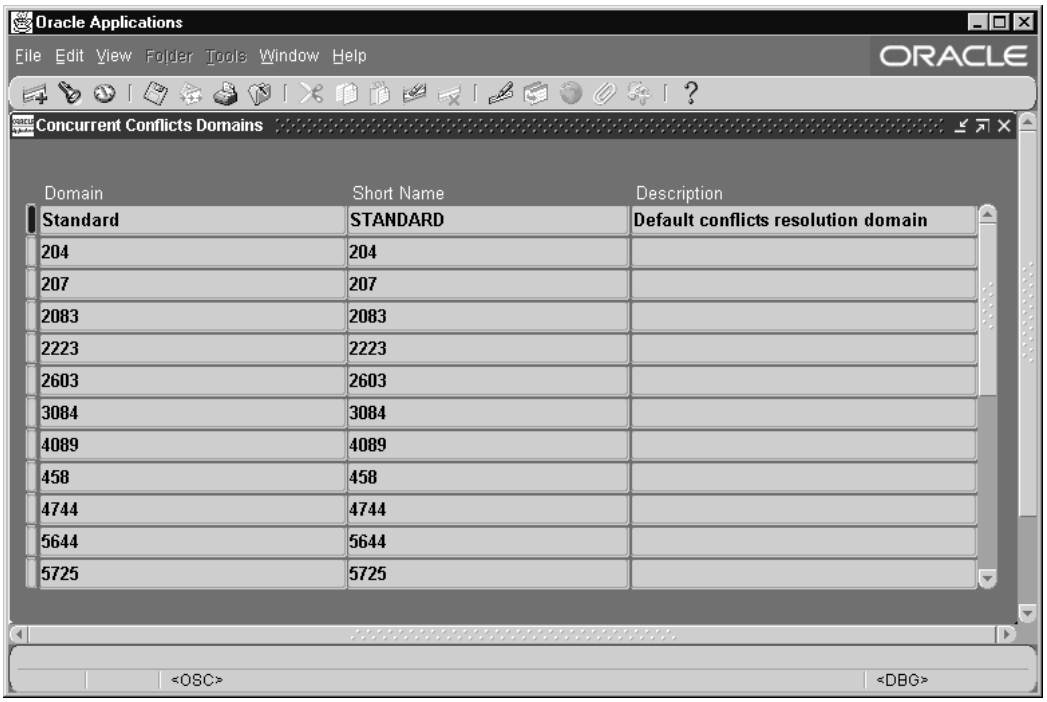
Copy Applications From...

Use this button to copy an existing data group, then add or delete application-ORACLE username pairs to create a new data group.

Related Topics

Overview of Concurrent Programs and Requests, page 6-1

Concurrent Conflicts Domains Window



Concurrent conflicts domains ensure that incompatible concurrent programs are not allowed to run simultaneously using related information.

For example, a conflict domain could be a range of numbers. Two concurrent programs could be incompatible if they used the same range of numbers, but compatible if they used different ranges of numbers.

Concurrent managers use concurrent conflicts domains to determine which concurrent programs cannot run at the same time. For example:

- When concurrent program A is defined as incompatible with concurrent program B, then A and B cannot run at the same time using the same concurrent conflict domain.

- If, for example, the programs A and B are assigned to the concurrent conflicts domains *Standard* when they are submitted, then programs A and B will not run together at the same time.

To define a conflict domain:

1. Enter a unique Domain name. The name you enter here may be used as a value for a parameter in the Submit Requests window.

2. Enter a unique Short Name for your domain. Limit the Short Name to 8 characters.

3. Optionally, you can provide a description for your domain.

Related Topics

Overview of Applications DBA Duties, page 11-1

ORACLE Usernames (Overview), page 11-1

ORACLE Users, page 11-8

Applications, page 11-12

Concurrent Programs HTML UI

Search for Concurrent Programs

Use this page to search for defined concurrent programs.

From this page you can create a new concurrent program or update an existing one.

Create Concurrent Program

The following are prerequisites to defining a concurrent program:

- Build the execution file for your concurrent program.

- Define a concurrent program executable for your operating system program file.

The combination of application name plus program name uniquely identifies your concurrent program.

Update Annotation

Optionally enter an annotation for the Integration Repository for the concurrent

program.

Enabled

Indicate whether users should be able to submit requests to run this program and the concurrent managers should be able to run your program.

Disabled programs do not show up in users' lists, and do not appear in any concurrent manager queues. You cannot delete a concurrent program because its information helps to provide an audit trail.

Program

You see this longer, more descriptive name when you view your requests in the Requests window. If this concurrent program runs through Standard Request Submission, you see this name in the Submit Requests window when you run this program.

Application

The program's application determines what ORACLE username your program runs in and where to place the log and output files.

Short Name

Enter a brief name that Oracle Applications can use to associate your concurrent program with a concurrent program executable.

Program Type

Available options are:

- Archive - reserved for future use only.
- Autoconfig Type
- Cloning - reserved for internal use only.
- Generic
- Purge - for concurrent programs listed in the Oracle Applications Manager dashboard used for purging data.
- Refresh
- Truncate

Executable

Enter the following:

Name

Select the concurrent program executable that can run your program. You define the executable using the Concurrent Program Executables window. You can define multiple concurrent programs using the same concurrent program executable.

Parameters

Parameters for the concurrent program are listed here. To add a parameter, click on the **Create** button.

Incompatibilities

Identify programs that should not run simultaneously with your concurrent program because they might interfere with its execution. You can specify your program as being incompatible with itself.

Conflict Domain Parameter

Enter the parameter which will hold the value of the conflict domain of the program.

For information on conflict domain parameters, see Concurrent Conflict Domains, page 6-29.

Run Alone

Indicate whether your program should run alone relative to all other programs in the same logical database. If the execution of your program interferes with the execution of all other programs in the same logical database (in other words, if your program is incompatible with all programs in its logical database, including itself), it should run alone.

You can enter any specific incompatible programs in the Incompatible Programs windows.

Application

Although the default for this field is the application of your concurrent program, you can enter any valid application name.

Name

The program name and application you specify must uniquely identify a concurrent program.

Your list displays the user-friendly name of the program, the short name, and the description of the program.

Scope

Enter Set or Program Only to specify whether your concurrent program is incompatible with this program and all its child requests (Set) or only with this program (Program Only).

Type

Choose the type of incompatibility, either Domain or Global.

For information on incompatibility types, see Incompatible and Run Alone Programs, page 6-27.

Request

Enter the following:

Request Settings

Enter the following:

Type

If you want to associate your program with a predefined request type, enter the name of the request type here. The request type can limit which concurrent managers can run your concurrent program.

Incrementor

For use by Oracle Applications internal developers only. The incrementor function is shown here.

MLS Function

The MLS (Multi-Lingual Support) function, if any, used by the program.

The Multilingual Concurrent Request feature allows a user to submit a request once to be run multiple times, each time in a different language. If this program utilizes this feature the MLS function determines which installed languages are needed for the request. See the *Oracle Applications Developer's Guide* for more information.

Activity Summarizer

For internal use only.

The Activity Summarizer is a PL/SQL subprogram summarizing about the purgable data in application tables for a concurrent program of type "Purge". It returns a list of table names and rows to be purged. Oracle developers register the PL/SQL procedure as summarizer procedure for the concurrent program.

Refresh Portlet

For internal use only.

Concurrent programs that produce data for a portlet can call a function to refresh the portlet's data. The value for Refresh Portlet indicates when the function should be called.

Allow Multiple Pending Requests

If this box is checked, multiple pending requests are allowed; otherwise, only one pending request is allowed.

Use in SRS

Check the SRS (Standard Request Submission) box to indicate that users can submit a request to run this program from a Standard Request Submission window.

If you check this box, you must register your program parameters, if any, in the Parameters window accessed from the button at the bottom of this window.

Allow Disabled Values

If you check the Use in SRS box, you can also check this box to allow a user to enter disabled or outdated values as parameter values.

Many value sets use special table columns that indicate whether a particular value is enabled (using ENABLED_FLAG, START_DATE_ACTIVE, and END_DATE_ACTIVE columns). These value sets normally allow you to query disabled or outdated values but not enter them in new data. For Standard Request Submission, this means that a user would not normally be allowed to enter disabled values as report parameter values when submitting a report, even if the report is a query-only type report.

Restart on System Failure

Use this option to indicate that this concurrent program should automatically be restarted when the concurrent manager is restored after a system failure.

NLS Compliant

The NLS (National Language Support) box is checked if the program allows for a user to submit a request of this program that will reflect a language and territory that are different from the language and territory that the users are operating in.

For example, users can enter orders in English in the United Kingdom, using the date and number formats appropriate in the United Kingdom, then generate invoices in German using the date and number formats appropriate to their German customers.

If this box is left blank then a user can associate any installed language with the request, but the territory will default to the territory of the concurrent manager environment.

Note that this option should be set only by the developer of the program. The program must be written as NLS Compliant to utilize this feature. See: *Oracle Applications Developer's Guide.*

Output Preferences

Enter the following:

Save

Indicate whether to automatically save the output from this program to an operating system file when it is run. This value becomes the default for all requests submitted for this program. The output of programs with Save set to No is deleted after printing.

If this is a Standard Request Submission program, users can override this value from the Submit Requests window.

Print

If you enter No, your concurrent program's output is never sent to the printer.

Format

Select the output format from the following:

The format that you select here is used by the concurrent manager to determine how to display your report output. You must ensure that the output format you choose matches the format generated by your report, unless the report is an Oracle Reports report in which case the format you select, determines the output generated.

- HTML
- PCL (HP's Printer Control Language)
- PDF
- PS (Post Script)
- Text

> **Important:** If you choose HTML or PDF as the output type with Oracle Report programs, you must use an appropriate printer driver that handles HTML or PDF files.

Columns / Rows

Enter the minimum column and row length for this program's report output. Oracle Applications uses this information to determine which print styles can accommodate your report.

Style

The print style you select depends on your system and printer setup. Print styles include:

- 132 columns and 66 lines (Landscape)
- 180 columns and 66 lines (Landwide)
- 80 columns and 66 lines (Portrait)
- 132 columns and 62 lines (A4)

Your list is limited to those styles that meet your program's columns and row length requirements.

Style Required

If your program requires a specific print style (for example, a checkwriting report), use this check box to enforce that print style.

Onsite Setting

The following fields are typically specific to the installation.

General

Enter the following:

Priority

You can assign this program its own priority. The concurrent managers process requests for this program at the priority you assign here.

If you do not assign a priority, the user's profile option Concurrent:Priority sets the request's priority at submission time.

Printer

If you want to restrict your program's output to a single printer, enter the name of the printer to which you want to send your output. If your program has minimum or maximum columns or rows defined, your list of values is limited to those printers that can support your program's requirements.

Users cannot override your choice of printer from the Submit Requests or Requests windows.

Template

The default layout template for the program. Values for this field are available only if the concurrent program has been registered as a data definition with XML Publisher and templates have been loaded to the Template Manager. For more information on XML Publisher and the Template Manager, see the Oracle XML Publisher documentation..

At the time of request submission, the default template is presented to the user. The user can override this value when submitting the request.

Retain for

This field indicates how many days the system should retain data for a request of this concurrent program after the request completes. The system will retain this data for this period even if the "Purge Concurrent Request and/or Manager Data" program is run during this time.

Security Group

This field is for HRMS security only. See: *Customizing, Reporting, and System Administration in Oracle HRMS.*

Log Level for Failure

The log level is used in diagnostics. If a request to run this concurrent program fails, the failure may be recorded in a log file with the specified log level. For more information on logging, see the *Oracle Applications Supportability Guide.*

Enable Trace

Turns on SQL tracing when program runs.

Enable Time Statistics

Enables the collection of timed statistics, such as CPU and elapsed times, by the SQL trace facility, as well as the collection of various statistics in the dynamic performance tables.

Delete Log File

By default, a log file is created for each concurrent request. If such log files are not necessary for requests for this concurrent program, you can specify that the log file is automatically deleted for each request of this program.

Session Controls

Use this region to specify options for the database session of the concurrent program when it is executed.

Consumer Group

Optionally specify the resource consumer group for the concurrent program. See: Resource Consumer Groups in Oracle Applications, page 11-3

Rollback Segment

Optionally specify a rollback segment to be used with the concurrent program. This rollback segment will be used instead of the default and will be used up until the first commit.

> **Important:** If you specify a rollback segment here, your concurrent program must use the APIs FND_CONCURRENT.AF_COMMIT and FND_CONCURRENT.AF_ROLLBACK to use the specified rollback segment. See: *Oracle Applications Developer's Guide*.

Optimizer Mode

Optionally specify an optimizer mode. You can choose ALL_ROWS, FIRST_ROWS, Rules, or Choose. You would specify an optimizer mode only for a custom program that may not perform well with the default cost-based optimizer (CBO) and needs tuning. You can use a different optimizer mode until your program is tuned for CBO.

Statistics

This region provides statistics on earlier requests for a defined concurrent program.

Summary information is collected when a request is completed, and stored in the table fnd_conc_prog_onsite_info.

Concurrent Program - Add Parameter

Enter and update the program parameters that you wish to pass to the program executable. Program parameters defined here should match the variables in your execution file.

General

Enter the following:

Enabled

Disabled parameters do not display at request submission time and are not passed to your execution file.

Sequence

Choose the sequence numbers that specify the order in which your program receives parameter values from the concurrent manager.

Parameter

Enter the parameter name. The value is case insensitive.

Validation

Enter the following:

Value Set

Enter the name of the value set you want your parameter to use for validation. You can only select from independent, table, and non-validated value sets.

The maximum size of your value set is 240 characters.

> **Important:** If you are using a value set of dates, this value set should have a format type of either Standard Date or Standard DateTime if you are using the Multilingual Request feature.

Default Type

If you want to set a default value for this parameter, identify the type of value you need.

Valid types include:

Constant The default value can be any literal value.

Profile The default value is the current value in the user profile option defined in the Default Value field. Use the profile option name, not the end-user name. You do not need to include $PROFILE$.

SQL Statement The default value is determined by the SQL statement you defined in the Default Value field.

Segment The default value is the value entered in a prior segment of

the same parameter window.

Default Value

You can enter a default value for the parameter. This default value for your parameter automatically appears when you enter your parameter window. You determine whether the default value is a constant or a context-dependent value by choosing the default type.

Your default value should be a valid value for your value set. Otherwise you see an error message when you enter your parameter window on the Run Request window and your default value does not appear.

Valid values for each default type include:

Constant Enter any literal value for the default value.

Profile The default value is the current value of the user profile option you specify here. Enter the profile option name, not the end-user name.

Segment The default value is the value entered in a prior segment of the same flexfield window. Enter the name of the segment whose value you want to copy.

SQL Statement The default value is determined by the SQL statement you enter here. Your SQL statement must return exactly one row and one column in all cases.

Required

If the program executable file requires an argument, you should require it for your concurrent program.

Enable Security

If the value set for this parameter does not allow security rules, then this field is display only. Otherwise you can elect to apply any security rules defined for this value set to affect your parameter list.

Range

Choose either Low or High if you want to validate your parameter value against the value of another parameter in this structure. Parameters with a range of Low must appear before parameters with a range of High (the low parameter must have a lower number than the high parameter). For example, if you plan two parameters named "Start Date" and "End Date", you may want to force users to enter an end date later than the start date. You could assign "Start Date" a range of Low and "End Date" a range of High. In this example, the parameter you name "Start Date" must appear before the parameter you name "End Date".

If you choose Low for one parameter, you must also choose High for another parameter in that structure (and vice versa). Otherwise you cannot commit your changes.

If your value set is of the type Pair, this field is display only. The value defaults to Pair.

Display

Enter the following:

Display

Indicate whether to display this parameter in the Parameters window when a user submits a request to run the program from the Submit Requests window.

You should provide a default type and value for any non-displayed parameter.

Token

For a parameter in an Oracle Reports program, the keyword or parameter appears here. The value is case insensitive. For other types of programs, you can skip this field.

Description Size

Enter the display length in characters for the parameter value description. Your window may show fewer characters of your description than you specify here if there is not enough room (determined by the sum of your longest prompt plus your display size for this parameter plus seven). However, your window does not display more characters of the description than you specify here.

Display Size

Enter the field length in characters for this parameter. The user sees and fills in the field in the Parameters window of the Submit Requests window.

You should ensure that the total of the value set maximum sizes (not the display sizes) for all of your parameters, plus the number of separators you need (number of parameters minus one), does not add up to more than 240. If your program values' concatenated length exceeds 240, you may experience truncation of your data in some forms.

The default is the name of the parameter.

Prompt

A user sees the prompt instead of the parameter name in the Parameters window of the Submit Requests window.

Concatenated Description Size

Enter the display length in characters for the parameter value description. The user sees the parameter value in the Parameter Description field of the Submit Requests and View Requests forms. The Parameter Description field concatenates all the parameter values for the concurrent program.

Tip: We recommend that you set the Concatenated Description Size for each of your parameters so that the total Concatenated Description Size for your program is 80 or less, since most video screens are 80 characters wide.

7

Defining Concurrent Managers

Defining Managers and their Work Shifts in the Oracle Forms UI

This section explains how you can define concurrent managers and specify when a manager is enabled.

A concurrent manager is itself a concurrent program that starts other concurrent programs running. When an application user submits a request to run a program, the request is entered into a database table that lists all of the requests. Concurrent managers read requests from the table and start programs running. See: Concurrent Managers, page 7-58.

In this section, we explain how to specify when a manager is enabled, how to use managers to balance your applications processing workload across different time periods, and how to associate a library of immediate concurrent programs to be called by your manager.

Defining new managers

You can define as many concurrent managers as you want. When you define a manager, you:

- Assign a predefined library of *immediate* concurrent programs to your manager.

  Immediate concurrent programs are subroutines associated with concurrent managers. All other concurrent programs are spawned as independent processes at run time.

- Assign work shifts to your manager, which determines what days and times the manager works.

- For each work shift, you define the maximum number of operating system processes the manager can run concurrently to read requests (start programs) during the work shift.

- Specialize your manager to read only certain kinds of requests.

Program Libraries

For a program that is spawned, a concurrent manager initiates or spawns another operating system process. A program that is immediate runs as part of the concurrent manager's operating system process.

A program library contains immediate concurrent programs that can be called by your manager.

An immediate concurrent program must be registered with a program library. Application developers using Oracle Application Object Librarycan register concurrent programs with a program library.

The Oracle Application Object Library FNDLIBR program library contains Oracle Applications immediate concurrent programs, and is assigned to the Standard concurrent manager. In most cases, you will include the FNDLIBR library with your manager's definition.

The Internal and the Standard concurrent managers

Oracle System Administration predefines two managers for you:

- The Internal Concurrent Manager, which functions as the "boss" of all the other managers. The Internal Concurrent Manager starts up, verifies the status of, resets, and shuts down the individual managers.

  You cannot alter the definition of the Internal Concurrent Manager.

- A manager named Standard. The Standard manager accepts any and all requests; it has *no specialization.* The Standard manager is active all the time; it works 365 days a year, 24 hours a day.

  > **Warning:** You should not alter the definition of the Standard concurrent manager. If you do, and you have not defined additional managers to accept your requests, some programs may not run. Use the Standard manager as a safety net, a manager who is always available to run any request. Define additional managers to handle your installation site's specific needs.

Transaction Managers

While conventional concurrent managers let you execute long-running, data-intensive application programs *asynchronously*, transaction managers support *synchronous* processing of particular requests from client machines. A request from a client program

to run a server-side program synchronously causes a transaction manager to run it immediately, and then to return a status to the client program.

Transaction managers are implemented as immediate concurrent programs. At runtime, concurrent processing starts a number of these managers. Rather than polling the concurrent requests table to determine what to do, a transaction manager waits to be signalled by a client program. The execution of the requested transaction program takes place on the server, transparent to the client and with minimal time delay. At the end of program execution, the client program is notified of the outcome by a completion message and a set of return values.

Communication with a transaction manager is automatic. The transaction manager mechanism does not establish an ongoing connection between the client and the transaction manager processes. The intent of the mechanism is for a small pool of server processes to service a large number of clients with real-time response.

Each transaction manager can process only the programs contained in its program library. Oracle Applications developers using *Oracle Application Object Library* can register transaction programs with a program library.

Related Topics

Administer Concurrent Managers, page 7-49

Concurrent Managers, page 7-58

Work Shift Definitions, page 7-3

Using Work Shifts to Balance Processing Workload, page 7-5

Using Time-Based Queues, page 7-7

Work Shift by Manager Report, page 7-16

Work Shifts Report, page 7-16

Work Shifts, page 7-65

Overview of Concurrent Processing, *Oracle Applications System Administrator's Guide - Maintenance*

Work Shift Definitions

When you define a concurrent manager, you assign one or more work shifts to it. Work shifts determine when the manager operates. You define work shifts using the Work Shifts form.

Disabling a work shift

If you define a period of time as a work shift, but do not necessarily want to use the work shift, you can:

- Not assign the work shift to a concurrent manager
- Assign the number of target processes for the work shift as zero (0), on the Define Manager form.
- Delete a work shift assignment using the Define Manager form.

Work Shifts and Hours of the Day

Work shifts can run twenty-four hours a day, from midnight till the next midnight. In military time this is defined as:

- 12:00am - 00:00:00
- 11:59:59pm - 23:59:59

Using work shifts to run through midnight

The military time clock for a twenty-four period starts and stops at midnight.

If you do not want a work shift to run twenty-four hours a day, but you do want to run programs continuously past 12:00 am, you must define two work shifts:

- The first work shift stops at 23:59 (11:59pm).
- The second work shift starts at 00:00 (12:00 am).

For example, you want to run some data-intensive programs during the night, when most of your employees are away from the job site. You define two work shifts which you assign to this manager.

- The first work shift starts at 20:00 (8:00pm) and stops at 23:59 (11:59pm).
- The second work shift starts at 00:00 (12:00am) and stops at 05:00 (5:00am).

Overlapping Work Shifts - Priority Levels

If you assign overlapping work shifts to a concurrent manager, the work shift with the **more specific time period** takes effect for the overlapping time period. For example, a work shift for July 4 overrides a work shift from 9:00 am to 5:00 pm on Monday through Friday.

The following table presents a descending list of priority levels for overlapping work shifts. A work shift with a specific date and range of times has the highest priority. The "Standard" work shift has the lowest priority.

Priority	Work Shift Definition	Example
1	Specific date and range of times	April 15, 2000 8:00am-5:00pm
2	Specific date and no range of times	April 15, 2000
3	Range of days and range of times	Monday-Friday 8:00am-5:00pm
4	Range of days and no range of times	Monday-Friday
5	Range of times and no date and no range of days	8:00am-5:00pm
6	Standard work shift. No date, days, or time defined.	Standard work shift is 365 days a year, 24 hours a day.

Overlapping Work Shifts with the same priority

When you have overlapping work shifts that have the same level of priority, the work shift with the **largest target processes** takes effect.

For example, you have two work shifts with a range of days and a range of times. You have a "Weekday" work shift from 9:00 am to 5:00 pm on Monday through Friday with 4 target processes.

You also have a "Lunch" work shift from 11:00 am to 1:00 pm on Monday through Friday with 8 target processes.

The "Lunch" work shift takes effect from 11:00 am to 1:00 pm (Mon.-Fri.) because it has the larger number of target processes.

Related Topics

Defining Managers and their Work Shifts, page 7-1

Using Work Shifts to Balance Processing Workload, page 7-5

Using Time-Based Queues, page 7-7

Using Work Shifts to Balance Processing Workload

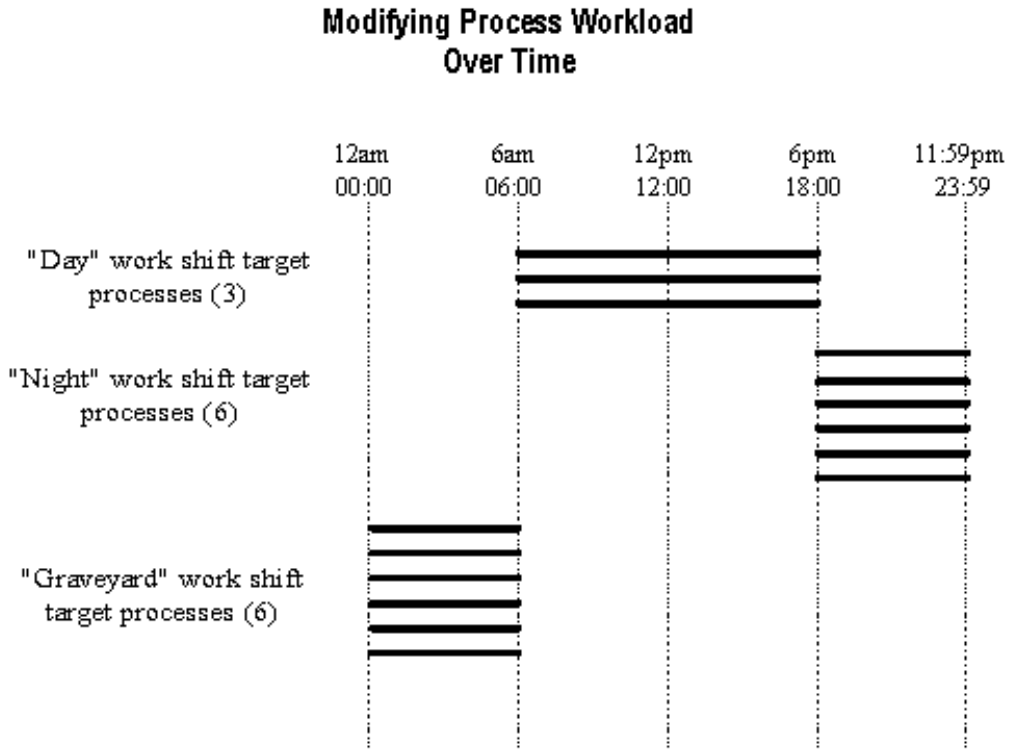
Part of a manager's definition is how many operating system processes it can devote to reading requests. For each of these processes, referred to as a *target process*, a manager

can start one concurrent program.

For each work shift you assign to a manager, you define a number of target processes.

By using work shifts with different numbers of target processes, you can modify your concurrent processing workload according to the day, time of day, and even specific dates.

The figure below illustrates how, by using three work shifts, a manager can be defined to run three programs concurrently from 6:00am-6:00pm, and six programs concurrently from 6:00pm-6:00am.



Failover Sensitivity for Work Shifts

Nodes can become overloaded when a middle-tier node fails and service instances on that node failover to their secondary nodes. The *Load Distribution* feature allows the System Administrator to control the allocation of resources during normal processing. The *Failover Sensitivity* feature allows Work Shifts to failover with fewer target processes than on the original node. This lessens the impact on the existing resources allocated on the secondary node.

The number of failover target processes is entered as part of the standard Work Shift settings in the *Service Instance Definition*. When failover occurs, the ICM uses the **Failover Processes** value in place of the normal running processes value as it iterates through service instances to perform queue sizing.

Related Topics

Defining Managers and their Work Shifts, page 7-1

Work Shift Definitions, page 7-3

Using Time-Based Queues, page 7-7

Work Shift by Manager Report, page 7-16

Work Shifts Report, page 7-16

Work Shifts, page 7-65

Using Time-Based Queues

You can create several time-based queues by defining managers to run programs based on how long those programs have typically run in the past. That is, you can specialize managers to segregate requests according to how long those requests take to run.

To do this, use the Completed Concurrent Requests Report in the System Administrator's report security group. This report lists the actual start date and time and actual completion date and time for concurrent programs that completed running. See: Completed Concurrent Requests Report, page 7-15.

> **Tip:** Run your concurrent programs at different times, perhaps, late at night and then again during the midafternoon, to determine processing time during different workload periods.

For example, based on actual time-to-completion, you can specialize different managers to run the following types of programs:

- inventory pick lists
- payable check runs
- postings
- invoice imports

Augment this approach by defining an "overflow" manager, for example, a manager who can accommodate programs directed to one (or more) of the managers above, but whose work shift is restricted to say, 2:00am-4:00am (02:00-04:00). If some of your long-running programs have not started running before the "overflow" work shift begins, then an additional manager is enabled to accommodate those programs.

Further augment this approach with an "exception" manager defined for *must have* requests. For example, a manager that can run:

- certain programs that must complete by a certain time. The "must-have" manager can be specialized to only read requests for certain programs.

- programs submitted by a particular user, for example, the Company Controller. You can specialize a manager to only read requests from a single application user. You can even define a second, higher-priority, username for a user to sign on with.

Related Topics

Defining Managers and their Work Shifts, page 7-1

Work Shift Definitions, page 7-3

Using Work Shifts to Balance Processing Workload, page 7-5

Work Shift by Manager Report, page 7-16

Work Shifts Report, page 7-16

Specializing Managers to run only certain programs, page 7-17

Grouping Programs by Request Type, page 7-33

Administer Concurrent Managers, page 7-49

Concurrent Managers, page 7-58

Work Shifts, page 7-65

Creating Services within Oracle Applications Manager

Creating and Editing a Concurrent Manager

Use this page to create a new concurrent manager.

Navigation: Site Map > Request Processing Managers (under Application Services) > (B) Create New or (B) Edit

You can define when a manager runs and how many programs the manager can start simultaneously when you assign work shifts to the manager. Specify which programs a manager can start by defining specialization rules.

General

Enter the following information:

Enabled

Check this box if the manager is enabled.

Manager

The name of the manager.

Short Name

The short name of the manager.

Application

The application name does not prevent a manager from starting programs associated with other applications. To restrict a manager to only running programs associated with certain applications, go to the Rules section.

The combination of an application and the name you define for your manager uniquely identifies the manager.

Cache Size

Enter the number of requests your manager remembers each time it reads which requests to run. For example, if a manager's work shift has 1 target process and a cache value of 3, it will read three requests, and try to run those three requests before reading any new requests.

> **Tip:** Enter a value of 1 when defining a manager that runs long, time-consuming jobs, and a value of 3 or 4 for managers that run small, quick jobs.

Program Library

Select a library of immediate concurrent programs to make available to your manager. Your manager can only run immediate concurrent programs that are registered in the selected program library. Concurrent managers can run only those immediate concurrent programs listed in their program library. They can also run concurrent programs that use any other type of concurrent program executable.

Resources Group

Optionally enter the resource consumer group for this manager.

Rules

Use the Rules section to specialize your manager to run only certain kinds of requests. Without specialization rules, a manager accepts requests to start any concurrent program.

A listing of available rules is displayed. Check the **Include** check box for a rule to include it.

The following information is also given for each rule:

- Type
- Application

- Name
- Description

To edit any of this information, use the **Edit** button. Use the **Remove** button to remove a rule from the list. To create a new rule, use the **Create New** dropdown list and click **Go**.

Work Shifts

Use the Work Shifts section to assign work shifts to your manager. A work shift defines the dates and times the manager is enabled, as well as the number of processes the manager can start running during the work shift.

To add a work shift, use the **Add from Available Shifts** button.

For each work shift listed, the following is displayed:

Sleep Seconds

The sleep time for your manager during this work shift. Sleep time is the number of seconds your manager waits between checking the list of pending concurrent requests (concurrent requests waiting to be started).

> **Tip:** Set the sleep time to be very brief during periods when the number of requests submitted is expected to be high.

Processes

The number of operating system processes you want your work shift to run simultaneously. Each process can run a concurrent request.

For example, if a work shift is defined with three (3) target processes, the manager can run up to three requests simultaneously.

Failover Processes

In the case of node failover, the maximum number of processes that the work shift can run simultaneously.

Nodes can become overloaded when a middle-tier node fails and service instances on that node failover to their secondary nodes. The Failover Processes value should be smaller than the normal Processes value, to lessen the impact on the existing resources allocated on a secondary node. When failover occurs, the ICM uses the Failover Processes value in place of the normal running processes value as it iterates through service instances to perform queue sizing.

Nodes

If you are operating in a parallel concurrent processing environment and you want your manager to operate on a specific node, select the name of the node.

The primary node, if available, is the node your concurrent manager operates on. If the primary node or the database instance on it goes down, your concurrent manager migrates to its secondary node. Your concurrent manager migrates back to its primary node when that node becomes available.

Nodes must be previously registered with Oracle Applications, using the Nodes form in Oracle Applications.

Creating and Editing a Transaction Manager

Use this page to create a new Transaction Manager. Transaction Managers handle synchronous requests from client machines.

Navigation: Site Map > Transaction Managers (under Application Services) > (B) Create New or (B) Edit

General

Enter the following information:

Enabled

Check this box if this transaction manager is enabled.

Manager

The name of the transaction manager.

Short Name

The short name for your transaction manager.

Application

The application associated with the transaction manager.

The combination of an application and the short name you specify here uniquely defines the transaction manager.

Program Library

Select a library of immediate transaction programs to make available to your manager. Your manager can only run immediate transaction programs that are registered in the selected program library. Transaction managers can run only those immediate transaction programs listed in their program library. They can also run transaction programs that use any other type of transaction program executable.

Work Shifts

Use the Work Shifts section to assign work shifts to your manager. A work shift defines the dates and times the manager is enabled, as well as the number of processes the manager can start running during the work shift.

To add a work shift, use the **Add from Available Shifts** button.

For each work shift listed, the following is displayed:

Sleep Seconds

The sleep time for a transaction manager determines how often a manager will check to see if it should shut down.

> **Tip:** Set the sleep time to be very brief during periods when the number of requests submitted is expected to be high.

Processes

The number of operating system processes you want your work shift to run simultaneously. Each process can run a concurrent request.

For example, if a work shift is defined with three (3) target processes, the manager can run up to three requests simultaneously.

Nodes

If you are operating in a parallel concurrent processing environment and you want your manager to operate on a specific node, select the name of the node.

The primary node, if available, is the node your concurrent manager operates on. If the primary node or the database instance on it goes down, your concurrent manager migrates to its secondary node. Your concurrent manager migrates back to its primary node when that node becomes available.

Nodes must be previously registered with Oracle Applications, using the Nodes form in Oracle Applications.

Creating and Editing an Internal Monitor

Use this page to create a new Internal Monitor.

Internal Monitors monitor the Internal Concurrent Manager in a parallel concurrent processing environment. If the Internal Concurrent Manager exits abnormally (for example, because its node or its database instance goes down), an Internal Monitor restarts it on another node.

General

Enter the following information:

Enabled

Check this box if this internal monitor is enabled.

Manager

The name of the internal monitor.

Short Name

The short name for your internal monitor.

Application

The application associated with the internal monitor.

The combination of an application and the short name you define for your internal monitor uniquely identifies the monitor.

Program Library

For an Internal Monitor, the program library is FNDIMON.

Work Shifts

Use the Work Shifts section to assign work shifts. A work shift defines the dates and times the manager is enabled, as well as the number of processes the manager can start running during the work shift.

To add a work shift, use the **Add from Available Shifts** button.

For each work shift listed, the following is displayed:

Sleep Seconds

The sleep time for your manager during this work shift. Sleep time is the number of seconds your manager waits between checking the list of pending concurrent requests (concurrent requests waiting to be started).

> **Tip:** Set the sleep time to be very brief during periods when the number of requests submitted is expected to be high.

Processes

The number of operating system processes you want your work shift to run simultaneously. Each process can run a concurrent request.

For example, if a work shift is defined with three (3) target processes, the manager can run up to three requests simultaneously.

Nodes

If you are operating in a parallel concurrent processing environment and you want your manager to operate on a specific node, select the name of the node.

The primary node, if available, is the node your concurrent manager operates on. If the

primary node or the database instance on it goes down, your concurrent manager migrates to its secondary node. Your concurrent manager migrates back to its primary node when that node becomes available.

Nodes must be previously registered with Oracle Applications, using the Nodes form in Oracle Applications.

Create a New Work Shift

Use this page to define work shifts for your services. Define work shifts to specify when your services can work.

Navigation: Site Map - Administration > [Service Instance type] (under Application Services) > Create [Service] > Workshifts region, (B) Add from Available Shifts > (B) Create New Site Map > Work Shift Library (under Application Services) > Create New

- Name - The name of your work shift should be intuitive, for instance "Week Days", "Weeknights" or "Weekends".
- Description - Add a description for your work shift.
- Schedule - For each work shift, specify a time period covering a range of days or a particular date. Specify if you are scheduling by day or by date.
- Day - Enter the first and last days of this shift. For instance, if your shift name is "Week Days", you could enter "Monday" in the "Days of Week From" field and "Friday" in the "Days of Week To" field. If you enter a value in the "Days of Week From" field, you must enter a value in the "Days of Week To field".
- Date - Enter a date here to create a date-specific work shift. Date-specific work shifts override work shifts that do not specify a specific date. If you want to enter a value in this field (specify a date), you may not enter values for the Days of Week fields for this row.
- Time - Enter the times of day at which your concurrent shift begins/ends. The time format is HH24:MM. For example, if your work shift name is "Week Days", you could enter "09:00" (9:00 am) as the start time and "17:00" (5:00 pm) as the end time. Note that Oracle Applications uses a 24-hour clock.

List of Work Shifts

This page displays the available work shifts.

Navigation: Site Map - Administration > Request Processing Managers (under Applications Systems) > Create New or Edit [Service] > Workshifts region, (B) Add from Available Shifts

The following columns are shown: Name, Start Day, End Day, Start Time, End Time, Date, and Description.

You can use the buttons to edit or delete an existing work shift, or create a new one.

Completed Concurrent Requests Report

This report displays how long concurrent programs actually run. Use this report to segregate requests, based on their typical time-to-complete, by specializing concurrent managers to only read requests for certain programs.

Use this report to record parameters and error messages associated with concurrent programs that have been run.

Report Parameters

If you do not enter any parameters, the report returns values for all completed concurrent requests.

Program Application Name

Choose the application name associated with the program whose completed concurrent requests you wish to report on.

Choose only an application name, without a program name, if you wish to run a report on all completed concurrent requests associated with an application.

Program Name

Choose the name of a program whose completed concurrent requests you wish to report on. You must enter a value for Program Application Name before entering a value for Program Name.

User Name

Choose the name of an application user whose completed concurrent requests you wish to report on.

Start Date/End Date

Enter the start date and end date for your report.

Report Headings

The report headings list the specified parameters and provide you with general information about the contents of the report.

Related Topics

Reviewing Requests, Request Log Files, and Report Output Files, *Oracle Applications System Administrator's Guide - Maintenance*

Managing Concurrent Processing Files and Tables, *Oracle Applications System Administrator's Guide - Maintenance*

Using Time-Based Queues, page 7-7

Specializing Managers to run only certain programs, page 7-17

Grouping Programs by Request Type, page 7-33

Work Shift by Manager Report

This report documents the work shifts assigned to each concurrent manager. Use the report when defining or editing concurrent managers.

Report Parameters

None.

Report Headings

The report headings provide you with general information about the contents of the report.

Related Topics

Defining Managers and their Work Shifts, page 7-1

Work Shifts Report, page 7-16

Work Shifts Report

This report documents all of your work shift definitions. Use this report when defining or editing concurrent manager work shifts.

Report Parameters

None.

Report Headings

The report headings provide you with general information about the contents of the report.

Related Topics

Defining Managers and their Work Shifts, page 7-1

Work Shift Definitions, page 7-3

Work Shifts Report, page 7-16

Work Shift by Manager Report, page 7-16

Work Shifts field help, page 7-65

Specializing Managers to Run Only Certain Programs

This essay explains how you can specialize managers to run only certain programs.

Introduction to Specialization Rules

Every time your users request a concurrent program to be run, their request is inserted into a database table. Concurrent managers read requests from this table, and start running programs if the manager is defined to read the particular request.

Without specialization rules, a manager reads requests to start *any* concurrent program.

Using specialization rules, you can specialize a manager to read only certain kinds of requests to start concurrent programs, for example, only requests to start Oracle General Ledger programs, or only requests to start programs requested by the user "Fred". See: Concurrent Managers, page 7-58.

A special type of specialization rule is the *combined* specialization rule, that can combine more than one action to define a single rule. See: Combined Specialization Rules, page 7-67.

Related Topics

Defining Specialization Rules, page 7-17

Examples - Using Specialization Rules, page 7-22

Defining Combined Specialization Rules, page 7-27

Controlling Concurrent Managers, page 7-35

Concurrent Managers field help, page 7-58

Defining Specialization Rules

A specialization rule associates an action with a type of request. There are two kinds of actions: Include and Exclude.

- Include defines a manager to only read requests of the type specified.
- Exclude defines a manager to read all requests except the type specified.

Requests to run concurrent programs may be allowed or disallowed on the basis of:

- the ORACLE ID of the request's Set of Books (for multiple installs) or Organization if you are using multiple organizations.

- the program itself or the program's application
- the request type of the program
- the user who submitted the request
- a combined rule, which combines more than one action to generate a single rule. The combined rule applies its actions to one or more types of request.

  For example, a combined rule can exclude an action from an Oracle ID and exclude another action from a specific program.

Using more than one rule

Each rule performs one action. When using more than one rule, the rules are evaluated as follows:

- Include rules are evaluated together using 'OR' statements as the binding logic.

  For example, If you use the rules:

  - Include X
  - Include Y

    The result of the rules allows the manager to run either X 'OR' Y but does not require that both programs be run.

- Exclude rules are evaluated together using 'AND' statements as the binding logic.

  For example, If you use the rules:

  - Exclude 1
  - Exclude 2

    The result of the rules prohibits the manager from running programs 1 'AND' 2 together or separately.

- Include rules are evaluated first, then Exclude rules are evaluated. Include rule(s) and Exclude rule(s) are evaluated together as an AND statement. For example, (Include X OR Y) AND (Exclude 1 AND 2).
- An Exclude rule overrides an Include rule.

Specialization rule actions, their binding logic, and examples are presented in the following two tables. See: Specialization Rule Logic - Examples, page 7-18.

Specialization Rule Logic - Examples

Include Rules	Result
Include *X*	Run only program X
Include *X*	Run program X
OR	...or
Include *User Sam*	Run requests by User Sam
	Net result: Run everyone's requests for program X, and run all of Sam's requests.

Exclude Rules	Result
Exclude *37*	Do not run program 37
Exclude *37*	Do not run program 37
AND	...and
Exclude *User Sam*	Do not run requests by User Sam
	Net result: Do not run anyone's requests for program 37, and do not run any of Sam's requests.

Include and Exclude Rules	Result
Include *User Sam*	Run only requests by User Sam
AND	...and
Exclude *37*	Do not run program 37
	Net result: Run all of Sam's requests except requests to run program 37.
Include *X*	(Run program X

Include and Exclude Rules	Result
OR	...or
Include *User Sam*	Run requests by User Sam)
---------- AND	...and
Exclude *37*	(Do not run program 37
AND	...and
Exclude *User Mary*	Do not run requests by User Mary)
	Net result: Run program X except when requested by Mary, and run all of Sam's requests except requests to run program 37.

The following table gives examples of the action types associated with specialization rules.

Rule Action	Type	Example	Explanation
INCLUDE	Combined Rule	Oracle Project Accounting - Tim's Budgets	Manager *only* reads requests to start programs defined by the Combined Rule "Tim's Budgets".
INCLUDE	ORACLE ID	APPS2	Manager *only* reads requests to start programs that connect to the APPS2 (a single install in a multiple install schema) Oracle ID.
INCLUDE	Program	Oracle Project Accounting - Sales Forecast	Manager *only* reads requests to start the concurrent program named "Sales Forecast".

Rule Action	Type	Example	Explanation
INCLUDE	Request Type	Oracle Inventory - Overnight Reports	Manager *only* reads requests to start programs belonging to the request type "Overnight Reports".
INCLUDE	User	Tim	Manager *only* reads requests to start programs submitted by the application user "Tim".
EXCLUDE	Combined Rule	Oracle General Ledger - Month End Reports	Manager reads *all* requests to start programs *except* those defined by the Combined Rule "Month End Reports".
EXCLUDE	ORACLE ID	APPS2	Manager reads *all* requests to start programs *except* those that connect to the APPS2 Oracle ID.
EXCLUDE	Program	Application Object Library - Purge Audit Tables	Manager reads *all* requests to start programs *except* requests for the program named "Purge Audit Tables".
EXCLUDE	Request Type	Oracle Purchasing - Weekend Programs	Manager reads *all* requests to start programs *except* those belonging to the request type "Weekend Programs".

Rule Action	Type	Example	Explanation
EXCLUDE	User	Margaret	Manager reads *all* requests to start programs *except* those submitted by the application user "Margaret".

Related Topics

Specializing Managers to run only certain programs, page 7-17

Examples - Using Specialization Rules, page 7-22

Defining Combined Specialization Rules, page 7-27

Using Combined Rules, page 7-29

Differences Between Specialization and Combined Rules, page 7-32

Concurrent Managers field help, page 7-58

Combined Specialization Rules field help, page 7-67

Examples - Using Specialization Rules

Following are examples of using specialization rules to define what requests a concurrent manager can read. When multiple rules are used to specialize a manager, the words OR and AND appear between each rule to clarify the relationship among multiple specialization rules.

Using Include and Exclude actions

Include Program - Oracle Assets, *No entry for Name field*.

Result The manager only reads requests to run concurrent programs for the application "Oracle Assets".

Include Program - Oracle Assets, *No entry for Name field*.

OR

Include Program - Oracle Payables, *No entry for Name field*.

Net Result The manager only reads requests to run concurrent programs for the application "Oracle Assets", or for the application "Oracle Payables".

The use of multiple Include actions expands the manager's ability to read requests beyond that of a single Program (single Include action).

Exclude Oracle ID - APPS2

Result The manager reads requests to run concurrent programs that connect to any Oracle ID, except those programs that connect to Oracle ID "APPS2".

Exclude Oracle ID - APPS2

AND

Exclude Program - Oracle Payables, *No entry for Name field*.

Net Result The manager reads requests to run concurrent programs that connect to any Oracle ID, except programs that connect to Oracle ID "APPS2", and programs for the application "Oracle Payables".

Simplifying your work

Multiple rules may not always be necessary, or the number or complexity of rules can be simplified. Consider the example below.

Include Program - Oracle Sales and Marketing, *No entry for Name field*.

OR

Include Request Type - Sales Forecasts

Net Result The manager only reads requests to run concurrent programs for the application "Oracle Sales and Marketing", or programs whose request type is "Sales Forecasts".

In this example, both rules are not necessary when programs belonging to the request type "Sales Forecasts" all connect to the Oracle ID "OSM". There is no need for the second Type Include rule.

Exclude rules override Include rules

Include Program - Oracle Payables, *No entry for Name field*.

AND

Exclude Program - Oracle Payables Invoice Aging Report

Net Result	The manager reads all requests for concurrent programs for the application "Oracle Payables", but does not read requests to run the Oracle Payables program "Invoice Aging Report".
Include	Program - Signon Audit Forms
AND	
Exclude	Request Type - Signon Audit Reports
Net Result	If the System Administrator program Signon Audit Forms belongs to the Request Type "Signon Audit Reports", the manager will not read requests to run the program, even though it has been specifically identified by an Include rule. The Exclude rule overrides the Include rule.

Specializing to only run a Program against specific Oracle IDs

In the following example, a manager can be specialized to only run a program against a specific Oracle ID. This is useful when there are multiple installations of an Oracle Application.

Include	Program - Oracle Payables Invoice Aging Report
AND	
Exclude	Oracle ID - APPS2
Net Result	The manager only reads requests to run the Oracle Payables program "Invoice Aging Report" when the program does not connect to the Oracle ID "APPS2". The Exclude action overrides the Include action. However, when the Invoice Aging Report runs against another Oracle ID, for example, "APPS", then this manager will read requests to run the program.

Distinguishing a Program from a Request Type

You can specialize a manager to read requests to run all the programs belonging to a Request Type, except for individual programs you wish to identify.

Include	Request Type - Oracle General Ledger Reports
AND	
Exclude	Program - Oracle General Ledger Account Analysis
Net Result	If the Account Analysis program belongs to the request

type Oracle General Ledger "Reports", then this manager will run every program in the request type Oracle General Ledger Reports, except the program Account Analysis.

Preventing specific programs from running

You can use an Exclude action more than once. For example, suppose your manager reads all requests to run concurrent programs for a particular application, but you want to prevent your manager from running two specific programs. You can:

Include	Program - Oracle General Ledger, N*o entry for Name field*.
AND	
Exclude	Program - Oracle General Ledger Consolidation Audit
AND	
Exclude	Program - Oracle General Ledger Consolidation Rules
Net Result	The manager reads requests for any concurrent programs for the application "Oracle General Ledger", except for the programs "Consolidation Audit" and "Consolidation Rules".

Specializing to run only specific programs at certain times

Using multiple Include rules, you can specialize a manager to run only specific programs. Then, when you define the manager's work shift, you can control when the manager reads requests to run the specific programs.

Include	Program - Oracle Payables Invoice Aging Report
OR	
Include	Program - Oracle Purchasing Receipt Accruals
Net Result	The manager only reads requests to run the Oracle Payables Invoice Aging Report, or the Oracle Purchasing Receipt Accruals program.

Tip: If you only wanted these two reports run during the night you can define the manager's work shift to run from 2:00am-6:00am (02:00-06:00).

Tip: When you first submit the requests to

run the programs, you can define a resubmission interval, for example, 1 month, to resubmit the programs to run every month.

Specializing according to Application User

You can specialize managers to only read requests from specific users.

Include	User - Markus Kalkin
Net Result	The manager only reads requests submitted by the application user "Markus Kalkin".
Include	User - Markus Kalkin

OR

Include	Program - Oracle Inventory Process Demand Interface

OR

Include	Program - Oracle Inventory Summarize Demand Histories
Net Result	The manager reads both requests submitted by user Markus Kalkin and requests to run the Oracle Inventory programs "Process Demand Interface" and "Summarize Demand Histories".

Tip: If you want specific programs submitted by a specific user to "jump ahead" of other requests waiting to be run, you can define and specialize a manager as in the example above, and set the user profile option Concurrent:Priority for the user to a high priority (Concurrent:Priority sets the priority of requests submitted by the user).

- Define a manager and give it a descriptive name.
- Specialize the manager as in the example above.
- Set the user profile option

Concurrent:Priority for user Markus to 10.

Related Topics

Specializing Managers to run only certain programs, page 7-17

Defining Specialization Rules, page 7-17

Defining Combined Specialization Rules, page 7-27

Concurrent Managers field help, page 7-58

Defining Combined Specialization Rules

A combined specialization rule combines more than one action to generate a single rule. The actions are combined as AND statements so that the rule is defined as:

Action 1 AND . . .

Action 2 AND . . .

Action 3 AND . . . so on.

You can create combined rules and use them with several managers, instead of duplicating a complex rule each time.

There are two kinds of Actions you may use to build a combined rule; Exclude and Include. Each action is defined by one line within the rule. Combining the specialization lines or individual actions defines the overall combined rule.

An Exclude action overrides a Include action.

For example, you can define an Exclude *application program x* action and a Include *user Yvonne Jones* action. Combining these two actions generates the combined rule "read all requests from user Yvonne Jones except requests to run program x". See: Combined Specialization Rules, page 7-67.

Combined specialization rule actions, their binding logic, and examples are presented in the following table.

Combination Rule Include Lines	Result
Include *Program X*	Run only program X
Include *Program X*	Run program X
AND	...and

Combination Rule Include Lines	Result
Include *User Sam*	Run requests by User Sam Net result: Run only Sam's requests for program X.

Combination Rule Exclude Lines	Result
Exclude *Program 37*	Do not run program 37
Exclude *Program 37*	Do not run program 37
AND	...and
Exclude *User Sam*	Do not run requests by User Sam Net result: Do not run anyone's requests for program 37, and do not run Sam's requests.

Combination Rule Include and Exclude Lines	Result
Include*User Sam*	Run requests by User Sam
AND	...and
Exclude *Program 37*	Do not run program 37 Net result: Run all of Sam's requests except requests to run program 37.
Include *Program Application General Ledger*	(Run General Ledger Programs
AND	...and
Include*User Sam*	Run requests by User Sam)
---------- AND	...and

Combination Rule Include and Exclude Lines	Result
Exclude *Program 37*	(Do not run program 37
AND	...and
Exclude *Program 38*	Do not run program 38)
	Net result: Run Sam's requests for programs from the application General Ledger, except programs 37 and 38.

Related Topics

Specializing Managers to run only certain programs, page 7-17

Defining Specialization Rules, page 7-17

Using Combined Rules, page 7-29

Differences Between Specialization and Combined Rules, page 7-32

Concurrent Managers field help, page 7-58

Combined Specialization Rules field help, page 7-67

Using Combined Rules

Using combined rules you can precisely specialize a manager.

A combined rule combines more than one action to generate a single rule. Each action is defined by one line within the rule. Combining the lines or individual actions defines the overall combined rule.

> **Tip:** You can use a combined specialization rule as one of many rules to specialize a manager.

Using single Exclude and Include actions

A single Exclude action within a combined rule acts the same way as a single Exclude action that defines a specialization rule. Both instruct a manager to read *all* requests to run concurrent programs *except* those identified by the action.

Exclude Oracle ID - APPS

Result The manager reads requests to run concurrent programs

that connect to any Oracle ID, except those programs that connect to Oracle ID "APPS".

A single Include action within a combined rule acts the same way as a single Include action that defines a specialization rule. Both actions instruct a manager to read *only* the requests that satisfy the action.

Include Oracle ID - APPS2

Result The manager only reads requests to run concurrent programs that connect to Oracle ID "APPS2".

Using multiple Exclude actions

Using multiple Exclude actions as multiple lines within a combined rule is equivalent to using multiple Exclude actions as multiple specialization rules.

You can exclude more kinds of requests by adding more Exclude lines to your combined rule.

Exclude Program - Oracle Sales & Marketing, *No entry for Name field*.

AND

Exclude Program - Oracle Inventory, *No entry for Name field*.

Net Result The manager reads all requests to run concurrent programs *except requests for* programs for the application "Oracle Sales & Marketing", *and requests for* programs for the application "Oracle Inventory".

Using multiple Include actions

Using multiple Include actions adds more requirements to a combined rule, and excludes more kinds of requests.

You cannot use two Include actions for the same action type. Each Include action is an exclusive statement for a particular type of action. For example, you cannot require a request to be for a program that connects to two different Oracle IDs.

Include Program - Oracle Payables, *No entry for Name field*.

AND

Include Program - Oracle Payables Confirm Receipt Batch

Net Result The manager only reads requests to run a single program, *Confirm Receipt Batch*, and only if that program is from the application "Oracle Payables".

Using Exclude and Include actions

You cannot use Exclude and Include actions for the same type of action. Each Include action is an exclusive statement for a particular type of action.

For example, it does not make sense to *require* a request to be for a program that connects to the Oracle ID "APPS" and disallow a request to connect to another Oracle ID.

Exclude overrides Include

When using multiple lines within a Combined Rule, the Exclude action always overrides a Include action.

Include	Program - Oracle Payables Invoice Import
AND	
Exclude	Oracle ID - APPS2
Net Result	The manager reads requests to run the Oracle Payables Invoice Import program, but will not run the program when it connects to the Oracle ID "APPS2". The Exclude action overrides the Include action.

Specializing a manager to run one program submitted by one user

You can define a combined rule that instructs a manager to only read requests to run a single program when submitted by a specific user.

Include	User - Sheryl
AND	
Include	Program - Oracle Project Accounting Distribute Usage Costs
Net Result	The manager only reads requests submitted by Sheryl to run the Distribute Usage Costs program.

Restricting the programs a manager will run for a specific user

You can define a combined rule that instructs a manager to ignore requests to run a certain programs when submitted by a specific user.

Include	User - Sheryl
AND	
Exclude	Program - Oracle Project Accounting Expenditure Status

Net Result	The manager only reads requests submitted by Sheryl, excluding requests to run the Oracle Project Accounting program *Accounting Expenditure Status*.

Specifying Oracle ID and excluding a program from a request type

Include	Request Type - Oracle Project Accounting Expenditure Reports
AND	
Include	Oracle ID - APPS2
AND	
Exclude	Program - Oracle Project Accounting Expenditure Status
Net Result	The manager only reads requests to run programs belonging to the Oracle Project Accounting request type " Reports", run against the Oracle ID "APPS2", excluding the program *Expenditure Reports*.

Related Topics

Specializing Managers to run only certain programs, page 7-17

Defining Combined Specialization Rules, page 7-27

Differences Between Specialization and Combined Rules, page 7-32

Concurrent Managers field help, page 7-58

Combined Specialization Rules field help, page 7-67

Differences Between Specialization and Combined Rules

The primary difference between a specialization rule and a combined specialization rule is in how the use of multiple actions affects the outcome of the rule, as described in the following table:

Rule	Action	Effect of Multiple Actions	Relationship to Other Rules
Specialization Rule	INCLUDE	With each additional Include rule, the manager can read MORE REQUESTS.	Each rule establishes an OR condition. *OR...INCLUDE...*

Rule	Action	Effect of Multiple Actions	Relationship to Other Rules
Specialization Rule	EXCLUDE	With each additional Exclude rule, the manager is excluded from, and reads, FEWER REQUESTS.	Each rule establishes an AND condition. *AND...EXCLUDE...*
Combined Rule Specialization Line	EXCLUDE	With each additional Exclude line, the manager is excluded from, and reads, FEWER REQUESTS.	Each line within a rule establishes an AND condition. *AND...EXCLUDE...*
Combined Rule Specialization Line	INCLUDE	With each additional Include line or additional requirement, the manager reads FEWER REQUESTS.	Each line within a rule establishes an AND condition. *AND...INCLUDE...*

Related Topics

Specializing Managers to run only certain programs, page 7-17

Examples - Using Specialization Rules, page 7-22

Defining Combined Specialization Rules, page 7-27

Using Combined Rules, page 7-29

Concurrent Managers field help, page 7-58

Combined Specialization Rules field help, page 7-67

Grouping Programs by Request Type

As System Administrator, you may want to group similar programs together. You do this by defining request types and assigning them to the programs that users request in Oracle Applications. You can define concurrent managers that only run programs that belong to a particular request type.

Using request types to specialize concurrent managers can help optimize the processing of Oracle Applications, by letting certain types of programs run without having to wait for other types of programs to finish processing. Using request types saves you time when you create a concurrent manager's specialization rules.

Using Request Types

Specializing a concurrent manage by request type involves three steps:

1. Define a Request Type using the Concurrent Request Types form.
2. Assign the Request Type to each concurrent program you want to identify as a member of this request type using the Concurrent Programs form.
3. Select the Request Type when you specialize a concurrent manager using the Concurrent Managers form.

Examples of using Request Types

Some example request types you may want to define are:

Quick
For concurrent programs that take a relatively short time to run.

Overnight
For concurrent programs that take a long time to run, which you typically schedule to run during the late night or early morning hours.

Month-End Reports
For concurrent programs that generate reports you run at the end of each month.

For example, if you run ten report programs at the end of each month, you could define a request type called " Month-End Reports" and assign it to your ten report programs.

Then you can use specialization rules to define a concurrent manager that only runs requests of type " Month-End Reports". This way, you do not have to specify your ten different report programs when you define your concurrent manager. You can also easily assign the ten programs to more than one manager.

Related Topics

Overview of Concurrent Processing, *Oracle Applications System Administrator's Guide - Maintenance*

Using Time-Based Queues, page 7-7

Completed Concurrent Requests Report, page 7-15

Specializing Managers to run only certain programs, page 7-17

Administer Concurrent Managers, page 7-49

Concurrent Managers, page 7-58

Controlling Concurrent Managers

This essay explains how to control your concurrent managers.

Manager States

Individual managers read requests to start concurrent programs and actually start programs running when certain conditions are satisfied, such as the manager's work shift definition, number of target processes, and specialization rules.

You can start, shut down, or reset the concurrent managers at any time. Oracle Applications provides an Internal Concurrent Manager that processes these commands. You can issue commands either to individual managers, or, by altering the state of the Internal Concurrent Manager, you can control every manager at once.

Starting Individual Managers

You can restart or activate managers on an individual basis. Restarting a concurrent manager forces the Internal Concurrent Manager to reread the definition for that concurrent manager. Activating a manager cancels a previous command to deactivate it, and allows the Internal Concurrent Manager to start that manager when its work shift starts.

You should restart an individual manager when you:

- modify its work shift assignments
- modify a work shift's target number of processes
- modify its specialization rules
- change a concurrent program's incompatibility rules

Deactivating Individual Managers

When you shut down an individual manager, you can choose whether to abort all requests and deactivate the manager immediately, or to allow it to finish processing its current requests before deactivating.

If you choose to Deactivate the manager, requests that are currently running are allowed to complete.

When you terminate requests and deactivate an individual manager, requests that are currently running are immediately stopped and marked for resubmission (when the manager is activated).

Oracle Applications concurrent programs are designed so that no data is lost or duplicated when a terminated request is resumed after a shut down. This applies for shutdowns that are normal (e.g., using the "Deactivate concurrent manager" request) or abnormal (e.g., after a hardware failure).

> **Important:** When a manager is selected and explicitly deactivated, it remains that way until you select and explicitly activate that manager. As a prerequisite, the Internal manager must be activated beforehand.

Controlling the Internal Concurrent Manager

When you activate the Internal Concurrent Manager, you activate all other managers as well, except those managers that were deactivated on an individual basis.

When you deactivate the Internal Concurrent Manager, it issues commands to deactivate all active managers. Managers that were deactivated on an individual basis are not affected.

If you terminate requests and deactivate the Internal Concurrent Manager, it issues commands to all other managers to terminate their requests and deactivate. Requests that are currently running are immediately stopped and marked for resubmission when the managers are activated.

Verify Concurrent Manager Status

The Internal Concurrent Manager continuously monitors each concurrent manager's operating system process. This process monitoring is referred to as the Internal Concurrent Manager's PMON cycle. The length of the PMON cycle is one of the arguments passed by the STARTMGR command, which starts up the Internal Concurrent Manager.

You can instruct the Internal Concurrent Manager to immediately verify the operating status of your individual concurrent managers, or to perform a PMON check.

Controlling Managers from the Administer Managers form

Use the Administer Concurrent Managers form to issue commands to your concurrent managers.

You can also have the Internal Concurrent Manager "manually" verify the status of your individual managers, and restart individual managers. See: Administer Concurrent Managers, page 7-49.

The following table describes control functions for the Internal Manager.

Control Function	Description
Activate concurrent manager	Activates the Internal manager and all other managers, except managers that were deactivated individually using "Deactivate concurrent manager".
Verify concurrent manager status	Manually executes the process monitoring (PMON) cycle.
Deactivate concurrent manager	Deactivates the Internal manager and all other managers.
Terminate requests and deactivate manager	All running requests (running concurrent programs) are terminated, and all managers are deactivated.

The following table describes control functions for any other manager.

Control Function	Description
Activate concurrent manager	If the manager is defined to work in the current work shift, it starts immediately. Cancels "Deactivate concurrent manager" and "Terminate requests and deactivate manager".
Restart concurrent manager	Internal manager rereads the manager's definition, and the rules for concurrent program incompatibilities. You should restart a manager when you: - Change work shift assignments - Modify the number of target processes - Modify specialization rules - Change concurrent program incompatibilities
Deactivate concurrent manager	Deactivates the manager. All requests (concurrent programs) currently running are allowed to complete before the manager shuts down. A manager will not restart until you select the manager and choose "Activate concurrent manager".

Control Function	Description
Terminate requests and deactivate manager	All running requests (running concurrent programs) handled by the manager are terminated. Once deactivated, a manager will not restart until you select the manager and choose "Activate concurrent manager".

Controlling the Internal Concurrent Manager from the Operating System

To start the Internal Concurrent Manager, use the shell script adcmctl.sh.

Alternatively, use one of two other commands you may use from the operating system to control the Internal Concurrent Manager: STARTMGR, which starts the Internal Concurrent Manager; and CONCSUB, which can be used to deactivate or abort the Internal Concurrent Manager, or to instruct the Internal Concurrent Manager to verify the operating system process for each individual manager.

The following table compares the Internal manager control states displayed by the Administer Concurrent Managers form with their corresponding operating system command. Not all arguments are shown.

From the Administer Concurrent Managers Form	From the Operating System (not all arguments shown)
Activate concurrent manager	STARTMGR (syntax may vary with platform)
Verify concurrent manager status	CONCSUB FND VERIFY
Deactivate concurrent manager	CONCSUB FND DEACTIVATE
Terminate requests and deactivate manager	CONCSUB FND ABORT

Starting the Internal Concurrent Manager from the Operating System

To start the Internal Concurrent Manager, use the shell script adcmctl.sh.

This command starts the Internal Concurrent Manager, which in turn starts any concurrent managers you have defined.

Alternatively, to start the concurrent managers, you can invoke the STARTMGR command from your operating system prompt.

You must have write privileges to the "out" and "log" directories of every application so

that the concurrent managers can write to these directories. You can start the concurrent managers with many different options. An option on some operating systems is to send an electronic mail note to a given user when the concurrent managers shut down. See your installation guide for a discussion of this command.

Use the STARTMGR command:

- during installation of Oracle Applications
- after you shut down the concurrent managers
- after MIS restarts the operating system
- after the database administrator restarts the database

The STARTMGR command takes up to ten optional parameters.

- Each parameter except PRINTER has a default.
- You can modify the STARTMGR command and your environment to set your own defaults.

Enter the following command at your system prompt to start the Internal Concurrent Manager:

```
$ startmgr  <optional parameters>
```

You can pass the parameters in any order. For example:

```
$ startmgr sysmgr="<username>/<password>"  mgrname="std"
 printer="hqseq1"  mailto="jsmith"  restart="N"
 logfile="mgrlog"  sleep="90"  pmon="5"  quesiz="10"
```

The startmgr script accepts an Oracle username/password as the sysmgr parameter. Alternatively, you could pass an Oracle Applications username/password as an appmgr parameter. If no sysmgr or appmgr parameter is provided on the command line, startmgr will prompt you for the Oracle password.

See: Setting Up Concurrent Managers, page 8-1

Viewing the Internal Concurrent Manager startup parameters

The Internal Concurrent Manager's log file displays startup parameter values executed by the STARTMGR command. An example is shown below. You cannot change the parameter values.

```
logfile=/fnddev/fnd/6.0/log/FND60.mgr  (path is port-specific)
 PRINTER=hqunx138
 mailto=appldev
 restart=N
 diag=N
  sleep=60 (default)
 pmon=20 (default)
 quesiz=1  (default)
```

Shutting down the Internal Concurrent Manager from the Operating System

From the operating system prompt, you can use the CONCSUB utility to submit a concurrent request, under the SYSADMIN username and the System Administrator responsibility.

The CONCSUB utility submits a concurrent request and returns you to the operating system prompt. You must wait until the concurrent request completes.

To check on the status of your concurrent request, use the Concurrent Requests form.

```
CONCSUB username/password 'Responsibility application shortname'
 'Responsibility name' 'Username' [WAIT={Y|N|n}] CONCURRENT
 'Program application shortname' PROGRAM
```

Parameters

username/password
The ORACLE username and password that connects to Oracle Application Object Library data. Alternatively, an Oracle Applications username and password for a user with the System Administrator responsibility.

Responsibility application shortname
The application shortname of the responsibility. For the System Administrator responsibility, the application shortname is SYSADMIN.

Responsibility name
The name of the responsibility. For the System Administrator responsibility, the responsibility name is *System Administrator*.

Username
The application username of the person who submits the request. For example, SYSADMIN is the username of the System Administrator.

WAIT={Y|N|*n*}
Set WAIT to Y if you want CONCSUB to wait until the request you submitted completes before CONCSUB returns you to the operating system prompt.

Set WAIT to N (the default value) if you do not want CONCSUB to wait.

You can also enter an integer value of *n* seconds for CONCSUB to wait before it exits.

When used, WAIT must be entered before CONCURRENT.

Program application shortname
The application shortname of the program. For the DEACTIVATE, ABORT, and VERIFY programs, the application shortname is FND.

PROGRAM
To submit the Shutdown All Managers concurrent request,

use the program DEACTIVATE.

To submit the Shutdown Abort Managers concurrent request, use the program ABORT.

To submit the Verify All Managers Status concurrent request, use the program VERIFY.

Example Syntax using CONCSUB

```
CONCSUB <Username/Password> SYSADMIN 'System Administrator'
 SYSADMIN  CONCURRENT FND DEACTIVATE

CONCSUB <Username/Password> SYSADMIN 'System Administrator'
 SYSADMIN  CONCURRENT FND ABORT

CONCSUB <Username/Password> SYSADMIN 'System Administrator'
 SYSADMIN  CONCURRENT FND VERIFY
```

Using CONCSUB to shut down your managers

Use CONCSUB to shut down the concurrent managers:

- before MIS shuts down the operating system
- before the database administrator shuts down the database
- when you want concurrent manager and concurrent program definitions to take effect

Then, use the STARTMGR command to restart the Internal Concurrent Manager, which starts the concurrent managers.

Example - nightly shutdown using CONCSUB

You can use the token WAIT with value Y (WAIT=Y) if you want to use CONCSUB to issue a concurrent request from within a shell script containing a sequence of steps. Using the token WAIT insures the managers deactivate, abort, or verify status before the shell script proceeds to the next step.

See: Controlling the Internal Concurrent Manager from the Operating System, page 7-38

1. Shell script customized for specific operating system starts.
2. CONCSUB *username/password* SYSADMIN'System Administrator' SYSADMIN WAIT=Y CONCURRENTFND DEACTIVATE

   When the shell script passes control to CONCSUB, CONCSUB waits until the program DEACTIVATE is complete before it returns control to the shell script.
3. Script issues the command to shut down the database.

4. Script issues the command to backup the database.

5. Script issues the command to startup the database.

6. $ startmgr sysmgr="apps/fnd" mgrname="std" printer="hqseq1" mailto="jsmith" restart="N" logfile="mgrlog" sleep="90" pmon="5" quesiz="10"

   The shell script passes control to STARTMGR, which starts up the Internal manager (and all the other managers).

7. Shell script completes.

Hiding the password using CONCSUB

If the username/password are still supplied, the CONCSUB utility will work as usual.

If username only is supplied (no '/password' in the first argument), it will prompt you for an Oracle Applications username and password.

In the following example, CONCSUB would connect using the .dbc file, and then only run if the Oracle Applications user "sysadmin" with password "sysadmin" is successfully authenticated.

```
CONCSUB Apps:User SYSADMIN 'System Administrator' SYSADMIN/sysadmin
 CONCURRENT FND VERIFY
```

The user can put the password in a file, and then redirect it to standard input (stdin). In UNIX the command would be executed as follows:

```
CONCSUB Apps:User SYSADMIN 'System Administrator' SYSADMIN
 CONCURRENT FND
FNDMNRMT Y 0 20221 < password.file
```

where password.file is an ASCII file that contains the password. This method is recommended for use in shell scripts or batch processes.

Overview of Parallel Concurrent Processing

This essay explains what parallel concurrent processing is, describes the environments it runs in, and explains how it works.

What is Parallel Concurrent Processing?

Parallel concurrent processing allows you to distribute concurrent managers across multiple nodes in a cluster, massively parallel, or networked environment. Instead of operating concurrent processing on a single node while other nodes are idle, you can spread concurrent processing across all available nodes, fully utilizing hardware resources.

Benefits of Parallel Concurrent Processing

Parallel concurrent processing provides Oracle Applications users with the following benefits:

- High performance - the ability to run concurrent processes on multiple nodes to improve concurrent processing throughput.
- Fault Tolerance - the ability to continue running concurrent processes on available nodes even when one or more nodes fails.
- Adaptability - the ability to integrate with platform-specific batch queue and load-balancing systems to maximize concurrent processing performance on a particular platform.
- Single Point of Control - the ability to administer concurrent managers running on multiple nodes from any node in a cluster, massively parallel, or networked environment.

Parallel Concurrent Processing Environments

Parallel concurrent processing runs in multi-node environments, such as cluster, massively parallel, and networked environments. In these environments, each node consists of one or more processors (CPUs) and their associated memory. Each node has its own memory that is not shared with other nodes And each node operates independently of other nodes, except when sharing a resource such as a disk.

With parallel concurrent processing, one or more concurrent managers run on one or more nodes in a multi-node environment. You decide where concurrent managers run when configuring your system.

You can define any set of concurrent manager specialization rules, and apply them across nodes in any way desired. For example, three "Oracle General Ledger" concurrent managers could be spread across three nodes. Or an "Oracle Payables" concurrent manager and an "Oracle General Ledger" concurrent manager could run simultaneously on the same node.

The following are examples of environments in which parallel concurrent processing can run:

Cluster Environments

In a cluster environment, multiple computers, each representing a single node, share a common pool of disks.

With parallel concurrent processing in a cluster environment, a single ORACLE database resides in the common disk pool, while multiple instances of Real Application Clusters (RAC) run simultaneously on multiple nodes in the cluster. Multiple concurrent managers are also distributed across the nodes in the cluster.

Massively Parallel Environments

In a massively parallel environment, multiple nodes are housed in a single computer. All nodes share access to a common pool of disks. The IBM SP/2, for example, is a massively parallel computer.

With parallel concurrent processing in a massively parallel environment, separate RAC instances run simultaneously on multiple nodes, with multiple concurrent managers also distributed across nodes.

Networked Environments

In networked environments, multiple computers of the same type are connected via a local area network (LAN) to a single database server, or alternatively, to a cluster of database servers.

For example, a simple networked environment could consist of multiple Sun SPARCstations connected via a LAN to a single Sequent server. In a more complex networked environment, multiple Sun SPARCstations could connect to a cluster of Sequent servers.

With parallel concurrent processing in a networked environment, concurrent managers run on multiple workstations. A single database server runs a single instance of ORACLE; or, a cluster of database servers runs multiple ORACLE instances using RAC.

How Parallel Concurrent Processing Works

Concurrent Managers

With parallel concurrent processing, each node with concurrent managers may or may not be running an ORACLE instance. On a node that is not running ORACLE, the concurrent manager(s) connect via Net8 to a node that is running ORACLE.

Parallel concurrent processing is activated along with Generic Service Management (GSM); it can not be activated independently of GSM. With parallel concurrent processing implemented with GSM, the Internal Concurrent Manager (ICM) tries to assign valid nodes for concurrent managers and other service instances. Primary and secondary nodes need not be explicitly assigned. However, you can assign primary and secondary nodes for directed load and failover capabilities.

> **Note:** In previous releases, you must have assigned a primary and secondary node to each concurrent manager.

Initially, a concurrent manager is started on its defined primary node, or if none exists, the node that the ICM assigns to it. In case of node or ORACLE instance failure, all concurrent managers on that node migrate to their respective secondary nodes if defined.

A concurrent manager on its secondary node migrates back to its primary node once that node becomes available. During migration, the processes of a single concurrent manager may be spread across its primary and secondary nodes.

Internal Concurrent Manager

The Internal Concurrent Manager can run on any node, and can activate and deactivate concurrent managers on all nodes. Since the Internal Concurrent Manager must be active at all times, it needs high fault tolerance. To provide this fault tolerance, parallel concurrent processing uses Internal Monitor Processes.

Internal Monitor Processes

The sole job of an Internal Monitor Process is to monitor the Internal Concurrent Manager and to restart that manager should it fail. The first Internal Monitor Process to detect that the Internal Concurrent Manager has failed restarts that manager on its own node.

Only one Internal Monitor Process can be active on a single node. You decide which nodes have an Internal Monitor Process when you configure your system. You can also assign each Internal Monitor Process a primary and a secondary node to ensure failover protection.

Internal Monitor Processes, like concurrent managers, have assigned work shifts, and are activated and deactivated by the Internal Concurrent Manager.

Log and Output File Access

The concurrent log and output files from requests that run on any node are accessible online from any other node. Users need not log onto a node to view the log and output files from requests run on that node.

Integration with Platform-Specific Queuing

Some cluster or massively parallel systems have their own mechanisms for queuing batch processes; for example, IBM LoadLeveler. Because users may wish to manage all processing with these mechanisms (and not just Oracle Applications processing), parallel concurrent processing is designed to integrate with them. Thus, you can match your concurrent process management to the specific capabilities of your operating platform.

For more information on integrating with platform-specific queuing, refer to the installation documentation for your platform.

Managing Parallel Concurrent Processing

This section describes how to manage parallel concurrent processing.

Parallel concurrent processing is always active when Generic Service Management (GSM) is active. Parallel concurrent processing can no longer be activated independently of Generic Service Management.

However, automatic activation of PCP does not additionally require that primary nodes be assigned for all concurrent managers and other GSM-managed services. If no primary node is assigned for a service instance, the Internal Concurrent Manager (ICM) assigns a valid concurrent processing server node as the target node. In general, this node will be the same node where the Internal Concurrent Manager is running. In the case where the ICM is not on a concurrent processing server node, the ICM chooses an active concurrent processing server node in the system. If no concurrent processing server node is available, no target node will be assigned.

Note that if a concurrent manager does have an assigned primary node, it will only try to start up on that node; if the primary node is down, it will look for its assigned secondary node, if one exists. If both the primary and secondary nodes are unavailable, the concurrent manager will not start (the ICM will not look for another node on which to start the concurrent manager). This strategy prevents overloading any node in the case of failover.

The concurrent managers are aware of many aspects of the system state when they start up. When an ICM successfully starts up it checks the TNS listeners and database instances on all remote nodes and if an instance is down, the affected managers and services switch to their secondary nodes. Processes managed under GSM will only start on nodes that are in Online mode. If a node is changed from Online to Offline, the processes on that node will be shut down and switch to a secondary node if possible.

Concurrent processing provides database instance-sensitive failover capabilities. When an instance is down, all managers connecting to it switch to a secondary middle-tier node.

However, if you prefer to handle instance failover separately from such middle-tier failover (for example, using TNS connection-time failover mechanism instead), use the profile option Concurrent:PCP Instance Check. When this profile option is set to OFF, Parallel Concurrent Processing will not provide database instance failover support; however, it will continue to provide middle-tier node failover support when a node goes down.

Defining Concurrent Managers

You define concurrent managers either in the Create New Request Processing Manager page in Oracle Applications Manager or in the Concurrent Managers form. When you define a manager, you specify the manager type, which may be either Concurrent Manager, Internal Monitor, or Transaction Manager.

There are three other types of managers that Oracle Applications predefines for you: the Internal Concurrent Manager, which describes the Internal Concurrent Manager process, the Conflict Resolution Manager, and the Scheduler. For the Conflict Resolution Manager and Scheduler you can assign the primary and secondary nodes.

For the Internal Concurrent Manager you assign the primary node only.

To each concurrent manager and each Internal Monitor Process, you may assign a primary and a secondary node. You may also assign primary and secondary system queue names, if a platform-specific queue management system is available on your platform. See: Concurrent Managers, page 7-58.

Administering Concurrent Managers

Target Nodes

Using the Services Instances page in Oracle Applications Manager (OAM) or the Administer Concurrent Managers form, you can view the target node for each concurrent manager in a parallel concurrent processing environment. The target node is the node on which the processes associated with a concurrent manager should run. It can be the node that is explicitly defined as the concurrent manager's primary node in the Concurrent Managers window or the node assigned by the Internal Concurrent Manager.

If you have defined primary and secondary nodes for a manager, then when its primary node and ORACLE instance are available, the target node is set to the primary node. Otherwise, the target node is set to the manager's secondary node (if that node and its ORACLE instance are available). During process migration, processes migrate from their current node to the target node.

Control Across Nodes

Using the Services Instances page in Oracle Applications Manager or the Administer Concurrent Managers form, you can start, stop, abort, restart, and monitor concurrent managers and Internal Monitor Processes running on multiple nodes from any node in your parallel concurrent processing environment. You do not need to log onto a node to control concurrent processing on it. You can also terminate the Internal Concurrent Manager or any other concurrent manager from any node in your parallel concurrent processing environment.

In an environment enabled with parallel concurrent processing, primary node assignment is optional for the Internal Concurrent Manager. The Internal Concurrent Manager can be started from any of the nodes (host machines) identified as concurrent processing server enabled. In the absence of a primary node assignment for the Internal Concurrent Manager, the Internal Concurrent Manager will stay on the node (host machine) where it was started. If a primary node is assigned, the Internal Concurrent Manager will migrate to that node if it was started on a different node.

If the node on which the Internal Concurrent Manager is currently running becomes unavailable or the database instance to which it is connected to becomes unavailable, the Internal Concurrent Manager will be restarted on a alternate concurrent processing node. If no primary node is assigned, the Internal Concurrent Manager will continue to operate on the node on which it was restarted. If a primary node has been assigned to

the Internal Concurrent Manager, then it will be migrated back to that node whenever both the node and the instance to which the Internal Concurrent Manager connects to from that node becomes available

Starting Up Managers

You start up parallel concurrent processing as you would ordinary concurrent processing, by running the adcmctl.sh script from the operating system prompt.

The Internal Concurrent Manager starts up on the node on which you run the adcmctl.sh script. If it has a different assigned node, it will migrate to that node if available.

After the Internal Concurrent Manager starts up, it starts all the Internal Monitor Processes and all the concurrent managers. It attempts to start Internal Monitor Processes and concurrent managers on their primary nodes, and resorts to a secondary node only if a primary node is unavailable.

Shutting Down Managers

You shut down parallel concurrent processing by issuing a "Stop" command in the OAM Service Instances page or a "Deactivate" command in the Administer Concurrent Managers form. All concurrent managers and Internal Monitor processes are shut down before the Internal Concurrent Manager shuts down.

Terminating Concurrent Processes

You can terminate running concurrent processes for a concurrent manager on the local node or on remote nodes by issuing an "Abort" command from the OAM Service Instances page or a "Terminate" command from the Administer Concurrent Managers form.

Migrating Managers

Most process migration occurs automatically in response to the failure or subsequent availability of a primary node. However, you may migrate processes manually by changing the node assignments for a concurrent manager or Internal Monitor Process using the Concurrent Managers form. To put your changes into effect, issue a "Verify" command against the Internal Concurrent Manager from the Administer Concurrent Managers form.

Related Topics

Concurrent Managers, page 7-58

Administer Concurrent Managers Window

View the status of your concurrent managers (including any transaction managers) and, if you wish, change the status of any manager by issuing a control command. For example, you can deactivate a manager that is currently active, then view its new status after the change takes effect.

Use the Refresh button to refresh the data shown.

Administer Concurrent Managers Block

Node

In a parallel concurrent processing environment, a manager's processes are targeted to run on this node.

If a concurrent manager is defined to use a platform-specific system queue, this field displays the name of the queue to which the manager submits its processes.

Processes Actual

Each manager process can run one concurrent request (start one concurrent program). Typically, the number of actual processes equals the number of target processes (the maximum number of requests a manager can run).

However, the number of actual processes may be less than the number of target processes due to manager deactivation or manager migration.

Processes Target

This field displays the maximum number of manager processes that can be active for this manager.

Requests Running/Requests Pending

Typically, when there are requests pending, this number should be the same as the number of actual processes. However, if there are no pending requests, or requests were just submitted, the number of requests running may be less thanthe number of actual processes.

Moreover, if a concurrent program is incompatible with another program currently running, it does not start until the incompatible program has completed. In this case, the number of requests running may be less than number of actual processes even when there are requests pending.

Controlling a Specific Manager - Status

This field displays the status of a manager after you have chosen a specific action for it using the top row of buttons near the bottom of the window.

You can control concurrent managers individually or collectively by controlling the Internal Concurrent Manager. This field is blank when managers have been activated by the Internal Concurrent Manager.

In a parallel processing environment, this field displays *Target node/queue unavailable* when the primary and secondary nodes (or system queues) are not available.

The actions you can choose for controlling a manager are:

Terminate

When you terminate requests and deactivate the Internal Concurrent Manager, all running requests (running concurrent programs) are terminated, and all managers are deactivated.

Managers previously deactivated on an individual basis are not affected.

You can terminate requests and deactivate individual managers. All running requests (running concurrent programs) handled by the manager are terminated.

Once deactivated, a manager does not restart until you select the manager and choose the Activate button.

Deactivate

When you deactivate the Internal Concurrent Manager, all

other managers are deactivated as well. Managers previously deactivated on an individual basis are not affected.

You can deactivate individual managers. Once deactivated, a manager does not restart until you select the manager and choose the Activate button.

When you deactivate a manager, including the Internal Concurrent Manager, all requests (concurrent programs) currently running are allowed to complete before the manager(s) shut down.

Verify

This choice appears only when you select the Internal Concurrent Manager.

The Internal Concurrent Manager periodically monitors the processes of each concurrent manager. You can force this process monitoring or PMON activity to occur by choosing the Verify button.

Another result of selecting this choice is that the Internal Concurrent Manager rereads concurrent program incompatibility rules.

Restart

This choice appears only when you select an individual manager.

When you restart a concurrent manager, the manager rereads its definition.

You should restart a manager when you have made the following changes using the Define Concurrent Manager form, and you wish those changes to take effect:

- Change work shift assignments
- Modify the number of Target Processes
- In a parallel concurrent processing environment, change node or system queue information

Activate

When you activate the Internal Concurrent Manager, you activate all other managers as well, except those managers that were deactivated on an individual basis.

You cannot activate the Internal Concurrent Manager from the PC client. The Internal Concurrent Manager is only activated from the server.

You can also activate an individual concurrent manager

that is currently deactivated, so long as the Internal manager is active. If the manager is defined to work in the current work shift, then the Internal manager starts it immediately.

Reviewing a Specific Manager

View details of a concurrent manager's operation

Processes You can view the details of the processes of a given concurrent manager. Processes that are currently active, migrating, or terminating, as well as processes that have been terminated or deactivated, are displayed.

Requests For a selected manager you can view all running and pending requests handled by the manager.

The following actions are available only for certain services managed Generic Service Management. These services must be defined to accept commands to suspend their operations.

Suspend Suspend the operations of the service.

Resume Resume the operations of the service.

Related Topics

Concurrent Processing Window, page 7-53

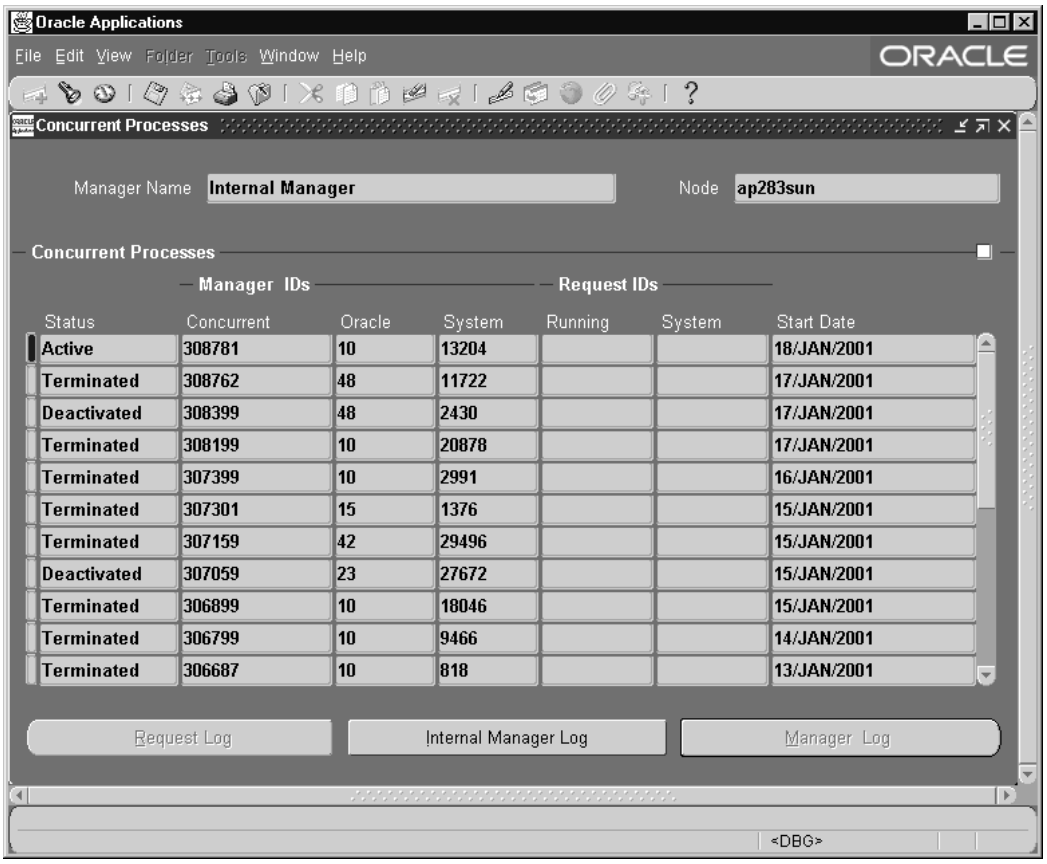
Defining Managers and their Work Shifts, page 7-1

Controlling Concurrent Managers, page 7-35

Overview of Parallel Concurrent Processing, page 7-42

Life cycle of a concurrent request, *Oracle Applications System Administrator's Guide - Maintenance*

Concurrent Processes Window



View status information about the processes of a specific concurrent manager, whose name and node are identified near the top of the window.

Displaying this window automatically queries all processes that are currently active, migrating, or terminating, as well as processes that have been terminated or deactivated.

Display order is by status value (Active, Migrating, Terminating, Terminated, Deactivated) and within status, by the order in which processes were started.

If you wish to reduce the number of displayed processes, you can delete records by submitting the "Purge Concurrent Request and Managers" report from the Run Requests form. You can delete records according to the number of days since the processes were started. However, you cannot delete the records of currently active managers.

Status

This field cannot be updated. The following are valid status values:

Active Currently running manager processes display as "Active".

Deactivated Manager processes that are no longer running display as "Deactivated".

These processes were deactivated by you choosing the Deactivate button in the Administer Concurrent Managers block, or by the Internal Concurrent Manager deactivating a concurrent manager at the end of that manager's work shift.

Migrating Managers that are migrating between primary and secondary nodes display as "Migrating".

In a parallel concurrent processing environment, concurrent managers run on either the primary or secondary node assigned to them. Managers migrate to the secondary node if the primary node or the database instance on the primary node is unavailable. Managers migrate back to the primary node once it becomes available.

Terminating Manager processes that are being terminated display as "Terminating".

These processes were terminated by you choosing the Terminate button in the Administer Concurrent Managers block, or by a user selecting "Terminate" in the Concurrent Requests form.

Terminated Manager processes that have been terminated display as "Terminated".

These processes were terminated by you choosing the Terminate button in the Administer Concurrent Managers block, or by a user selecting "Terminate" in the Concurrent Requests form.

Manager Identifiers Concurrent

This field displays a number generated by the individual concurrent manager that identifies the process. This field cannot be updated.

This number may be referenced if an operating system process ID is not available.

You can use this number to view the log file associated with the process. (This is the same log file you view when you select Manager Log from the View field of the Concurrent Requests form):

- At the operating system level, locate yourself in the log directory

$FND_TOP/APPLLOG.

- For concurrent managers, use w<*number*>.mgr.
- For Internal Monitor processes, use i<*number*>.mgr.

Manager Identifiers Oracle

This field displays the ORACLE process ID associated with the manager process. This field cannot be updated.

Manager Identifiers System

This field displays the operating system process ID associated with the manager process. This field cannot be updated.

Request Identifiers Running

Please note the following about this field:

- Normally this field is blank, as the run-time of a request is typically very short.
- For a terminated manager, the ID of the request being processed at the time of termination is displayed.

Request Identifiers System

This field displays the operating system process ID for a spawned concurrent process.

Viewing Log Files

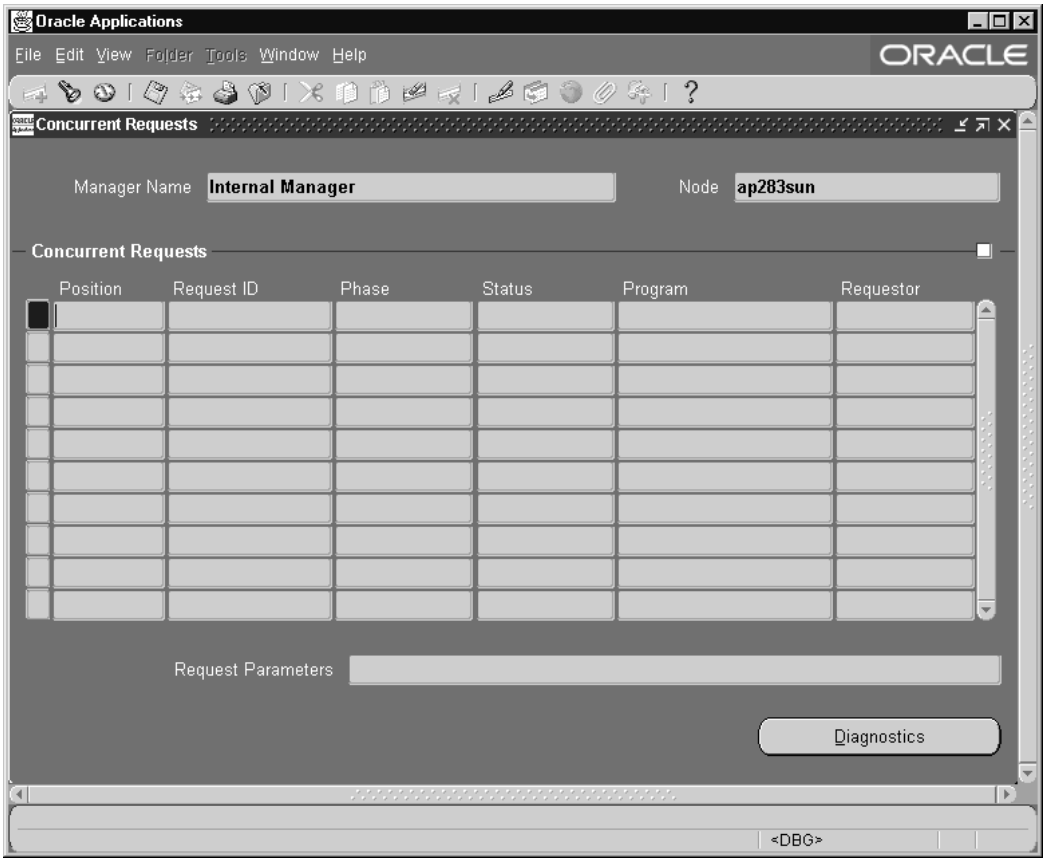
Use the three buttons near the bottom of the window to view log files. Log files record information that may be helpful when diagnosing problems.

Request Log Choose this button to view the log file of the process associated with the running request.

Internal Manager Log Choose this button to view the Internal Concurrent Manager's log file.

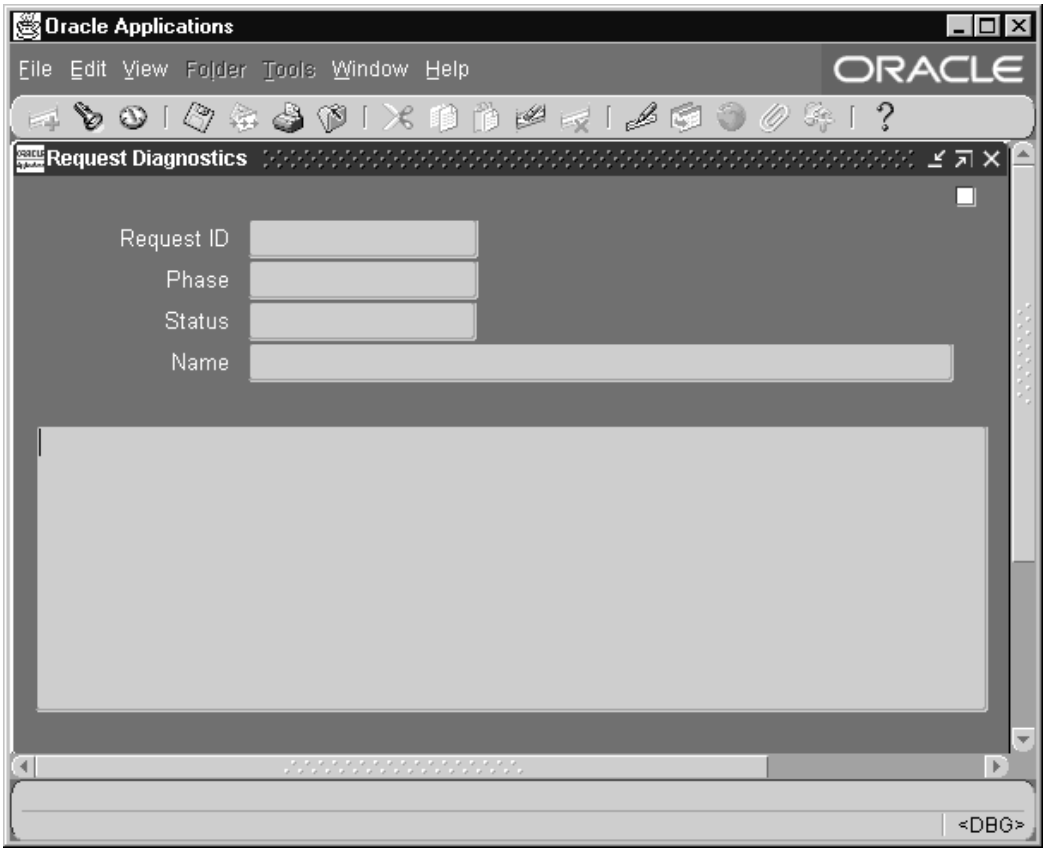
Manager Log Choose this button to view the log file of the concurrent manager who started running the request.

Concurrent Requests Window



View all running and pending requests for a selected manager, whose name and node are identified near the top of the window.

Request Diagnostics Window



This window informs you when the request completed or if it did not complete, shows you a diagnostic message indicating why.

Related Topics

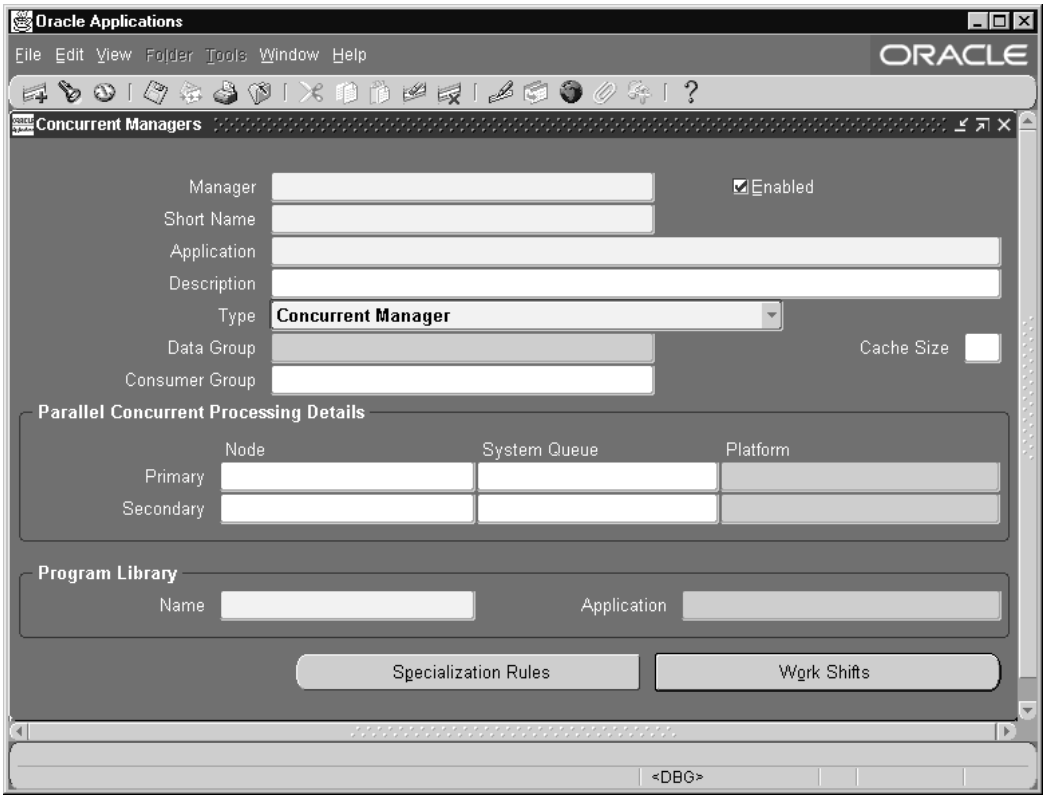
Concurrent Manager field help, page 7-58

Defining Managers and their Work Shifts, page 7-1

Controlling Concurrent Managers, page 7-35

Overview of Parallel Concurrent Processing, page 7-42

Concurrent Managers Window



Use this window to define your concurrent managers. You can determine when a manager runs and how many programs a manager can start simultaneously when you assign workshifts to the manager. Determine which programs a manager can start by defining specialization rules.

Concurrent Managers Block

The combination of an application and the name you define for your manager uniquely identifies the manager.

Application

The application name does not prevent a manager from starting programs associated with other applications. To restrict a manager to only running programs associated with certain applications, go to the Specialization Rules window.

Type

Once you define a concurrent manager, you cannot update this field. There are several types of managers:

Concurrent Manager	Concurrent Managers start concurrent programs running.
Internal Monitor	Internal Monitors monitor the Internal concurrent manager in a parallel concurrent processing environment. If the Internal Concurrent Manager exits abnormally (for example, because its node or its database instance goes down), an Internal Monitor restarts it on another node.
Transaction Manager	Transaction managers handle synchronous requests from client machines.

Cache Size (Concurrent Manager only)

Enter the number of requests your manager remembers each time it reads which requests to run. For example, if a manager's workshift has 1 target process and a cache value of 3, it will read three requests and wait until these three requests have been run before reading new requests.

In reading requests, the manager will only put requests it is allowed to run into its cache. For example, if you have defined your manager to run only Order Entry reports then the manager will put only Order Entry requests into its cache.

If you enter 1, the concurrent manager must look at its requests list each time it is ready to process another request.

By setting the cache size at a higher number, the concurrent manager does not have to read its requests list each time it runs a request. However, the manager does not recognize any priority changes you make for a particular request if it has already read that request into its cache. Further, even if you give a higher priority to a new request, that new request must wait until the buffer is empty and the manager returns to look at the requests list. That request may have to wait a long time if you set the buffer size to a high number.

You should use cache size to tune your concurrent managers to work most efficiently for you site's needs. If your organization tends to reprioritize jobs going to a certain manager, that manager should have its buffer size set fairly low.

> **Tip:** Enter a value of 1 when defining a manager that runs long, time-consuming jobs, and a value of 3 or 4 for managers that run small, quick jobs.

Data Group (Transaction Manager only)

The data group the transaction manager uses to connect to the database. Transaction managers only run programs submitted from responsibilities that use the same data group as the transaction manager.

Note: Data groups are no longer supported by Oracle Applications. This section is provided for reference only.

Resource Consumer Group

The resource consumer group for the manager. For more information on resource consumer groups, see: Resource Consumer Groups in Oracle Applications, page 11-3.

(Parallel Concurrent Processing Details) Node

If you are operating in a parallel concurrent processing environment and you want your manager to operate on a specific node, select the name of the node.

The primary node, if available, is the node your concurrent manager operates on. If the primary node or the database instance on it goes down, your concurrent manager migrates to its secondary node. Your concurrent manager migrates back to its primary node when that node becomes available.

Nodes must be previously registered with Oracle Applications, using the Nodes form. See: Nodes, page 7-73.

(Parallel Concurrent Processing Details) System Queue

If you are operating in a parallel concurrent processing environment and you want your manager to use a platform-specific queue management system instead of generic concurrent processing queue management, specify the queue or class name of that system. For example, you may choose a system queue name from a platform-specific queue management system like NQS or IBM Load Leveler.

The primary system queue is the queue you associate with the primary node. The secondary system queue is the queue you associate with the secondary node.

Important: To ensure that your manager uses your platform-specific queue management system, you should start the concurrent managers in the proper mode. Refer to platform-specific documentation to determine if your platform supports interfacing with system queues. For UNIX platforms, refer to the appropriate Oracle Applications Installation Update. For all other platforms, refer to the appropriate Oracle Applications installation guide.

Program Library

Concurrent managers can run only those immediate concurrent programs listed in their program library. They can also run concurrent programs that use any other type of concurrent program executable as long as the specialization rules include them.

Immediate concurrent programs must be registered in a program library by an

applications developer using Oracle Application Object Library.

Transaction Managers can only run programs listed in their program library.

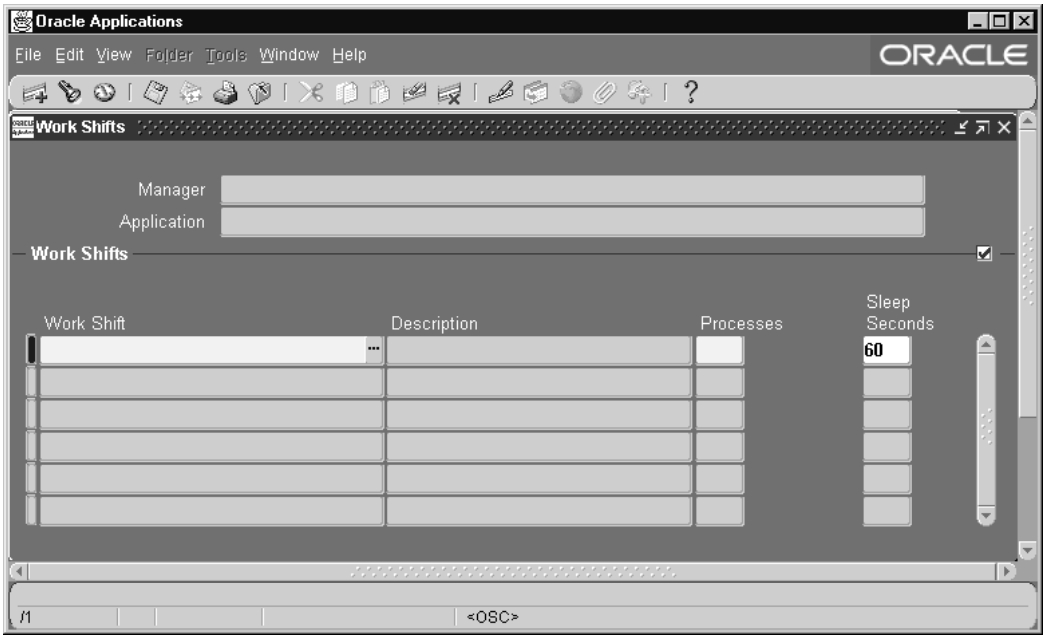
Defining Manager Operations

The two buttons near the bottom of the window display additional windows for defining when your manager operates, and, if you wish, specializing your manager to run only certain kinds of programs.

Specialization Rules You define what kinds of requests the manager reads by defining specialization rules for your manager.

Work Shifts You define when the manager operates by assigning one or more work shifts to your manager. With each work shift, you can vary the number of programs the manager can run simultaneously.

Work Shifts Window



Assign work shifts to a concurrent manager. A work shift defines the dates and times the manager is enabled. For each work shift you define the number of processes the manager starts running.

Work shifts are defined using the Work Shifts form. See: Work Shifts, page 7-65.

Processes

Enter the number of operating system processes you want your work shift to run

simultaneously. Each process can run a concurrent request.

For example, if a work shift is defined with three (3) target processes, the manager can run up to three requests simultaneously.

Parameter

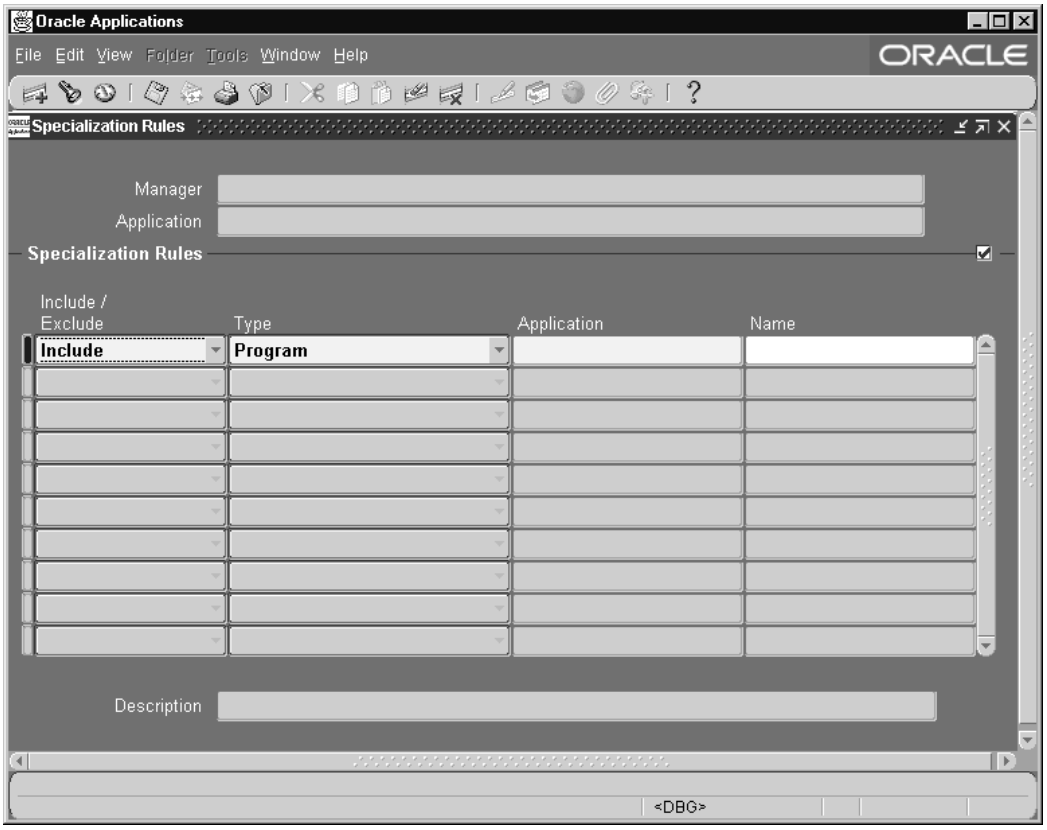
Enter the parameter string for a service under Generic Service Management. The value of this field is dependent on the service type definition.

Sleep Seconds

Enter the sleep time for your manager during this work shift. Sleep time is the number of seconds your manager waits between checking the list of pending concurrent requests (concurrent requests waiting to be started).

> **Tip:** Set the sleep time to be very brief during periods when the number of requests submitted is expected to be high.

Specialization Rules Window



Specialize your manager to run only certain kinds of requests. Without specialization rules, a manager accepts requests to start *any* concurrent program.

Include/Exclude

Select from the poplist whether or not to include or exclude those requests that are based on the rule to run.

Type

Select the type of specialization rule you want to assign to your manager. Based on the rule's action you selected, allow or disallow, requests can be run by your manager according to a:

- Combined Rule

  For example, only requests that satisfy the combined rule you select are allowed to be run by your manager. Or conversely, requests that satisfy a certain combined rule are excluded from running.

  Combined specialization rules, which combine more than one logical statement, are

defined using the Combined Specialization Rules form. See: Combined Specialization Rules, page 7-67.

- ORACLE ID

  For example, programs with a certain ORACLE ID are excluded from running. Or conversely, a concurrent manager only includes programs with a specific ORACLE ID.

- Program

  For example, only the program you select is excluded from running. Or conversely, a concurrent manager only includes the programs you select. You can also include or exclude all programs belonging to a specific application using the Program type by entering the application in the Application field and leaving the Name field empty.

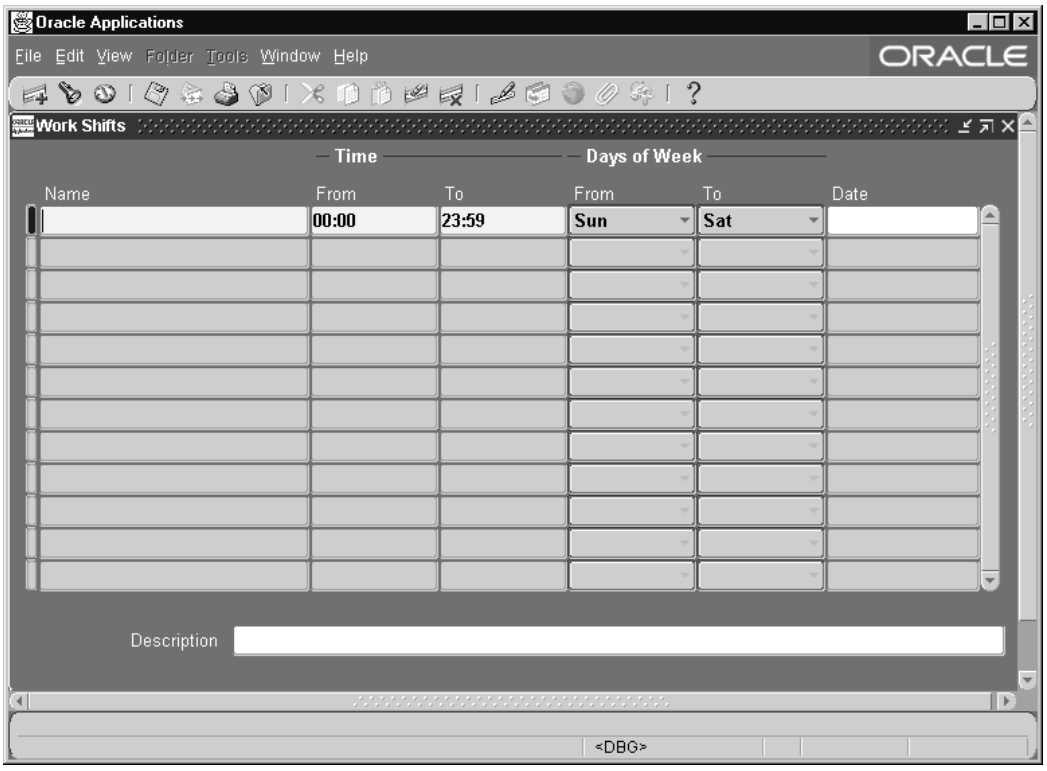
- Request Type (of the program)

  For example, programs of a certain request type are excluded from running. Or conversely, a concurrent manager only includes programs with the request type you select.

- User (application username at sign on)

  For example, all programs submitted by a certain user are excluded from running. Or conversely, a concurrent manager includes only programs submitted by the user you select.

Work Shifts Window



Use this window to name and define your concurrent manager work shifts. Define work shifts to specify when your concurrent managers can work.

For each work shift, specify a time period covering a range of days or a particular date. See: Work Shifts Definitions, page 7-3.

Name

The name of your concurrent work shift should be intuitive, for instance "Week Days", "Weeknights" or "Weekends".

From/To

Enter the times of day at which your concurrent shift begins/ends. The time format is HH24:MM. For example, if your work shift name is "Week Days", you could enter "09:00" (9:00 am) as the start time and "17:00" (5:00 pm) as the end time. Note that Oracle Applications uses a 24-hour clock.

Days of Week From/Days of Week To

Enter the first and last days of this shift. For instance, if your shift name is "Week Days",

you could enter "Monday" in the "Days of Week From" field and "Friday" in the "Days of Week To" field. If you enter a value in the "Days of Week From" field, you must enter a value in the "Days of Week To field". You may not use the Date field for this row.

Date

Enter a date here to create a date-specific workshift. For instance, you can name a workshift "Memorial Day", and enter the date in this field to enable this workshift only on the Memorial Day holiday.

Date-specific workshifts override workshifts that do not specify a specific date. If you want to enter a value in this field (specify a date), you may not enter values for the Days of Week fields for this row. See: Overlapping Work Shifts - Priority Levels, page 7-4.

Related Topics

Defining Managers and their Work Shifts, page 7-1

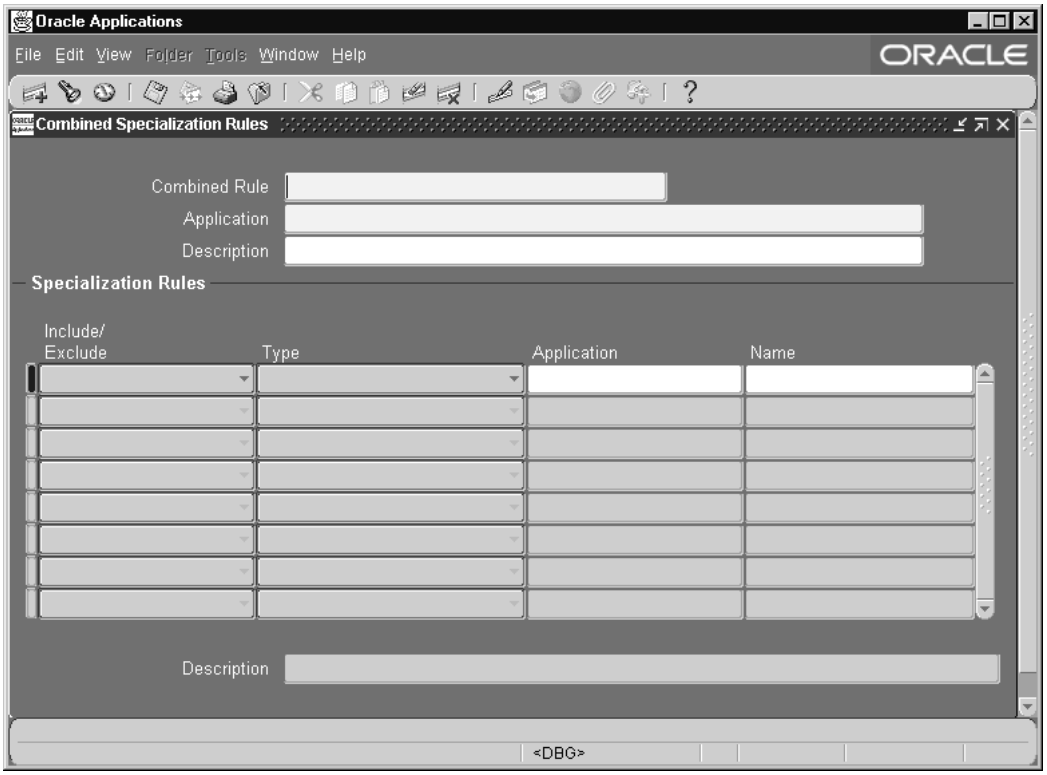
Work Shift by Manager Report, page 7-16

Work Shifts Report, page 7-16

Administer Concurrent Managers field help, page 7-49

Concurrent Managers field help, page 7-58

Combined Specialization Rules Window



Define rules identifying which requests a concurrent manager can read. With the rules you define here, you may specialize the function of a concurrent manager.

Using this window, you can define several Include and Exclude statements, each referred to as a specialization line, and combine the lines into a single specialization rule referred to as a *Combined Rule*.

Unlike the individual rules you define using the Specialization Rules window from within the Concurrent Managers window, the combined rules you define here differ in two ways:

- You can combine Include and Exclude statements. This enables you to identify very specific requests for running concurrent programs.
- Within a combined rule, using multiple Include statements restricts a concurrent manager more.

  With individual rules you define using the Specialization Rules window (within the Concurrent Managers window), the more "Include" rules you define, the less restricted a manager becomes.

  See: Concurrent Managers, page 7-58

Combined Specialization Rules Block

Together, the application name and the name you define for your combined specialization rule uniquely identifies the rule.

Application

The application name does not prevent a concurrent manager from starting programs associated with other applications.

Specialization Rules Block

Define the individual rules (statements) that make up your combined specialization rule.

- Each rule in this block defines one statement.
- The sum of all the specialization rules defines your combined specialization rule.

Include/Exclude

Select from the poplist whether to include or exclude those requests that are based on the rule to run.

Type

Select the type of specialization rule you want to enforce on a concurrent manager.

You cannot combine two Include rules of the same type. For example, you cannot include programs to be associated with an ORACLE ID, then, on another line, include programs to be associated with a second, different ORACLE ID.

Based on a rule's action, exclude or include, programs can be run by your manager according to a:

- ORACLE ID

  For example, programs with a certain ORACLE ID are excluded from running. Or conversely, a concurrent manager only includes programs with a specific ORACLE ID.

- Program

  For example, only the program you select is excluded from running. Or conversely, a concurrent manager only includes the programs you select. You can also include or exclude all programs belonging to a specific application using the Program type by entering the application in the Application field and leaving the Name field empty.

- Request Type (of the program)

  For example, programs of a certain request type are excluded from running. Or conversely, a concurrent manager only includes programs with the request type you select.

- User (application username at sign on)

  For example, all programs submitted by a certain user are excluded from running. Or conversely, a concurrent manager includes only programs submitted by the user you select.

Related Topics

Specializing Managers to run only certain programs, page 7-17

Defining Specialization Rules, page 7-17

Defining Combined Specialization Rules, page 7-27

Using Combined Rules, page 7-29

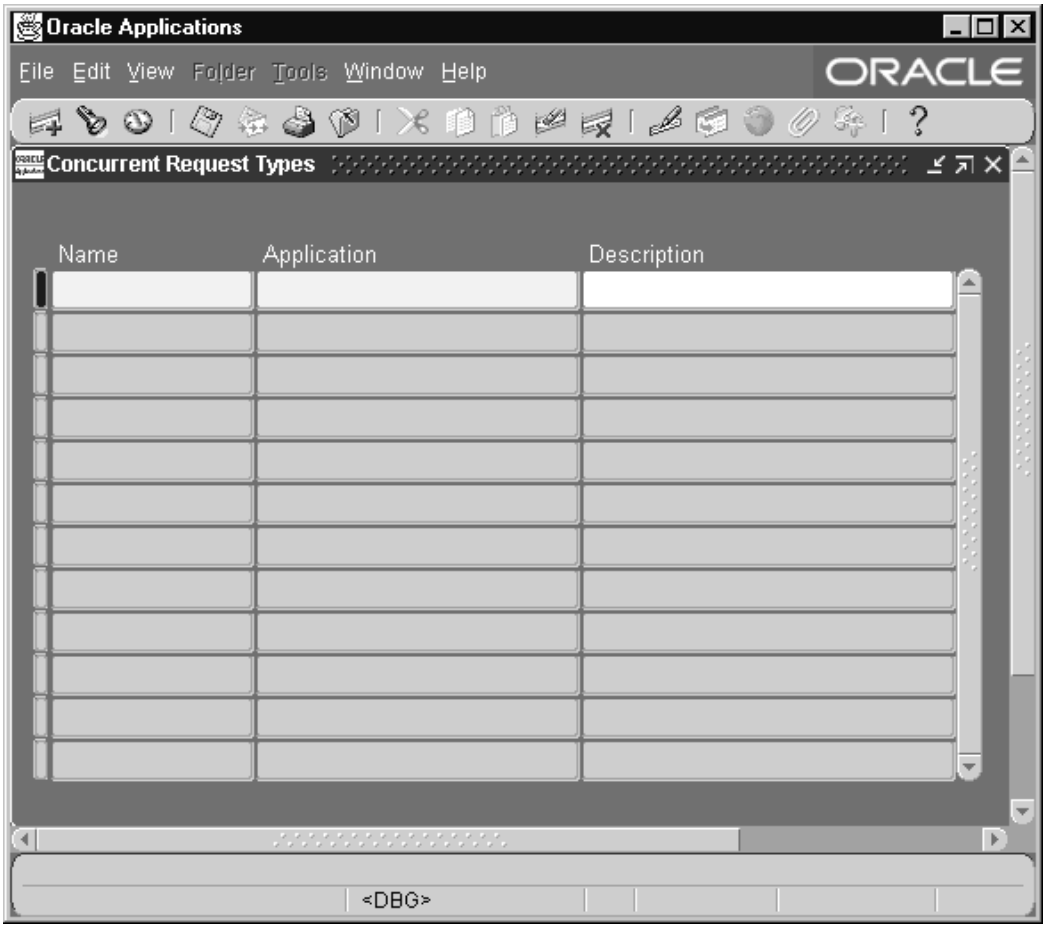
Differences Between Specialization and Combined Rules, page 7-32

Grouping Programs by Request Type, page 7-33

Administer Concurrent Managers field help, page 7-49

Concurrent Managers field help, page 7-58

Concurrent Request Types Window



Use this window to identify several concurrent programs as a group by assigning each program a common *request type*.

You assign a request type defined here to a concurrent program using the Concurrent Programs window. Then, when you define a concurrent manager using the Define Concurrent Manager window, you can define the manager to run (Allow) or not run concurrent programs based on their request type.

For example, you could define a request type as "end-of-month reports", assign that request type to several concurrent programs, then define a concurrent manager to only run "end-of-month" requests.

Concurrent Request Types Block

Name and describe each type of concurrent request you want to define. The combination of application name plus request type uniquely identifies your concurrent request type.

This application name does not prevent you from assigning this request type to concurrent programs associated with other application names.

Related Topics

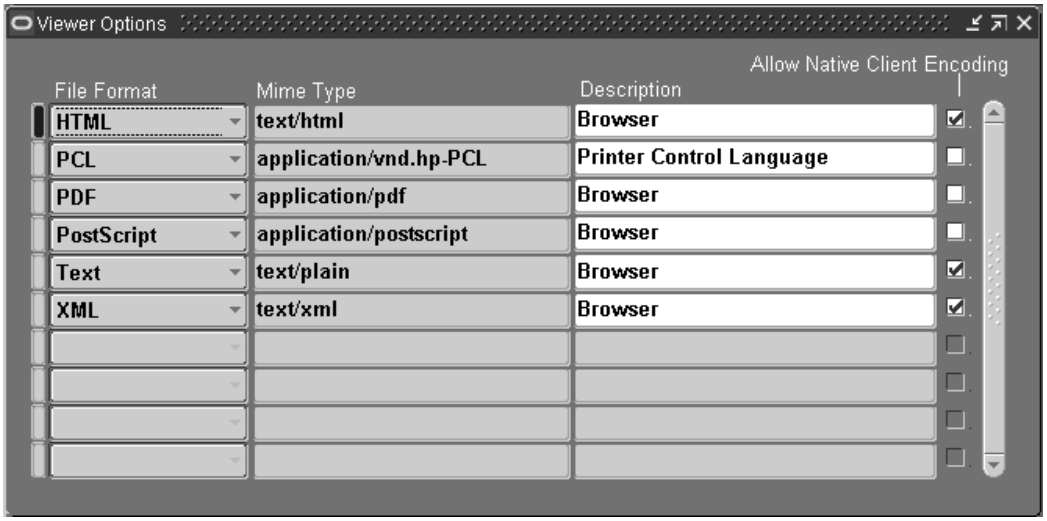
Specializing Managers to run only certain programs, page 7-17

Grouping Programs by Request Type, page 7-33

Concurrent Managers field help, page 7-58

Combined Specialization Rules field help, page 7-67

Viewer Options Window



Use this form to define the MIME types for the output formats of your concurrent requests. These MIME types are used in viewing reports.

For each file format, you can associate one or more MIME types.

A user can use one MIME type to view reports of a certain format. For example, a user can view all text format reports in Microsoft Word. The MIME types for supported formats for a particular user are set by several profile options. They are:

- Viewer: Application for HTML
- Viewer: Application for PCL
- Viewer: Application for PDF
- Viewer: Application for PostScript
- Viewer: Application for Text

This MIME type is sent to a browser window when the user views a report of that file format.

Viewer Options Block

Associate one or more MIME types with each supported file format. By defining viewer options, you can specify the application or applications that are available for displaying files of each format.

File Format

The file format.

MIME Type

The MIME type to use for the file output.

Allow Native Client Encoding

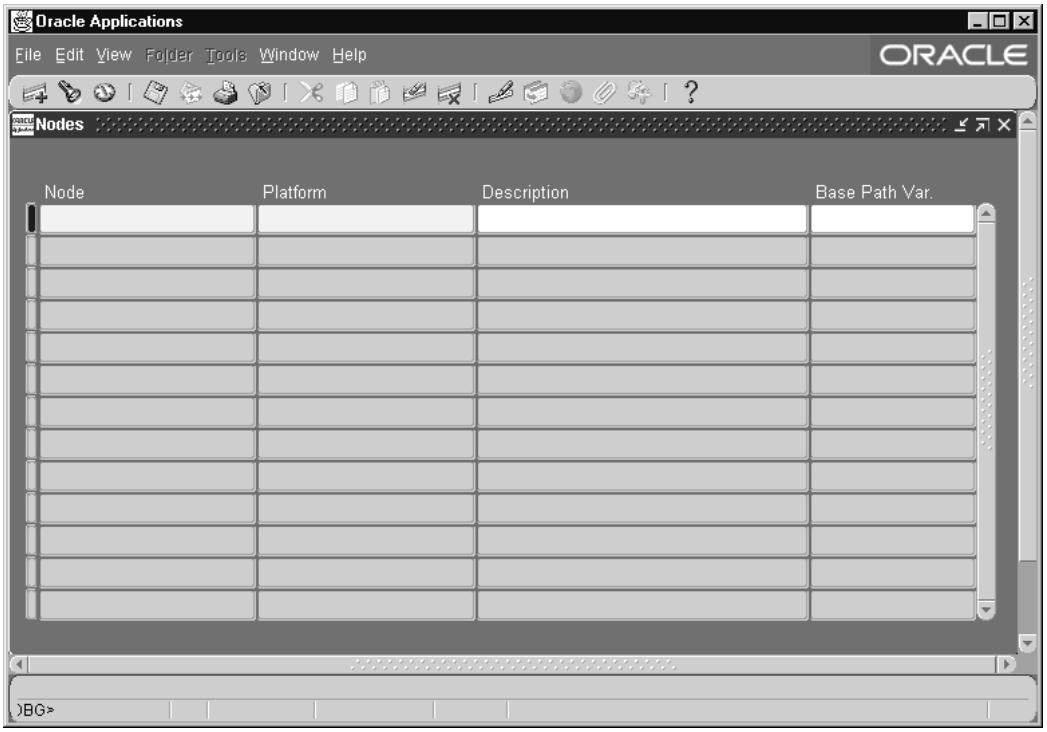
If this box is checked, the Report Viewer will convert the output file into the character set specified by the FND: Native Client Encoding profile option.

Related Topics

Defining the Reports Viewer, *Oracle Applications System Administrator's Guide - Maintenance*

Reviewing Requests, Request Log Files, and Report Output Files, *Oracle Applications System Administrator's Guide - Maintenance*

Nodes Window



A node consists of one or more processors and their associated memory. In parallel concurrent processing environments (such as cluster, massively parallel, and homogeneous networked environments) each node operates independently of other nodes except when sharing resources, such as a disk.

You can assign concurrent managers to different nodes to spread your concurrent processing workload and increase throughput. A concurrent manager runs its processes on the nodes to which it is assigned.

Nodes Block

Node

Enter the operating system name of a node.

Platform

Select the operating system platform that your node resides on.

Base Path Variable

Consult your installation manual to determine the correct base path variable for your

platform to determine the location of the concurrent managers' log and out files for this node.

Related Topics

Overview of Parallel Concurrent Processing, page 7-42

Concurrent Managers field help, page 7-58

8

Setting Up and Starting Concurrent Managers

Concurrent Managers

Oracle Applications concurrent managers run processes in the background on a server machine. You must set up and start the concurrent managers for each product group before you can use your Oracle Applications products. Refer to the *Maintaining Oracle Applications* documentation set for more information on the adcmctl script for starting and stopping concurrent managers. The instructions in this section apply whether you are installing or upgrading.

Setting Up Concurrent Managers

For UNIX

Keep the following in mind when you start the concurrent managers:

- Concurrent managers inherit directory privileges from the user who starts them. If you plan to start the managers from a login other than the main applications login, applmgr, ensure that the login has the appropriate directory privileges.

  For more information on directory privileges, see *Oracle Applications Concepts*.

Parameter values set in startmgr override any other values. Command line values override environment and default values, and so on. Another section in this manual contains more information on editing the startmgr script.

You can change directory privileges while the managers are running, and the changes will be effective immediately. You can change environment variables and startup parameters while the managers are running, but your changes will not take effect until the concurrent managers are restarted. To put changes into effect, shut down the managers, make the necessary modifications, and restart the managers.

For Windows

On Windows platforms, the OracleConcMgr<SID> service (where <SID> is the database <SID> and the value of the LOCAL variable) spawns the concurrent manager programs. The OracleConcMgr<SID> service is initially created by the Rapid Install program. If you need to recreate the service, use one of the following two methods:

Method 1

From the command prompt:

```
C:\> cd %COMMON_TOP%\admin\install
C:\> adsvcm.cmd <NT User> <NT Password>
```

where NT User is the user that runs the concurrent manager service, and NT Password is the password of the user that runs the service.

Method 2

Invoke the GUI program ccmsetup.exe (located in %FND_TOP%\bin), and choose the option to create the service.

Parameters

The following entries describe the concurrent manager startup parameters, which are read by the concurrent manager service and passed to the concurrent manager programs. The GUI program ccmsetup.exe (located in %FND_TOP%\bin) should be used to create and set these parameters in the Windows registry. Note that the registry settings override defaults. These parameters can also be used by the adcmctl.cmd script (see the Maintaining Oracle Applications documentation set for more information).

Database Connection

Schema Name The APPS schema name should be set to the APPS user ID.

Password This is the password to the Oracle Applications account in the database.

TNS Aliasname This is the database <SID>.

Server Startup

Use this Account Check the "Use this Account box" to specify the OS account (<NT User> and <NT Password>) that runs the CCM service. If you do not check this box, the system account launching the service will not have permission to access network resources.

Startup Options

Sleep
Number of seconds (integer) the internal concurrent manager waits between times it looks for Queue Controlled (in statuses Deactivate/Abort/Verify/Activate) requests. The default value is 60.

Restart
Number of minutes (integer) the internal concurrent manager waits before attempting to restart after abnormal termination. The default value prevents the manager from restarting after abnormal termination. The default value is N.

Queue Size
Number of pmon cycles (integer) the internal concurrent manager waits between times it checks for normal changes in concurrent manager operation. Normal changes include the start or end of a work shift and changes to concurrent manager definitions entered in the Concurrent Managers form. The default value is 1.

PMON cycle
Number of sleep cycles (integer) the internal concurrent manager waits between times it checks for concurrent managers that have failed while running a concurrent request. The default value is 20.

Printer Name
Name of the printer to which the concurrent managers send request output, where the requests are submitted from within a concurrent program if the submitting program (parent request) does not have a printer associated with it.

Enable Distributed Concurrent Processing
Check this box to enable distributed concurrent processing.

Shutdown

Deactivate/ Normal
Shuts down the concurrent managers using normal shutdown methods.

Terminate/Abort Processes
When shutting down the concurrent managers, aborts the concurrent managers.

> **Note:** This option is currently disabled. Checking this box will have no effect.

File Logging

Activation Log The name of the log file generated upon startup of the Internal Concurrent Manager. Defaults to CM_<SID>.log.

Deactivation Log The name of the log file generated upon shutdown of the Internal Concurrent Manager. Defaults to CS_<SID>.log.

Record Diagnostic Messages Checking this box will cause concurrent managers to produce diagnostic output regularly. Note that leaving this box unchecked prevents large log files.

Save Log History Checking this box prevents log files from being overwritten when the concurrent manager is started.

Starting the Concurrent Managers

Before starting the concurrent managers, you must start the Oracle Applications TNS listener on all nodes. The TNS listener must be started by the applmgr user.

For UNIX

You can start the concurrent managers from by running the script startmgr from the operating system command line. To start the concurrent manager from the operating system prompt, use the following syntax:

```
$ startmgr \
sysmgr="<APPS username>/<APPS password>" \
mgrname="<name>" \
PRINTER="<printer>" \
mailto="<userid1 userid2...>" \
restart="N|<minutes>" \
logfile="<filename>" \
sleep="<seconds>" \
pmon="<cycles>" \
quesiz="<cycles>" \
diag="Y|N"
```

All parameters are optional. You can pass parameters to the script in any order. These parameters can also be used by the adcmctl.sh script (see the Maintaining Oracle Applications documentation set for more information).

Parameters

The following entries describe the concurrent manager startup parameters. The default values apply if you do not specify different values in the startmgr script, on the command line when you run startmgr, or in your environment.

sysmgr APPS schema name should be set to the APPS schema user ID and password. You will be prompted for the password

if you omit the parameter and use the default value. The default value is $FNDNAM.

mgrname Name of the internal concurrent manager (alphanumeric characters only). The default value is std.

PRINTER Name of the printer to which the concurrent managers send request output, where the requests are submitted from within a concurrent program if the submitting program (parent request) does not have a printer associated with it.

mailto List of users who receive mail when the internal concurrent manager stops running. The default value is the user who starts managers.

restart Number of minutes (integer) the internal concurrent manager waits before attempting to restart after abnormal termination. The default value is N. The default value prevents the manager from restarting after abnormal termination.

logfile The name of the internal concurrent manager's log file. The default value is <mgrname.mgr>.

sleep Number of seconds (integer) the internal concurrent manager waits between times it looks for Queue Controlled (in statuses Deactivate/Abort/Verify/Activate) concurrent requests. The default value is 60.

pmon Number of sleep cycles (integer) the internal concurrent manager waits between times it checks for concurrent managers that have failed while running a concurrent request. The default value is 20.

quesiz Number of pmon cycles (integer) the internal concurrent manager waits between times it checks for normal changes in concurrent manager operation. Normal changes include the start or end of a work shift and changes to concurrent manager definitions entered in the Concurrent Managers form. The default value is 1.

diag diag=Y tells all concurrent managers to produce diagnostic output regularly. The default value diag=N prevents large log files.

Example

```
$ startmgr sysmgr="apps/apps" \
mgrname="std" \
PRINTER="hqseq1" \
mailto="jsmith" \
restart="N" \
logfile="mgrlog" \
sleep="30" \
pmon="5" \
quesiz="2"
```

For Windows

The OracleConcMgr<SID> service is launched from the Control Panel Services applet. It can also be launched from the command line using the following command:

```
C:\> net start OracleConcMgr<SID>
```

- Concurrent managers inherit directory privileges from the user who installs and starts them. If you plan to install and start the managers from a login other than the main applications login (applmgr) ensure that the login has the appropriate directory privileges. The Windows account that starts the concurrent manager service must be the same one that installed it.

- Startup parameter values apply in this order:
  - Values set in the registry (through the use of ccmsetup)
  - Default values

    This means that values set in the registry override the default values.

  You cannot change startup parameters while the managers are running. To put changes into effect, shut down the managers, make the necessary modifications, and restart the managers.

Restarting the Concurrent Managers

You must restart the concurrent managers whenever you start the Oracle8*i* Server database or change the concurrent manager startup parameters.

On UNIX platforms, concurrent managers append to their own log file if the log files exist when they restart. Therefore, the user who restarts the concurrent managers must either own the existing files, have write privilege for them, or delete them before restarting.

On Windows, the concurrent manager logs are overwritten when the concurrent managers are restarted. Checking the "Save Log History" option in the ccmsetup GUI tool saves the previous log files.

The concurrent managers delete temporary files when each concurrent process finishes.

If the concurrent managers stop abnormally, however, they may not delete these files.

> **Tip:** Delete temporary files only if they have not been accessed more recently than a few days ago. This helps to prevent the loss of files required by the operating system or the concurrent managers.
>
> Ideally, delete temporary files during maintenance windows when the database is down and no applications programs are active.

Network Failure Recovery

As part of its shutdown process, the ICM determines if it's being forced to shutdown due to losing its database connection. This is done by looking for specific error messages ORA-3113, ORA-3114, or ORA-1041. If one of these error messages is detected, the ICM spawns the reviver process, which attempts to make a database connection. If unsuccessful, it sleeps for a period before trying again. This continues until either a successful connection is made or it receives a signal to shut itself down.

When a successful connection is made, the process kills the old ICM database session, and then starts a new ICM using the normal start manager script. Once the ICM is restarted, it starts up any other managers that had also shut down, and normal processing resumes.

Shutting Down the Concurrent Manager Service (Windows)

The OracleConcMgr<SID> service may be shut down from the Control Panel Services applet. It can also be stopped from the command line using the following command:

```
C:\> net stop OracleConcMgr<SID>
```

Although you can shut down concurrent managers from Oracle Applications System Administrator's responsibility, this does not stop the concurrent manager service. You must still stop the concurrent manager service from the Windows Control Panel Services applet before you can restart the concurrent managers.

> **Note:** The OracleConcMgr<SID> service may take several minutes to shut down because it needs to finish processing currently running requests.

> **Warning:** Do not use the Task Manager to stop the concurrent manager service or other Applications processes unless you are advised to do so by Oracle Worldwide Support.

Removing the Concurrent Manager Service (Windows)

If you need to remove the concurrent manager service, ensure that it is not running.

To remove or delete the OracleConcMgr<SID> service, use one of the following 2 methods:

Method 1

At the command prompt, type:

```
C:\> cd %COMMON_TOP%\admin\install
C:\> adsvcm.cmd -deinstall
```

Method 2

Invoke the GUI program ccmsetup.exe, and choose the option to remove the service.

Once you have done this, you will need to reinstall the concurrent manager service in order to process concurrent requests.

File Conventions

The following tables list the locations and file naming conventions for log, output, and temporary files. The location of product log and output files depends on whether you have set up a common directory.

For UNIX

File Type	Location	Filename
Internal Concurrent Manager Log	$FND_TOP/$APPLLOG with Common Directory: $APPLCSF/$APPLLOG	<mgrname>.mgr
Concurrent Manager Log	$FND_TOP/$APPLLOG with Common Directory: $APPLCSF/$APPLLOG	w<nnn>.mgr
Request Log	Default: $<PROD>_TOP/$APPLLOG with Common Directory: $APPLCSF/$APPLLOG	l<request ID>.req
Request Output	Default: $<PROD>_TOP/$APPLOUT with Common Directory: $APPLCSF/$APPLOUT	o<request ID>.out

File Type	Location	Filename
Temporary	$APPLTMP or $REPORTS60_TMP	OF<abcd12345>.t where <abcd12345> is a random OS-generated string

Parameters

The variable parameters have the following values:

mgrname For UNIX, the name specified with the mgrname parameter in the startmgr command. If no name is specified, the filename is std.mgr.

For Windows, the name specified with the mgrname parameter in the ccmsetup.exe program. If no name is specified, the filename is std.mgr.

nnn A sequence number between 1 and 999 is generated by the concurrent processing facility.

<PROD>_TOP The product's top environment variable, such as GL_TOP.

request ID The number that identifies the concurrent request.

USERNAME Up to eight characters (uppercase) of the application username of the user that requested the concurrent process.

<abcd12345> Naming convention in which <abcd> are random letters and <12345> designate the operating system process ID of the concurrent process that generated the file.

Directory Privileges

Oracle recommends that you start the managers from the applmgr login to ensure that they inherit the correct directory privileges. The user who starts the concurrent managers then owns the log and output files that the concurrent managers create.

Oracle recommends that you start the managers from the applmgr login to ensure that they inherit the correct directory privileges.

For UNIX, any user who runs an environment file and has access to the startmgr script can start the concurrent managers.

Warning: Always start the concurrent managers from the applmgr

login if you are using parallel concurrent processing on multiple nodes.

For UNIX

This section describes directory privileges for UNIX.

Setting the startmgr User ID with setuid in UNIX

To ensure that startmgr inherits the applmgr directory privileges, you can use the UNIX setuid facility to set startmgr to the applmgr login's UNIX user ID. The concurrent managers then inherit the applmgr privileges no matter which login runs startmgr. This allows you to start or restart concurrent managers using the Administer Concurrent Managers form, regardless of the originating UNIX login. Note that you must reset the user ID with setuid if you modify or copy startmgr. Refer to your online UNIX documentation for information on setuid.

> **Note:** The use of the setuid command may cause unexpected behavior on certain platforms that employ dynamic linking of libraries. Please refer to the Oracle Applications Installation Update for your platform for any information regarding this problem.

Directory Privileges for Logins other than applmgr

If you do not set the startmgr script to the applmgr user ID and you start the managers from a login other than applmgr, that login needs to have these privileges:

- Read and execute privileges on all Oracle Applications directories
- Write privilege for all directories defined by the following variables:
  - APPLLOG (typically log directory or directories)
  - APPLOUT (typically out directory or directories)
  - APPLCSF (common directory for log and output files)
  - APPLTMP (temporary directory)
  - REPORTS60_TMP (temporary directory for Oracle Reports files)
- Write privilege for these directories: /tmp and /usr/tmp

You can verify that a login has the necessary privileges on a certain directory with this command:

```
$ ls -ld < directory>
```

Here is an example:

```
$ ls -ld /usr/tmp
drwxrwxrwx 3 root 22880 Mar 10 11:05 /usr/tmp
      ^^^
```

The three letters marked in the sample response indicate that all users have read, write, and execute privilege for the directory.

Printing

This section contains printer reference material including information on how to create and register executable printing programs.

To register printers in the Printers form of Oracle Applications, your Oracle Applications System Administrator needs to know each printer's operating system name. Your installation update tells you where to find the printer names for your platform. Installation updates may also contain other information on setting up your printers.

Printing (for UNIX)

This section contains printer reference material specific to the UNIX operating system, including information on how to create and register executable printing programs.

Standard Print Subroutine

The standard printing subroutine that you can select in the Printer Drivers form uses Oracle Application Object Library routines to print reports. This requires fewer machine resources than printing through a customized executable program or a shell command such as lp or lpr.

When you use the subroutine, there may be options available through the descriptive flexfield at the bottom of the form. These options vary by platform and may include the following: mail Notify user by electronic mail when report finishes printing.

- Mail: Notify user by electronic mail when report finishes printing.
- Priority: Set the priority for reports in the print queue.

Check your installation update for any additional options available on your platform.

Executable Printing Programs

Oracle Applications supports the use of executable programs for printing. However, we recommend that you use executable programs only to provide features unavailable through Oracle Applications printer drivers, such as:

- Interpreting special characters in the text passed to the printer. For example, you need a program to interpret 8-bit characters sent to a 7-bit compatible printer
- Interpreting arguments passed by Oracle Applications. For example, you need a program to perform different actions based on different output filenames.

If you do not need to support special features such as these, print through the standard printing subroutine and printer drivers defined in the Oracle Applications database. This makes the most efficient use of machine resources.

Upgrading to Existing Executable Programs

Because printing through the standard printer subroutine uses machine resources more effectively than printing through executable programs, we recommend the following if you used executable printing programs in the previous release of Oracle Applications:

- If predefined printer drivers can replace the executable program, simply register the drivers along with the printer types in the Printer Types form.

  For example, Oracle Applications provides a print style Landscape and the printer driver LANDSCAPESUB. They perform the same function as the program land, which enables DEC LN03 printers to print 132 characters per line.

- If no predefined drivers will work, you may be able to create a customized driver that can replace the executable. You create drivers with the Printer Drivers form.

- If you cannot replace the executable with a simple printer driver definition, you can continue to use the executable by registering it or the shell script that calls it with Oracle Applications.

Writing an Executable Program

Executable printing programs can format report output through escape sequences or a printer programming language. Creating them requires a thorough knowledge of both printer operation and a computer programming language. Follow the guidelines in this section if you need to create an executable printing program.

Printer Styles

An executable program should be able to format report output for various print styles, including these:

- Portrait: 80 columns wide, 66 lines per page

- Landscape 132 characters wide, 66 lines per page (62 lines per page on A4 style paper)

- Landwide 180 characters wide, 66 lines per page (62 lines per page on A4 style paper)

Formatting Arguments

If the program handles formatting for various print styles internally, you can pass arguments from the printer drivers to the program to determine which print style to use.

If the program does not contain print style formatting commands, you can define the commands in a shell script that calls the program. You then define the shell script as the printing program in a printer driver and pass arguments that determine the print style

from the driver to the script.

The printer driver that calls the executable program or shell script must be able to pass the following arguments:

- Name of the destination printer
- Number of copies to print
- Banner on title page
- Filename

Initialization and Reset

You do not have to add printer initialization and reset strings to your program if you can define these strings in the Printer Drivers form.

Character Mode Oracle Reports Commands

We recommend that you design your executable programs to work with the standard Oracle Reports print drivers. The following standard drivers are located in the $FND_TOP/$APPLREP directory:

- P.prt - Portrait style
- L.prt - Landscape style
- W.prt - Landwide style
- A.prt - A4 style

The program should not misinterpret the commands for bold on, bold off, and page size that the standard drivers imbed in Oracle Applications reports. If necessary, you can create customized Oracle Reports drivers as described below.

Location of Program

When you have compiled and linked the source code or written a shell script, move the program to the $APPLBIN subdirectory under the top directory of your custom development area. Keep copies of the source file in your custom development area as a backup.

Creating Customized Character Mode Oracle Reports Print Drivers

The Oracle Reports print drivers set the font styles for italics, underlining, and bolding. If your executable printing program cannot use the standard Oracle Reports drivers, create a customized driver for each print style you will use with the program.

To create a customized driver, copy L.prt, P.prt, A.prt, or W.prt from $FND_TOP/$APPLREP to your custom development area. Modify a standard driver as needed for your executable printing program. Give the customized driver a new filename but keep the .prt extension. Then copy the customized driver to $FND_TOP/$APPLREP.

Tip: Use the executable program name and print style letter as the driver name. For example, use HPLJ3P.prt for portrait style printing with the executable program HPLJ3. When you print in portrait style with this program, the concurrent managers pass the HPLJ3P.prt driver to Oracle Reports as DESFORMAT=HPLJ3P.

Registering Executable Programs

When you have created your executable programs and, optionally, your shell scripts and Oracle Reports drivers, register them in the Printer Drivers form.

1. Navigate to the Printer Drivers window and create a new printer driver name. Also add the user name, description, and platform.
2. In the SRW Driver field, enter the name (without the .prt extension) of a standard or customized Oracle Reports driver. All drivers must be in the directory $FND_TOP/$APPLREP.
3. Enter Program as the driver method.
4. Enter No in the Spool File field.
5. Enter No in the Standard Input field.
6. In the program name field, enter the name of the executable program or the shell script that calls it. Include the full path name if this file is not in the $FND_TOP/bin directory.
7. Add the arguments that Oracle Applications passes to the program or shell script. The driver must pass the following to the executable program:
   - Name of the destination printer
   - Number of copies to print
   - Banner on title page
   - Filename
   - Add the initialization and reset strings to the appropriate fields if the program does not send these strings to the printer.

Printing (For Windows)

This section contains printer reference material specific to the Windows operating system. It also explains how to create and register executable printing programs.

Operating System Names for Printers

To register printers in the Printers window of Oracle Applications, your Oracle Applications system administrator needs to know each printer's operating system name. For Windows, you can obtain the printer's name from the Printers folder in My Computer. If you register an invalid printer, the operating system's default printer will be used instead.

Standard Print Subroutine

The standard printing subroutine that you can select in the Printer Drivers form uses Oracle Application Object Library routines to print reports. This requires fewer machine resources than printing through a customized executable program DOS command such as PRINT.

Executable Printing Programs

Oracle Applications supports the use of executable programs for printing. However, we recommend that you use executable programs only to provide features unavailable through Oracle Applications printer drivers, such as:

- Interpreting special characters in the text passed to the printer. For example, you need a program to interpret 8-bit characters sent to a 7-bit compatible printer.

- Interpreting arguments passed by Oracle Applications. For example, you need a program to perform different actions based on different output filenames.

If you do not need to support special features such as these, print through the standard printing subroutine and printer drivers defined in the Oracle Applications database. This makes the most efficient use of machine resources.

Upgrading Existing Executable Programs

Because printing through the standard printer subroutine uses machine resources more effectively than printing through executable programs, we recommend the following if you used executable printing programs in the previous release of Oracle Applications:

- If predefined printer drivers can replace the executable program, simply register the drivers along with the printer types in the Printer Types form.

  For example, Oracle Applications provides a print style Landscape and the printer driver LANDSCAPESUB. They perform the same function as the program land, which enables DEC LN03 printers to print 132 characters per line.

- If no predefined drivers will work, you may be able to create a customized driver that can replace the executable. You create drivers with the Printer Drivers form.

- If you cannot replace the executable with a simple printer driver definition, you can continue to use the executable by registering it or the shell script that calls it with Oracle Applications.

Executable printing programs can format report output through escape sequences or a printer programming language. Creating them requires a thorough knowledge of both printer operation and a computer programming language. Follow the guidelines in this section if you need to create an executable printing program.

Printer Styles

An executable program should be able to format report output for various print styles, including these:

- Portrait: 80 columns wide, 66 lines per page
- Landscape 132 characters wide, 66 lines per page (62 lines per page on A4 style paper)
- Landwide 180 characters wide, 66 lines per page (62 lines per page on A4 style paper)

Formatting Arguments

If the program handles formatting for various print styles internally, you can pass arguments from the printer drivers to the program to determine which print style to use.

If the program does not contain print style formatting commands, you can define the commands in a command file that calls the program. You then define the .cmd file as the printing program in a printer driver and pass arguments that determine the print style from the driver to the script.

The printer driver that calls the executable program or .cmd file must be able to pass the following arguments:

- Name of the destination printer
- Number of copies to print
- Banner on title page
- Filename

Initialization and Reset

You do not have to add printer initialization and reset strings to your program if you can define these strings in the Printer Drivers form.

Character Mode Oracle Reports Commands

We recommend that you design your executable programs to work with the standard Oracle Reports print drivers. The following standard drivers are located in the %FND_TOP%\%APPLREP% directory:

- P.prt - Portrait style

- L.prt - Landscape style
- W.prt - Landwide style
- A.prt - A4 style

The program should not misinterpret the commands for bold on, bold off, and page size that the standard drivers imbed in Oracle Applications reports. If necessary, you can create customized Oracle Reports drivers as described in the next section.

Location of Program

When you have compiled and linked the source code or written a command file, move the program to the %APPLBIN% subdirectory under the top directory of your custom development area. Keep copies of the source file in your custom development area as a backup.

Creating Customized Character Mode Oracle Reports Print Drivers

The Oracle Reports print drivers set the font styles for italics, underlining, and bolding. If your executable printing program cannot use the standard Oracle Reports drivers, create a customized driver for each print style you will use with the program.

To create a customized driver, copy L.prt, P.prt, A.prt, or W.prt from %FND_TOP%\%APPLREP% to your custom development area. Modify a standard driver as needed for your executable printing program. Give the customized driver a new filename but keep the .prt extension. Then copy the customized driver to %FND_TOP%\%APPLREP%.

> **Tip:** Use the executable program name and print style letter as the driver name. For example, use HPLJ3P.prt for portrait style printing with the executable program HPLJ3. When you print in portrait style with this program, the concurrent managers pass the HPLJ3P.prt driver to Oracle Reports as DESFORMAT=HPLJ3P.

For more information, see the Oracle Reports Developer documentation.

Registering Executable Programs

When you have created your executable programs and, optionally, your Oracle Reports drivers, register them in the Printer Drivers form.

Define a printer driver and corresponding print style for each print style that your executable program supports. Complete the following steps to register an executable program for a printer driver:

1. Navigate to the Printer Drivers form and create a new printer driver name. Also add the user name, description, and platform.
2. In the SRW Driver field, enter the name (without the .prt extension) of a standard or customized Oracle Reports driver. All drivers must be in the directory

%FND_TOP%\%APPLREP%.

3. Enter Program as the driver method.

4. Enter No in the Spool File field.

5. Enter No in the Standard Input field.

6. In the Program Name field, enter the name of the executable program or the command file that calls it. Include the full path name if this file is not in the %FND_TOP%\bin directory.

7. Add the arguments that Oracle Applications passes to the program or command file. The driver must pass the following to the executable program:

   - Name of the destination printer
   - Number of copies to print
   - Banner on title page
   - Filename
   - Add the initialization and reset strings to the appropriate fields if the program does not send these strings to the printer.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Drivers Window, page 9-53

9

Printers

Printers and Printing

Oracle Applications offers two printing solutions to handle all your printing requirements. For most printing needs, the Pasta Utility offers quick setup and easy maintenance. For additional flexibility, Oracle Applications allows you to define your own printer drivers and print styles.

- To set up your printers using Pasta, see Printer Setup with Pasta, page 9-10.

  Note: Pasta is required to print using UTF8.

- To set up your printers using a custom solution, see Customizing Printing Support in Oracle Applications, page 9-31.

Overview

When you run an Oracle Applications report, Oracle Reports generates and formats the output. A completed report is sent to the operating system by the concurrent manager, which issues an operating system print command, or calls a custom print program or subroutine that issues an operating system print command.

Oracle Reports and Report Generation

Oracle Reports includes page break, carriage return, line feed, text bold on/off, and text underline on/off instructions within the output file. The values are retrieved from a SQL*ReportWriter (SRW) driver file.

When the report is generated for online viewing, Oracle Reports uses the SRW driver named by the print style in the Print Styles form.

When the report is to be printed, Oracle Reports uses the SRW driver named by the Oracle Applications printer driver in the Printer Drivers form.

The dimensions of a report are determined by the columns and rows values in the print style, defined using the Print Styles form. These values override the width and height values in an SRW driver file.

Concurrent Manager Issues or Calls a Print Command

When a report program finishes running, the concurrent manager prepends an initialization string, and appends a printer reset string to the output file. Both strings are defined using the Printer Drivers form.

An Oracle Applications printer driver is invoked by issuing a print command or by calling a print program or subroutine.

When the printer driver method is *Command*, the concurrent manager can issue an operating system print command and arguments, entered in the Arguments field of the Printer Drivers form.

When the printer driver method is *Program*, the concurrent manager can call a custom print program, named (along with its path) in the Name field of the Printer Drivers form. Arguments to the program may be entered in the form's Arguments field.

When the printer driver method is *Subroutine*, the concurrent manager calls a predefined Oracle Applications subroutine that passes a print command and arguments to the printer via the operating system. The subroutine name is entered in the Program Name field of the Printer Drivers form.

The concurrent manager may provide values for four arguments to an operating system print command or custom print program:

- the name of the file to be printed
- the operating system name of the target printer
- the title of the file, which appears on a header page if it is printed
- the number of copies to be printed

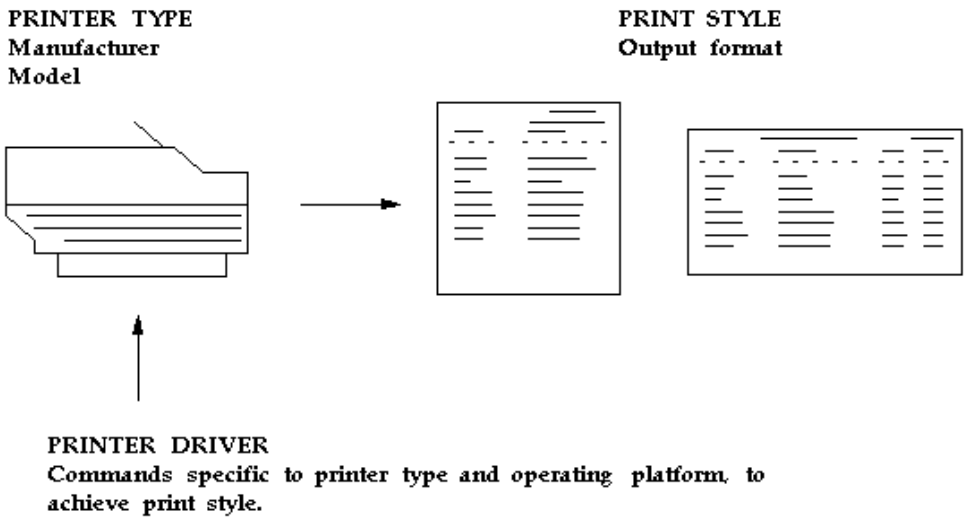
Printer Types, Print Styles, and Printer Drivers

The commands that a printer can understand vary from one type of printer to another. A *printer type* identifies a printer by manufacturer and model.

A *print style* tellsthe printer how a printed output should look. A *printer driver* delivers commands that tell the printer how to output the specified print style.

The ability to print a report in a particular print style depends on the type of printer the report file is sent to.

For each print style that a particular type of printer can print, a printer driver specific to the printer type and the operating system is required.



Printer Types

The printer type is the printer manufacturer and model. Two examples are a DEC LN03 printer and an HP Laserjet III printer.

Print Styles

A Print style defines the page format for a printer, such as the number of lines per page, the width of each line, and whether a header page should be printed.

Each printer type can have one or more associated print styles.

Print styles allow you to set up report dimensions on a variety of printers. You can tailor your page setups while providing consistent-looking reports from printer to printer.

For example, users may wish to print a menu report with a wider left margin to allow for hole punching the paper. As System Administrator, you register this new style, which users can then access if the printer type supports it.

At report submission time, users select the style in which to output the report.

- Only styles available on the destination printer are displayed.
- Some concurrent programs predefine either the printer or the print style, and these values cannot be changed.

Printer Drivers

To print in a particular style from a particular printer type, you define a printer driver.

A printer driver is the mechanism that delivers a report's output along with its commands to the target printer.

Concurrent managers determine what drivers to use depending on what the print style is and what printer type the report is to be sent to.

You need to define a printer driver for each print style that you want to use with a specific printer type on a specific platform.

Sequence of Printing Events

The concurrent manager associates a print style and a printer driver with the destination printer's printer type. This combination of print style and printer driver is defined in the Printer Types form.

A printer driver tells the destination printer how to interpret the format. An SRW Driver formats text and sets page breaks within an Oracle Reports file.

Sequence of Printing Events - Example

The following is an example of the sequence of printing events.

1. A user submits a request to run a report from the Run Reports form.
2. A request to run the report is added to the requests table.
3. A concurrent manager reads the request.
4. The concurrent manager calls Oracle Reports to run the report, and passes the SRW Driver name. If Report Copies = 0 and the Printer field is blank, the Print Style's SRW Driver is used. If Report Copies > 0 and Printer is required, then the Printer Driver's SRW Driver is used.

   The concurrent manager passes Print Style information (Columns and Rows) to Oracle Reports (overrides SRW Driver width and height if the report is to be printed).
5. A report is created using Oracle Reports. The concurrent manager attaches Printer Driver information to the file. It prepends the initialization string and appends the reset string.

   The concurrent manager also passes suppress header option information from the Printer Styles form.
6. The concurrent manager issues an operating system print command with the arguments Destination Printer, Filename (including path), Number of Copies to print, and Filename for the Title on the banner page.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers , page 9-2

Setting Character-Mode vs. Bitmap Printing, page 9-5

Setting up Your Printers, page 9-8

Fast-Track Printing with Pasta , page 9-10

Customizing Printing Support in Oracle Applications, page 9-31

Postscript Printing in UNIX, page 9-43

Setting Character-Mode vs. Bitmap Printing

Running Character mode Oracle Reports Concurrent Programs

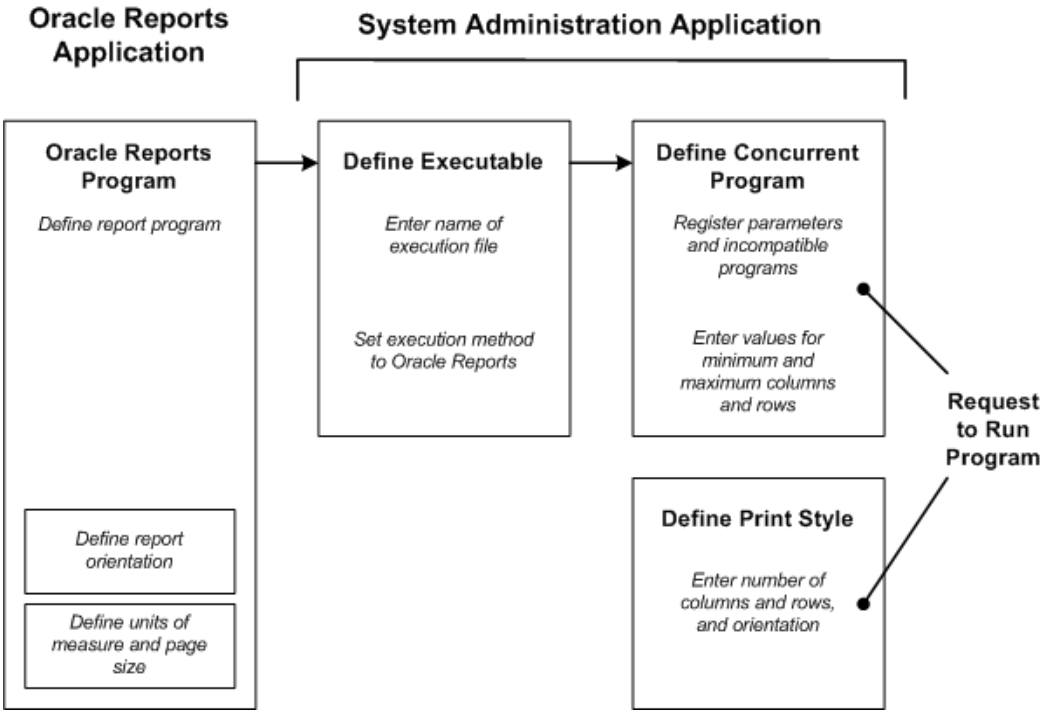
Character mode Oracle Reports programs take their page dimensions and orientation from the print style associated with the request to run the program.

Some print styles are predefined, and a System Administrator can define additional styles, if necessary.

After you create an Oracle Reports program, you create a corresponding concurrent program executable with the Oracle Reports execution method.

You then define a concurrent program for that executable, registering any parameters and incompatible programs. You also enter the minimum column and row length, orientation, and print style.

Running Character Mode Oracle Reports Programs



Running Bitmap Oracle Reports Concurrent Programs

Bitmap Oracle Reports programs are defined similarly in Oracle Reports and in the Concurrent Program Executable form.

To run an Oracle Reports program in bitmap mode, query the concurrent program's definition in the Concurrent Programs form, and choose PostScript in the Format field.

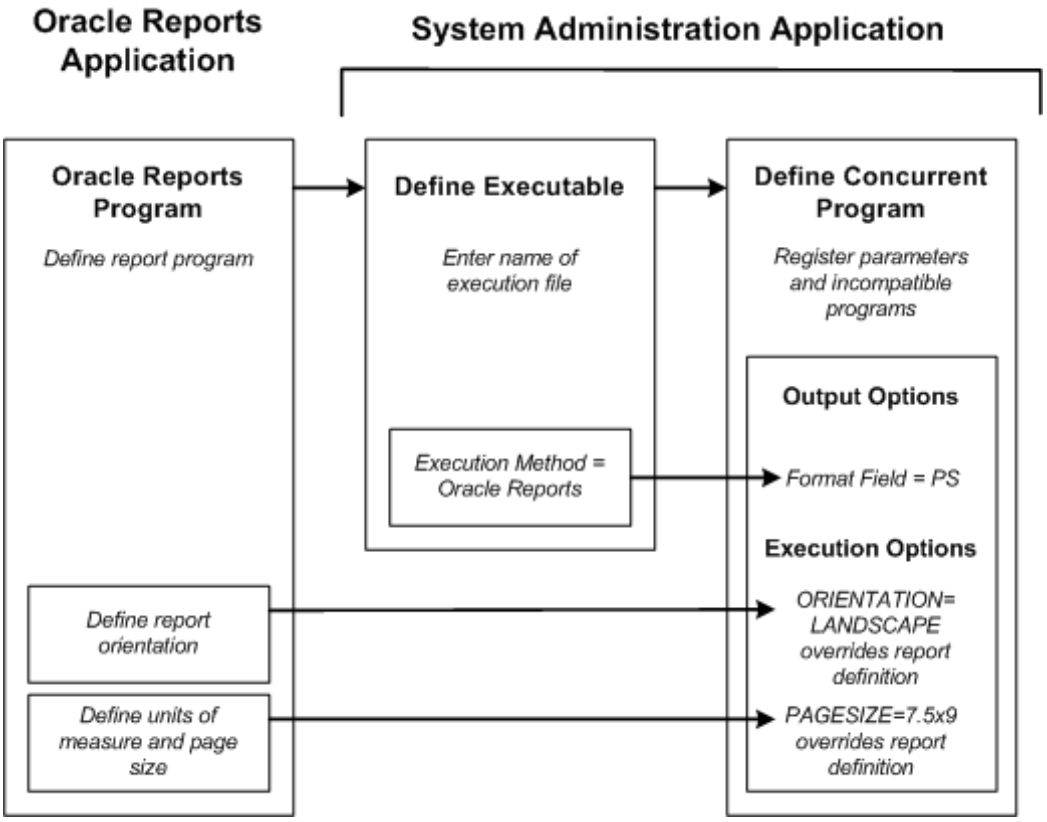
Bitmap Oracle Reports programs take their page dimensions and orientation from the program's definition (note: when printing a bitmap report, a print style is still required).

If you wish to override the program's definitions, you can enter values in the Execution Options field for ORIENTATION and PAGESIZE.

When entering more than one execution option, each option should be separated by a *single* space. There should be no spaces before or after the options. For example:

```
ORIENTATION=LANDSCAPE PAGESIZE=7.5x9
```

Running Bitmap Oracle Reports Programs



Notes about PAGESIZE in the Execution Options field

In Oracle Reports, when defining a report the units and size of the report are specified in the menu under Report->Global Properties->Unit of Measurement.

For bitmapped reports, <width>x<height> for PAGESIZE is usually in inches; however, this depends on the particular report definition.

You can enter the PAGESIZE parameter in the Execution Options field of the Concurrent Programs form (for bitmapped reports only) when you want to override the values specified in the report definition. For example:

```
PAGESIZE=7.5x9
```

If the dimensions specified with the PAGESIZE parameter are smaller than what the report was designed for, you will generate a "REP-1212" error.

Related Topics

Overview of Printers and Printing, page 9-1

Setting up Your Printers, page 9-8

Creating Custom Printer Drivers, page 9-31

Printer Drivers field help, page 9-53

Setting Up Your Printers

Oracle Applications provides you with predefined printer types, print styles, and printer drivers. Use the Printer Types form to query the combinations of print style and printer driver that support each type of printer you have. Customize the predefined components as necessary. See: Customizing Printing Support in Oracle Applications, page 9-31.

> **Important:** Predefined printing components may have to be modified for different printer types and/or operating platforms.

Forms for Defining Printer Support

You use four forms to define printer support.

Printer Types

You must define any printer types used at your site that are not shipped with Oracle Applications. It is on this form that you associate the print style with a printer driver for the particular printer type.

Printers

You register a printer so that Oracle Applications recognizes the printer and can forward to it the output from a report program.

To register a printer you specify the printer's operating system name, which uniquely identifies the printer, and select the printer type. The printer type must already be defined.

For example, if you want users of Oracle Applications to be able to print to a newly purchased printer, you:

- Register the operating system name of the new printer (for example, printer39), and select the printer type (for example, LN03).
- If the correct printer type is not defined, you must define the new printer type before you can register the printer.

Print Styles

To generate a report, the print style values for columns and rows are passed by the concurrent manager to Oracle Reports. A print style determines the dimensions of your report, or the number of rows and columns per page.

Printer Drivers

A printer driver includes the initialization and reset strings that format and restart a printer. You need a defined printer driver for each print style that you plan to use with a specific printer type on a specific platform.

Printing Setup Interrelationships

- Many printers can be registered as the same printer type.
- A printer type can support multiple print styles.
- A printer driver must be assigned to a printer type for each print style.
- Many printer drivers can support the same print style.
- Many printer drivers can support the same printer type.

See: Overview of Printers and Printing, page 9-1

Printer Setup Information Is Cached On Demand

Printer setup information such as Printer Type definitions, Print Style definitions, and Printer Driver definitions, are read into memory (cached) the first time the information is required to print a program's output.

The cache area that holds printer setup information is private to the concurrent managers. Printer setup information remains cached in memory until the concurrent managers are restarted, when the values are erased and new values are cached.

> **Important:** You should issue a *Restart concurrent manager* command for all currently active managers whenever you edit an existing Printer Type, Print Style, or Printer Driver (unless the type, style or driver has not been referred to or cached yet).

See: Controlling Concurrent Managers, page 7-35

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers, page 9-2

Customizing Printing Support in Oracle Applications, page 9-31

Creating Custom Printer Drivers, page 9-31

Printer Setup with Pasta

Overview

Pasta is an Oracle Applications utility that converts text report files to PostScript and also enables the printing of custom PostScript reports from Oracle Applications. The reports can then be directed to any PostScript printer.

Setting up your system to use Pasta is much simpler than the standard Oracle Applications printer setup procedure. The Printer Type, Printer Driver, and SRW driver files are provided. The only setup required to begin printing is the registration of the printer with Oracle E-Business Suite.

Many printing options can be defined using the Pasta configuration file (pasta.cfg). You no longer need to maintain multiple drivers and styles for each printer.

Pasta is provided as an executable named FNDPSTAX.

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Setup for Basic Printing with Pasta

The following setup can be used to enable any PostScript printer to print text or PostScript reports in the following styles: Landscape, Landwide, Portrait, or Dynamic.

Use the Printers window to register your printer:

1. Enter your printer's name as defined in the operating system and applications.
2. Select "--Pasta Universal Printer" from the list of values for the printer Type.

You are now ready to print text and PostScript reports from your PostScript printer using the default Pasta configuration.

For more information on the Printers window, see Printers Window, page 9-49.

For more information on setting options in the Pasta configuration file, see: Configuration File Options, page 9-18.

Related Topics

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Defining Configuration Files for Specific Printers

The pasta.cfg file controls many printing options. You can use the default file for many printers for multiple languages. However, if you have printers that require special setup, you can customize these options by creating a configuration file for each printer.

Copy the pasta.cfg file to pasta_<printer name>.cfg. Make the necessary changes to the file. Pasta automatically looks for a printer-specific file name. If it does not find one, Pasta then uses the default file, pasta.cfg.

For example, suppose you have a printer named "hqprinter" for which you want to set the page height and width to letter size:

1. Copy $FND_TOP/resource/pasta.cfg to $FND_TOP/resource/pasta_hqprinter.cfg where hqprinter is the name of the printer as defined on the operating system.

2. In the pasta_hqprinter.cfg file, edit the paper size options:

   Set pagewidth=8.5

   Set pageheight=11

Pasta will now use the options as defined in the pasta_hqprinter.cfg file when printing to the hqprinter.

For more information on setting options in the Pasta configuration file, see: Configuration File Options, page 9-18.

Using a Different Configuration File as the Default

You can change the file that is defined as the default configuration file for Pasta by using the -F command line parameter.

For example, suppose you create a PCL print-specific configuration file named pcl.cfg. Set the FNDPSTAX command line option as follows:

```
-Fpcl.cfg
```

Pasta will look for pcl_<printer>.cfg first, and if it is not found, Pasta will use pcl.cfg as the default.

These files must be placed under the $FND_TOP/resource directory.

The -F command line parameter can be set in the Arguments field of the Printer Drivers window. See Printer Drivers Window, page 9-53.

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Configuration File Options, page 9-18

Command Line Parameters, page 9-24

Modify an Existing Printer Type to Use Pasta

If your printer is already assigned to a printer type that contains styles and drivers that you want to maintain, you can add Pasta to the existing printer type.

To add Pasta to an existing printer type, you must associate one of the Pasta printer drivers with a print style on the Printer Types window.

The seeded Pasta printer drivers are:

- PASTA_LANDSCAPE
- PASTA_PORTRAIT
- PASTA_LANDWIDE
- PASTA_DYNAMIC

You can associate a Pasta driver with an existing print style, or you can create a new print style. To create a new print style, use the Print Styles window. For more information on defining a print style, see Print Styles Window, page 9-50.

1. Query your existing printer type in the Printer Types window.
2. In the Style field, select the style to which you want to assign a Pasta driver.

   Or, if you are assigning Pasta to a style already defined for the printer type, delete the driver in the Driver Name field currently associated with the style.
3. In the Driver Name field, select the appropriate Pasta driver.

   For more information on the Printer Types window, see Printer Types Window, page 9-47.

Add a New Printer Type to Use Pasta

If you want to add a new Printer Type, you can also add Pasta to your new printer type.

1. Navigate to the Printer Types window.
2. Enter the Type of printer.
3. In the Style field use the list of values to select the style to which you want to assign a Pasta driver.
4. In the Driver Name field, select the appropriate Pasta driver from the list of values:
5. PASTA_LANDSCAPE
6. PASTA_PORTRAIT
7. PASTA_LANDWIDE
8. PASTA_DYNAMIC

   For more information on the Printer Types window, see Printer Types Window, page 9-47.

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Configuration File Options, page 9-18

Command Line Parameters, page 9-24

Setting Margins

The margins on your printed output are determined by the margin settings in the pasta.cfg file and the printable area defined by your printer. In order to set your margins properly you must know the printable area for your specific printer and adjust the margin parameter settings in the pasta.cfg file accordingly. The margin parameters are leftMargin, rightMargin, topMargin, and bottomMargin.

For example, suppose you want to set the left margin to one inch. If the printable area for your printer begins at .25 inches from the left, then you must set the leftMargin option to .75 in the pasta.cfg file.

For more information on setting options in the Pasta configuration file, see: Configuration File Options, page 9-18.

Refer to your printer's documentation for information on its printable area.

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Command Line Parameters, page 9-24

Printing a Report Generated Using the noprint Option

When you use the "noprint" printer option to generate a report, you can still have the option of printing it later using Pasta.

Before running the report, associate the noprint printer with the "--Pasta Universal Printer Type" in the Printers form.

> **Note:** You must restart the concurrent manager for this to take effect. See Printer Setup Information is Cached on Demand, page 9-9.

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Configuration File Options, page 9-18

Command Line Parameters, page 9-24

Generating Other Formats Using the Preprocessing Option

Pasta can use a preprocessing option to invoke any executable that supports an input file and an output file (filter program). Pasta will invoke the filter program to preprocess the Pasta output before passing it to the printing command. By using the preprocessing option, you can generate output formats other than the formats that Pasta currently supports. For example, by invoking products such as Adobe Acrobat Distiller Server or Ghostscript, you can generate PDF output or PCL output.

> **Important:** Ensure that the executable for the preprocess program is placed in your path.

The preprocessing command is a configuration file (pasta.cfg) option. This option uses {infile} and {outfile}.

- {infile} is the file generated by Pasta to be used as input to the preprocessing command. It is a temporary file and will be deleted after it is passed to the preprocessing command.
- {outfile} is the output file generated by the preprocessing command. Pasta names it temporarily and it will be deleted after it is passed to the printing command.

  If you want to keep the {outfile}, you can name it by using the "outFile" pasta configuration file option (see Configuration File Options, page 9-18), or the "-o" command line option (see Command Line Parameters, page 9-24). Pasta will copy {outfile} to the file you specify.

Example for Generating PCL Output

In this example, "gs" is Ghostscript and "pxlmono" is a device used with HP black and white PCL XL printers (Laserjet 5 and 6 family).

In the pasta.cfg file, enter the following for the preprocess option:

```
preprocess=gs -q -dNOPAUSE -dBATCH -sDEVICE=pxlmono
-sOutputFile={outfile} {infile}
```

To get a list of output devices available in Ghostscript go to http://www.gnu.org/software/ghostscript/devices.html

Example for Generating PDF Output

In this example, "ps2pdf" is a shell script bundled with Ghostscript. The ps2pdf script can convert a PostScript file to a PDF file.

In most cases you cannot send a PDF file to the printer command because the printer command cannot understand PDF. Set the noPrint option to "y" or use the "-np" (no print) command line option if you do not want Pasta to send the PDF file to the printer.

Use the outFile option to define the destination on the middle tier for the output file. You can use {inputfile} in the outFile option. Pasta will replace it with the actual input file name (without the path) specified by the "-f" (input file) command line option.

1. Using the basic Pasta setup procedure, define a printer in the Printers window called "PDFfile". Assign the "--Pasta Universal Printer" type to the printer.

   For more information about defining a printer, see: Setup for Basic Printing with Pasta, page 9-10.

2. Create a configuration file for the PDFfile printer called "pasta_PDFfile.cfg".

   For more information about defining a configuration file, see: Defining Configuration Files for Specific Printers, page 9-11.

3. In the pasta_PDFfile.cfg file, enter the following for the preprocess option:

   ```
   preprocess=ps2pdf {infile} {outfile}
   noPrint=y
   outFile=<APPLTMP>/{inputfile}.pdf
   ```

Related Topics

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Font Source

If your printer does not have the necessary fonts installed, Pasta embeds the required glyphs as a small font in your report from the font files.

If you do not want to use the fonts provided by Oracle you can specify the font you want by using the font name option in the pasta.cfg file. You can use any TrueType fonts on your middle tier; or, if your printer has a font installed suitable for the language of your report, you can utilize the printer font.

The form of the pasta.cfg option is:

```
Font.Default.<Style>=<TrueType font file name>
```

or

```
Font.Default.<Style>=printer:<Printer font file name>
```

Example using TrueType font file:

```
Font.Default.Plain=<FND_TOP>/<APPLRSC>/ADUO.ttf
Font.Default.Bold=<FND_TOP>/<APPLRSC>/ADUOB.ttf
```

Example using Printer Font:

```
Font.Default.Plain=printer:Courier
Font.Default.Bold=printer:Courier-Bold
```

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Language-Specific Font Support, page 9-17

Configuration File Options, page 9-18

Command Line Parameters, page 9-24

Language-Specific Font Support

The default pasta.cfg file contains font settings for languages. You override the default setting in the language-specific section of the pasta.cfg file.

For example, if you want to override the default setting of Courier font for French language reports to use Helvetica instead, add the following to the end of the pasta.cfg file:

```
[FRENCH]
Font.Default.Plain = printer:Helvetica
Font.Default.Bold = printer:Helvetica-Bold
```

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Configuration File Options, page 9-18

Command Line Parameters, page 9-24

Configuration File Options

The configuration file pasta.cfg governs many printing options. The file is a normal ASCII text file that has a simple format described below.

The options are listed in the following two tables. They are divided into Generic Options and Arabic, Hebrew, and Thai options. For each option are listed the Key Name, the Default Value (if applicable), the Description, and the Equivalent Command Line Option (if applicable).

> **Note:** For options that also have command line equivalents, Oracle recommends that you set the value in the configuration file.

Generic Options

Key Name	Default Value	Description	Command Line Equivalent
outputFormat	ps	Two output formats are supported: PostScript ("ps") and text ("text"). When the output format is text, you can specify the output character set by the Oracle character set name (for example, text.WE8ISO8859P1). If you use "auto" as the output character set (text.auto), Pasta uses the appropriate character set according to the NLS_LANGUAGE value in the FND_LANGUAGES table.	-x

Key Name	Default Value	Description	Command Line Equivalent
textAutoCharset	The default value is taken from the FND_LANGUAGES table.	When the outputFormat is set to "text.auto", Pasta uses a default character set for each language based on the language and character set mappings in the FND_LANGUAGES table. To override the default setting for a language, use the textAutoCharset option in the Language-Specific section of the pasta.cfg file. For example, to override the default character set for Japanese to use JA16EUC instead, enter the following: [Japanese] textAutoCharset=JA16SJIS	N/A
preprocess	N/A	Use this option to convert the output file. Enter a preprocessing command to invoke any executable that supports an input file and an output file (filter program). Pasta will invoke the filter program before passing the file to the printing command.	N/A
printCommand	N/A	Specific print command for Unix platform.	N/A

Key Name	Default Value	Description	Command Line Equivalent
ntPrintCommand	N/A	Specific print command for Windows platform.	N/A
outFile	N/A	If you want to save the output file, use this option to define the ouput file name and its destination on the middle tier.	-o
noPrint	N/A	Set this option to "y" if you do not want Pasta to produce printed output.	-np
duplex	default	Specifies duplex printing. Options are "y", "n", or "default" (uses the printer-side setting).	N/A
embednumcopies	y	Set this option to "y" to embed the number of copies to be printed in a PostScript file. Using this option will eliminate the header page normally printed between reports.	N/A
copysort	y	If you set embednumcopies to "y", you can choose to have the copies collated, by setting this option to "y".	N/A
heightScaleRate	1.0	Adjusts the space between lines.	-h

Key Name	Default Value	Description	Command Line Equivalent
widthScaleRate	1.0	Adjusts the space between characters.	-w
pagewidth	8.27	Adjusts the page width.	-pw
pageheight	11.64	Adjusts the page height.	-ph
topMargin	.25	Adjusts the top margin.	N/A
bottomMargin	.25	Adjusts the bottom margin.	N/A
rightMargin	.25	Adjusts the right margin.	N/A
leftMargin	.25	Adjusts the left margin.	N/A
Font.<Face>.<Style>	N/A	Specify the TrueType font file name. The <Face> must be either "Default" or the actual font face name (such as Helvetica). The <Style> must be either "Plain", "Bold", "Italic", or "BoldItalic".	N/A
fontsize	7.8 (for landscape) 10.0 (for portrait)	Font size in points. If this is not set, the font size is calculated automatically.	-s

Key Name	Default Value	Description	Command Line Equivalent
boldfontsize	Size in points (Default value from the fontsize parameter)	Font size for bold font in points. This option is used mostly when Font.Default.Bold is used for specifying a barcode font.	-bs
psEmbeddedFontType	type3 (default) type42	A type of font that will be embedded in PostScript.	
tabsize	8	Pasta replaces a tab with the number of spaces specified in this option.	N/A
errorlogfile	(standard error output)	Set this option to have Pasta create a log file.	-el

Arabic, Hebrew and Thai Options

Key Name	Default Value	Description	Equivalent Command Line Option
thai_space_compensation	n	In the Thai language, some characters are combined into one glyph. If this option is set to "y", Pasta will align your report by adding spaces at the end of the column that includes combined characters.	N/A

Key Name	Default Value	Description	Equivalent Command Line Option
bidi_algorithm	oracle	If you set this option to "unicode", Pasta follows the Unicode BiDi algorithm. Setting the value to "oracle" will use Oracle's algorithm.	N/A
direction	default	Options are "ltr" (left-to-right), "rtl" (right-to-left), and "default" (depends on NLS_LANGUAGE setting).	N/A
dolayout	y	To layout the text, set this option to "y". If not, set it to "n".	N/A
doshaping	y	To shape the text, set this option to "y". If not, set it to "n". This option is for Arabic only.	N/A
numerals	context	Possible values are "arabic" for Arabic numerals, "hindi" for Hindi numerals, or "context" to use Arabic or Hindi depending on the context. Required for Arabic data only.	N/A

Related Topics

Setup for Basic Printing with Pasta, page 9-10

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Command Line Parameters

When using the Pasta utility from the command line, you can use the options below.

```
FNDPSTAX [options]
```

-c<number>	<number> specifies the number of copies to print.
-el<logfile>	Specifies the error log file's path and name. The path is the absolute path to the error log file.
-f<filename>	<filename> specifies the input file name. Example: -fmyfile.txt
-F<cfgfile>	Specifies the configuration file's path and name.
-h<rate>	Adjusts the space between lines. The default value is 1.0. If <rate> is larger than 1, the space between lines will be larger.
-w<rate>	Adjusts the space between characters. The default value is 1.0. If <rate> is larger than 1.0, the space between characters will be larger.
-l	Print in landscape mode. (The default is portrait).
-o<filename>	<filename> specifies the output file name.
-s<size>	Overrides the font size option in pasta.cfg.
-bs<size>	Overrides the font size in -s and fontsize for bold font.
-np	No print option.
-ph<height>	<height> specifies the paper height in inches.
-pw<width>	<width> specifies the paper width in inches.
-pform<psfile><pf file>	Converts a PostScript file <ps file> to a PrintForm file <pf file>.

-pf<pf file>	<pf file> specifies a PrintForm file to be merged in a Pasta output at runtime.
-pn<printername>	<printername> specifies the printer name.
-t<banner title>	Banner option for use with the Unix lp command.
-v	Displays the version number.
-x<ps|text.[charset|auto]>	Specifies the output format. Two output formats are supported: PostScript ("ps") and text ("text").

Related Topics

Defining Configuration Files for Specific Printers, page 9-11

Modify/Add Printer Type to Use Pasta, page 9-12

Setting Margins, page 9-13

Printing a Report Using the noprint Option, page 9-14

Generating Other Formats Using the Preprocessing Option, page 9-15

Font Source, page 9-16

Language-Specific Font Support, page 9-17

Configuration File Options, page 9-18

Using PrintForms

PrintForms is a feature of the E-Business Suite that allows you to create your own background template for a report and use it at print-time either as a background image (such as your company logo) or as a standard form (such as an invoice).

The PrintForm process combines the input report file and the background template file to generate one PostScript file, which is then passed to a PostScript printer.

Because this process is executed at print-time, the need to customize report definitions is eliminated, which also simplifies maintenance. Moreover, using the PrintForm as a standard form can replace printing solutions that require expensive preprinted stationery and specialized printers.

There are two phases to using PrintForms:

1. Create the PrintForm
2. Deploy the PrintForm to Oracle Applications

Note: The following instructions are general guidelines for creating

and using PrintForms. For more detailed instructions see the "Pasta User's Guide Release 3.0" available in Note 239196.1 on Oracle *MetaLink*.

Create the PrintForm

Creating a PrintForm is a five-step process:

1. Create an image file using the supported editor of your choice.
2. Convert the image file to a PostScript file (.ps).
3. Convert the PostScript file to a PrintForm file (.pf) to use as your template.
4. Place the PrintForm file in the appropriate directory.
5. Update printer driver information (in the Printer Drivers window) to use the PrintForm.

Create an image file

The following editors can be used to create the image file that will become your PrintForm template.

- Oracle Reports
- Microsoft Word, Excel, or Powerpoint (with Adobe PDFMaker)
- Any Windows Application
- Any Application that generates PDF

When creating your image file ensure that the paper size, margin settings, and style of the document are appropriate for the report you are fitting the template to.

> **Note:** Even if you need only one type of image (such as a company logo added to all your reports), you must create one PrintForm for each print style that you plan to use (such as portrait, landscape, and landwide) with the appropriate placement of the image for the style.

Convert the image file to a PostScript file

The PrintForm template file must be created from a PostScript file. The PostScript file can be generated in one of the following ways:

1. Generate a PostScript file from Oracle Reports.

2. Convert any Windows application file (such as Microsoft Word or PowerPoint) to PostScript using the Adobe Generic Printer Driver "save to file" option.

3. Convert any PDF file to PostScript using a third-party utility, such as Xpdf (available for Windows and Unix platforms).

   Note: For information about downloading and using Xpdf see http://www.foolabs.com/xpdf/

Convert the PostScript file to a PrintForm file

Use a Pasta command line parameter to convert the PostScript file to a PrintForm (.pf) file. During this process comments are added to the PostScript file.

The syntax for the command line is as follows:

```
FNDPSTAX -pform <PostScript template file>.ps <PrintForm file>.pf
```

For more information about Pasta command line parameters, see Command Line Parameters, page 9-24.

Deploying the PrintForm to Oracle Applications consists of the following steps:

1. Place the PrintForm in the APPL_TOP.

2. Update the printer driver information.

3. Restart the Concurrent Manager.

Place the PrintForm in the appropriate directory

You can place the PrintForm file anywhere under APPL_TOP in the Concurrent Processing node, but Oracle recommends that you put the file under the FND_TOP/resource/<lang> directory, where <lang> is the language code (such as US for English or KO for Korean).

Update the printer driver information

1. Navigate to the Printer Drivers form.

2. Query up the printer driver to which you want to add the PrintForm option.

3. Edit the Arguments field for the printer driver to use the PrintForm in the appropriate directory.

   For example, if you place a PrintForm called logo_ls.pf in the $FND_TOP/resource/US directory and you want to apply the PrintForm to the PASTA_LANDSCAPE driver, edit the Arguments field as follows:

```
-pn$PROFILES$.PRINTER -f$PROFILES$.FILENAME -c$PROFILES$.CONC_COPIES -l
-pf<FND_TOP>/resource/US/logo_ls.pf
```

where <FND_TOP> is the actual path for your $FND_TOP.

For more information about this form, see Printer Drivers Window, page 9-53.

Restart the Concurrent Manager

You must restart the concurrent manager for this to take effect. See Printer Setup Information is Cached on Demand, page 9-9.

Common UNIX Printing System (CUPS) Integration

CUPS is a third-party product that provides printing service to most PostScript printers and raster printers. It is based on the Internet Printing Protocol (IPP) technology making it open to any system that supports IPP. CUPS can run on various kinds of Unix systems, such as Solaris, HP-UX, AIX, Tru64, and major Linux distributions. CUPS is provided under the GNU GPL license.

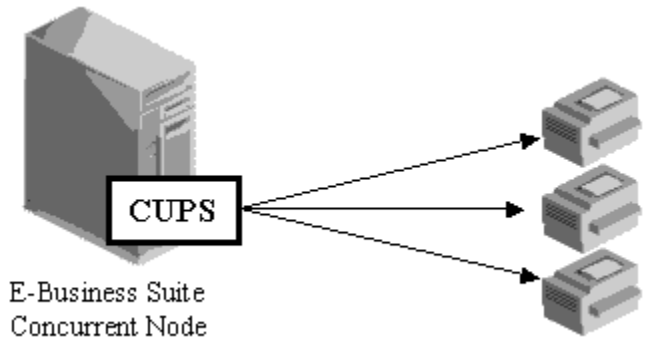
For more details about CUPS, refer to http://www.cups.org/

Two integration scenarios are described in the following sections:

- All printers connected to a single CUPS server residing on the E-Business Suite concurrent processing node
- Multiple CUPS distributed servers with printers connected to each CUPS server

Scenario 1: All printers connected to a single CUPS server residing on the E-Business Suite concurrent processing node

This scenario is graphically depicted in the following diagram:



There are no special configuration steps required if the printer system has been replaced with CUPS on the server.

If you maintain both the System V printing system and CUPS, you must define a new

printer type with the required print styles and printer drivers.

1. Create a new Pasta configuration file for CUPS in the $FND_TOP/resource directory (example: pasta_cups.cfg).

2. Ensure that the lp command that is set for the printCommand parameter in the Pasta configuration file is a CUPS lp command. You can specify the full path to the command if you maintain both the UNIX standard lp command and the CUPS lp command.

3. Create a set of printer drivers for each print style with the configuration file created in Step 1.

   Example:

   Printer Driver Name: PASTA_LANDSCAPE_CUPS

   Arguments:

   ```
   -pn$PROFILES$.PRINTER -f$PROFILES$.FILENAME -cPROFILES$.CONC_COPIES
   -l -Fpasta_cups.cfg
   ```

   You can refer to the existing printer driver definitions created by Oracle, such as PASTA_LANDSCAPE, PASTA_PORTRAIT, or PASTA_LANDWIDE.

4. Create a printer type for the CUPS printer.

   Example:

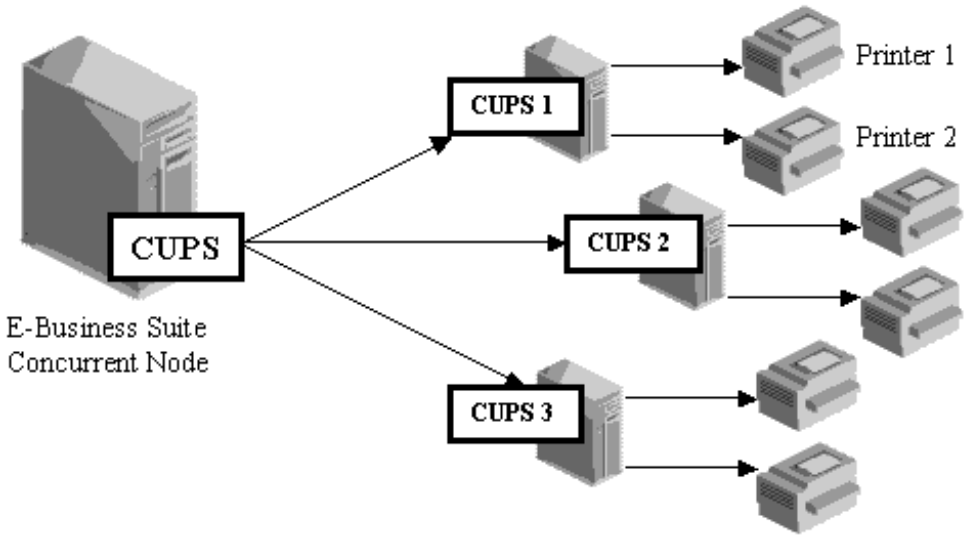
   Printer Type: Pasta printer for CUPS

Print Style	Printer Driver
LANDSCAPE	PASTA_LANDSCAPE_CUPS
PORTRAIT	PASTA_PORTRAIT_CUPS
LANDWIDE	PASTA_LANDWIDE_CUPS

5. Register your printer with the print style.

Scenario 2: Multiple CUPS distributed servers with printers connected to each CUPS server

The following figure displays multiple CUPS distributed servers with multiple printers attached to each:



In this scenario, one printer type is required for each server on which a CUPS process is running with a physical printer.

1. Create a Pasta configuration file in the $FND_TOP/resource directory for each server on which the CUPS process is running .

   For example: pasta_cups1.cfg, pasta_cups2.cfg, pasta_cups3.cfg

2. For each configuration file, make the following change:

   ```
   printCommand=lp -h<cupshost> -d{printername}
   ```

   where <cupshost> is replaced with the actual server name.

   Make sure that the lp command is a CUPS lp command. You can specify the full path to the command if you maintain both the UNIX standard lp command and the CUPS lp command.

   Leave {printername} as is. This parameter will be replaced by Pasta at runtime with the actual printer name.

3. Create a set of printer drivers for each print style with the configuration file created in Step 1.

   Example:

   Printer Driver Name: PASTA_LANDSCAPE_CUPS1

   Arguments:

   ```
   -pn$PROFILES$.PRINTER -f$PROFILES$.FILENAME -cPROFILES$.CONC_COPIES
   -l -Fpasta_cups1.cfg
   ```

   You can refer to the existing printer driver definitions created by Oracle, such as PASTA_LANDSCAPE, PASTA_PORTRAIT, or PASTA_LANDWIDE.

4. Create a printer type for each server in Oracle Applications.

   For example: Pasta printer CUPS1, Pasta printer CUPS2, Pasta printer CUPS3

   Add the required print styles associated with the printer drivers to the printer type.

   Example:

   Printer Type: Pasta printer for CUPS1

Print Style	Printer Driver
LANDSCAPE	PASTA_LANDSCAPE_CUPS1
PORTRAIT	PASTA_PORTRAIT_CUPS1
LANDWIDE	PASTA_LANDWIDE_CUPS1

1. Register your printer with the print style.

   For example, if printer1 is defined on the server CUPS1 and you have created Pasta printer CUPS1 printer type for the server, register printer1 with the Pasta printer CUPS1 printer type.

Related Topics

Printers Window, page 9-49

Printer Types Window, page 9-47

Print Styles Window, page 9-50

Printer Drivers Window, page 9-53

Defining Configuration Files for Specific Printers, page 9-11

Configuration File Options, page 9-18

Customizing Printing Support in Oracle Applications

Oracle Applications provides numerous predefined printer types with which you can identify your printers, as well as print styles that define the dimensions of Oracle Reports output files, and printer drivers that instruct the various printer types how to output the selected print style.

Use the Printer Types form to query the combinations of print style and printer driver that support each type of printer you have.

Important: Predefined printing components may have to be modified for different printer types and/or operating platforms.

For example, if a blank page is being printed after each printed page, the number of rows defined for the print style may need to be reduced, or an escape sequence that is being interpreted differently, creating a page eject command, may have to be rewritten.

Verify and, if necessary, Customize Printer Driver Definitions

Upon installation for any printer type you are using, verify your printer driver definitions, particularly the following:

- Initialization string

  Print a short report to verify the page's printing orientation. If you want to change the printer's default font for the report, you would include that information in the Initialization string.

- Reset string

  Print two short reports with different printing orientations, for example, one that is landscape and another that is portrait, to verify the printer is resetting itself properly.

- Arguments

  Print a short report to verify the arguments to the operating system's print command or a custom print program are being interpreted correctly.

If you need to define a new print style, verify the printer driver you assign to the new print style, for any printer type you use.

Verify and, if necessary, Customize Oracle Reports SRW Drivers

If you have a printer type that does not properly interpret the control characters in the SRW driver files that set page breaks, bold on/off and underline on/off attributes in your Oracle Reports files, you can copy the SRW driver file and modify it.

Creating Custom Printer Drivers

If necessary, edit the Initialization string and the Reset string for the printer type you are using. Refer to your printer's user guide for instructions. The Initialization and Reset fields appear on the Printer Drivers form.

Edit your Initialization string or Reset string if:

- Your printer type requires different control characters.

- The control characters have a different meaning due to your operating system and platform.
- Language translation changes the meaning of the control characters. The printer needs special control characters to select different character sets.
- You want to change the printer's default font for the report (Initialization string only).

Printer Driver Methods

There are three methods to invoke a printer driver:

Command

The concurrent manager can issue an operating system print command and its arguments.

An operating system print command, along with all its arguments, is entered in the Arguments field of the Printer Drivers form.

Program

The concurrent manager can call a custom print program and pass arguments to the program.

The name of a custom print program is entered in the Program Name field and any arguments to be passed to the program are entered in the Arguments field of the Printer Drivers form.

Subroutine

The concurrent manager can call a predefined Oracle Applications subroutine that passes a print command and arguments to the printer via the operating system.

A subroutine is predefined by Oracle Applications, and the name is entered in the Program Name field of the Printer Drivers form.

The arguments field is disregarded when the driver method is *Subroutine*. However, the concurrent manager reads the Initialization and Reset escape sequences.

On UNIX systems, the subroutine method, unlike the command method, does not start an operating system shell along with the print command.

Example - Using the Program Driver Method

The Program driver method allows customers to define their own custom print programs. For example, your company might want to write a custom program that opens a file, allows the file to be edited and saved under a second filename, then sends the second (edited) file on to the printer by issuing the print command. This method of

issuing print commands is called a *filter*.

Location for Custom Print Programs

To call a custom print program using the Printer Drivers form, the program name, including the full pathto the program, should be entered in the Program Name field.

The path to the program name is not necessary if the program's location can be identified by the operating system's PATH environment variable (i.e., is in the $PATH variable name).

For platforms where the equivalent of a $PATH variable doesn't exist, then use the full path name. A path can be up to 255 characters.

Custom print programs are not registered as concurrent programs with Oracle Application Object Library, but are called after the concurrent process has completed.

Using Operating System Shell Scripts

For operating system shell scripts, the printer driver method can be either command or program, as long as you populate the argument field correctly.

The script for a command shell procedure should reside in:

```
$FND_TOP/$APPLBIN.
```

Arguments That a Concurrent Manager Can Supply Values For

The concurrent manager can supply four different values as arguments to the operating system print command it issues, or to a custom print program that it calls. An example of using these values as arguments follows.

Example - Entering a Print Command and Arguments

In this example, the UNIX print command "lp" is entered along with the arguments that a concurrent manager can supply values for. While print commands vary, the tokens for which values are retrieved are always the same.

Because print commands are operating system dependent, please refer to *Installing Oracle Applications*.

Example - Printer Drivers Form's Arguments field:

```
lp -d$PROFILES$.PRINTER -n$PROFILES$.CONC_COPIES -t"$PROFILES$.TITLE"
$PROFILES$.FILENAME
```

The following table lists arguments and their contents for the UNIX lp print command:

Argument Syntax	Token and Value Retrieved
-d$PROFILES$.PRINTER **-d** calls out the destination printer.	**$PROFILES$.PRINTER** retrieves the operating system name of the printer associated with the request.
-n$PROFILES$.CONC_COPIES **-n** calls out the number of copies to print.	**$PROFILES$.CONC_COPIES** retrieves the value of the profile option *Concurrent:Report Copies*, unless this value is updated at runtime.
-t"$PROFILES$.TITLE" **-t** calls out the report title to print on a banner or header page.	"**$PROFILES$.TITLE**" retrieves the title of the output file, typically titled as *Application username.Request ID*. For example, if user John Smith ran a report whose concurrent request ID was 64225, the title would be JSMITH.64225. This is operating system dependent.
$PROFILES$.FILENAME	**$PROFILES$.FILENAME** calls out the filename of the report to be printed. The value retrieved is the output file name, including the path to the file. Note that this file is a temporary file created from information from the Printer Driver definition (from the Printer Drivers window) and the actual output for the report.

In addition, the following table lists arguments and their contents in the cases of an original print request and reprint requests:

Argument Syntax	Original Print Request Value Retrieved	Reprint Value Retrieved
$PROFILE$.ORIGREQID	Request ID.	Original request ID.
$PROFILE$.ORIGUSERNAME	User name.	Original user name.
$PROFILE$.REPREQID	0 (zero)	Reprint request ID.
$PROFILE$.REPUSERNAME	NULL	Reprint request user name.

Argument Syntax	Original Print Request Value Retrieved	Reprint Value Retrieved
$PROFILE$.OUTFILENAME	Output file path and filename.	Output file path and filename. Note that the file here contains only the report output.
$PROFILE$.OUTFILEHOST	Host name.	Host name.

Using Standard Input

When Standard Input is set to Yes, the printer driver accepts standard input, so you can feed a report's output directly to the printer from standard input. Two examples of using standard input are:

- when you run a pipe in UNIX such as "cat myfile | lpr" rather than "lpr myfile", the output file is sent to the stdin (standard input).
- the UNIX command lpr, which accepts standard input when a filename is not specified.

The Standard Input field should be set to No when the Driver Method is set to Program or Subroutine. Unless the program accepts standard input, the Standard Input field should always be set to No.

> **Important:** When Standard Input is set to No, the print command issued by the concurrent manager runs asynchronously. That is, the concurrent manager issues the command, and does not wait for an operating system response.

Using Initialization and Reset Strings

Use the initialization and reset strings to set and reset the orientation, character set and line density for your printer.

Initialization and reset strings consist of control characters and escape sequences.

- A control character can be represented by " ^ " followed by another character.
- An escape sequence can be identified by either " /e " or " \e ".

> **Important:** You see "/e" for escape sequences defined using the

Printer Drivers form (because you cannot enter the backslash (\ ) character into a form when your terminal definition uses backslash as the [Menu] key). You see "\e" for escape sequences originally defined in .pdf files that were later upgraded in Oracle Applications printer drivers.

For nonprintable characters, you may represent their value in octal mode. For example, 0x26 is represented as " /046 ". As an example, if you need to represent the escape sequence:

^ [^ L ^ [l 6 D (0 x 26)

you can represent it as:

/ e ^ L / e l 6 D / 0 4 6

Using a Spool File

When Spool File is set to No, then a temporary file is created where the initialization and reset strings are inserted, and the file is sent to the print command or program.

Set the Spool File to Yes only if the print program creates its own temporary file. This option is recommended when using the Program driver method and the print program creates its own temporary file.

This option helps to reduce the creation of temporary files, since the concurrent manager will not create a temporary file when Spool File is set to Yes.

When Spool File is set to Yes, it is recommended that the:

- Standard Input be set to No
- Initialization and reset fields are null.

This option does not apply to the Subroutine driver method.

Creating Custom SRW Drivers

SRW drivers are read by Oracle Reports when a report is generated, and insert control characters that tell the destination printer where to set page breaks, and which characters to format as bold or underlined.

SRW drivers only pertain to Oracle Reports output files. An SRW driver is used during the generation of a report. A printer driver is used when the completed output file is sent to the printer.

SRW drivers are designed for the DEC LN03 printer, and all printers that understand the same control characters as the LN03.

Location and Content of SRW Driver Files

SRW driver files reside in $FND_TOP/$APPLREP, and have the file extension ".prt". The predefined SRW file names are:

- A.prt
- P.prt
- L.prt
- PD.prt
- W.prt

Creating a Custom SRW Driver

You can customize any of the SRW driver files to support a printer type that is not correctly interpreting the control characters used to set page breaks and format text as bold or underlined in Oracle Reports files.

For example, you may need to change the control characters that instruct the printer to set a page break.

```
on an LN03 on an XYZ LaserInk
new page ... ^L  ^[E
```

If you need to change formatting control characters for page breaks, underlined text, or bold text in Oracle Reports:

- Copy the .prt file (SRW driver) and rename the copy.
- Modify the new file with new control characters.
- Place the modified copy of the SRW driver file in $FND_TOP/$APPLREP.
- Associate the new driver with a print style and/or printer driver definition.

> **Important:** Copy the SRW driver (.prt file) and rename it before starting any text editing.

SRW Drivers - Print Styles and Printer Drivers

When the concurrent manager calls Oracle Reports to run a report, the SRW driver name is passed as a parameter to Oracle Reports.

The SRW driver is not required because some customers might be using styles or printer drivers for non-Oracle Reports programs.

The SRW driver name you enter in the Print Styles and Printer Drivers forms is used in slightly different ways depending on whether you are printing or simply viewing the report.

If you run an Oracle Reports program without printing the output file, the SRW driver associated with the report's print style is used.

If you run an Oracle Reports program and print the output file, the SRW driver that is correct for the destination printer type is chosen by selecting the SRW driver associated with the printer driver.

Related Topics

Overview of Printers and Printing, page 9-1

Creating Custom Printer Drivers, page 9-31

Print Styles field help, page 9-50

Printer Drivers field help, page 9-53

Hierarchy of Printer and Print Style Assignments

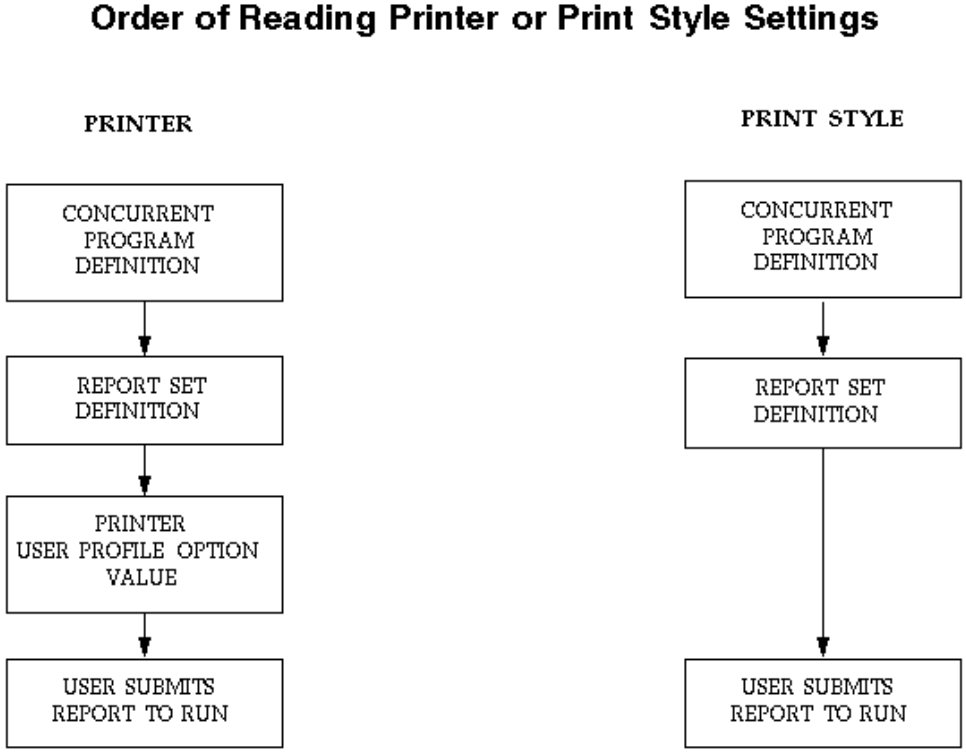
A printer and a print style can be chosen and their identities can be included in a concurrent program's definition. When a concurrent program is defined to send its output to a specific printer, or is required to generate its output in a specific print style, those values cannot be overridden by users, or by report set default settings, or by user profile default settings.

Often, a default value can be set in more than one way. This leads to a hierarchical relationship among the various default settings, where one default takes precedence over another. The diagram below illustrates the order of how printer or print style values are read by the concurrent manager when submitting a report program to run.

> **Important:** Defining a concurrent program with a default print style, or requiring a concurrent program to output a specific print style, does not make that style available at a printer. You must assign the print style, and its corresponding printer driver, to each printer type you wish to print from.

The concurrent manager reads the printer value using the following hierarchy: concurrent program definition, report set definition, printer user profile option value, and value specified by user during report submission.

The concurrent manager reads the print style value using the following hierarchy: concurrent program definition, report set definition, and value specified by user during report submission.



Hierarchy of Printer Assignments

As System Administrator, you can restrict concurrent programs and reports to direct their output to a specific printer. Restricting a program or report's output to a specific printer overrides user profile option settings and prevents report set or user runtime printer choices.

If a printer is not included as part of a concurrent program's definition, then default printer settings may be entered, as indicated in the table below. Users can override any default setting at runtime.

The following table describes the printer assignment hierarchy:

Form	Explanation
Concurrent Programs *System Administrator*	As System Administrator, you can define a concurrent program to always direct its output to only one specific printer. This setting cannot be overridden at runtime or when defining a report in a report set.

Form	Explanation
Request Set *System Administrator*	As System Administrator, you can assign a default printer to a report within a report set.
Request Set *Application Users*	Users can assign a default printer to a report within a report set, when they own the report set. This default setting can be changed by the System Administrator.
Personal Profile Values *Application Users*	Users can assign a default printer for all their reports using their Personal Profile Values form. This assignment overrides the default Printer profile option set by the System Administrator.
System Profile Values *System Administrator*	As System Administrator, you can assign a default printer to an installation site, Oracle application, responsibility, or user. Users can override this setting at runtime.

Hierarchy of Print Style Assignments

As System Administrator, you can require concurrent programs and reports to generate their output in a specific print style. Requiring a program's or a report's output to be in a specific print style prevents report set or user runtime print style choices.

Requirements for alternate print styles

All concurrent programs whose execution method is "Oracle Reports" require a print style to be selected when the program is defined. When the print style is not designated as a *required* print style, then other print styles may be selected, either as a default for a report in a report set, or at runtime when submitting the report, if two conditions are satisfied:

- The print style complies with the concurrent program's minimum values for columns and rows (entered on the Concurrent Programs form).
- The print style has been assigned to the destination printer's printer type (entered on the Printer Types form).

The following table describes the print style assignment hierarchy:

Form	Explanation
Concurrent Programs *System Administrator*	As System Administrator, you can require a concurrent program to generate its output in a specific print style. This setting cannot be overridden at runtime or when defining a report in a report set. If a Print Style is entered in a program definition, but is not required, it serves as the first default setting to be read.
Request Set *System Administrator*	As System Administrator, you can assign a default print style to a report within a report set.
Request Set *Application Users*	Users can assign a default print style to a report within a report set, when they own the report set. This default setting can be changed by the System Administrator.

System Administrator Printer and Print Style Settings

Program Definitions, Printers and Print Styles

As System Administrator you can restrict programs to send their output files only to a specified printer, for example, a printer in a secure office, using the Concurrent Programs form. You can also require a report to generate its output in a specific print style.

Assigning Default Printers and Print Styles to Reports in a Set

As System Administrator you can identify a default printer for each report within a report set, and assign a default print style for each report, using the Request Set form.

Assigning Default Printers Using Profile Options

As System Administrator you can identify a printer as a *default printer* for your installation site, a specific Oracle Application, a specific responsibility, or any of your end users, by setting the "Printer" user profile option in the System Profile Values window.

Users can override a default profile option value by:

- Setting their own personal "Printer" profile option using their Personal Profile Values form.
- Selecting another (available) printer at runtime when submitting a report.

End User Printer and Print Style Settings

End users may:

- Set default print styles for reports in their report sets, using their Request Set form.
- Identify a default printer of their own by using the Personal Profile Values form.

  Users may override the default profile option setting their System Administrator defines.
- Choose any available printer and print style when running reports, when using the Run Reports form.

  If a default printer or print style displays, users may override the default if other printers or print styles are available.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers, page 9-2

Setting up Your Printers, page 9-8

Printer Setup with Pasta: , page 9-10

Printers, page 9-49

Print Styles, page 9-50

Printer Drivers, page 9-53

Postscript Printing in UNIX

You can convert your report output files into postscript format when printing in some UNIX environments by using the *enscript* UNIX utility.

> **Important:** Refer to your UNIX documentation before using enscript. Usage and the arguments employed by enscript may be specific to your platform.

Concurrent Manager Arguments

The concurrent manager can supply four different values as arguments to an operating system print command or custom print program. See the example of using all four values provided by the concurrent manager. See: Example - Entering a Print Command and Arguments, page 9-34

See the example of using the enscript UNIX utility and two of the values the concurrent manager supplies as arguments. See: Example - Using the UNIX Enscript Command, page 9-44.

Enscript Arguments and Print Styles

The following table lists some sample enscript arguments, using the Courier font, for converting a report's output into postscript for the portrait, landscape, landwide, and A4 print styles.

Print Style	Enscript Arguments	Explanation	Result
Portrait	-fCourier10	Font is Courier 10 point.	80 characters portrait
Landscape	-r -fCourier8	**-r** rotates the printer's output 90 degrees to print in landscape mode. Font is Courier 8 point.	132 characters landscape
Landwide	-r -fCourier6	**-r** rotates the printer's output 90 degrees to print in landscape mode. Font is Courier 6 point.	180 characters landscape
A4	-fCourier10	Font is Courier 10 point.	132 characters landscape (A4 paper)

Example - Using Enscript to Print Postscript

In this example, the enscript command, followed by its arguments, is entered in the Arguments field of the Printer Drivers window, and the Driver Method is set to *Command*.

Printer Drivers window Arguments field:

```
enscript -r -fCourier8 -B -P$PROFILES$.PRINTER $PROFILES$.FILENAME
```

The following table explains the syntax for the enscript command.

Syntax	Explanation
-r	Enscript argument. Rotates the printer's output 90 degrees to print in landscape mode.
-fCourier8	Enscript argument. **-f** selects the font, in this example the font is Courier with a point size of 8.
-B	Enscript argument. Omits page headings.
-P$PROFILES$.PRINTER	Enscript argument. **-P** precedes the name of the printer which the output is sent to. Concurrent manager token. **$PROFILES$.PRINTER** retrieves the operating system name of the printer associated with the request.
$PROFILES$.FILENAME	Concurrent manager token. **$PROFILES$.FILENAME** calls out the filename of the report to be printed. The value retrieved is the output file name, including the path to the file. Note that this file is a temporary file created from information from the Printer Driver definition (from the Printer Drivers window) and the actual output for the report.

In this example, the UNIX enscript command is entered along with two arguments that a concurrent manager can supply values for.

- Since the argument "$PROFILE$.CONC_COPIES" is not used, the number of copies to be printed is set by the enscript default (which is usually one).
- Since the argument "$PROFILE$.TITLE" is not used, the concurrent manager does not provide a value for printing the report title on a banner or header page.

The following table lists additional arguments and their contents in the cases of an original print request and reprint requests:

Argument Syntax	Original Print Request Value Retrieved	Reprint Value Retrieved
$PROFILE$.ORIGREQID	Request ID.	Original request ID.
$PROFILE$.ORIGUSERNAME	User name.	Original user name.
$PROFILE$.REPREQID	0 (zero)	Reprint request ID.
$PROFILE$.REPUSERNAME	NULL	Reprint request user name.
$PROFILE$.OUTFILENAME	Output file path and filename.	Output file path and filename. Note that the file here contains only the report output.
$PROFILE$.OUTFILEHOST	Host name.	Host name.

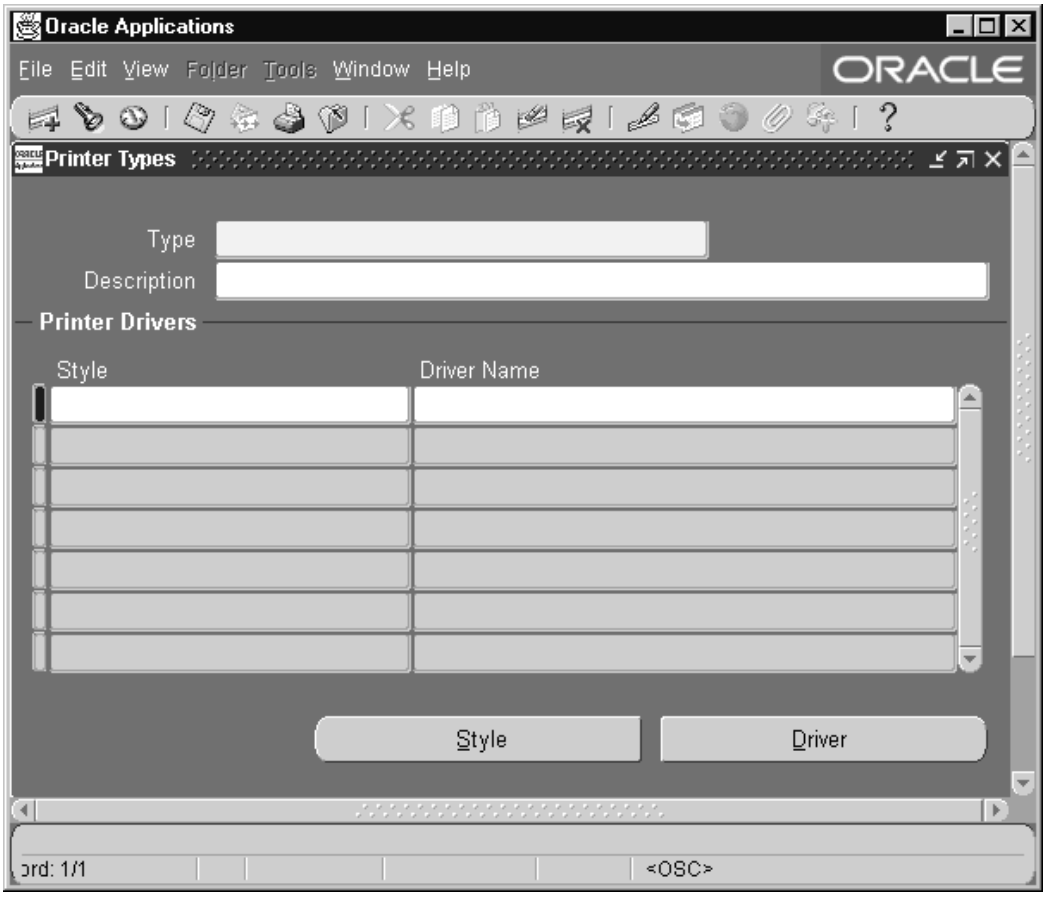
Related Topics

Overview of Printers and Printing, page 9-1

Setting up Your Printers, page 9-8

Hierarchy of Printer and Print Style Assignments, page 9-39

Printer Types Window



Use this window to define a printer type and to assign print styles and their corresponding printer drivers to the printer type.

Defining printer types allows you to assign print style and printer driver definitions to any number of printers by registering the printers as a specific "type".

When users choose a printer to send a report to, the available print styles are normally determined by the printer type.

Concurrent programs, however, can be defined to require their report output in a specific print style. For example, some Oracle Reports programs may require a specific print style in order to print correctly.

> **Important:** You should issue a *Restart concurrent manager* command for all currently active managers whenever you edit an existing Printer Type, Print Style, or Printer Driver.

See: Controlling Concurrent Managers, page 7-35

Printer Types Block

Type

Enter a name for a printer type. Example printer types might be "LINE" for a line printer or "LN03" for an LN03 model printer.

You select this printer type when you register a printer using the Printers window.

Printer Drivers Block

Use this block to assign print styles and printer drivers to your printer types.

The Style button opens the Printer Styles window.

The Driver button opens the Printer Drivers window.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers , page 9-2

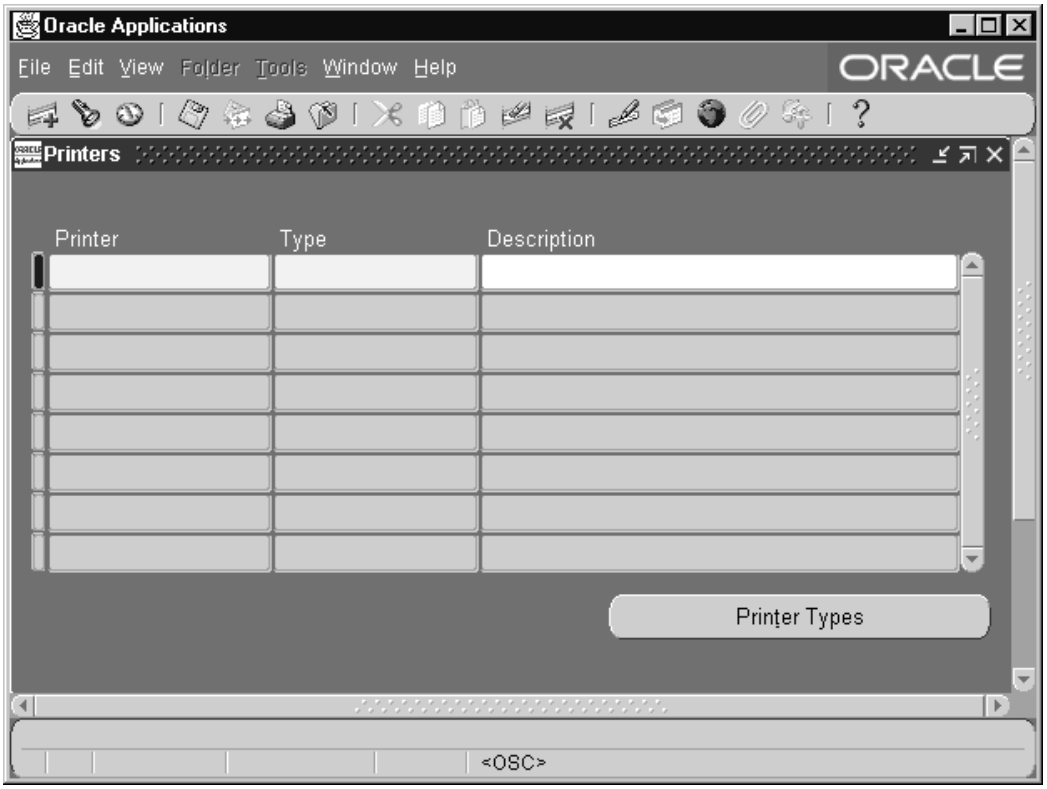
Setting up Your Printers, page 9-8

Printers, page 9-49

Print Styles, page 9-50

Printer Drivers, page 9-53

Printers Window



- Register printers with Oracle Applications by entering the operating system's name for the printer and assigning it a printer type.
- You must register a printer before you can print reports from it, using Oracle Applications.
- You can only register a printer with a previously defined printer type. Use the Printer Types window to define printer types.
- You can specify the default printer to which a user submits reports by setting the " Printer" user profile option.

Printers Block

Printer

Enter the name your operating system specifies for the printer.

Type

Select your printer type (i.e., manufacturer and model). Some reports require a printer of a specific type in order to print correctly.

You can only select a previously defined printer type. Use the Printer Types button to open a window to define a printer type.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers, page 9-2

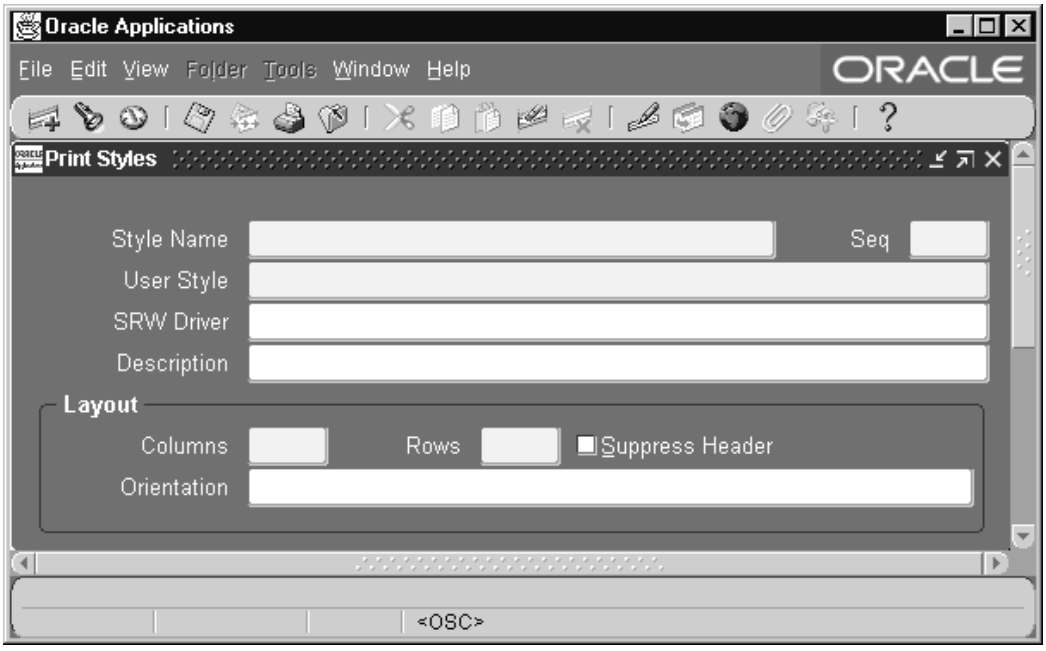
Setting up Your Printers, page 9-8

Printer Types field help, page 9-47

Print Styles field help, page 9-50

Printer Drivers field help, page 9-53

Print Styles Window



Use this window to define print styles. A print style describes how your report should be printed. For example, print style determines the:

- Number of lines per page
- Width of each line

- Page orientation (e.g., portrait or landscape)

Oracle Applications reports are designed to work with standard, shipped print styles. The following print styles are predefined:

- Portrait
- Landscape
- Landwide
- A4
- Dynamic Portrait

Not all reports work with all print styles. You may define additional print styles to customize your reports.

Once defined, a print style cannot be deleted.

Print Styles Block

Define a print style. The combination of Name and User Name uniquely identifies a print style.

> **Important:** You should issue a *Restart concurrent manager* command for all currently active managers whenever you edit an existing Printer Type, Print Style, or Printer Driver.

See: Controlling Concurrent Managers, page 7-35.

Style Name

Multiple print styles display alphabetically in a list window.

You cannot update a print style's name.

Sequence

Enter a number that determines the display sequence for your print style when performing a query in this window. A negative sequence number appears before zero, and zero appears before a positive sequence number.

User Style

This user name does not appear anywhere except this window.

SRW Driver

Enter the name of the Oracle Reports (SRW) driver to be called when printing an applications report generated by Oracle Reports. This field is used only by applications reports generated by Oracle Reports.

Layout Block

Columns

Enter the number of columns your print style defines.

Rows

Enter the number of rows your print style defines.

Suppress Header

Reports may print with a header page that indicates who requested the report and when. Check the Supress Header check box to define a print style that suppresses printing of this header page.

For example, suppressing the header page when printing checks prevents a check from being overwritten and maintains the orderly sequence of check numbers.

Orientation

Enter the orientation of your printed page, for example, portrait or landscape.

Related Topics

Overview of Printers and Printing, page 9-1

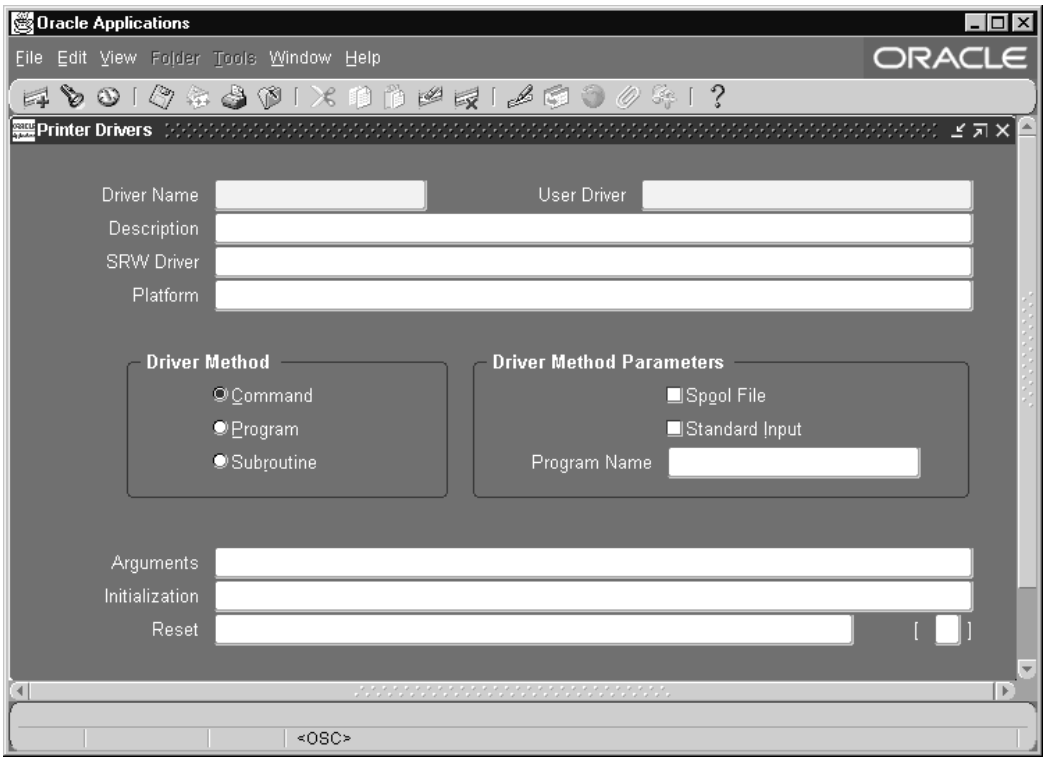
Printer Types, Print Styles, and Printer Drivers , page 9-2

Printer Types field help, page 9-47

Printers field help, page 9-49

Printer Drivers field help, page 9-53

Printer Drivers Window



Use this window to define your printer driver and printer commands.

> **Important:** You should issue a Restart concurrent manager command for all currently active managers whenever you edit an existing Printer Type, Print Style, or Printer Driver.

Oracle Applications ships printer drivers for the following print styles:

- Portrait
- Landscape
- Landwide
- A4
- Dynamic Portrait

Oracle Applications also ships printer drivers for specific printer types, including the following:

- Apple

- DEC LN03
- HP Laserjet II, HP Laserjet III, HP Laserjet 4
- HP line printer, HP 256X line printer
- EPOCH
- EPSON FX1050 and DMTX1
- QMS PS 825/925

Define additional printer drivers if you have different types of printers, or define additional print styles.

Printer Drivers Window Fields

Driver Name

The printer driver name must be unique for a given platform.

User Driver

This user name is referenced by Oracle Applications and must be unique for a given platform.

SRW Driver

Enter the name of the Oracle Reports (SRW) printer driver, if any, that will be invoked by your printer driver. Only Oracle Reports programs require this information.

Enter the entire path to the file, or just the file name. If you enter only the file name, Oracle Applications assumes the file is located in the $FND_TOP/$APPLREP directory.

Platform

Select the platform for which the printer driver is defined. Do not assign platform codes to printer drivers unless you have multiple drivers of the same name. If it cannot find a specific platform code associated with a driver, the concurrent manager will default to the driver with a null platform code.

Driver Method Region

Command The printer driver executes within an operating system shell. An example is the lpr command in UNIX.

Program The printer driver executes directly as a program, not through an operating system shell.

- An example is a C standalone program for printing.
- This method executes faster than the Command method, but cannot access shell commands like PRINT on MS-DOS.

Subroutine

The printer driver executes a predefined Oracle Applications routine.

Subroutines are specific to operating platforms and are invoked directly by a system call from the concurrent manager

Driver Method Parameters Region

Spool File

Select whether the printer driver creates its own copy of a file for printing. If this check box is checked when the Driver Method is set to Program, the print program creates its own spool file.

- An example of spool files is the UNIX lpr command, which creates its own copy of a file if you do not specify the -s option.

Standard Input

Select whether the printer driver accepts standard input. Uncheck this check box when the Driver Method is set to Program. Unless the program accepts standard input, this check box should always be unchecked.

- An example is the UNIX command lpr, which accepts standard input when a filename is not specified.

Program Name

If the driver method is *Program*, enter the full path to the program that the driver invokes. The path is not necessary if the program's location can be identified by the operating system's PATH environment variable. See Location for Custom Print Programs, page 9-34.

If the driver method is *Subroutine*, enter the subroutine name that the driver invokes.

Arguments

When the Driver Method is set to Program, enter any generic arguments that must be supplied to the print program.

When the Driver Method is set to Command, enter the full command and its arguments.

Initialization

Enter the initialization string that must be sent to the printer before the printer driver can begin printing.

Reset

Enter the reset string that returns the printer to its ready state when printing is complete.

[ ]

The double brackets ([ ]) identify a descriptive flexfield that you can use to add data fields to this form without programming.

This descriptive flexfield allows you to define special commands specific to your printer driver and/or the platform it runs on.

Related Topics

Overview of Printers and Printing, page 9-1

Printer Types, Print Styles, and Printer Drivers , page 9-2

Creating Custom Printer Drivers, page 9-31

10

Oracle Applications Help

Setting Oracle Applications Help Profile Options

Oracle Applications Help uses several profile options (in the profile category Help System):

- Applications Help Web Agent - set this profile option if you want to have online help launched on a web server other than that specified by the Apps Servlet Agent profile.
- Help Utility Upload Path - used in uploading help files. See: Downloading and Uploading Help Files, page 10-2.
- Help Utility Download Path - used in downloading help files. See: Downloading and Uploading Help Files, page 10-2.
- Help System Root - determines which Contents navigation tree appears when the online help first launches. At the site level, this should be set to 'fnd:library'. To have an application's navigation tree appear in context-sensitive help, set this profile at the application level to '<application short name>:<root node key>'. For example, set it to '<ap:contents>' for the Oracle Payables main tree.
- Help Localization Code - used to group and differentiate help documentation within a language.

Customizing Oracle Applications Help

This section describes how you can create and upload custom Oracle Applications help files for the online help system.

Oracle Applications help files are formatted as HTML allowing easy modification using commercial HTML text editors. You can also add customized files of your own to the help system.

If you have licensed Oracle Tutor, you can use it to edit your Oracle Applications help files. See: Using Oracle Tutor, page 10-17.

> **Caution:** With each new release of Oracle Applications and each patch you accept, you may need to reapply your changes to any updated help files you have modified, if you want access to the latest information.

Customizing Oracle Applications Help includes the following topics:

- Downloading and Uploading Help Files (Help System Utility), page 10-2
- Linking Help Files, page 10-5
- Updating the Search Index, page 10-8
- Customizing Help Navigation Trees, page 10-8
- Customizing Help in a Global Environment, page 10-16

Customizing help files involves two utilities, the Oracle Applications Help System Utility and the Help Builder. These utilities are available under "Help Administration" from the seeded System Administration responsibility.

Downloading and Uploading Help Files

The Generic File Manager Access Utility (FNDGFU) is used for downloading and uploading help files. For an overview of this utility, see: Generic File Manager Access Utility (FNDGFU), page B-24

Oracle Applications help files are stored in the database. The Oracle Applications Help Administration pages of the Help System Utility are provided for retrieving and replacing them in the course of customization.

Setting Help System Utility Profile Options

Before using the Help Download or Help Upload pages you should define the upload and download directory paths. Oracle Applications provides profile options for you to set these paths.

Use the profile option Help Utility Download Path to define the directory location to which the Help System Utility will download files. Use Help Utility Upload Path to define the directory location from which your customized files will be transferred back into the Oracle Applications Help System.

Identifying Help Files for Customization

Help files are downloaded by file name. To identify the specific file that you want to customize, open the document in the Oracle Applications Help System. Use the view source function of your browser to view the HTML source code. The source information

will include the file name.

To identify the language and product of the help file, use the source document URL. The final three nodes of the source document URL are the language, the product name, and the anchor or target name.

Using this document again as an example, you will see the final three nodes of the URL are /US/FND/@ht_updown#ht_updown. This identifies the language as US, the product group as FND (Applications Object Library), and the target name as ht_updown.

> **Note:** The syntax in the URL, @ht_updown#ht_updown, is an example of the Oracle Applications special syntax used to link documents by anchorname. For more information about this syntax see Linking Help Files, page 10-5.

The Oracle Applications Help System Utility also provides reports to cross-reference target names and help file names. See Creating Reports, page 10-4.

Downloading Help Files

You download help files by language and by product. That is, you select the language (for example, US for U.S. English) and you select the product (for example, AR Oracle Receivables). It is important to note the two-letter code for the product (in this example, the two-letter code is AR) because the product code determines the download directory.

To download files, navigate to the Help Download page. This page is listed under Help Administration available under the seeded System Administration responsibility.

Follow these steps to download a single help file:

1. Select Single File Download.
2. Enter search criteria for the file you want.
3. Select the file you want from the search results, and select Download.

Follow these steps to download multiple help files:

1. Select Bulk Files Download.
2. Enter in the Target Directory and Filter Options as appropriate.
3. Select Download.

Uploading Help Files

Once you have customized the help files, use the Help System Utility to upload the documents into the help system. Your files are uploaded from the upload directory specified in the profile option Help Utility Upload Path.

You can upload different types of files, including:

- HTML files (all HTML files must have a .htm extension)
- GIF graphics files (must have a .gif extension)
- Adobe® Acrobat files (must have a .pdf extension)
- Cascading Style Sheets (must have a .css extension)

Follow these steps to upload a single help file.

1. Copy the customized file to a directory accessible from your desktop. .
2. Navigate to the Help Upload page. This page is listed under Help Administration available under the seeded System Administration responsibility.
3. Select Single File Upload.
4. Specify the application under which this help file belongs.
5. Specify the customization level (100 and above indicates a custom file; below 100 indicates a file from Oracle).
6. Select Upload.

Follow these steps to upload multiple help files:

1. Copy the customized files to the appropriate product folder in the upload directory.
2. Specify the application under which these help files belong.
3. Specify the customization level (100 and above indicates a custom file; below 100 indicates a file from Oracle).
4. Select Upload.

Creating Reports

The Help System Utility provides two reports for you to cross-reference help targets and file names.

Help Target to File Names Report This report lists by target, each file that contains the target, the document title of the file, and the product.

File Name to Help Targets Report This report lists every file name and document title by language and product and all the targets found within each file.

Follow these steps to run these reports:

1. Navigate to the Help Reports page. This page is listed under Help Administration

available under the seeded System Administration responsibility.

2. Select the report you want to run.

3. Select the desired application, language, and output format.

4. Click Go.

Linking Help Files

The Oracle Applications help system supports a special syntax for hypertext links that keeps them working even when files are renamed or split into parts. The special syntax, which is explained in detail below, looks like this:

```
For more about widgets, see  <A HREF="@widgets#widgets">All About
Widgets</A>.
```

Oracle Applications help files use this syntax, and you can use it too in your custom help files. Or if you prefer, you can always use conventional hypertext links based on filename.

Linking Help Files includes the following topics:

Special Link Syntax

Links in Oracle Applications help files point, not at a particular filename, but rather at one of the named anchors contained in the file. The Oracle Applications help system resolves anchorname to file dynamically, every time a link is negotiated.

Information on which files contain which anchornames is put into the help system automatically on upload. Authors must ensure that anchornames are unique across an application's help files to prevent duplicate links. In return, they need never worry about a change in filename breaking their links.

Named Anchors in Conventional HTML

By **named anchor** is meant the following kind of HTML tag:

```
<A NAME="anchorname"></A>
```

Named anchors can be placed anywhere in the body of an HTML file, and are typically used for links internal to the file in question. A pound sign (#) is placed before the anchorname in the link that points at it.

For example, you would use HTML like the following to allow users to jump forward to a section with the anchorname of "widgets":

```
For more about widgets, see  <A HREF="#widgets">All About Widgets</A>
below.

<A NAME="widgets"></A> <H2>All About Widgets</H2>
```

Extended to Support Interdocument Links

Oracle Applications help files extend this conventional HTML syntax to create links not only within, but also **between** help files. To link to a file that contains a particular named anchor, you simply place an at sign (@) before the anchorname. To link to the precise spot within the file where this anchor appears, you append a pound sign followed by the anchorname, just as you would in conventional HTML. This results in the following special syntax:

```
<A HREF="@anchorname#anchorname">link text</A>.
```

For example, to link to the file that contains the "widgets" anchor illustrated above, at the point in the file that this anchor occurs, you would use HTML like the following:

```
For more about widgets, see  <A HREF="@widgets#widgets">All About
Widgets</A>.
```

If you simply want to link to the top of the file that contains this anchor, you can omit the pound-sign segment "#widgets."

Links in Oracle Applications help files rarely omit the pound-sign segment. This means that however topics are rearranged within or among files, links to these topics from other files always go to the proper file, and to the precise spot within the file where the topic occurs.

> **Caution:** Do not use case to make distinctions between anchornames. Unlike most web browsers, the Oracle Applications help system treats anchornames in a case-insensitive fashion.

Cross-Application Links

In the Oracle Applications help system, all help files associated with a particular application exist in the same directory, as far as their URL is concerned. Help files associated with other applications exist in directories named after the application's short name. All these application directories are at the same level in the help system.

To create a link that goes to a help file associated with a different application, you create a relative link that goes up a level to the parent of all help application directories, and then back down through the other application's directory, before concluding with Oracle Applications' special link syntax. This results in the following cross-application link syntax:

```
<A HREF="../shortname/@anchorname#anchorname"> link text</A>
```

For example, if the "All About Widgets" topic illustrated above were an Oracle Payables help topic, and you wanted to link to it from an Oracle General Ledger help file, you

would use a link like the following, where AP is Oracle Payables' short name:

```
For more about widgets in Oracle Payables, see  <A
HREF="../AP/@widgets#widgets">All About Widgets</A>.
```

When used in this fashion, application short names are case insensitive.

> **Note:** These application help directories are merely "virtual" directories recognized by the Oracle Applications help system when used in URLs. All files are actually stored in the database, with application short name being one attribute among many associated with them.
>
> Oracle Payables' official short name is SQLAP. This has been shortened to AP for the virtual directory used in the Oracle Applications help system. Similarly, Oracle General Ledger's official short name of SQLGL has been shortened to GL, and Oracle Assets short name of OFA has been shortened to FA.

Related Topics Links

Links are not limited to a single target in the Oracle Applications help system. You can point your links at multiple topics and files by using the following syntax:

```
<A HREF = "@anchorname1,anchorname2,anchorname3"> Related Topics</A>
```

When a user negotiates the link, a page headed "Related Topics" appears, containing a list of the page titles corresponding to these anchornames, with each title linked to the file in question.

To include cross-application links, simply prefix the application short name and a colon to the anchorname:

```
<A HREF = "@anchorname1,shortname:anchorname2,anchorname3"> Related
Topics</A>
```

Context-Sensitive Help

When you ask for help in Oracle Applications, the topic for your current window opens. If you ask for help from a report parameters window, your help file opens to a discussion of that report.

Oracle Applications help files contain special anchornames to enable these context-sensitive links. When calling help from an Oracle Applications Forms-based window, Oracle Applications looks for an anchorname based on the form name and the window name combined as follows:

```
<A NAME="form_name_window_name"></A>
```

You can override the form_name portion of the anchorname by specifying a HELP_TARGET parameter in the parameter field of the Form Functions window. Use the syntax HELP_TARGET = "alternative_form_name". See: Form Functions, *Oracle Applications System Administrator's Guide - Security*.

When calling help from an Oracle Application Framework (HTML-based) window from the global Help button, the anchorname follows this syntax:

```
<A NAME="appShortName_packageFunctionalComponent_pageName"></A>
```

For more information on coding for the global Help button, refer to the documentation listed in "Oracle Application Framework Documentation Resources, Release 12", Note 391554.1, on Oracle*MetaLink*.

When calling help from a report parameter window, Oracle Applications looks for an anchorname constructed as follows:

```
<A NAME="SRS_concurrent_program_shortname"></A>
```

Updating the Search Index

Oracle interMedia Text enables the search feature provided by the Oracle Applications help system. Run the concurrent program Rebuild Help Search Index to rebuild the search index after uploading customized documents. This ensures that they will be included in any searches your users perform.

Customizing Help Navigation Trees

You use the Help Builder applet to customize the help navigation trees that appear in your browser window's navigation frame when help is invoked.

> **Caution:** With each new release of Oracle Applications and each patch you accept, you will need to reapply your changes to any updated help navigation trees you have modified, if you want access to the latest information. In addition, Oracle does not provide any mechanism for identifying changes between releases of Oracle Applications help navigation trees.

You can use the Help Builder to perform the following tasks:

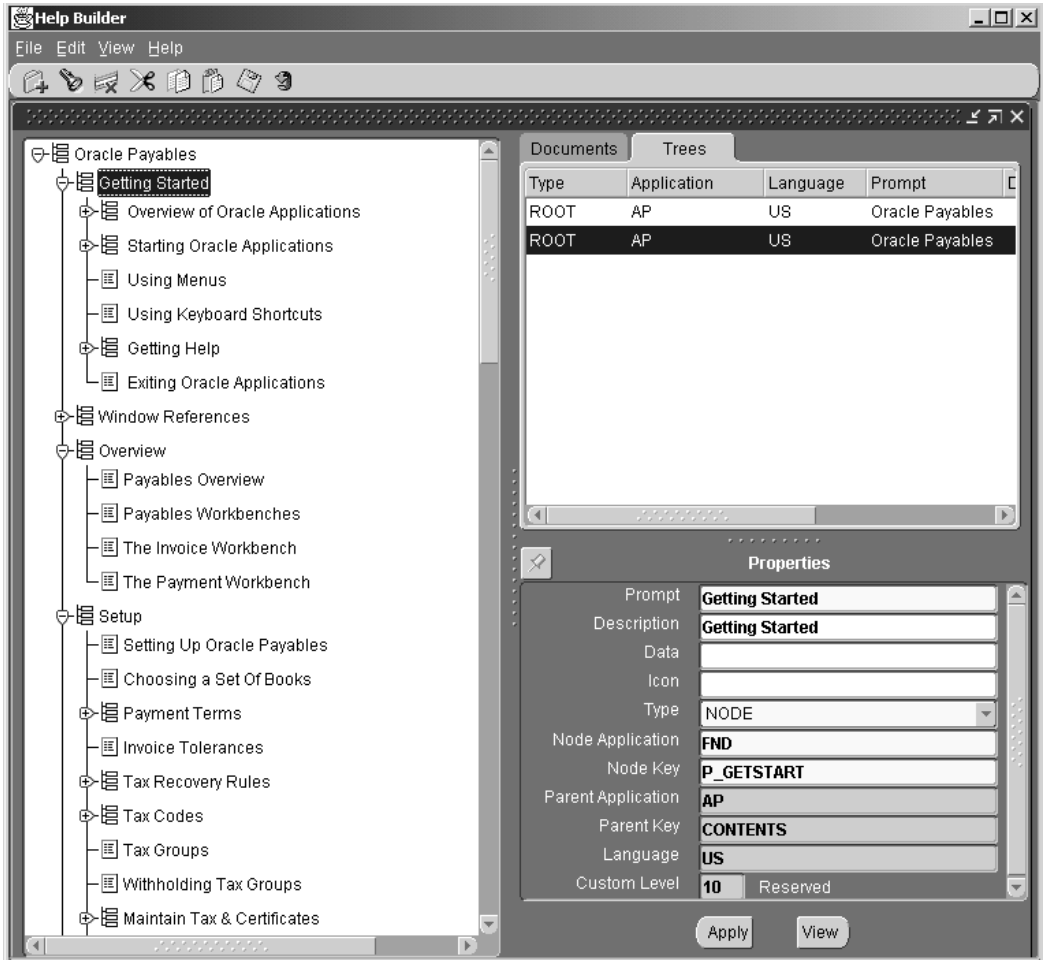
- Open a tree for editing, page 10-11
- Add new help files to a tree, page 10-12
- Add new nodes to a tree, page 10-13
- Add nodes to one tree from another, page 10-14
- Change the organization of a tree, page 10-14
- Create a new navigation tree, page 10-15

For help understanding the information associated with each of the Help Builder's fields, see Help Builder Window Reference, page 10-15.

Accessing the Help Builder

To access the Help Builder, navigate from Oracle Self-Service Web Applications as follows: **System Administration - Help Builder**.

The Help Builder User Interface



The Help Builder window default view consists of three panes. The left pane displays the tree that is currently selected. Use this area to manipulate your tree by adding nodes, deleting nodes, and dragging nodes into the positions desired. The top right pane displays items matching the searches you have performed using the Find Documents or Find Trees functions. Click on the Trees or Documents Tab as appropriate. The bottom right pane is the Properties Pane. This area displays the properties of the item (root, node, or document) currently selected. Those properties with enabled fields can be updated.

Help Builder Menus

Help Builder has the following menus:

File Menu

The File Menu provides the following functions:

New Creates a new root node. Selecting this option will open the Root Node Properties window for you to enter the appropriate values for your new root node.

Open Opens a tree. Selecting this option will open the Find Trees window for you to enter selection criteria to find the appropriate tree.

Save Saves the current changes.

Reload Reloads the current tree to apply all changes throughout the tree hierarchy.

Print Prints the current tree pane.

Exit Exits the Help Builder.

Edit Menu

The Edit Menu provides the following functions:

New Node Creates a new node beneath the selected node on the tree. If the selected node is a document, a document node is created. If the selected node is a branch node, a branch node is created.

Cut Cuts the selected item (document or branch node).

Copy Copies the selected item (document or branch node).

Paste Pastes an item beneath the currently selected node on the tree. If the node selected is a document, the item will be pasted beneath it on the tree. If the node selected is a branch that has been expanded, the item will be pasted as a child of the selected node. If the node selected is a branch that has not been expanded, the item will be pasted as a sibling beneath the selected node.

Delete Deletes the selected item.

Properties... Allows you to update the properties of the selected item via the Properties window. The Properties window is identical to the Properties Pane.

See Help Builder Window Reference, page 10-15 for descriptions of the Properties fields.

Preferences... Allows you to set interface preferences via the Preferences window. The preferences you can set are:

Background Color

Line and Box Color

Default Node Color

Default Node Text Color

Font Name

Font Size

View Menu

The View Menu provides the following functions

Node Properties Enable the check box to display the Node Properties pane (enabled is the default).

Toolbar Enable the check box to display the Toolbar (enabled is the default).

Statusbar Enable the check box to display the Status Bar (enabled is the default).

Find Documents Opens the Find Documents window.

Find Trees Opens the Find Trees window.

Help Menu

The View Menu provides the following functions:

About... Displays information about the Help Builder.

Library Opens the Oracle Applications Help Library.

Help Builder Tasks

Opening a tree for editing

To open a tree for editing:

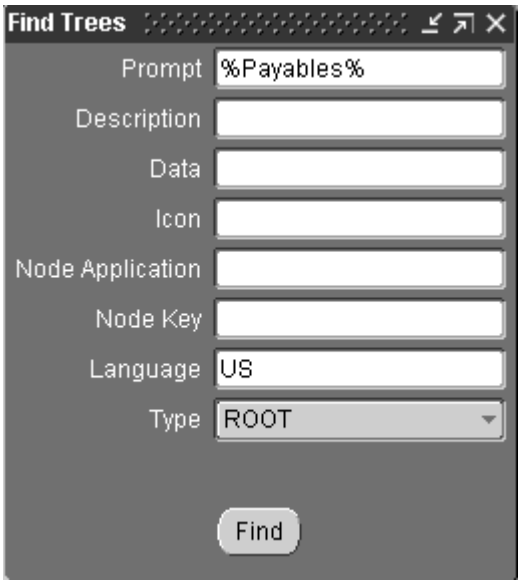
1. Open the Find Trees window using one of the following menu options:

   (M) File > Open

   (M) View > Find Trees

2. Enter your search criteria in the **Find Trees** window, and click **Find**.

   For example, enter %Payables% in the **Prompt** field to find the Oracle Payables tree.



See Help Builder Window Reference, page 10-15 for descriptions of the Find Trees search fields.

3. Trees matching your criteria will be displayed in the upper right pane of the Help Builder window on the Trees tab. Select a tree in the list to see its properties displayed in the Properties pane below. Double-click a tree to open it.

   The tree's top-level node will appear in the left pane. Expand and contract nodes to display the part of the tree you want to edit.

Adding new help files to a tree

To add new help files to a tree:

1. Upload the help files to the database, page 10-2.

2. Open the tree for editing, page 10-11.

3. Click the **Find Documents** icon on the toolbar, or select Find Documents from the View Menu.

   The **Find Documents** window appears.

See Help Builder Window Reference, page 10-15 for descriptions of the Find Documents search fields.

4. Enter your search criteria, select **Exclude documents already on a tree**, and click **Find**.

   Files corresponding to the information you entered appear on the **Documents** tab of the upper right pane of the Help Builder window. Select a document to view its properties in the Properties Pane.

5. To add the file to the tree, drag it from the **Documents** tab and drop it on the tree in the the position desired.

   Note: Files containing multiple named anchors appear multiple times. Each Target will have its own listing. Choose the Target that corresponds to the topic you want to add.

   If the topic you want to add is not the header target of the file, but a target within the document, you must supply the special link syntax in the **Data** field of the Properties Pane.

   For example, the target name might appear in the Data field as @ht_updown. To link directly to this anchor you would add #ht_updown to the end of the anchorname. The resulting entry in the **Data** field will be @ht_updown#ht_updown.

   For more information about this syntax see Linking Help Files, page 10-5.

Adding new nodes to a tree

To add new nodes to a tree:

1. Open the tree for editing, page 10-11.

2. Select the node beneath which you want to add a new node, and click **New Node** on the toolbar or select New Node from the Edit Menu.

   Note: The New Node feature will add a node that is like the node selected. For example, if a branch node is selected a branch node will be added beneath it. It a leaf node is selected, a leaf node will be added beneath it.

   When adding a branch node as a sibling to an existing branch node, be sure that the selected branch node is not expanded. If the existing branch node is expanded, the new node will be added as a child to the selected branch.

3. Enter information for the new node in the Properties Pane, and click **Apply**.

4. Click **Save** to save your changes.

Adding nodes from one tree to another

To add nodes from one tree to another:

1. Open the tree for editing, page 10-11.

2. From the View menu select **Find Trees**.

3. In the Find Trees window select **Node** from the **Type** poplist and enter search criteria for the nodes you want to add. Click **Find**.

   Nodes corresponding to the information you enter appear on the **Trees** tab of the main Help Builder window.

4. Drag nodes from the **Trees** tab and drop them on the tree.

5. Click **Save** to save your changes.

Changing the organization of a tree

To change the organization of a tree:

Caution: Changes made to nodes added from another tree are reflected in the original tree and all other trees that include them.

1. Open the tree for editing, page 10-11.

2. To move a node, drag the node from its current location and drop it at its new location in the tree.

3. To delete a node, select it and click **Delete** on the toolbar or select

Delete from the Edit Menu.

4. To change a prompt, select the node, enter the new prompt in the **Prompt** field of the **Properties** pane, and click **Apply**.

   Other node properties can be changed in a similar fashion.

Note: If the same node appears elsewhere in the tree, your changes will not appear there until you click the **Reload** button on the toolbar. For one node to be the same as another, the Node Key and Node Application of their parent nodes must be the same, as well as all their own properties. Their grandparent nodes and above can be different.

Creating a new navigation tree

To create a new navigation tree:

1. Choose **File** >**New**.

   The **Root Node Properties** window appears.

2. Enter information for the tree's root node, and click **Apply**.

3. Add new nodes to the tree, page 10-13.

4. Add new help files to the tree, page 10-12.

5. Add nodes from other trees to the tree, page 10-14.

6. To view the new tree with context-sensitive help, enter its root as the Help Tree Root for some application, responsibility, or user.

   To view it stand alone, substitute its root in the root= parameter at the end of your site's Oracle Applications Help URL. Include the application short name. For example, for a root named ROOT_INV belonging to the Oracle Inventory product, you would use root=INV:ROOT_INV in the URL.

Help Builder Window Reference

Field names and descriptions for the Help Builder window are given below.

Application Application shortname of application that owns the help file.

Custom Level Customization level of the node. 100 is the default for customer use. Levels under 100 are reserved for system use.

Data	If the node links to a help file, the file name or a target name preceded by an "@" symbol, or an absolute URL.
Description	Longer description of the node, if the Prompt is terse. Otherwise may simply repeat the Prompt.
Filename	Pre-upload filename of the help file.
Icon	Not used.
Language	Language code of help files covered by the node.
Node Application	Application shortname of the application that owns this node. If different from the value given for the ROOT, this node and all the nodes it branches into have been grafted into the tree from another application.
Node Key	String that uniquely identifies the node in this Node Application. The node key can be generated automatically or typed into the field (for a new node). However, once references to the node exist, the node key cannot be changed.
Prompt	The text that appears on the tree for this node.
Target	Anchorname contained in the help file. Do not precede with an @ sign in the Find Documents window.
Title	Title of the help file.
Type	ROOT: Top-most node of a navigation tree. NODE: Node that branches into other nodes, but is not the ROOT. If **Data** is not null, it links to a help file as well. DOCUMENT: Node that does not branch into other nodes, but simply links to a help file.
Version	Version identifier of the help file.

Customizing Help in a Global Environment

The Oracle Applications help system contains files translated into many different languages, and localized for diverse countries and regions. If your enterprise crosses linguistic and cultural boundaries, or if you use Oracle Human Resources, the following information may apply when customizing your help files.

Linking Between Different Languages

One level up the virtual directory hierarchy used in Oracle Applications help URLs are the application directories used to construct cross-application links. Two levels up are the language directories, which you can use to construct cross-language links.

To create a link that goes to a help file in a different language, use the following link syntax:

```
<A HREF = "../../language_code/shortname/@anchorname#anchorname"> link
text</A>
```

For example, to link to "All About Widgets" in the French version of Oracle Payables help, you would use the following link, where AP is Oracle Payables' short name and F is the French language code:

```
For more about French widgets, see  <A
HREF="../../F/AP/@widgets#widgets">Qu'est-ce qu'un widget?</A>.
```

When used in this fashion, language codes are case insensitive.

> **Note:** After following a link to a different language, users stay in that language until they follow a link back out to their original language. This can be either a link they encounter within a help file, or a link from the navigation tree, which remains in their original language throughout.

Using Oracle Tutor

Oracle Tutor is a documentation tool designed to help companies create and maintain their end user process documentation. Oracle Tutor defines end user process documentation as policies and procedures, instructions, and support documents. Oracle Tutor also provides a system for

- distributing process documentation to employees by job role
- customizing Oracle Applications Online Help
- integrating process documentation with Oracle Application courseware
- building customized Oracle Application training materials based on job title or topic
- auditing process documentation to ensure accuracy and effectiveness

Oracle Tutor is comprised of three components:

- the documentation methodology
- model documents

- software

The Tutor **method** defines all aspects of the documentation process: the content and format of the different document types, as well as the relationship between documents; the process by which documents are created and maintained; and the methods by which documents are distributed and used for training and reference purposes.

Tutor **model documents** are complete process documents and courseware that provide a starting point -- document "owners" edit the models instead of creating documents from scratch.

Tutor **software** includes tools for keeping documents up to date and distributing them on a need to know basis.

- Author is the tool used by individual document owners to create and edit MS Word documents quickly and easily.

- Publisher is the tool used to

  1. build desk manuals by job title,

  2. build student and instructor guides by job title or topic, and

  3. ensure document integrity through the generation of a wide range of cross-reference reports.

Together, these three components help companies overcome the operational challenges of documenting and maintaining business practices.

For more information, please see the *Tutor Author User Manual*, the *Tutor Publisher User Manual,* and the *Tutor Implementation Guide.*

11

Applications DBA Duties

Overview of Applications DBA Duties

Applications database administration (DBA) combines the efforts of an Oracle Applications System Administrator and an ORACLE database administrator.

ORACLE Schemas

Upon installation of Oracle Applications, a number of schemas (sometimes called ORACLE schemas) are present in the database. You do not need to create these schemas; however, you should change the default passwords.

These schemas come from different sources and can by described as being of the following types:

1. Schemas that exist in every Oracle database (whether used by Oracle Applications or not) [ex: SYS,SYSTEM].

2. A small set of schemas used by shared components of Oracle Applications [for example, APPLSYSPUB,APPLSYS,APPS].

3. A large set of schemas provided by the individual products of Oracle Applications [for example, ABM,AHL,...,ZSA,ZX].

4. A set of schemas that belong to optional database features or third party products these fall into three subtypes:

    1. Used by and patched with Oracle Applications [for example, CTXSYS, PORTAL30].

    2. Used by Oracle Applications but patched only with the RDBMS [for example, MDSYS,ORDSYS]

    3. Not used by Oracle Applications [for example, SCOTT]

At no time do any of the schemas provided with Oracle Applications relate to a particular Oracle Applications user.

All types of schemas are used during runtime operations of Oracle Applications and the schemas of type 2, 3 & 4.1 are accessed during initial installation and patching.

For schema passwords, Oracle Applications concerns itself with mainly three passwords for its schemas:

1. A password for APPLSYSPUB (also known as the GATEWAY user). The default password is 'PUB'.

2. A password shared between APPLSYS and APPS (also known as FNDNAM). The default password is 'APPS'.

3. A password for all of the product-specific base schemas (type 3). The default password for these schemas is same as the schema name.

Important: You should change these passwords upon installation.

Note that the Oracle database schemas and passwords connect to the ORACLE database, while application usernames and passwords access Oracle Applications.

Related Topics

ORACLE Users (Schemas) Window, page 11-8

Applications Window, page 11-12

Registering an ORACLE Schema

The installation process automatically registers Oracle Applications ORACLE schemas, so you only need to register any additional ORACLE schemas that you need using the ORACLE Users window.

You must register an ORACLE schema with Oracle Applications if:

- you create a custom application using Oracle Application Object Library

- you want to associate an additional ORACLE schema with an Oracle Applications product

Important: Before you can register an ORACLE schema, your database administrator must first create an ORACLE schema that connects to the ORACLE database. You then use the ORACLE Users window to register your ORACLE schema.

Reregistering ORACLE schemas

You should also reregister ORACLE schemas associated with custom applications built using Oracle Application Object Library each time you upgrade Oracle Application Object Library

Initialization Code

You can add in custom initialization SQL code to be executed when a database session starts up or when it is re-initialized. You specify this code using a profile option.

The code is executed by FND_GLOBAL.INITIALIZE and APPS_INITIALIZE immediately after initializing global variables, profiles, and the contents of client_info on session startup.

Profile Option Initialization SQL Statement - Custom

Using the profile option Initialization SQL Statement - Custom, you can add site-specific initialization code, such as optimizer settings. This profile value must be a valid SQL statement, or a PL/SQL block for more than one statement, that is to be executed once at the startup of every database session.

This profile option can be set at any level by the System Administrator, and is reserved for use by customers.

Profile Option Initialization SQL Statement - Oracle

This profile option is used by Oracle Applications. This profile option and its value settings are delivered as seed data, and must not be modified.

Related Topics

Oracle Applications and Cost-Based Optimization, page 12-1

Overview of Applications DBA Duties, page 11-1

ORACLE Users Window, page 11-8

Resource Consumer Groups in Oracle Applications

The Database Resource Manager introduced in Oracle8*i* is used to allocate and manage resources among database users and applications.

Resource consumer groups and resource plans provide a method for specifying how to partition processing resources among different users. A resource consumer group defines a set of users who have similar resource usage requirements. An overall resource plan specifies how resources are distributed among the different resource consumer groups.

Oracle Applications allows the system administrator to assign individual Oracle

Applications users to resource consumer groups. In addition, concurrent programs and concurrent managers can be assigned to resource consumer groups.

Note: These resource consumer groups apply to CPU resources only.

For additional information, see the Oracle database documentation.

Assigning Resource Consumer Groups

The system administrator can assign a user to a resource consumer group by setting the value of the user profile option FND:Resource Consumer Group for that particular user. The user can see this profile option but cannot update it.

The system administrator can assign a concurrent program to a resource consumer group in the Parameters window of the Define Concurrent Program form. See: Concurrent Programs Parameters Window, page 6-73.

The system administrator can assign a concurrent manager to a resource consumer group in the Define Concurrent Manager form. See: Concurrent Managers Window, page 7-58.

Hierarchy of Resource Consumer Group Assignments

Conflicts can arise between the resource consumer groups associated with a single session. For example, a concurrent manager assigned to one resource consumer group may run a concurrent program assigned to another. A similar situation arises when a user performs a transaction managed by a transaction manager that has a different resource consumer group than the user. To resolve such conflicts, Oracle Applications uses a hierarchy.

In the case of a concurrent program, the system first checks to see if the program has an assigned resource consumer group and if so, uses that. If not, the system checks the concurrent manager running the program and uses its resource consumer group. If the concurrent manager is not assigned to a resource consumer group the system uses the default group "Default_Consumer_Group".

In the case of a transaction manager running a transaction program, the system once again checks the resource consumer group assigned to the program, if any, and if there is none, checks the transaction manager. If the transaction manager has no assigned resource consumer group the system then checks the profile option value for the user whose session began the transaction. If there is no resource consumer group defined the system uses the default resource consumer group.

For a user running a form, the system first checks the profile option value for that user and uses that if it is defined. Otherwise the system uses the default resource consumer group.

Oracle Applications Schema Password Change Utility (FNDCPASS)

Changing passwords frequently helps ensure database security. Oracle Applications provides a command line utility, FNDCPASS, to set Oracle Applications schema passwords. This utility changes the password registered in Oracle Applications tables and changes the schema password in the database. This utility can also change user passwords.

> **Note:** You cannot change a schema name, such as APPLSYS or GL, after a product is installed, with FNDCPASS.

> **Important:** Ensure that the entire Oracle Applications system has been shut down before changing any schema passwords.
>
> All users should log out and the Applications system should be down before running this utility.
>
> If Oracle Applications user passwords are being changed then the relevant users should not be logged in.

> **Important:** Before changing any passwords, you should make a backup of the tables FND_USER and FND_ORACLE_USERID.

FNDCPASS Command and Arguments

To change the APPS and APPLSYS (type 2) schema password:

Use this command to change passwords for schemas that are used by shared components of Oracle Applications.

```
FNDCPASS <logon> 0 Y <system/password> SYSTEM \
 <username> <new_password>
```

Use the above command with the following arguments. When specifying the SYSTEM token, FNDCPASS expects the next arguments to be the APPLSYS username and the new password.

logon — The Oracle username/password.

system/password — The username and password for the SYSTEM DBA account.

username — The APPLSYS username. For example, 'applsys'.

new_password — The new password.

This command does the following:

1. Validates APPLSYS.
2. Re-registers password in Oracle Applications.
3. Changes the APPLSYS and all APPS passwords (for multi-APPS schema installations) to the same password.

   Because everything with a Privilege Level [set to any of ('E', 'U', 'D')] in the FND_ORACLE_USERID table must always have the same password, FNDCPASS updates these passwords as well as APPLSYS's password. For example, the APPS password will be updated when the APPLSYS password is changed.
4. ALTER USER is executed to change the ORACLE password for the above ORACLE users.

For example, the following command changes the APPLSYS password to 'WELCOME'.

```
FNDCPASS apps/apps 0 Y system/manager SYSTEM APPLSYS WELCOME
```

To change an Oracle Applications schema password (type 3) (other than APPS/APPLSYS):

Use this command to change the password of a schema provided by an individual product in Oracle Applications.

```
FNDCPASS <logon> 0 Y <system/password> ORACLE \
 <username> <new_password>
```

Use the above command with the following arguments. When specifying the ORACLE token, FNDCPASS expects the next arguments to be an ORACLE username and the new password.

logon — The Oracle username/password.

system/password — The username and password for the SYSTEM DBA account.

username — The Oracle username. For example, 'GL'.

new_password — The new password.

For example, the following command changes the GL user password to 'GL1'.

```
FNDCPASS apps/apps 0 Y system/manager ORACLE GL GL1
```

To change all ORACLE (type 3) schema passwords:

Use this command to change the passwords of all schemas provided by Oracle Application products.

```
FNDCPASS <logon> 0 Y <system/password> ALLORACLE \
 <new_password>
```

Use the above command with the following arguments. When specifying the ALLORACLE token, FNDCPASS expects the next argument to be the new password.

logon The Oracle username/password.

system/password The username and password for the SYSTEM DBA account.

new_password The new password.

For example, the following command changes all ORACLE schema passwords to "WELCOME":

```
FNDCPASS apps/apps 0 Y system/manager ALLORACLE WELCOME
```

To change an Oracle Applications user's password:

You can use this command to change an individual Oracle Applications user's password.

```
FNDCPASS <logon> 0 Y <system/password> USER \
 <username> <new_password>
```

Use the above command with the following arguments. When specifying the USER token, FNDCPASS expects the next arguments to be an Oracle Applications username and the new password.

logon The Oracle username/password.

system/password The username and password for the System DBA account.

username The Oracle Applications username. For example, 'VISION'.

new_password The new password.

For example, if you were changing the password for the user VISION to 'WELCOME', you would use the following command:

```
FNDCPASS apps/apps 0 Y system/manager USER VISION WELCOME
```

Using the FNDCPASS Utility

Here is an example of changing an Oracle user's password, where <username> is the Oracle schema name.

1. Use the FNDCPASS utility to change the password.

   ```
   FNDCPASS <APPS username>/<APPS password> 0 Y \
    <SYSTEM username>/<SYSTEM password> ORACLE \
   <username> <new_password>
   ```

   When changing the APPS or APPLSYS passwords, replace ORACLE with SYSTEM.

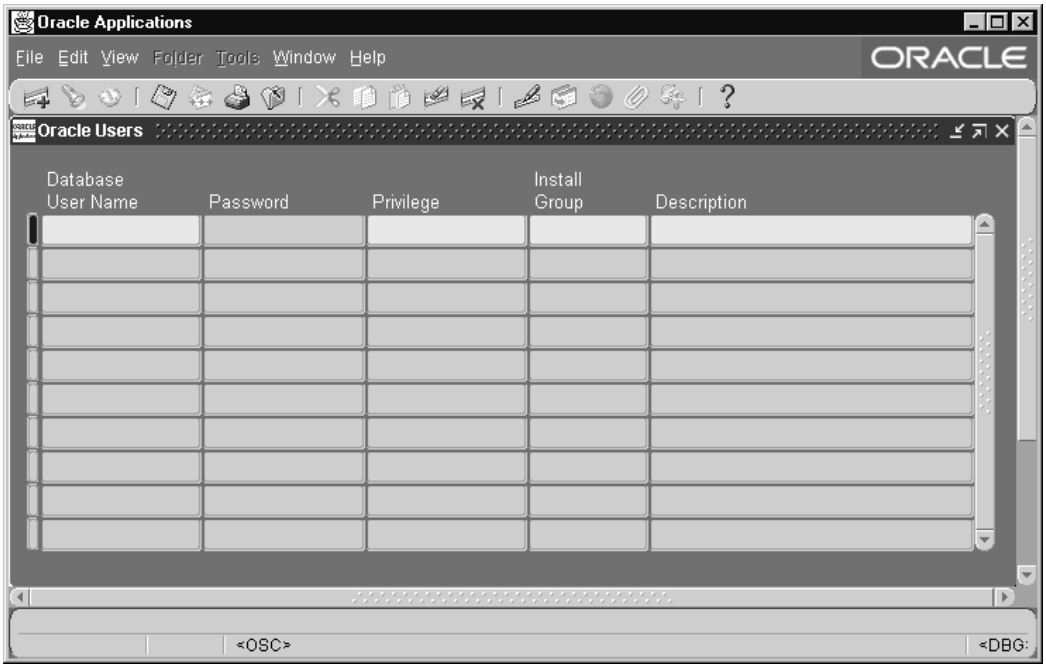
Important: Passwords for APPLSYS and the APPS schemas -- including the MRC schema -- must be the same. If you change the password for one, FNDCPASS automatically changes the others.

2. Verify the new password.

   If you changed the password for APPS (and APPLSYS), restart all concurrent managers, then log on to Oracle Applications to test the new password.

   Note: When changing APPS (or APPLSYS) and APPLSYSPUB passwords, do not restart the system until the entire password change process has been completed.

ORACLE Users Window



Use this window to register an ORACLE username with Oracle Applications. An ORACLE username grants access privileges to the ORACLE database.

Before registering an ORACLE username, create the ORACLE username that matches your application needs (this function is usually performed by a database administrator). The ORACLE username must include the *create session* privilege.

The installation process always registers your ORACLE username, so you need not register it unless you create a custom application using Oracle Application Object Library, or if you wish to associate an additional ORACLE username with Oracle

Applications.

If you register an ORACLE username as a "restricted" ORACLE username, you submit a concurrent request to set up read-only privileges to the Oracle Application Object Library tables. An "enabled" ORACLE username has all privileges to those tables. A " disabled" ORACLE username has no privileges to those tables.

If you do not register and enable your ORACLE username or if you disable a registered ORACLE username, your user cannot use Oracle Application Object Library features such as menus and flexfields.

You should not change the registration of any ORACLE usernames that the installation process registers, other than changing the passwords.

If you are registering a change to an existing ORACLE password, make the password change in the database immediately AFTER you register the password change in Oracle Applications. Until you register the password changes in Oracle Applications and implement them in the database, responsibilities using this ORACLE username cannot connect to the database.

Your password must follow the guidelines for creating passwords discussed in the Oracle database documentation. Remember that if you use non-character values in your password, you may need to use quotation marks around your password when changing it in the database.

> **Note:** Use FNDCPASS to change the password, not the ORACLE Users window>

> **Warning:** If you are changing the password to the *applsys* ORACLE username, which contains the Oracle Application Object Library tables, you must *not* change the passwords to any other ORACLE usernames at the same time.
>
> As soon as you change and save the password, you should immediately log out of the Oracle Applications, make the *applsys* password change in the database, and then sign on again before you do anything else. You should also ensure that no other users are logged on to the Oracle Applications while you are changing the *applsys* password.

> **Important:** For passwords for the APPS accounts, the *applsys* password must be identical to the password for the APPS accounts (APPS, APPS2, APPS3). The uniform passwords enable the different sets of books to operate correctly.

ORACLE Users Block

Password

Enter the password of your ORACLE username. Your password is not displayed. If you are registering a change to an existing ORACLE password, make the password change in the database immediately AFTER you register the password change in Oracle Applications.

Until you register the password changes in Oracle Applications and implement them in the database, responsibilities using this ORACLE username cannot connect to the database.

> **Warning:** If you are changing the password to the *applsys* ORACLE username, which contains the Oracle Application Object Library tables), you must *not* change the passwords to any other ORACLE usernames at the same time.
>
> As soon as you change and save the password, you should immediately log out of the Oracle Applications, make the *applsys* password change in the database, and then sign on again before you do anything else. You should also ensure that no other users are logged on to the Oracle Applications while you are changing the *applsys* password.
>
> Using FNDCPASS is the preferred method for changing passwords. Note that FNDCPASS automatically updates all instances of the password for you.

Privilege

Enter the type of privilege to the Oracle Application Object Library database tables that you want this ORACLE username to have. The Oracle Application Object Library tables contain information for Oracle Application Object Library features such as menus, help text, and flexfields. If you do not have access to these tables, you cannot use these features.

The default value for this field is Enabled.

- Enabled - An enabled ORACLE username has full privileges (insert, query, update, and delete) to the Oracle Application Object Library database tables.

- Restricted - A restricted ORACLE username has only query privileges to the Oracle Application Object Library database tables. This ORACLE username can view Oracle Application Object Library data, but cannot insert, update, or delete information.

- Disabled - A disabled ORACLE username has no privileges to the Oracle Application Object Library database tables. This ORACLE username cannot insert, query, update, or delete Oracle Application Object Library information and cannot use Oracle Application Object Library features.

Two additional privilege types appear, associated with ORACLE usernames configured at installation. However, these privilege types cannot be selected from your list of values.

- Public - The installation process registered an ORACLE username with the Public privilege, allowing all users to access the Application Sign-On Security form where they must enter a valid Oracle Applications username and password.

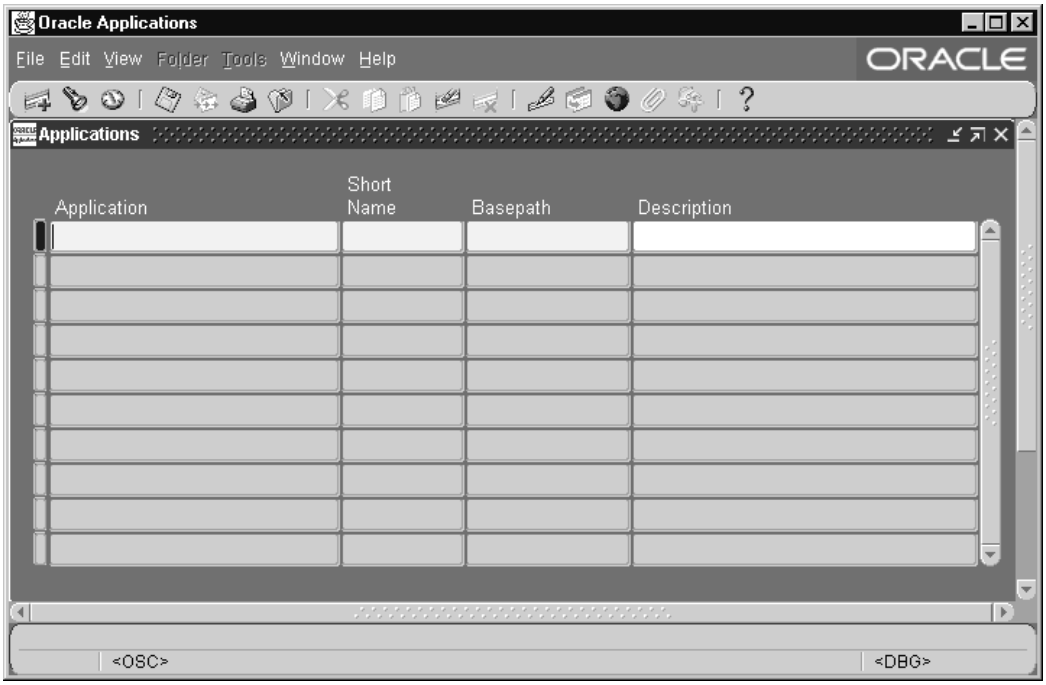
- Applsys - The installation process registered the Oracle Application Object Library ORACLE username with the Applsys privilege.

Install Group

The value of the installation group associated with your ORACLE username. Install group numbers should be consecutive whole numbers, where 1 represents the first set of books (or first set of product installations), 2 is the second set of books, 3 is the third set of books, and so on. Install group number 0 represents products that need only single installations.

> **Important:** Since the installation process does not affect ORACLE usernames (also known as "schemas") for custom applications, this value is for your reference only and is currently not used.

Applications Window



Use this window only if you are creating a custom application to isolate custom code and/or data from shipped Oracle Applications.

When you define a custom application, you supply several pieces of information to Oracle Applications. You must register your application name, application short name, application basepath, and application description with Oracle Application Object Library. Oracle Application Object Library uses this information to identify application objects such as responsibilities and forms as belonging to your application. This identification with your custom application allows Oracle Applications to preserve your application objects and customizations during upgrades. The application basepath tells Oracle Application Object Library where to find the files associated with your custom application.

You can use your custom application to name your custom menus, concurrent programs, custom responsibilities, and many other custom components. For some objects, the application part of the name only ensures uniqueness across Oracle Applications. For other components, the application you choose has an effect on the functionality of your custom object.

Prerequisites

- If you are creating a custom application: Define an environment variable that translates to your application's basepath (see *Oracle Applications Concepts* for your

operating system).

- If you are creating a custom application: Set up a directory structure for your application (see *Oracle Applications Concepts* for your operating system)
- If you are using this window with Oracle Alert: If your application resides in a database other than the database where Oracle Alert resides, you must create a database link.

Applications Block

When you register a custom application, you provide the information Oracle uses to identify it whenever you reference it. Although you can change the application short name of an application, doing so may cause a change in the application code where you hardcode your application short name. For example, if you pass program arguments through the menu that have application name hardcoded, you will also have to update them.

> **Important:** You should not change the name of any application that you did not develop, as you cannot be sure of the consequences. You should never change the name of any Oracle Applications application, because these applications may contain hardcoded references to the application name.

Application

This user-friendly name appears in lists seen by application users.

Short Name

Oracle Applications use the application short name as an internal key; for example, when identifying forms, menus, concurrent programs and other application components. The short name is stored in hidden fields while the name displays for users.

Your short name should not include spaces. You use an application short name when you request a concurrent process from a form, and when you invoke a subroutine from a menu.

> **Tip:** Although your short name can be up to 50 characters, we recommend that you use only four or five characters for ease in maintaining your application and in calling routines that use your short name. To reduce the risk that your custom application short name could conflict with a future Oracle Applications short name, we recommend that your custom application short name begins with "XX".

Basepath

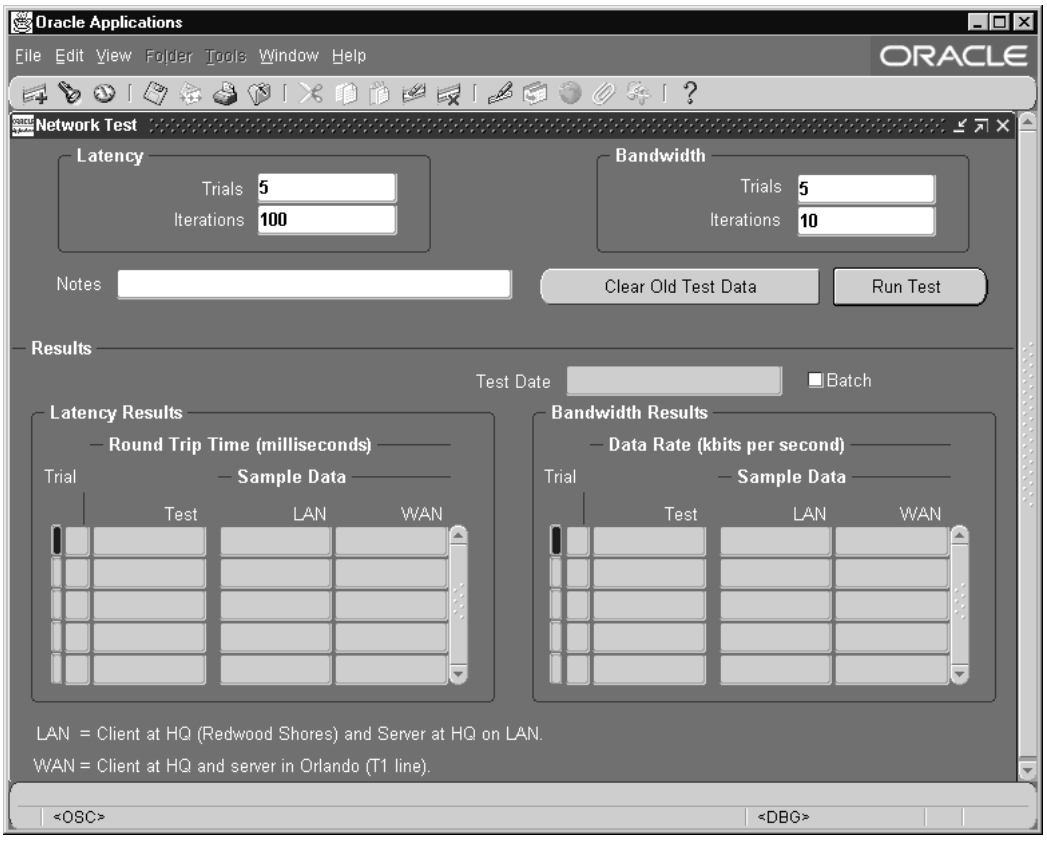
Enter the name of an environment variable that represents the top directory of your application's directory tree. Oracle Applications searches specific directories beneath the basepath for your application's files and scripts.

In general, your application's basepath should be unique so that separate applications do not write to the same directories.

However, you may define custom applications that will be used only for naming your custom responsibilities, menus and other data components. In this case, you can use the basepath of the Oracle application that uses the same forms as your application. For example, if you are defining a Custom_GL application, you could use the GL_TOP basepath for your custom application. In this case, however, you should not create custom components in the directory structure, such as custom forms and reports, because they will be difficult to isolate for maintenance and upgrading.

See: *Oracle Applications Concepts*

Network Test Window



You can use the Network Test form to monitor the latency and bandwidth of the network for forms applications, or to help create a baseline for use in comparing response times from within the application. This information enables you to make comparisons between locations, or at different times of day at the same location. The form shows the time taken to perform one or more Oracle Forms round trips, and the throughput used.

The latency shown on the form represents a combination of the round trip time needed to traverse the physical network (including any devices), and the Forms overhead to process a packet. The network test form is designed to more closely measure the network latency and bandwidth of an actual forms user. Note that the results are not expected to match the times returned by `ping`, `traceroute`, or other diagnostic network commands.

To test the network latency, a short sequence of packets is sent from the client application to the application server, then on to the database server, and back to the client. You need to specify the number of sequences (iterations) you want to send, and the number of times you want to send each set of iterations (trials). The default setting is 5 trials of 100 iterations each. The average latency is the total time for all round trips in a trial, divided by the number of iterations.

The bandwidth test (or more accurately, throughput test), examines the data transfer rate, and shows how many bytes per second your network transferred between the client, application server and database server.

Running a Test

Click the Run Test button to perform the test.

You can provide notes to indicate the conditions for each test you run.

Evaluating the Test Results

If one test result varies significantly from the other trials, discard that information.

Use the Clear Old Test Data button to purge previous test results from your database.

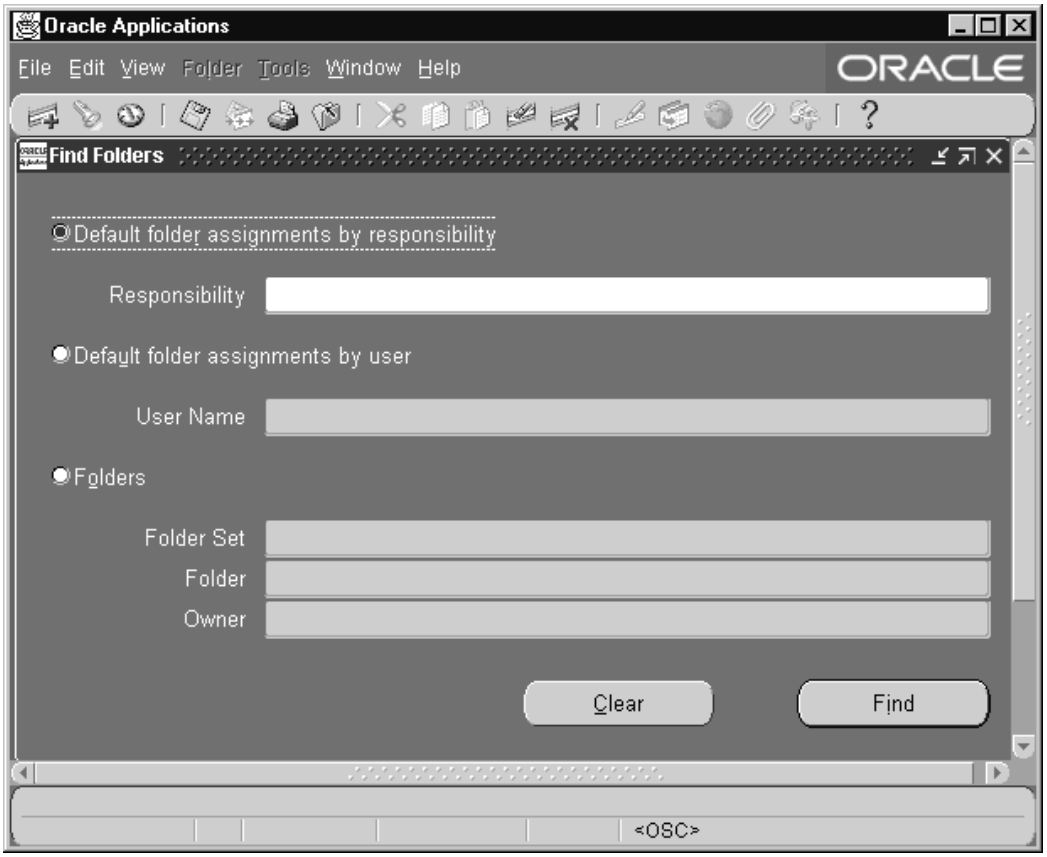
The results of both the latency and throughput tests are displayed in the Results block.

- *Latency Results* displays the minimum, average, and maximum round trip time for a single round trip from a PC client to the server.

- The *Bandwidth* window shows the throughput results, and displays the minimum, average, and maximum data rate in bytes per second.

For comparison, the sample data fields show the results of tests completed at Oracle Headquarters in Redwood Shores, California.

Administering Folders



Administer folders by assigning default folder definitions either to a specific user or to a responsibility. Manage folder definitions by assigning them to new owners, determining which folder definitions should be public (accessible by anyone), and setting the AutoQuery behavior of the folders.

You can do different tasks depending on how you search for folders or folder assignments in the Find Default Folders window.

You can assign a default folder to a user or responsibility in "restricted mode" such that all folder functionality is disabled at runtime for the user. In this way you can, for example, prevent users from seeing specific fields, or control which records they can query. This behavior is controlled by the Behavior Mode poplist value, set when the folder is assigned.

You must have default folders before you perform these steps.

To Assign a Folder to a Responsibility:

Follow these steps to assign a folder to a responsibility:

1. Navigate to the Find Default Folders window. Use "Default folder assignments by responsibility" to view the responsibilities for which to assign default folders.

2. You can assign default folders for each responsibility. When users of this responsibility navigate to this folder block, they see the default folder you specify, unless it is overridden by a user-level default.

   From the Folder field, enter the name of the default folder. The name of the folder set to which the folder belongs is filled in automatically.

   If you do not know the name of the folder, enter the folder set first, then view the folders that belong to that set.

   After you save a default folder definition for a folder set, that folder set no longer appears in the list of values.

Folder Set: Every folder set is associated with a particular folder block, and a user or responsibility can have one default folder within each folder set. The folder set name generally describes the records shown in the block; some blocks may have multiple sets of folders associated with them.

To Assign a Folder to a User:

Follow these steps to assign a folder to a user:

1. Navigate to the Find Default Folders window. Use "Default folder assignments by user" to view a list of eligible users.

2. You can assign default folders for each responsibility. When users navigate to this folder block, they see the default folder you specify.

   From the Folder field, enter the name of the default folder. The name of the folder set to which the folder belongs is filled in automatically.

   If you do not know the name of the folder, enter the folder set first, then view the folders that belong to that set.

   After you save a default folder definition for a folder set, that folder set no longer appears in the list of values.

Folder Set: Every folder set is associated with a particular folder block, and a user or responsibility can have one default folder within each folder set. The folder set name generally describes the records shown in the block; some blocks may have multiple sets of folders associated with them.

Source Type: Either User or Responsibility. Records entered in this window use the source type of User. If one of the current user's responsibilities has default folders defined, the default folders are listed with a source type of Responsibility.

User defaults override Responsibility defaults. You cannot delete Responsibility default folders in this window.

Responsibility: The responsibility which uses this default folder definition.

To Assign Ownership of a Folder:

Follow these steps to assign ownership of a folder:

1. Navigate to the Find Default Folders window. Use "Folders" to view general information about folders.

2. Select the folder(s) that requires a change of ownership.

3. Choose "Change Owner" and enter the new owner for the selected folders, or change the value in the Owner field to change the owner of a single folder.

Folder Set: Every folder set is associated with a particular folder block, and a user or responsibility can have one default folder within each folder set. The folder set name generally describes the records shown in the block; some blocks may have multiple sets of folders associated with them.

Public: Whether this folder definition is public; whether users besides the owner can use it. Use this field to determine whether to make folder definitions generally available.

Anyone's Default: Whether this folder definition is used as a default by a user or a responsibility. If it is a default definition, use Default Assignments to view the users and responsibilities for which it is the default folder definition.

Default Assignments: The users and responsibilities that use this folder definition as a default.

To Delete a Folder Definition

Follow these steps to delete a folder definition:

1. Navigate to the Find Default Folders window. Use "Folders" to view general information about folders.

2. If you queried up multiple folders, select the folder(s) to delete.

3. Delete the folder. Deleting folders deletes the folder definition along with any user and responsibility default assignments for the folder.

To Create and Assign a Folder in "Restricted Mode"

Use the steps below to create and assign a folder in a "restricted mode". When user opens a folder in restricted mode, all folder functionality is disabled.

1. Run the folder form and navigate to the folder block.

   Hide or show fields as you wish. Take care in choosing the appropriate fields, as the

fields that are hidden will not be accessible for users or responsibilities of this folder block after it is assigned to them as a default folder in restricted mode.

2. Save the folder.

3. Assign the folder as a default folder to a responsibility or user.

4. Set the value of the new Behavior Mode poplist to "Restrict fields and folder functions".

   A default folder can have one of the following values for Behavior Mode:

   - No restrictions - End user can perform all folder functions.
   - Restrict fields and folder functions - End user cannot perform any folder functions. This is "Restricted Mode".

Runtime Scenarios with Restricted Mode

End user runs folder form with restrictions

When the user opens the restricted default folder form, all folder functions are disabled. For example, the user cannot open any other folders, or move or resize fields.

Within a folder block, once a restricted default folder loads, all folder functionality will become disabled even if that block supports other folder objects.

System Administrator wants to change the default restricted folder

Once a default folder is assigned with the Behavior Mode "Restrict fields and folder functions" to any user or responsibility, it no longer appears in the list of available folders for opening by any user (even though this folder is defined as "Public").

To change this default folder, you should first assign the default folder to yourself. Then run the folder form and navigate to the folder block so that the default folder will load. You can then make modifications and save the folder. Even though the Behavior Mode is restricted, the folder functions can still be performed since you have become the owner of the folder.

12

Query Optimization in Oracle Applications

Oracle Applications and Query Optimization

Oracle Applications Release 12 uses cost–based optimization in order to choose the most efficient execution plan for SQL statements. Using this approach, the optimizer determines the most optimal execution plan by costing available access paths and factoring information based on statistics for the schema objects accessed by the SQL statement.

Oracle Applications requires several database initialization parameters to be set correctly in order to ensure optimal performance. Refer to Oracle*Metalink* Note 396009.1, and ensure that you have configured the parameters according to this note.

For the query optimizer to produce an optimal execution plan, the statistics in the data dictionary should accurately reflect the volume and data distribution of the tables and indexes. To this end, database statistics should be refreshed periodically. However, that does not necessarily imply that you should gather statistics frequently. Systems that are close to going live typically experience inserts of a large amount of data, as data from legacy systems is migrated. In that scenario, the statistics would probably need to be refreshed quite frequently (for instance, after each major load), as large loads could change the data distribution significantly. Once the system reaches steady state, the frequency of statistics collection at the schema/database level should be reduced to something like once a month. However, statistics on some volatile tables can be gathered as frequently as required.

Oracle Applications provides a set of procedures in the FND_STATS package to facilitate collection of these statistics. FND_STATS uses the DBMS_STATS package to gather statistics.

Warning: You should not run DBMS_STATS directly.

FND_STATS.GATHER_SCHEMA_STATS uses a parameter called OPTIONS. If set to GATHER AUTO, this option allows FND_STATS to determine automatically the tables for which statistics should be gathered, based on the change threshold. The

Modifications Threshold can be adjusted by the user by passing a value for modpercent, which by default is equal to 10. GATHER AUTO uses a database feature called Table Monitoring, which needs to be enabled for all the tables. A procedure called ENABLE_SCHEMA_MONITORING has been provided to enable monitoring on all tables for a given schema or all Applications schemas.

Gathering Statistics for the CBO

Oracle Applications provides concurrent programs that use the package FND_STATS to gather statistics for your applications database objects. For information on DBMS_STATS, see Oracle Supplied PL/SQL Packages Manual.

The following concurrent programs are available for collecting and maintaining statistics:

- Gather Table Statistics
- Backup Table Statistics
- Restore Table Statistics
- Gather Schema Statistics
- Purge FND_STATS History Records

Gather Table Statistics

The Gather Table Statistics program gathers the table statistics for the specified table. This program can optionally backup the existing statistics in the FND_STATTAB table before gathering new statistics. If the value of backup_flag is BACKUP, then FND_STATS exports the old statistics using dbms_stats.export_table_stats before gathering the new statistics. The exported data is stored in FND_STATTAB. If the value of backup_flag is anything other than BACKUP then the old table statistics are not saved. This program also gathers index statistics for the table by default. For a detailed description of the procedure used by this concurrent program, see: GATHER_TABLE_STATS Procedure.

Parameters

Owner Name The owner of the table.

Table Name The name of the table.

Estimate Percent The sampling percentage. If left blank, a default value of 10 is used. The valid range is from 0 to 100.

Degree The degree of parallelism to be used for gathering statistics. If a Degree is not provided, it defaults to the minimum of

parallel_max_servers and cpu_count.

Partition Name The name of the partition.

Backup Flag The backup flag indicates whether to backup statistics. Set this flag to "BACKUP" to back up your statistics.

Granularity The granularity of statistics to collect (only relevant for tables that are partitioned). Valid values are:

- DEFAULT - Gather global and partition–level statistics.
- SUBPARTITION - Gather subpartition–level statistics.
- PARTITION - Gather partition–level statistics.
- GLOBAL - Gather global statistics.
- ALL - Gather all (subpartition, partition, and global) statistics.

History Mode This parameter controls the amount of history records that are created. Valid modes are LASTRUN, FULL and NONE. The default is LASTRUN. For an explanation of the different modes, please refer to the GATHER_TABLE_STATS Procedure

Invalidate Dependent Cursors This flag indicates whether cursors dependent on the table being analyzed should be invalidated or not. This parameter is ignored if you are running a database prior to Oracle 9i Release 2 (9.2.x).

Backup Table Statistics

This concurrent program backs up the current statistics of the given table into the FND_STATTAB table. This program also backs up the related index and column statistics by default.

An identifier, commonly referred to as STATID, can be associated with the backup up statistics. This STATID allows you to restore a particular version of the statistics using the Restore Table Statistics concurrent program. Statistics for the same object can be backed up with different STATIDs. You can even backup different versions of the statistics for the same object by assigning different STATIDs.

For a detailed description of the procedure used by this concurrent

program, see: BACKUP_TABLE_STATS Procedure.

Parameters

Schema Name The name of the schema. The value ALL means all Oracle Applications schemas.

Table Name The name of the table.

Statistics ID An optional identifier to associate with these statistics within FND_STATTAB. The default STATID is BACKUP.

Partition Name Name of the table partition. If the table is partitioned and if the partition name is NULL, then global and partition table statistics are exported.

Restore Table Statistics

This concurrent program allows you to restore the previously backed up table statistics for a given statistics identifier, commonly referred to as the STATID.

All index and column statistics associated with the specified table are restored as well.

For a detailed description of the procedure used by this concurrent

program, see: RESTORE_TABLE_STATS Procedure

Parameters

Schema Name The name of the schema. The value ALL means all Oracle Applications schemas.

Table Name The name of the table.

Statistics ID An optional identifier to associate with these statistics within FND_STATTAB. The default STATID is BACKUP.

Partition Name Name of the table partition. If the table is partitioned and if the partition name is NULL, then global and partition table statistics are imported.

Gather Schema Statistics

This concurrent program gathers the specified schema level statistics.

Before gathering the statistics, this program can also create a backup of the current statistics, depending on the value of the Backup Flag. If for some reason, the earlier statistics need to be restored, that can be done using the Restore Schema Statistics concurrent program. The STATID used for this backup is NULL.

This program also creates histograms on the columns seeded in the FND_HISTOGRAM_COLS table.

For a detailed description of the procedure used by this concurrent program, see information on the GATHER_SCHEMA_STATS procedure.

Parameters

Schema Name

Schema for which statistics are to be gathered. Specify ALL for all Oracle Applications schemas (all schemas that have an entry in the FND_PRODUCT_INSTALLATIONS table).

Percent

The sampling percentage. If left blank, the default value of 10 is used. The valid range is from 0 to 100.

Degree

The degree of parallelism to be used for gathering statistics. If a Degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.

Backup Flag

The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current statistics into the FND_STATTAB table. If NOBACKUP is used, then the GATHER_SCHEMA_STATS procedure will not backup the current statistics. This way the GATHER_SCHEMA_STATS procedure will run faster.

Restart Request ID

In the case where the Gather Schema Statistics run fails due to whatever reasons, the concurrent request can be re-submitted and it will pick up where the failed run left off, if you provide the concurrent request_id of the failed run.

History Mode

This parameter controls the amount of history records that are created. The history records, stored in FND_STATS_HIST can be queried to find out when stats were gathered on a particular object and the amount of time it took to gather statistics on that object.

- Last Run - History records for each object are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the object. This is the default behavior.

- Full - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records " concurrent program should be run periodically to purge the FND_STATS_HIST table.

- None - This mode does not generate any history information. If this mode is used, the run cannot be restarted.

Gather Options

This parameter specifies how objects are selected for statistics gathering.

- GATHER : All tables and indexes of the schema **schemaname** are selected for stats gathering. This is the default.
- GATHER AUTO : Tables of the schema schemaname for which the percentage of modifications has exceeded modpercent are selected for statistics gathering. Indexes of these tables are selected by default. Table monitoring needs to be enabled before using this option.
- GATHER EMPTY : Statistics are gathered only for tables and indexes that are missing statistics.
- LIST AUTO : This option does not gather statistics. It only provides a listing of all the tables that will be selected for statistic gathering, if the GATHER AUTO option is used.
- LIST EMPTY : This option does not gather statistics. It only provides a listing of all the tables that will be selected for statistics gathering, if the GATHER EMPTY option is used.

Modifications Threshold

Applicable only to GATHER AUTO and LIST AUTO options. This parameter specifies the percentage of modifications (with respect to the total rows) that have to take place on a table before it can be picked up for AUTO statistics gathering.

Invalidate Dependent Cursors

This flag indicates whether cursors dependent on the table being analyzed should be invalidated or not. By default, dependent cursors are invalidated. This parameter is ignored if you are running a database prior to Oracle 9i Release 2 (9.2.x).

Gather Column Statistics

This concurrent program should be used for gathering the Column Statistics, i.e.

creating a histogram on a given column.

The procedure takes a backup into the FND_STATTAB table before gathering the statistics.

For a detailed description of the procedure used by this concurrent

program, see: GATHER_COLUMN_STATS Procedure

Parameters

Table Owner	The owner of the table.
Table Name	The name of the table.
Column Name	The name of the column.
Estimate Percent	The sampling percentage. If left blank, a default value of 10 is used. The valid range is from 0 to 100.
Parallel Degree	The degree of parallelism to be used for gathering statistics. If a Degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.
Bucket Size	The number histogram buckets.
Backup Flag	The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current column statistics into the FND_STATTAB table. If left blank, it defaults to NOBACKUP.

Gather All Column Statistics

This concurrent program is obsolete.

Purge FND_STATS History Records

This program can be run to purge the history records from the FND_STATS_HIST table. This program should be scheduled to run periodically if statistics are being gathered with History Mode as FULL. You do not need to run this program if you gather statistics with History Mode as NONE or the default – LASTRUN.

Parameters

Purge Mode	The Purge Mode can take one of the two values: DATE or REQUEST. If the mode chosen is DATE, history records are purged based on the date range, otherwise, if it is REQUEST, records are purged based on the Request ID.
From Value	Start Date or Request ID

To Value End Date or Request ID.

FND_STATS Package

The FND_STATS package provides procedures for gathering statistics for Oracle Applications database objects. It also provides procedures for backing up the current statistics into the table - FND_STATTAB, and restoring them back if desired. This package also allows users to specify the degree of parallelism. That helps speed up statistics gathering for large objects. FND_STATS can also maintain a history of its actions in a table called FND_STATS_HIST. The data in this table is used to provide restart ability, and can also be queried to find out the time taken to gather statistics on each object.

FND_STATS relies on the Oracle-supplied package DBMS_STATS to perform the actual statistics gathering. For more information on DBMS_STATS, refer to the Oracle database Tuning and Supplied Packages Reference manuals.

CREATE_STAT_TABLE Procedure

This procedure creates the table that is required for backing up the statistics.

There are two versions of this procedure. The first one does not need any arguments and creates the table with the default name - FND_STATTAB in the schema corresponding to the FND product. The second version allows you to provide the schema name, table name and the tablespace for the statistics table.

Syntax

```
FND_STATS.CREATE_STAT_TABLE ;

FND_STATS.CREATE_STAT_TABLE (
   schemaname  IN  VARCHAR2,
   tabname     IN  VARCHAR2,
   tblspcname  IN  VARCHAR2);
```

Parameters

schemaname Name of the schema.

tabname Name of the table.

tblspcname Tablespace in which to create the statistics tables. If none is specified, then the tables are created in the user's default tablespace.

BACKUP_TABLE_STATS

This procedure backs up the statistics for the given table in the FND_STATTAB table.

Setting cascade to TRUE results in all index and column statistics associated with the specified table to be stored as well. An identifier, commonly referred to as STATID, can be associated with the backup up statistics. This STATID allows you to restore a particular version of the statistics using the RESTORE_TABLE_STATS procedure.

Syntax

```
FND_STATS.BACKUP_TABLE_STATS (
   schemaname  VARCHAR2,
   tabname     VARCHAR2,
   statid      VARCHAR2 DEFAULT 'BACKUP',
   partname    VARCHAR2 DEFAULT NULL,
   cascade     BOOLEAN  DEFAULT TRUE);
```

Parameters

schemaname Name of the schema.

tabname Name of the table.

statid Optional identifier to associate with these statistics within FND_STATTAB.

partname Name of the table partition. If the table is partitioned and if partname is NULL, then global and partition table statistics are exported.

cascade If TRUE, then column and index statistics for this table are also exported.

BACKUP_SCHEMA_STATS Procedure

This procedure can be used to backup the statistics for an entire schema. The statistics are backed up into the FND_STATTAB table. A different version can be stored by specifying a different statid. An identifier, commonly referred to as STATID, can be associated with the backup up statistics. This STATID allows you to restore a particular version of the statistics using the RESTORE_SCHEMA_STATS procedure.

Syntax

```
FND_STATS.BACKUP_TABLE_STATS (
   schemaname  VARCHAR2,
   statid      VARCHAR2 DEFAULT NULL);
```

Parameters

schemaname Name of the schema. ALL means all Oracle Applications schemas.

statid Optional identifier to associate with these statistics within FND_STATTAB.

RESTORE_SCHEMA_STATS Procedure

This procedure restores statistics for the given schema, that were previously backed up in the FND_STATTAB table, into the dictionary. Statid can be provided to distinguish between different sets of statistics for the same object.

Syntax

```
FND_STATS.RESTORE_SCHEMA_STATS (
   schemaname  VARCHAR2,
   statid VARCHAR2 DEFAULT NULL
   );
```

Parameters

schemaname Name of the schema. ALL means all Oracle Applications schemas.

statid Optional identifier to associate with these statistics within FND_STATTAB.

RESTORE_TABLE_STATS Procedure

This procedure restores statistics for the given table from the FND_STATTAB table for the given statid (optional) and transfers them back to the dictionary. Setting cascade to TRUE results in all index and column statistics associated with the specified table being imported also.

Syntax

```
FND_STATS.RESTORE_TABLE_STATS (
   ownname  VARCHAR2,
   tabname  VARCHAR2,
   statid   VARCHAR2 DEFAULT NULL,
   partname VARCHAR2 DEFAULT NULL,
   cascade  BOOLEAN  DEFAULT TRUE,
   );
```

Parameters

ownname Name of the schema.

tabname Name of the table.

statid Optional identifier to associate with these statistics within FND_STATTAB.

partname Name of the table partition. If the table is partitioned and if partname is NULL, then global and partition table statistics are exported.

cascade If TRUE, then column and index statistics for this table are also exported.

RESTORE_COLUMN_STATS Procedure

This procedure restores statistics for the given column from the FND_STATTAB table for the given statid (optional) and transfers them back to the dictionary. There are two versions of this procedure. One first one requires the table owner, table name and column name to be supplied. The second version restores the statistics for all the columns seeded in the FND_HISTOGRAM_COLS table.

Syntax

```
FND_STATS.RESTORE_COLUMN_STATS (
   ownname  VARCHAR2,
   tabname  VARCHAR2,
   colname  VARCHAR2,
   partname VARCHAR2 DEFAULT NULL,
   statid   VARCHAR2 DEFAULT NULL
   );

FND_STATS.RESTORE_COLUMN_STATS (
   statid   VARCHAR2 DEFAULT NULL
   );
```

Parameters

ownname Name of the schema.

tabname Name of the table.

colname Name of the column. Optional identifier to associate with these statistics within FND_STATTAB.

partname Name of the table partition. If the table is partitioned and if partname is NULL, then global and partition table statistics are exported.

statid Optional identifier to associate with these statistics within FND_STATTAB.

ENABLE_SCHEMA_MONITORING Procedure

This procedure should be used for enabling the Monitoring option for all tables in the specified schema. Monitoring option should be enabled before using the GATHER

AUTO or LIST AUTO option of GATHER_SCHEMA_STATS. If the value of the schemaname argument is ALL, then the Monitoring option is enabled for all tables that belong to all schemas registered in Oracle Applications

Syntax

```
FND_STATS.ENABLE_SCHEMA_MONITORING (
    schemaname  VARCHAR2 DEFAULT 'ALL');
```

Parameters

schemaname Name of the schema for which Monitoring should be enabled.

DISABLE_SCHEMA_MONITORING Procedure

This procedure should be used for disabling the Monitoring option for all tables in the specified schema. If the value of the schemaname argument is ALL, then the Monitoring option is disabled for all tables that belong to all schemas registered in Oracle Applications.

Syntax

```
FND_STATS.DISABLE_SCHEMA_MONITORING (
    schemaname  VARCHAR2 DEFAULT 'ALL');
```

Parameters

schemaname Name of the schema for which Monitoring should be disabled.

GATHER_SCHEMA_STATS Procedure

This procedure gathers statistics for all objects in a schema. Statistics are gathered with the granularity of DEFAULT. This procedure is also available through the concurrent program "Gather Schema Statistics." If this procedure fails at any time during operation, supplying the request ID for the request that failed can restart it. The request ID can be captured when the program is started from concurrent manager or can be queried from the FND_STATS_HIST table.

GATHER_SCHEMA_STATS cannot be executed directly in sqlplus because of an OUT parameter. The procedure GATHER_SCHEMA_STATISTICS has been provided for gathering schema statistics from the sqlplus prompt.

Syntax

```
FND_STATS.GATHER_SCHEMA_STATS (
    schemaname        VARCHAR2,
    estimate_percent  NUMBER DEFAULT NULL,
    degree            NUMBER DEFAULT NULL,
    internal_flag     NUMBER DEFAULT NULL,
    Errors  OUT Error_Out,
    request_id   NUMBER default null,
    hmode        VARCHAR2 default 'LASTRUN',
    options in   VARCHAR2 default 'GATHER',
    modpercent   NUMBER default 10,
    invalidate   VARCHAR2 default 'Y'
);

FND_STATS.GATHER_SCHEMA_STATISTICS (
    schemaname        VARCHAR2,
    estimate_percent  NUMBER DEFAULT NULL,
    degree            NUMBER DEFAULT NULL,
    internal_flag     NUMBER DEFAULT NULL,
    request_id        NUMBER DEFAULT NULL,
    hmode             VARCHAR2 DEFAULT 'LASTRUN',
    options in        VARCHAR2 DEFAULT 'GATHER',
    modpercent        NUMBER DEFAULT 10,
    invalidate        VARCHAR2 DEFAULT 'Y'
);
```

Parameters

schemaname — Schema to analyze. ALL means all Oracle Applications schemas.

estimate_percent — The sampling percentage. If a value is not provided, the default value of 10 is used. The valid range is from 0 to 100.

degree — The degree of parallelism to be used for gathering statistics. If a degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.

internal_flag — The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current statistics into the FND_STATTAB table. If NOBACKUP is used, then the GATHER_SCHEMA_STATS procedure will not backup the current statistics. This way the GATHER_SCHEMA_STATS procedure will run faster.

errors — User defined Type for holding the Error messages .

Request_id — A request_id can be provided to identify the history records for a given statistics gathering run. This parameter is also used for providing restart ability. In case, a statistics gathering run fails due to whatever reasons, subsequent

submission can pick up where the failed run left off, if you provide the request_id of the failed run.

Hmode

This parameter controls the amount of history records that are created. The history records, stored in FND_STATS_HIST can be queried to find out when statistics were gathered on a particular object and the amount of time it took to gather statistics on that object.

LASTRUN - History records for each schema are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the index. This is the default behavior.

FULL - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records" concurrent program should be run periodically to purge the FND_STATS_HIST table.

NONE - This mode does not generate any history information. If this mode is used, the run cannot be restarted.

Options

This parameter specifies how objects are selected for statistics gathering.

GATHER - All tables and indexes of the schema <schemaname> are selected for stats gathering. This is the default.

GATHER AUTO - Tables of the schema schemaname for which the percentage of modifications has exceeded modpercent are selected for statistics gathering. Indexes of these tables are selected by default. Table monitoring needs to be enabled before using this option.

GATHER EMPTY - Statistics are gathered only for tables and indexes that are missing statistics.

LIST AUTO - This option does not gather statistics. It only provides a listing of all the tables that will be selected for statistic gathering, if the GATHER AUTO option is used.

LIST EMPTY - This option does not gather statistics. It only provides a listing of all the tables that will be selected for statistics gathering, if the GATHER EMPTY option is used.

Modpercent

Applicable only to GATHER AUTO and LIST AUTO

options. This parameter specifies the percentage of modifications (with respect to the total rows) that have to take place on a table before it can be picked up for AUTO statistics gathering.

Invalidate This flag indicates whether cursors dependent on the table being analyzed should be invalidated. By default, dependent cursors are invalidated. This parameter is ignored if you are running a database prior to Oracle 9i Release 2 (9.2.x).

Exceptions

```
ORA-20000: Schema does not exist or insufficient privileges.
ORA-20001: Bad input value.
```

GATHER_INDEX_STATS Procedure

This procedure gathers statistics for the specified index.

Syntax

```
FND_STATS.GATHER_INDEX_STATS (
    ownname   VARCHAR2,
    indname   VARCHAR2,
    percent   NUMBER DEFAULT NULL,
    partname   VARCHAR2 DEFAULT NULL,
    backup_flag  VARCHAR2 DEFAULT NULL,
    hmode      VARCHAR2 DEFAULT 'LASTRUN',
    invalidate   VARCHAR2 DEFAULT 'Y'

);
```

Parameters

ownname Schema of index to analyze.

indname Name of index.

percent The sampling percentage. If left blank, the default value of 10 is used. The valid range is from 0 to 100.

partname Partition name.

backup_flag The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current column statistics into the FND_STATTAB table. If left blank, it defaults to NOBACKUP.

Hmode This parameter controls the amount of history records that

are created.

LASTRUN - History records for each index are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the index. This is the default behavior.

FULL - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records" concurrent program should be run periodically to purge the FND_STATS_HIST table.

NONE - This mode does not generate any history information. If this mode is used, the run cannot be restarted.

Invalidate This flag indicates whether cursors dependent on the index being analyzed should be invalidated. By default, dependent cursors are invalidated.

GATHER_TABLE_STATS Procedure

This procedure gathers table, column and index statistics. It attempts to parallelize as much of the work as possible. This operation does not parallelize if the user does not have select privilege on the table being analyzed.

Syntax

```
FND_STATS.GATHER_TABLE_STATS (
    ownname   VARCHAR2,
    tabname   VARCHAR2,
    percent   NUMBER DEFAULT NULL,
    degree   NUMBER DEFAULT NULL,
    partname   VARCHAR2 DEFAULT NULL,
    backup_flag  VARCHAR2 DEFAULT NULL,
    cascade   BOOLEAN DEFAULT TRUE,
    granularity VARCHAR2 DEFAULT 'DEFAULT',
    hmode   VARCHAR2 DEFAULT 'LASTRUN',
    invalidate VARCHAR2 DEFAULT 'Y'
);
```

Parameters

ownname Owner of the table.

tabname Name of the table.

percent The sampling percentage. If left blank, the default value of

10 is used. The valid range is from 0 to 100.

degree The degree of parallelism to be used for gathering statistics. If a degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.

partname Name of the partition.

backup_flag The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current table statistics into the FND_STATTAB table. If left blank, it defaults to NOBACKUP.

cascade When set to TRUE index statistics are gathered in addition to gathering statistics for the specified table.Index statistics gathering is not parallelized. Using this option is equivalent to running the GATHER_INDEX_STATS procedure on each of the table's indexes

granularity The granularity of statistics to collect (only relevant for tables that are partitioned). Valid values are:

DEFAULT - Gather global and partition–level statistics.

SUBPARTITION - Gather subpartition–level statistics.

PARTITION - Gather partition–level statistics.

GLOBAL - Gather global statistics.

ALL - Gather all (subpartition, partition, and global) statistics.

Hmode This parameter controls the amount of history records that are created.

LASTRUN - History records for each index are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the index. This is the default behavior.

FULL - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records" concurrent program should be run periodically to purge the FND_STATS_HIST table.

NONE - This mode does not generate any history information. If this mode is used, the run cannot be

restarted.

Invalidate This flag indicates whether cursors dependent on the index being analyzed should be invalidated. By default, dependent cursors are invalidated.

GATHER_COLUMN_STATS Procedure

This procedure should be used for gathering the Column Statistics, i.e. creating a histogram on a given column.

There are two versions of this procedure. The first one gathers statistics on all columns seeded in the FND_HISTOGRAM_COLS for the given appl_id. If NULL, all seeded histograms are created. The other version gathers column statistics for the specified column.

Syntax

```
FND_STATS.GATHER_COLUMN_STATS (
    appl_id     NUMBER DEFAULT NULL,
    percent    NUMBER DEFAULT NULL,
    degree      NUMBER DEFAULT NULL,
    backup_flag VARCHAR2 DEFAULT NULL,
    Errors      OUT Error_Out,
    hmode       VARCHAR2 DEFAULT 'LASTRUN',
    invalidate  VARCHAR2 DEFAULT 'Y'
);

FND_STATS.GATHER_COLUMN_STATS (
    ownname      VARCHAR2,
    tabname      VARCHAR2,
    colname      VARCHAR2,
    percent      NUMBER DEFAULT NULL,
    degree       NUMBER DEFAULT NULL,
    hsize        NUMBER DEFAULT 254,
    backup_flag  VARCHAR2 DEFAULT NULL,
    partname     VARCHAR2 DEFAULT NULL,
    hmode        VARCHAR2 DEFAULT 'LASTRUN',
    invalidate   VARCHAR2 DEFAULT 'Y'
);
```

Parameters

appl_id Application ID.

ownname Owner of the table.

colname Column name.

tabname Table name.

partname Name of the partition.

percent	The sampling percentage. If left blank, the default value of 10 is used. The valid range is from 0 to 100.
degree	The degree of parallelism to be used for gathering statistics. If a degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.
hsize	Number of buckets in the histogram.
backup_flag	The backup flag indicates whether to backup statistics. Set this flag to BACKUP if you wish to back up the current column statistics into the FND_STATTAB table. If left blank, it defaults to NOBACKUP.
errors	User defined Type for holding the Error messages.
hmode	This parameter controls the amount of history records that are created. LASTRUN - History records for each index are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the index. This is the default behavior. FULL - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records" concurrent program should be run periodically to purge the FND_STATS_HIST table. NONE - This mode does not generate any history information. If this mode is used, the run cannot be restarted.
Invalidate	This flag indicates whether cursors dependent on the index being analyzed should be invalidated. By default, dependent cursors are invalidated.

GATHER_ALL_COLUMN_STATS Procedure

This procedure gathers column statistics, i.e. creates histograms on all columns that are seeded in the FND_HISTOGRAM_COLS, belonging to the specified schema .

Syntax

```
FND_STATS.GATHER_ALL_COLUMN_STATS (
    ownname    VARCHAR2 ,
    percent    NUMBER DEFAULT NULL,
    degree     NUMBER DEFAULT NULL,
    hmode      VARCHAR2 DEFAULT 'LASTRUN',
    invalidate VARCHAR2 DEFAULT 'Y'

);
```

Parameters

ownname Schema for which seeded histograms have to be created. ALL means all Applications schemas.

percent The sampling percentage. If left blank, the default value of 10 is used. The valid range is from 0 to 100.

degree The degree of parallelism to be used for gathering statistics. If a degree is not provided, it defaults to the minimum of parallel_max_servers and cpu_count.

Hmode This parameter controls the amount of history records that are created.

LASTRUN - History records for each index are maintained only for the last gather statistics run. Each subsequent run will overwrite the previous history record for the index. This is the default behavior

FULL - This mode does not overwrite any history information. History records are created for each run and are identified by the Request ID. If a Request ID is not provided, one is generated automatically. If this mode is used, the "Purge FND_STATS History Records" concurrent program should be run periodically to purge the FND_STATS_HIST table.

NONE - This mode does not generate any history information. If this mode is used, the run cannot be restarted.

Invalidate This flag indicates whether cursors dependent on the index being analyzed should be invalidated. By default, dependent cursors are invalidated.

ANALYZE_ALL_COLUMNS Procedure

This procedure has been made obsolete.

LOAD_XCLUD_STATS Procedure

This procedure has been made obsolete.

PURGE_STAT_HISTORY Procedure

This procedure should be used for purging the unwanted history records from the fnd_stats_hist table. There are two versions of this procedure. The first one takes in a range of request ids and deletes all history records that fall within that range. The second version takes a range of dates as arguments and all the history records falling in-between that range are deleted. The delete takes place as an autonomous transaction.

Syntax

```
FND_STATS.PURGE_STAT_HIST (
    From_req_id  NUMBER,
    To_req_id   NUMBER);

FND_STATS. PURGE_STAT_HIST(
    Purge_from_date   VARCHAR2,
    Purge_to_date   VARCHAR2);
```

Parameters

from_req_id Start Request ID.

to_req_id End Request ID.

purge_from_date Start Purge Date.

purge_to_date End Purge Date.

CHECK_HISTOGRAM_COLS Procedure

For a given list of comma-separated tables, this procedure checks the data in all the leading columns of all the non-unique indexes of those tables and determines if histograms need to be created for those columns. The algorithm for this procedure is:

```
select
decode(floor(sum(tot)/(max(cnt)*FACTOR)),0,'YES','NO') HIST
from (select count(col) cnt , count(*) tot
from tab sample (PERCENT)
where col is not null
group by col);
```

The decode statement determines whether a single value occupies 1/FACTOR or more of the sample PERCENT.

If sum(cnt) is very small (a small non-null sample), the results may be inaccurate. A count(*) of at least 3000 is recommended. The procedure is run from a SQL prompt after setting the server output on.

Syntax

```
FND_STATS.CHECK_HISTOGRAM_COLS (
 tablelist VARCHAR2,
 factor  NUMBER DEFAULT 75,
 percent  NUMBER DEFAULT 10,
 degree  NUMBER DEFAULT NULL);
```

Parameters

tablelist A comma separated list of tables. It should be of the form schema.tablename. A wildcard in the tablename is also allowed. For example, tablelist=>'oe.so%head% , pa.pa_exp% , ar.ra_customers'. The owner part is mandatory.

factor The factor for calculating the histograms.

percent Sample percent.

degree Degree of parallelization.

VERIFY_STATS Procedure

For a given list of comma-separated tables, or for a given schema name, this procedure reports the statistics in the data dictionary tables for the tables, indexes, and histograms.

Syntax

```
FND_STATS.VERIFY_STATS (
 schemaname    VARCHAR2 DEFAULT NULL,
 tablelist     VARCHAR2 DEFAULT NULL,
 days_old      NUMBER DEFAULT NULL,
 column_stat   BOOLEAN DEFAULT FALSE);
```

Parameters

schemaname The name of a schema. If schemaname is NULL (which is the default), then the procedure reports on the given list of tables.

tablelist A comma-separated list of tables. If the tablename is not of the form <schema>.<tablename> then the schema is the value of the schemaname parameter. If the tablelist is NULL (the default), then the procedure reports on all the tables for the specified schemaname.

days_old The procedure only reports those tables whose statistics are older than the days_old number of days. The default is

	NULL, which means the procedure will report on all the tables.
column_stat	If TRUE, the procedure reports column statistics for the export_table_stats table. The default is FALSE.

13

Oracle Applications and Oracle Real Application Clusters

Introduction to Oracle Real Application Clusters

This section gives an overview of the steps required to install Oracle Applications in an environment that uses Oracle Real Application Clusters (Oracle RAC).

For full details of using Oracle Applications Release 12 with Oracle RAC, see Oracle *MetaLink* Note 388577.1, *Oracle E-Business Suite Release 12 with 10g Release 2 Real Application Clusters and Automatic Storage Management*.

Prerequisites

Several steps must be followed to prepare your system for utilizing Oracle RAC to support Oracle E-Business Suite. In summary, these are as follows:

1. Install requisite Oracle Clusterware
2. Install requisite Oracle database software
3. Install requisite database components from Oracle 10g Companion CD
4. Upgrade Cluster Ready Services and database software (if applicable)
5. Create Automatic Storage Management (ASM) instances (optional)

Migrating to Oracle RAC

Conversion of an Oracle 10g database to utilize Oracle RAC is accomplished by running the `rconfig` utility against a sample XML file, ConvertToRAC.xml, which you modify to meet the particular requirements of your system.

In order to validate and test the settings specified for converting to Oracle RAC with rconfig, `rconfig` should be run in "Verify only" mode prior to running the actual conversion. This will perform a test run, in which `rconfig` will validate parameter settings, and report any issues that need to be resolved before the actual conversion is undertaken.

Note: See Oracle*MetaLink* Note 388577.1 for full details.

Establishing the Oracle E-Business Suite Environment for Oracle RAC

On the Oracle E-Business Suite side, a number of steps are needed to prepare the environment for Oracle RAC. Follow the detailed instructions in Oracle*MetaLink* Note 388577.1, *Configuring Oracle Applications Release 12 with 10g R2 RAC*.

In summary, you need to:

1. Run AutoConfig on the Applications database tier.
2. Check the tnsnames.ora and listener.ora files in the $INST_TOP/ora/10.1.2/network/admin and $INST_TOP/ora/10.1.3/network/admin/<context_name>.
3. Ensure that the correct TNS aliases have been generated for load balance and failover.
4. Ensure that all the aliases are defined using the virtual hostnames.
5. Verify the dbc file located at $FND_SECURE. Ensure that the parameter APPS_JDBC_URL is configured with all instances in the environment, and load_balance is set to YES.
6. Run the AutoConfig context editor (from Oracle Applications Manager) and specify values for the requisite load-balancing variables for Forms-based applications and HTML-based applications.
7. Run AutoConfig again.
8. Restart the Applications processes using the new scripts generated by AutoConfig.

Configuring Parallel Concurrent Processing with Oracle RAC

To take full advantage of Parallel Concurrent Processing (PCP), you will need to have more than one Concurrent Processing node in your environment. If you do not, refer to Oracle*MetaLink* for details of cloning the application tier.

The key steps in configuring concurrent parallel processing are as follows:

1. Configure PCP on all concurrent processing nodes.
2. Set up transaction managers.
3. Set up load balancing on concurrent processing nodes by running AutoConfig.

For further details, see Oracle*MetaLink* Note 388577.1, *Configuring Oracle Applications Release 12 with 10g R2 RAC*.

Related Topics

Overview of Parallel Concurrent Processing, page 7-42

Managing Parallel Concurrent Processing, page 7-45

14

Document Sequences

What is a Document Sequence?

A document sequence uniquely numbers documents generated by an Oracle Applications product. Using Oracle Applications, you initiate a transaction by entering data through a form and generating a document, for example, an invoice. A document sequence generates an audit trail that identifies the application that created the transaction, for example, Oracle Receivables, and the original document that was generated, for example, invoice number 1234.

Document sequences can provide proof of completeness. For example, document sequences can be used to account for every transaction, even transactions that fail.

Document sequences can also provide an audit trail. For example, a document sequence can provide an audit trail from the general ledger into the subsidiary ledger, and to the document that originally affected the account balance.

Document sequences generate audit data, so even if documents are deleted, their audit records remain.

Related Topics

Defining a Document Sequence, page 14-1

Defining Document Categories, page 14-4

Assigning a Document Sequence, page 14-5

Document Numbering vs. Document Entry, page 14-7

Defining a Document Sequence

To define a sequence, you select a sequence name and an application to "own" the sequence.

- A sequence can number documents stored in database tables belonging to its

owning application.

- Audit records for a sequence are stored in the application's audit table, titled *Application Short Name*_DOC_SEQUENCE_AUDIT. For example, the audit table for a sequence owned by Oracle Payables is AP_DOC_SEQUENCE_AUDIT.

  Important: Your database administrator must grant access to an application's audit table for all ORACLE usernames associated with responsibilities that will use the sequence (responsibilities that access forms using the sequence).

You can set start and end dates for when the sequence is available. The start date defaults to the current date. By default, there is no end date, so the sequence definition does not expire.

You can choose whether a sequence numbers documents automatically, or accepts numbers manually entered by a user.

Automatic, Gapless, and Manual Numbering

Automatic numbering assigns a unique number to each document as it is generated. Automatic numbering is sequential by date and time of creation.

Gapless numbering also automatically generates a unique number for each document, but ensures that the document was successfully generated before assigning the number. With Gapless numbering, no sequence numbers are lost due to incomplete or failed document creation.

Important: We recommend that you choose this type only when gapless numbering is essential, as it may affect the performance of your system.

Manual numbering requires a user to assign a unique number to each document before it is generated. With manual numbering, numerical ordering and completeness is not enforced. Users can skip or omit numbers when entering the sequence value.

Automatic Numbering - Initial Value and Message Display

If you define a sequence to automatically number documents, you can:

- Enter an initial value for your sequence. The default is "1".

- Choose whether you want to display a message when a document is generated, telling the user the name of the sequence, and the sequence value (document number).

Two examples of sequence definitions, one with automatic numbering and the other with manual numbering, are represented in the table below.

Field in Document Sequences form	EXAMPLE 1 Sequence with Automatic Numbering	EXAMPLE 2 Sequence with Manual Numbering
(Sequence) NAME	AUTOPAY	ADJUSTMENTS
(Owning) APPLICATION	ORACLE PAYABLES - Sequence can number documents stored in an Oracle Payables database table.	ORACLE RECEIVABLES - Sequence can number documents stored in an Oracle Receivables database table.
EFFECTIVE DATE - START	CURRENT DATE & TIME (Default value)	OCT-01-94 User defines sequence "Adjustments" not to be available until Oct 1, 1994.
EFFECTIVE DATE - END	Field left blank. Sequence does not expire.	DEC-31-94 User defines sequence "Adjustments" to no longer be available after Dec 31, 1994.
(Numbering) TYPE	AUTOMATIC - Unique numbers are automatically generated in sequence. GAPLESS No omissions or gaps in numbers are possible, due to a rollback if the document creation is unsuccessful.	MANUAL - User must enter a unique number before transaction can be completed, and document is generated. User may skip or omit numbers.
INITIAL VALUE	1 (Default value) User could enter their own initial value, for example, 5700.	**Not Available when numbering type is Manual.**
MESSAGE	YES - When a document that is automatically numbered is created, a message displays the sequence name and the sequence value (document number).	**Not Available when numbering type is Manual.**

Related Topics

What is a Document Sequence?, page 14-1

Defining Document Categories, page 14-4

Defining Document Categories

Document categories organize documents into logical groups.

- A document category (also called a document type) is one of the rules you use to define which documents a sequence assigns numbers to.
- You can separately number each document category by assigning a different sequence to each category.

A document category identifies the database table that stores documents resulting from transactions your users enter.

- When you assign a sequence to a category, the sequence numbers the documents that are stored in a particular table.

Use categories to more precisely classify your documents. For example, you can categorize accounts receivable invoices into several different categories, such as:

- Chargebacks
- Deposits
- Guarantees
- Debit Memos
- Credit Memos
- Sales Invoices
- Customer Service Invoices

Similarly, you can categorize accounts payable or purchase invoices into several different categories, such as:

- Standard
- Expense Report
- Prepayment
- Interest
- Credit Memo
- Debit Memo

Related Topics

What is a Document Sequence?, page 14-1

Defining a Document Sequence, page 14-1

Assigning a Document Sequence, page 14-5

Assigning a Document Sequence

Before you can assign a sequence to number documents, you must define which documents are to be numbered.

Sequences versus Assignments

Defining a sequence is different from assigning a sequence to a series of documents.

- A sequence's definition determines whether a document's number is automatically generated or manually entered by the user.
- A sequence's assignment, that is, the documents a sequence is assigned to, is defined in the Sequence Assignments form.

Defining Documents for numbering by Assigned Sequences

You specify a combination of four rules that define any given document for assignment to a specific sequence name.

You can then assign a different (numbering) sequence to each document definition.

The four rules, that when combined, define what documents a selected sequence assigns numbers to are:

Application

You select the application that generates the documents you wish to number.

For example, to number sales invoices, you select Oracle Receivables.

Category

You select a document category to identify a logical subset of documents.

For example, if you do not want to number all invoices in Oracle Receivables, you can choose to number only the category of sales invoices.

A category identifies a table that stores transactions entered (documents generated) using an Oracle Application.

The Category values you can choose from to define a document are dependent upon the application you select.

Set of Books

You select the chart of accounts for your business that is affected by the documents you wish to number. You may optionally enable this rule through the Document Flexfield.

Method

You select the method that your documents are entered, automatic or manual. You may optionally enable this rule through the Document Flexfield.

Automatic is when a concurrent process, such as an external program, is set up to enter transaction data into an Oracle Application.

Manual is when a document is manually entered using a form in an application.

Assignment of Sequences to Document Definitions

For each unique document definition there can only be one active sequence assignment. A document definition consists of the Application, Category, and the optional Document Flexfield segments Set of Books and Method

> **Important:** When assigning sequences to a document definition, each active sequence can be assigned to only one unique combination of application and category (i.e., application table).

Active Assignments and Active Sequences

An active sequence assignment does not have a post dated end date. That is, the assignment's end date is not before the current date.

- An active sequence *assignment* either has no end date, or an end date that is not before the current date.
- A sequence assignment and its dates of effectivity are defined on the Sequence Assignments form.

A sequence *definition* must be active as well. That is, the sequence definition's end date (as opposed to its assignment's end date) must not be before the current date.

- A sequence definition and its dates of effectivity are defined on the Document Sequences form.

When you define a document sequence, you give the sequence a name, and define how the sequence numbers each document by:

- Choosing whether numbers are automatically generated in sequence, or entered manually by the user.

- Entering the initial value or first number in the sequence.

Related Topics

What is a Document Sequence?, page 14-1

Defining a Document Sequence, page 14-1

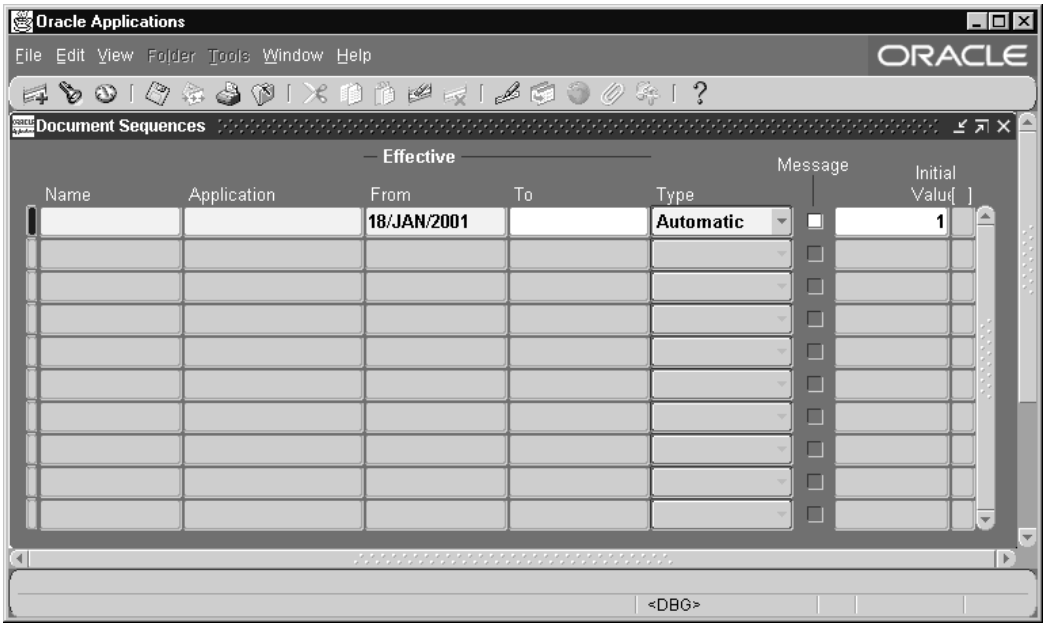
Defining Document Categories, page 14-4

Document Numbering vs. Document Entry, page 14-7

Document Numbering vs. Document Entry

Do not confuse the type of document numbering a sequence employs, which can be automatic or manual, with the method of entering documents, which is also defined as either automatic or manual.

Document Sequences Window



Name a new document sequence and define how the sequence numbers each document.

A document sequence uniquely numbers documents generated by an Oracle Applications product (for example, invoices generated by Oracle Receivables). Using the Sequence Assignments window, you assign your sequence to number only those documents that satisfy rules you define.

A document sequence numbers documents generated by an Oracle application, for example, journal entries generated by Oracle General Ledger.

Document sequences ensure that every document your users create can be accounted for. See: Sequences Assignments, page 14-12.

Document Sequences Block

Define the name, type of numbering scheme, effective dates, and initial value for your document sequence.

Name

Once entered, sequence names cannot be changed.

Application

Once selected, the application associated with your sequence cannot be changed.

Audit records for your sequence are stored in the application's audit table, titled *Application Short Name*_DOC_SEQUENCE_AUDIT. For example, the audit table for a sequence owned by Oracle Payables is AP_DOC_SEQUENCE_AUDIT.

Effective From/To

Enter the dates on which your document sequence takes effect/is no longer enabled. The Start on field automatically defaults to the current date, and once a sequence is defined, the start date cannot be changed. If you leave the End on field blank, your document sequence does not expire; and if you enter an end date and define your sequence, the end date cannot be modified later. If there is no end date defined and there are no active assignments for a sequence, you can disable the sequence by entering the current date as the end date. Once disabled, a sequence cannot be reactivated.

Type

Once defined, you cannot change the type of document numbering sequence.

Automatic Sequentially assigns, by date and time of creation, a unique number to each document as it is generated.

Manual Manual numbering requires a user to assign a number to each document before it is generated. You must enter unique values. However, please note that numerical ordering and completeness is not enforced.

Important: The Automatic-By-User type is currently not supported, and is reserved for a future version of Oracle Applications.

Warning: The Gapless Numbering type is valid only in the context of

certain localizations. We recommend that you choose this type only after consulting with Worldwide Support, as it may affect the performance of your system.

Message

Check the Message check box if you want each document to display a message (in the message line near the bottom of the screen) informing the user of the sequence name and value (number).

This check box only applies to sequences with the automatic type of numbering. Messages appear only on form displays, and are not written to a request's log file.

Once a sequence is defined, the message choice cannot be changed.

Initial Value

Enter a value for the first document in your sequence. This field only applies to sequences with automatic or gapless numbering type. Sequence numbers should not be greater than eight (8) digits.

If you leave this field blank, the first document is automatically assigned a value of "1".

Once a sequence is defined, this initial value cannot be changed.

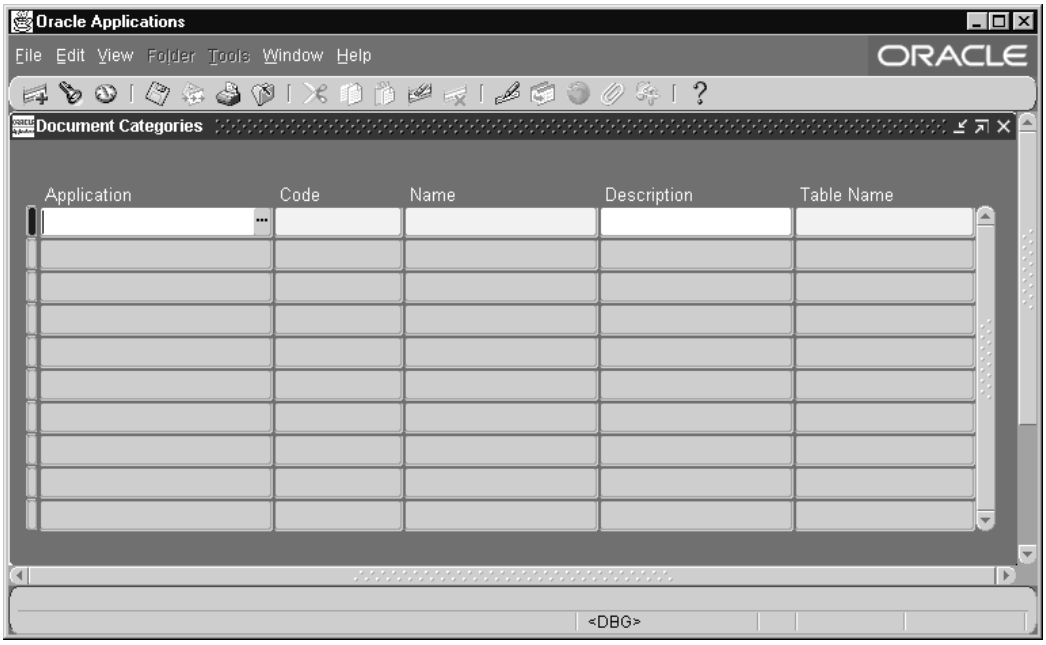
Related Topics

What is a Document Sequence?, page 14-1

Document Categories, page 14-10

Sequence Assignments, page 14-12

Document Categories Window



Define categories for your documents in order to divide your documents into logical groups, which you can number separately by assigning different sequences.

A document sequence uniquely numbers each document the sequence is assigned to.

- Using the Sequence Assignments form, you assign your sequence to number only documents that satisfy rules you define.
- Document category, or type, as it may be titled on some forms, is one of the rules that define which documents a sequence assigns numbers to.

Each category identifies a table that stores documents resulting from transactions your users generate.

- When you assign a sequence to a category, the sequence numbers the documents that are stored in the table.

Document Categories Block

Name a document category and associate a table with the category.

When you enter this block, Oracle automatically queries for any existing document categories.

Application

Once a category is defined, you cannot change the choice of application. Only tables belonging to the selected application can be assigned to a category.

Code

Category code must be unique within an application. Once a category is defined, you cannot update its code.

Name

You can update the name, if you wish. For example, if the category name is predefined, you can change the name to a more familiar value.

Description

You can update the description, if you wish. For example, if the category description is predefined, you can change the description to a more familiar value.

Table Name

Select the name of the table that stores the documents you want to identify by your category.

- When the sequential numbering feature checks for completeness or generates a report, it locates the category's documents in the table.
- Only tables belonging to the application associated with the category can be chosen.
- Once a category is defined, you cannot change the choice of table.

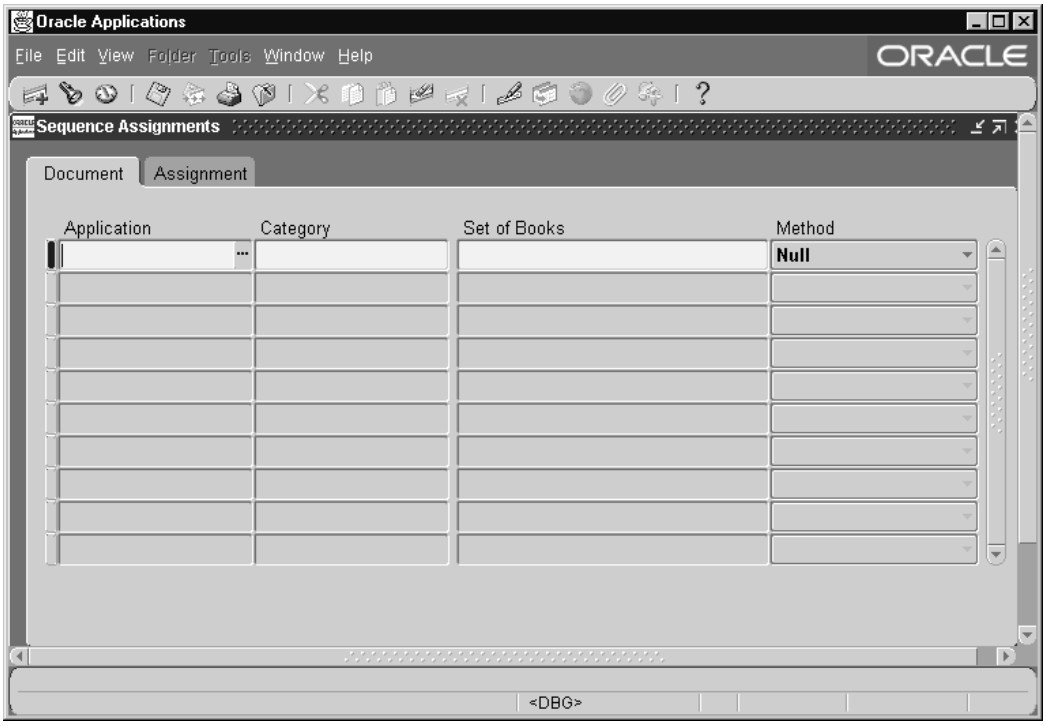
Related Topics

What is a Document Sequence?, page 14-1

Document Sequences, page 14-7

Sequence Assignments, page 14-12

Sequence Assignments Window



Define which documents a document sequence can number, and then assign the document sequence to your definition.

A document sequence numbers documents generated by an Oracle Applications product (for example, invoices generated by Oracle Receivables).

Documents can be defined by the application that generates them and their category (the table in which they are stored). Additional fields appear when the optional rules for defining documents (Set of Books and Method of document entry) are enabled.

Besides entering a document definition and assigning a sequence to it, you can, if you wish, enter effective dates for the assignment.

Prerequisites

Define the document sequence using the Document Sequences window. See: Document Sequences, page 14-7.

Sequence Assignments Block

Specify documents by the application that generates them and the category of the document (table where the documents are stored). You can also include in your document definition the set of books they affect, and the method by which the

document is entered.

Once a document definition is entered, you select a sequence to assign it to, and if you wish, enter effective dates for the assignment.

There can only be one active sequence assigned to each unique combination of Application, Category, Set of Books, and Method. The last two criterion are optional, and are set in the Document Flexfield.

However, the same sequence, the same numbering scheme, and initial value can be assigned to more than one combination of Application, Category, Set of Books, and Method as long as the Application and Category remain the same.

Application

Select the application that generates the documents you wish to number.

For example, to number sales invoices, you select Oracle Receivables.

Category

Select a document category to identify a logical subset of documents.

For example, if you do not want to number all invoices in Oracle Receivables, you can choose to number only the category of sales invoices.

Assignment Region

Since the effective dates for an assignment must fall within the sequence's start and end dates, the list of available sequences depends on the start and end dates specified for the assignment.

Start Date/End Date

Enter the dates on which the sequence assignment to your document definition takes effect/is no longer enabled. The Start Date field automatically defaults to the current date, and once a sequence assignment is defined, the start date cannot be changed.

If you leave the End Date field blank, your sequence assignment does not expire; and if you enter an end date and define your sequence assignment, the end date cannot be modified later.

If there is no end date defined and there are no active assignments for a sequence, you can disable the sequence assignment by entering the current date as the end date. Once disabled, a sequence assignment cannot be reactivated.

Sequence

Select a sequence to assign to your document definition. The sequence's application and the document's application must be the same.

Once you define a sequence assignment, the sequence name cannot be updated later.

If you want to disable the sequence assignment and assign a new sequence to the document definition (Document Flexfield combination), you must first, enter an End Date to disable the current sequence assignment, then, second, create a new record (row) for the new assignment.

Document Flexfield

The Document Flexfield may consist of none, one or two segments.

Set of Books Select the chart of accounts for your business that is affected by the documents you wish to number.

Method Select the method that your documents are entered, automatic or manual.

Automatic is when a concurrent process (e.g., an external program) enters transaction data into an Oracle Application, which generates documents.

Manual is when a document is manually entered using a form in an application.

Once defined, a Document Flexfield definition cannot be updated. You may not define additional segments for the Document Flexfield.

> **Important:** To enable this descriptive flexfield, use the Descriptive Flexfield Segments window. Select the application *Application Object Library*, and the title "Document Flexfield". Be sure to unfreeze the flexfield; then, navigate to the Segments window and enable the segments. Freeze your flexfield after you set it up, and save and compile the new definition.

Related Topics

Descriptive Flexfields Concepts, *Oracle Applications Flexfields Guide*

What is a Document Sequence?, page 14-1

Document Sequences, page 14-7

Document Categories, page 14-10

15

Developer Tools

Developer Tools

Oracle Applications provides several tools to help developers create and debug custom forms and programs. Described in this chapter are the Forms Personalization and Work Directory features.

In addition, the CUSTOM library allows extension of Oracle Applications without modification of Oracle Applications code. For information on the CUSTOM library, see the *Oracle Applications Developer's Guide*.

Also, the *Oracle Applications Supportability Guide* provides information for system administrators and developers regarding diagnostic and logging features.

Forms Personalization

The Form Personalization feature allows you to declaratively alter the behavior of Forms-based screens, including changing properties, executing builtins, displaying messages, and adding menu entries.

For each function (a form running in a particular context based on parameters passed to it), you can specify one or more Rules. Each Rule consists of an Event, an optional Condition, the Scope for which it applies, and one or more Actions to perform.

An Event is a trigger point within a form, such as startup (WHEN-NEW-FORM-INSTANCE), or when focus moves to a new record (WHEN-NEW-RECORD-INSTANCE). There are standard events that almost every form sends, and certain forms send additional product-specific events.

The Scope is evaluated based on the current runtime context to determine if a Rule should be processed or not. The Scope can be at the Site, Responsibility, User, or Industry level. Each Rule can have one or more Scopes associated with it.

The Condition is an optional SQL code fragment that is evaluated when the Event occurs; if it evaluates to TRUE then the Actions are processed.

Each Action consists of one of the following:

- setting a Property, such as making a field Required or hiding a Tab page
- executing a Builtin, such as GO_BLOCK, DO_KEY or EXECUTE_FUNCTION
- displaying a Message
- enabling a Special menu entry

Once Rules are defined, when the target function is run then the Rules are automatically applied as events occur within that form.

Although the Form Personalization feature is declarative, the intended audience is a person familiar with Oracle Forms including the PL/SQL programming language, and the *Oracle Applications Developer's Guide*. Additionally, any change made could interfere with the base code of a form (the code that Oracle ships).

Please refer to "Form Personalizations in Oracle Applications", Note 395117.1, on Oracle *MetaLink* for more information.

Work Directory

The Work Directory feature enables a developer, support consultant, or other technical specialist to test modifications to forms and concurrent programs in Oracle Applications without affecting users of the same code tree.

Using the Work Directory, a user can be logged into an Oracle Applications system but access a version of a form or concurrent program that is not within the standard $PROD_TOP directory. For example, an on-site developer can test out a new version of a custom form without affecting other testing on the system.

You can use the Work Directory feature for alternate files of forms and concurrent programs only.

Implementation

To implement this feature, set up a directory to hold the alternate files for your forms or concurrent programs.

To use an alternate file, set the profile option FND:Override Directory with the path for the directory containing the alternate file.

> **Important:** This profile option should usually be set at the User level only. If you set FND:Override Directory at the Site level, for example, you will affect all users at that site using the particular forms.

Using the Work Directory

After you have created the alternate directory and set the profile option FND: Override Directory with the appropriate value, you can use files in that alternate directory.

In searching for the appropriate file path for a form or concurrent program, Oracle Applications will first check to see if the profile option FND: Override Directory is set and if a given file exists in the specified directory. If the above two conditions are true then the alternate file is used. If the profile option is not set or if the necessary file does not exist in that directory, then the default (usual) file path is used.

> **Note:** The Oracle Applications Navigator caches the paths to files that have been successfully opened. If the standard form has been opened, then that form will be used for the remainder of the session. To switch to a different file path, you must exit and restart Oracle Applications.

To provide a visual indication that an alternate form is in use, the developer of the form should specify a different version number for the form in the PRE-FORM trigger. This version number appears during runtime using Help > About Oracle Applications. For more information, see the *Oracle Applications Developer's Guide.*

Web Enabled PL/SQL Window

Use this form to maintain the FND_ENABLED_PLSQL runtime registry.

PL/SQL Object Block

Name

The name of the PL/SQL object.

Type

The type of the PL/SQL object. The object may be a package, package procedure, or procedure.

Enabled

Check this box to enable the PL/SQL object., and the Logging Service

16

Administering Process Navigation

Overview of Process Navigation

A "process" is a series of actions taken to achieve a specific result. The Process Navigator utilizes Oracle Workflow to depict each of your business processes with a workflow diagram. A process diagram contains an icon for each step in the process; each icon acts as a visual cue and as an access point for the actual form associated with each step. You can navigate to any form involved in the process simply by clicking on the appropriate icon.

What is Oracle Workflow?

Oracle Workflow allows you to define business processes using a drag-and-drop designer. You can route relevant information to decision makers, automate processes, deliver electronic notifications to users in a given workflow, and monitor your processes as they are implemented. You can display any workflow diagram as a process in the Process Navigator. For more information, see the Oracle Workflow documentation.

What are Seeded Processes?

A seeded process is one that is delivered to you ready to use. Oracle Applications includes several seeded business processes which you can use as they are.

Modifying Your Menu

Before you begin, you should be aware that simply referencing a form from a process does not provide the required permissions for the responsibility to access the forms in the process. Form Functions for each form referenced from a process must be added to the Function Security Menu for the responsibility. If the Form Function is not accessible, the user will receive an error when attempting to access the form from the process in the Process Navigator.

Creating Process Navigator Processes

You must use Oracle Workflow Builder to create or customize any of the processes that are displayed in the Process Navigator. These instructions describe how to create new processes for the Process Navigator.

The following table lists the terms/components of a Process Navigator process and the corresponding components in Oracle Workflow Builder that define them.

Process Navigator Component	Description	Controlling Oracle Workflow Builder Component(s)
Process	The diagram that appears in the Process Navigator.	Process activity and process diagram
Process description	A description of the displayed process.	Process activity
Step	An icon in the process, which takes you directly to an Oracle Applications form when you double-click on it.	Notification activity
Step description	A description of the selected process step.	Message
Form associated with a step.	The Oracle Applications form that appears when you double click on a step in a Process Navigator process.	Form-type Message attribute

Note: The following procedures do not address most of the functionality of Oracle Workflow Builder, but are tailored to creating processes for the Process Navigator. The Oracle Workflow Builder is a tool used to design workflow processes. Workflow processes can range from routing documents through an approval process to setting up your Oracle Applications. See the Oracle Workflow documentation for more information.

Creating Process Navigator Processes

To create a new process for the Process Navigator, you must first create the necessary components in Oracle Workflow Builder. The components you create make up the

process definition, which is then saved to the database or to a flat file. The Process Navigator then reads the process definition from the database to display the process and its information and provide you access to the related Oracle Applications forms.

Creating a New Process Navigator Process

Note: For more information on creating a process, see the *Oracle Workflow Developer's Guide*.

1. Open Oracle Workflow Builder.
2. Create an item type. An item type is a repository that will contain all the components associated with the process you wish to build.
3. Create an Item Attribute of type role, whose internal name is USER_NAME.

   Note: Enter a new display name for the message using the format <Verb><Form Title>. If the form title already contains a verb, then simply use the form title as the display name. If the form title does not contain a verb, then consider using one of the following verbs:

   ```
   Define / Assign / Run / Load / Convert / Open / Set /

   Generate / Review
   ```

4. Create a message to describe the task that is to be accomplished by a Process Navigator process step.
5. Create a form-type for the message. The seeded processes generally assign these message attributes an internal name of Open Form, but this is not required.
6. Create a notification activity to represent a Process Navigator process step.
7. Create a process activity to represent a Process Navigator Process.

   Note: Enter a display name for your process. This name appears in the Process Navigator's process list. The naming convention for the process should be a functional name followed by the word "Process."

   Enter a description for your process. The description appears when a user selects a process in the Process Navigator, For Oracle Workflow Builder Release 2.5 and higher, the description is limited to 240 characters.

8. Draw the Process Diagram. Once you create a process activity, you can draw the

process diagram that is associated with it. The process diagram is what appears when you display a process in the Process Navigator.

> **Note:** The Performer type of the Notification Activity you include in a process diagram for the Process Navigator must be set to the item attribute USER_NAME.

9. Save your changes. When you save your work to a database, you actually save everything in the current data store that has been modified. When you save your work to a flat file, you actually save everything in the current data store to the file.

   > **Note:** It is highly recommended that for new processes created for the Process Navigator that you always save a copy of your workflow process definition as a flat file and check that file into a source control system to maintain a working version of your process definition. Then when you want to update your definition in the database, you can pull up the flat file and save it directly to the database. Avoid using the process definition stored in your database as your source controlled version, as others with access to the database can update the definition.

10. Enable access to your process.

Enable access to your process

Before a process may be accessed in the Navigator you must complete the following two steps. Create a new function for your process in the Form Functions window, and add your process to a responsibility by adding the function you just created, to the responsibilities top menu in the Menus window.

Create a function for your process

Use the following procedure to create a function:

1. As the System Administrator navigate to the Form Functions window (Application->Function).

2. Enter a Function Name for your process using the format:

   `<app>_<processname>`

   Where <app> can be any application short name and <processname> is the internal name you entered when you created your process activity.

3. Enter a User Function Name. The name you enter here appears in the Navigator.

4. Enter "PROCESS" as your function type.

5. In the tabbed region 'Form' use the following format to enter a value in the Parameters field:

   ```
   <itemtype>:<processname>
   ```

6. Save your work. No other fields are required to create your process function.

Add your function to a menu

In order for a user to access a process in the Navigator, the process must be added to a menu referenced by the user's responsibility. To determine the menu referenced by a particular responsibility use the Responsibilities window (Security->Responsibility->Define).

1. As the System Administrator navigate to the Menus window (Application->Menu).

2. Use the Find window to access the desired menu.

3. In a new row use the LOV to select the function you created for your process in the Functions field. You may optionally enter a description for the function. DO NOT enter any other fields. The Sequence field is automatically populated and the Navigator Prompt and Submenu fields must remain empty.

4. Save your work.

Access the Seeded Processes from the Database

To access the seeded processes, use the following procedure:

1. Run the Oracle Workflow Builder from you client.

2. Select Open from the File menu.

3. Choose Database.

4. For User, enter the FNDNAM of your database

5. For Password, enter the FNDNAM password of your database

6. For Connect, enter the alias for your database which should be entered in your tnsnames.ora file under the following directory on your client:

   ```
   Local drive (i.e. "C"):orant\network\admin
   ```

   Note: If you are using Windows 95, then the "orant" should be replaced with "orawin" in the directory structure above.

7. In the Show Item Types window, select the item type(s) associated with the seeded

processes you wish to view. To select more than one item type, hold down your control key as you select the item types. Choose Show, and then choose OK.

Find the Form Function Name

Use the following procedure to find the form function name:

1. Log into Oracle Applications and navigate to the form of interest.
2. Choose About Oracle Applications... from the Help menu. Scroll down to Form Information and make note of the form name.
3. Now log into Oracle Applications using the Implementation System Administration responsibility and navigate to /Application/Form. Within the Form window, query for the form name you just made a note of in the Form field.
4. Make note of the value in the User Form Name field once your query completes.
5. Close the Form window and navigate to /Application/Function. Within the Function window, query for the User Form Name value that you just made a note of in the Form field.
6. The value that is returned in the Function field is the form function name that you need to associate a Process Navigator process step to a form.

17

Administering Globalization

Overview of Globalization Support

This chapter describes some of the features of globalization (formerly internationalization) support in Oracle Applications. Topics covered in this chapter include language values used in user sessions, behavior of month name abbreviations, and multilingual external documents. In addition, the Oracle Application windows for Languages, Natural Languages, and Territories are described.

For additional information on globalization concepts and features in Oracle Applications, see Oracle*MetaLink* Notes 393861.1, "Oracle Applications Release 12 Globalization Guide", and 393320.1, "Oracle Applications Release 12 Internationalization Update Notes". Also see *Oracle Applications Concepts Guide*.

Language Values for User Sessions

Language Values for User Sessions using AppsLocalLogin.jsp

This section describes how the language is determined for a user's session and other interactions with the system.

Login Page Language

If the language is a licensed Oracle Applications language, the initial login page language is determined from one of the following sources, in the order shown:

- A language value passed on the AppsLocalLogin.jsp URL
- The language used in an earlier login session
- The browser language

If the language is not a licensed Oracle Applications language, the initial login page is in

the Oracle Applications base language.

> **Note:** The default for Single Sign-On (SSO) SSO logins will fall back to browser language preferences, in order. The default for non-SSO logins will fall back to the base language if the first language preference is not available.

The user can change the login page language selection by choosing one of the other language name icons on the login page. The language name icons on the login page represent the active languages in the Applications installation. If the user selects an alternative language, the login page is refreshed in that alternative language. Logging in and logging back out will display a URL with the correct language value. The URL can be bookmarked.

> **Important:** In Oracle E-Business Suite Release 11.5.10, the language name icons could be hidden by setting a Local Login Mask profile option. In Release 12, this profile option is no longer available. To hide the language name icons, use the OA Framework Personalization feature.

Login Page Language and Runtime Session Language

Users must ensure that the language used for the login page is the language they want to use for their Applications runtime session. The runtime session will start in the language used in the login page. After logging in, users can, if they wish, change the runtime session language from the Preferences screen. By default, the ICX:Language profile value is not used to determine the runtime session language when logging in through the login page. However, this behavior can be modified by setting the "Applications Override SSO Server Language" profile option.

ICX:Language Profile Option

The ICX:Language profile option is used for the following:

- Default language to use for offline with other users on the system. For example, the language to use for an e-mail notification to be sent to a user on the system, and which may be read offline from the system.
- Default language to use for other Oracle Applications login methods besides the login page. For example, the language to use for the session language when logged in through JDeveloper.
- Session language when the "Applications Override SSO Server Language" profile option is set to "Override SSO Server Language".

The ICX:Language profile is normally not set at user level when a given user initially

starts using Oracle Applications. Regardless of whether a value exists for the profile at user level, the language chosen on the login page (the runtime session value) is not automatically saved to the ICX:Language profile value for the user. The user can either specifically set a value for the ICX:Language profile at user level after logging in, or continue to inherit the value from the higher profile value levels. A value always exists for the ICX:Language profile at site level.

Preferences Page Language Support

The current values for the runtime session language and the saved ICX:Language profile value for the user are displayed on the Preferences page.

The following updates are allowed:

- Change only the runtime session language
- Change only the saved ICX:Language profile value for the user
- Change both the runtime session and the ICX:Language profile values to either the same values or different values

Language Value from Login External to Oracle Applications

This section describes language value behavior when logging into Oracle Applications from non-Oracle Applications based login systems.

For Oracle Applications, the language used in such a non-Applications system cannot be relied on to determine the runtime session language for the Oracle Applications session. That language may not be an Oracle Applications-supported active language, and it may not be the preferred language of the user for the runtime Oracle Applications session. Therefore, the ICX:Language profile value for the user determines the Oracle Applications runtime session for that user.

Once the Applications runtime session is initiated, the user can use the Preferences page to change the runtime session language for the session. The user can also choose to change his saved ICX:Language profile value.

For an Oracle Applications environment integrated with Oracle Portal, the Applications session will inherit the portal language if the session is launched from Oracle Portal. If the portal language is not supported or enabled in Oracle Applications, the Applications session language will fall back to the Oracle Applications base language.

Language Values for Oracle Workflow Notifications

Oracle Workflow email notifications use the user-level language set in the Preferences page and also stored in the ICX:Language profile option.

Date Formats in NLS Implementations

The storage space for month name abbreviations is limited to 3 bytes. For some language and character set combinations, the month name abbreviation requires more than 3 bytes. For this reason, when a language is used in which any month name abbreviation exceeds 3 bytes in the current character set, all month name abbreviations for that language are automatically replaced by numeric representations that conform to the 3-byte space limit.

The languages affected in a UTF8 implementation are: Canadian French, Croatian, Czech, German, Greek, Hungarian, Polish, Russian, Simplified Chinese, Traditional Chinese, Slovak, Thai, and Turkish.

The languages affected in their local character sets are: Arabic, Simplified Chinese, Traditional Chinese, and Thai.

Multilingual External Documents

Oracle Applications ships with a set of external documents, or those documents directed toward your customers and trading partners, for which we model the data multilingually. Any document for which the data model is multilingual can be submitted, through a single request, to run in one language or in any subset of the installed languages. Your Italian customer, for example, can receive invoices printed in Italian, while your Korean customer receives packing slips printed in Korean.

Oracle Shipping

- Bill of Lading
- Commercial Invoice
- Pack Slip

Oracle Order Management

- Price List
- Sales Order Acknowledgment

Oracle Receivables

- Dunning Letter Print
- Print Statements

- Transaction Print

Oracle Purchasing

- Printed Change Order Report (Landscape)
- Printed Change Order Report (Portrait)
- Printed Purchase Order Report (Landscape)
- Printed Purchase Order Report (Portrait)
- Printed RFQ Report (Landscape)
- Printed RFQ Report (Portrait)
- Dispatch Purchase Order

Oracle Sourcing

- Negotiation PDF

Oracle Payables

- Invalid PO Supplier Notice
- Prepayment Remittance Notice
- Print Invoice Notice
- Supplier Open Balance Letter

Oracle Human Capital Management (Human Resources)

- Full Person Details
- Full Applicant Details
- Full Assignment Details
- Full Work Details

Oracle Payroll

- Check Writer
- Deposit Advice
- Third Party Checks

Translations Window

In windows with an enabled translation icon, users can click on the icon to bring up a Translations window to enter or update translated values for specific records in the database. See: Creating Translations for a Record, *Oracle Applications User's Guide*.

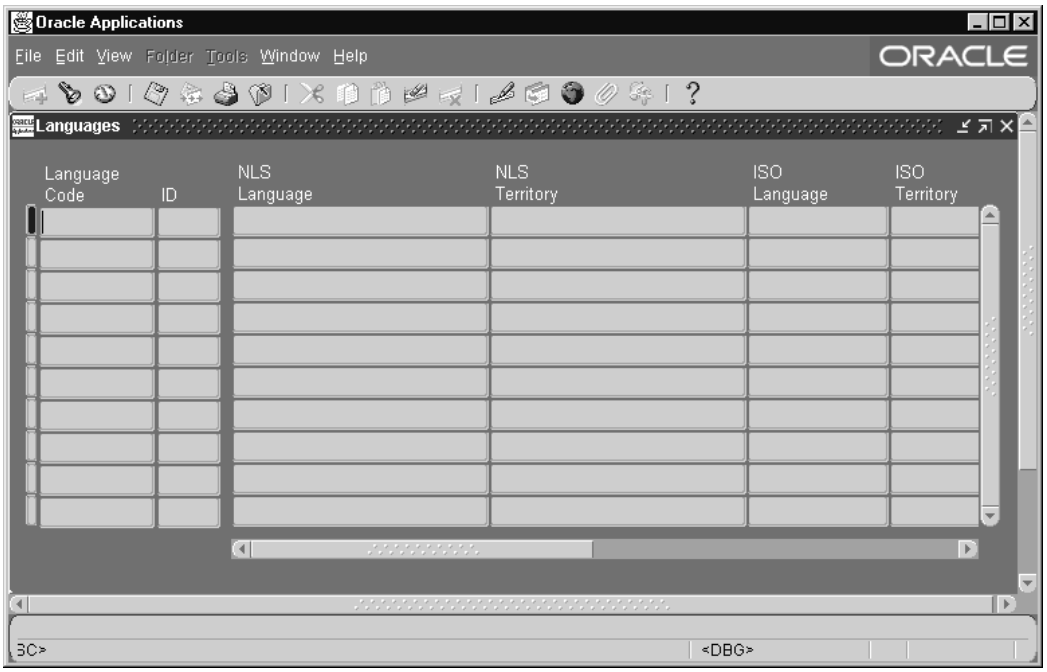
Users can enter translations for multiple languages at once in the Translations window. Translated values should be entered using the Translations window.

Users should be aware of which fields are allowed to be translated. Users can click on the translation icon to identify if a field is translatable. If a user types into a field that was not meant to be translated, the value in the source language will be overwritten, and the translated value will be the only entry for that field. All users will see the data in that language, and the value in the original language will be lost.

Currencies Window

For information on how to define currencies using this window, see: Defining Currencies, *Oracle General Ledger User's Guide*.

Languages Window



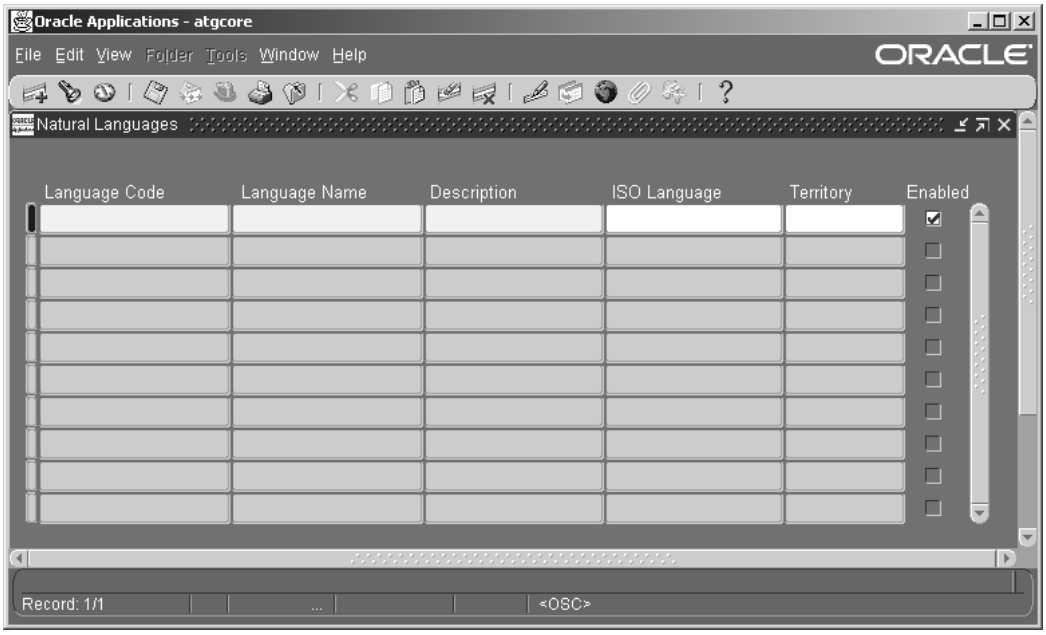
Use the Languages window to review information about the languages available for use in Oracle Applications.

Languages Record

Each record includes the Language Code, ID, NLS Language, NLS Territory, ISO Language, ISO Territory, Default Code Set, Installed flag, Local Date Language, UTF8 Date Language, and Description.

The columns Local Date Language and UTF8 Date Language are used in date rendering. For some languages, the length in bytes of the month short name is longer than the prepared buffer in a UTF8 database; for these languages, the UTF8 Date Language is used as the date language in UTF8 environments. In databases using other character sets, the Local Date Language is used.

Natural Languages Window



The Natural Languages window allows you to review information about the languages available for use in Oracle Applications.

The Natural Languages table allows you to associate the name of a language with another entity, such as a person or company.

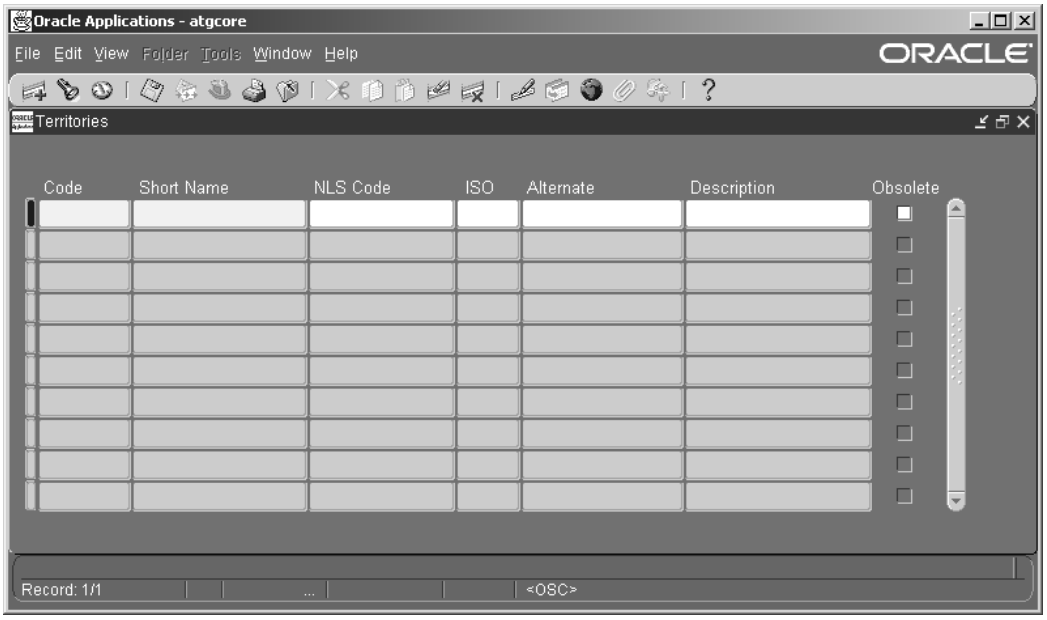
- You can add new languages in this window. Prefix the language code for new languages with an 'x'.
- You can update the enabled status of seeded languages.
- You cannot delete saved records.

Natural Languages Record

Each record includes the Language Code, Language Name, ISO Language, and Territory for each language.

You can use Oracle Applications Manager (OAM) License Manager to enable a licensed language..

Territories Window



Use the Territories window to review and modify information for the country values used in Oracle Applications.

Territories Block

Each record includes the two-letter uppercase territory Code such as "US", the Short Name for the territory such as "United States", the NLS Code, the ISO numeric entity code, an Alternate territory code, and a longer description (Description), such as "United States of America".

Normally you would not want to update the seeded data that comes with your products, but you may wish to modify the way the country is represented in List of Values through out your applications.

You can update the description of the Territory to change the territory value displayed in List of Values used in Oracle Application products.

A

Oracle Application Server with Oracle E-Business Suite

Introduction

This appendix lists documents that describe the use of Oracle Application Server with an Oracle E-Business Suite Release 12 environment.

Using Oracle Application Server 10*g* with Oracle E-Business Suite

Oracle E-Business Suite Release 12 utilizes Oracle Application Server 10*g* as a core part of the technology stack. In addition, you can optionally deploy Oracle Discoverer, Oracle Internet Directory, Oracle Portal, and Oracle Single Sign-On with E-Business Suite Release 12.

Benefits of utilizing optional Application Server components include:

- Performance, scalability and high-availability via distributed architectures
- The ability to connect a single Enterprise Portal to web providers running on multiple Release 12 instances
- Uniform Single Sign-On support for all Release 12 applications
- Bidirectional Oracle Internet Directory-to-FND_USER synchronization, with selective cardinality of synchronization for individual user attributes
- "Link-on-the-fly" support - for environments where the Single Sign-On user IDs in Oracle Internet Directory differ from the Release 12 user IDs
- "One-to-many" support - for environments where a Single Sign-On user ID may be associated with one or more Release 12 user IDs

- Integration with third-party single sign-on services (e.g. Netegrity, Entrust) and LDAP V.3 directories (e.g. SunONE/iPlanet, Microsoft Active Directory)

- Integration with third-party and legacy applications via Oracle Integration

A detailed description of Oracle Single Sign-On integration options with Oracle E-Business Suite Release 12 can be found in the *Oracle System Administrator's Guide - Security*.

For information on using OracleAS 10*g* with Oracle E-Business Suite, refer to the following documents on Oracle*MetaLink*:

- Note 376811.1 - Using Oracle Application Server 10g with Oracle E-Business Suite Release 12

- Note 380483.1 - Integrating Oracle E-Business Suite Release 12 with Oracle Internet Directory and Oracle Single Sign-On

- Note 380484.1 - Using Oracle Portal 10g with Oracle E-Business Suite Release 12

- Note 380486.1 - Installing and Configuring Web Cache 10g and Oracle E-Business Suite 12

- Note 373634.1 - Using Discoverer 10.1.2 with Oracle E-Business Suite Release 12

B

Loaders

Generic Loader

The Generic Loader (FNDLOAD) is a concurrent program that can move Oracle Applications data between database and text file representations. The loader reads a configuration file to determine what data to access. For information on specific configuration files consult the *Open Interfaces Guide* for your product group. The following sections describe the operation of the Generic Loader.

> **Warning:** Use only the loader files provided by Oracle Applications. If you use files not provided by Oracle Applications or modify the provided scripts you risk corrupting your database. Oracle does not support the use of custom loader files or modified Oracle Applications loader files.

Overview

The Generic Loader can download data from an application entity into a portable, editable text file. This file can then be uploaded into any other database to copy the data. Conversion between database store and file format is specified by a configuration file that is read by the loader.

The Generic Loader downloads data from a database according to a configuration (.lct) file, and converts the data into a data file (.ldt file). The Generic Loader can then upload this data to another database using a configuration file.

The loader operates in one of two modes: download or upload. In the download mode, data is downloaded from the database to a text file; in the upload mode, data is uploaded from a text file to the database.

Data structures supported by the loader include master-detail relationships and foreign key reference relationships.

In both downloading and uploading, the structure of the data involved is described by

a configuration file. The configuration file describes the structure of the data and also the access methods to use to copy the data into or out of the database. The same configuration file may be used for both uploading and downloading.

When downloading, the Generic Loader creates a second file, called the data file, that contains the structured data selected for downloading. The data file has a standard syntax for representing the data that has been downloaded. When uploading, the Generic Loader reads a data file to get the data that it is to upload. In most cases, the data file was produced by a previous download, but may have come from another source. The data file cannot be interpreted without the corresponding configuration file available.

Download database information to a text file

The text file is human-readable and portable, and can be examined and modified with any editor. Generally, a "developer key" is used to identify records written out to text files. For example, the PROFILE_OPTION_NAME, not the PROFILE_OPTION_ID, is used to identify records in the Profiles configuration file.

Upload (merge) the information in a text file to the database

In uploading, if a row exists, but has different attributes, the row is updated. If a row does not exist, a new row is inserted.

Depending on the configuration file, a row that exists in the database but not in the text file may or may not be deleted when the text file is uploaded. Refer to the configuration file to determine how such rows are handled.

These download and upload capabilities allow profile value information that is defined in one database to be easily propagated to other databases. This is useful for delivering Oracle Applications seed data to customers, as well as for copying customer profile definitions from a primary site to other sites.

The text file version of profile value data is also useful for bulk editing operations, which can be accomplished more efficiently with a text editor than with a form.

Preservation of data

FNDLOAD uses the OWNER and LAST_UPDATE_DATE attributes to determine whether to overwrite pre-existing data. The rules applied are:

1. If the entity being uploaded is not present in the database, a new entity is always inserted.

2. Entities uploaded from a file with OWNER=SEED never overwrite entities with OWNER=CUSTOM in the database.

3. Entities with OWNER=CUSTOM uploaded from a file always update entities with OWNER=SEED in the database.

4. If the owner of the entity is the same in the file and database, the entity is updated only if the LAST_UPDATE_DATE in the file is later than the LAST_UPDATE_DATE in the database.

FNDLOAD Executable

The Generic Loader is a concurrent program named FNDLOAD. The concurrent executable takes the following parameters:

```
FNDLOAD apps/pwd 0 Y mode configfile datafile entity [ param ... ]
```

where

<apps/pwd>	The APPS schema and password in the form username/password[@connect_string]. If connect_string is omitted, it is taken in a platform-specific manner from the environment using the name TWO_TASK.
< 0 Y >	Concurrent program flags
mode	UPLOAD or DOWNLOAD. UPLOAD causes the datafile to be uploaded to the database. DOWNLOAD causes the loader to fetch rows and write them to the datafile.
<configfile>	The configuration file to use (usually with a suffix of .lct, but not enforced or supplied by the loader).
<datafile>	The data file to write (usually with a suffix of .ldt, but not enforced or supplied by the loader). If the data file already exists, it will be overwritten.
<entity>	The entity(ies) to upload or download. When uploading, you should always upload all entities, so specify a "-" to upload all entities.
< [param] >	Zero or more additional parameters are used to provide bind values in the access SQL (both UPLOAD and DOWNLOAD). Each parameter is in the form NAME=VALUE. NAME should not conflict with an attribute name for the entities being loaded.

File Specifications

The configuration file and data file parameters are specified in one of two ways:

```
@<application_short_name>:[<dir>/.../]file.ext
```

For example,

```
@fnd/120/loader/fndapp.lct
@po:install/data/poreq.ldt
```

Alternatively, the parameters can be specified as such:

```
<native path>
```

For example,

```
mydata.ldt
c:\loader\config\cfg102.lct
```

Examples

An example of downloading is:

```
FNDLOAD apps/apps@devdb 0 Y
     DOWNLOAD testcfg.lct out.ldt FND_APPLICATION_TL APPSNAME=FND
```

This command does the following:

- connects to apps/apps@devd
- downloads data using the configuration file testcfg.lct
- writes data to data file out.ldt
- downloads the FND_APPLICATION_TL entity with APPSNAME parameter defined as value 'FND'

An example of uploading is:

```
FNDLOAD apps/apps@custdb 0 Y
     UPLOAD fndapp.lct fnd1234.ldt -
```

This command does the following:

- connects to apps/apps@custdb
- uploads data using the configuration file in fndapp.lct from data file in fnd1234.ldt
- The contents of the entire data file is uploaded.

Configuration File

Operation of the Generic Loader is controlled by the specified configuration file. The configuration file contains the following:

- DEFINE block
- DOWNLOAD block
- UPLOAD block

The contents of the configuration file specify the structure of the data and the access methods to use to move the data between the data file and a database.

DEFINE Block

The DEFINE block specifies the structure of the datafile records. The define block format is identical to that already generated by existing Oracle Application Object Library loaders. The structure of this section is

```
DEFINE <entity> KEY <key_attribute_name> <datatype> ...
 (BASE|TRANS|CTX) <attribute_name> <datatype> ...
 [DEFINE <child_entity> ...]
END <entity>
```

Example

```
DEFINE MENU
  KEY MENU_ID           NUMBER
  TRANS USER_MENU_NAME  VARCHAR2(80)
  TRANS DESCRIPTION     CLOB
  DEFINE ENTRY
    BASE  SUBMENU               REFERENCES MENU
    TRANS USER_SUB_MENU_NAME    VARCHAR2(60)
    BASE  DESCRIPTION_DOCUMENT  CLOB
  END ENTRY
END MENU
```

One or more KEY attributes defines the primary key of each entity. BASE and CTX attributes are those that do not require translation. TRANS attributes do. Note that BASE and CTX attributes are treated identically. That is, CTX is just a synonym for BASE. The CTX attribute type is provided merely to allow users to optionally differentiate between BASE attributes. For example, translators may wish to simplify their .ldt files by stripping out the BASE attributes. However, they may also want to keep some BASE attributes for context. By denoting some attributes as BASE and some as CTX, they can control which attributes to remove.

Data types can be standard Oracle scalar types, except that only VARCHAR2(size), NUMBER, and CLOB are currently supported. An attribute can also be defined as a foreign key reference to another entity in your configuration file. The foreign key entity must be a "top-level" entity and its download statement must include filter parameters in its WHERE clause for each of its key attributes. Also, the parameter names must match the key attribute names exactly.

Note that entity definitions can be nested to indicate master-detail relationships. Nested entity definitions inherit the key attributes of their parent entities and should not redefine them.

DOWNLOAD Statement

The DOWNLOAD statement is a SQL statement that selects rows to download. The statement can join to other tables to resolve sequence generated ID numbers into developer keys where possible. The DOWNLOAD statement may also contain bind values of the form ':NAME' which are substituted with parameter values from the command line. DOWNLOAD statements have the form

```
DOWNLOAD <entity> "select <attribute expressions> from ..."
```

Example

```
DOWNLOAD FND_LOOKUP_TYPE
  "select VA.APPLICATION_SHORT_NAME VIEW_APPSNAME,
   LT.LOOKUP_TYPE,
   OA.APPLICATION_SHORT_NAME,
   LT.CUSTOMIZATION_LEVEL,
   decode(LT.LAST_UPDATED_BY, 1, 'SEED', 'CUSTOM') OWNER,
   LT.MEANING,
   LT.DESCRIPTION
   from   FND_LOOKUP_TYPES_VL LT,
   FND_APPLICATION VA,
   FND_APPLICATION OA,
   FND_SECURITY_GROUPS SG
   where  VA.APPLICATION_ID = LT.VIEW_APPLICATION_ID
   and    OA.APPLICATION_ID = LT.APPLICATION_ID
   and    (:VIEW_APPSNAME is null or
     (:VIEW_APPSNAME is not null
   and VA.APPLICATION_SHORT_NAME like :VIEW_APPSNAME))
   and    (:LOOKUP_TYPE is null or
      (LOOKUP_TYPE is not null and LT.LOOKUP_TYPE like :LOOKUP_TYPE))
   and    SG.SECURITY_GROUP_ID = LT.SECURITY_GROUP_ID
   and    ((:SECURITY_GROUP is null and SG.SECURITY_GROUP_KEY =
'STANDARD') or
     (:SECURITY_GROUP is not null
   and SG.SECURITY_GROUP_KEY = :SECURITY_GROUP))
   order by 1, 2 "
```

Download statements for child entities may reference any key attribute of the parent entity, or any command line parameter.

UPLOAD Statement

The UPLOAD statement is a SQL statement or PL/SQL anonymous block which accepts file data and applies it to the database. The statement is executed once for each record read from the data file. Bind values in the statement are satisfied by attributes from the file data or command line parameters.

Example

```
UPLOAD FND_LOOKUP_TYPE
 BEGIN
 " begin
   if (:UPLOAD_MODE = 'NLS') then
     fnd_lookup_types_pkg.TRANSLATE_ROW(
  x_lookup_type  => :LOOKUP_TYPE,
  x_security_group      => :SECURITY_GROUP,
  x_view_application    => :VIEW_APPSNAME,
  x_owner        => :OWNER,
  x_meaning      => :MEANING,
  x_description  => :DESCRIPTION);
   else
  fnd_lookup_types_pkg.LOAD_ROW(
  x_lookup_type  => :LOOKUP_TYPE,
  x_security_group      => :SECURITY_GROUP,
  x_view_application    => :VIEW_APPSNAME,
  x_owner        => :OWNER,
  x_meaning      => :MEANING,
  x_description  => :DESCRIPTION);
   end if;
   end; "
```

As in the DOWNLOAD, the UPLOAD statement for child entities may reference any attributes from the parent record.

Example

```
UPLOAD FND_LOOKUP_VALUE
  " begin
 if (:UPLOAD_MODE = 'NLS') then
   fnd_lookup_values_pkg.TRANSLATE_ROW(
     x_lookup_type  => :LOOKUP_TYPE,
     x_lookup_code  => :LOOKUP_CODE,
     x_security_group      => :SECURITY_GROUP,
     x_view_application    => :VIEW_APPSNAME,
     x_owner        => :OWNER,
     x_meaning      => :MEANING,
     x_description  => :DESCRIPTION);
 else
   fnd_lookup_values_pkg.LOAD_ROW(
     x_lookup_type  => :LOOKUP_TYPE,
     x_lookup_code  => :LOOKUP_CODE,
     x_security_group      => :SECURITY_GROUP,
     x_view_application    => :VIEW_APPSNAME,
     x_owner        => :OWNER,
     x_meaning      => :MEANING,
     x_description  => :DESCRIPTION,
     x_tag   => :TAG);
 end if;
  end;"
```

Data File

A data file is a portable text file. The data file created from a download using the above configuration file would include:

```
# -- Begin Entity Definitions --

DEFINE FND_LOOKUP_TYPE
   KEY   VIEW_APPSNAME VARCHAR2(50)
   KEY   LOOKUP_TYPE VARCHAR2(30)
   BASE  OWNER VARCHAR2(6)
   TRANS MEANING VARCHAR2(80)
   TRANS DESCRIPTION VARCHAR2(240)
   DEFINE FND_LOOKUP_VALUE
     KEY   LOOKUP_CODE VARCHAR2(30)
     BASE  END_DATE_ACTIVE VARCHAR2(10)
     BASE  OWNER VARCHAR2(6)
     TRANS MEANING VARCHAR2(80)
     TRANS DESCRIPTION VARCHAR2(240)
     BASE  TAG VARCHAR2(30)
   END FND_LOOKUP_VALUE
 END FND_LOOKUP_TYPE

# -- End Entity Definitions --

BEGIN FND_LOOKUP_TYPE "FND" "YES_NO"
   OWNER = "SEED"
   MEANING = "Yes or No"

   BEGIN FND_LOOKUP_VALUE Y
     OWNER = "SEED"
     MEANING = "Yes"
   END FND_LOOKUP_VALUE

  BEGIN FND_LOOKUP_VALUE N
     OWNER = "SEED"
     MEANING = "No"
   END FND_LOOKUP_VALUE
END FND_LOOKUP_TYPE
```

Oracle Application Object Library Configuration Files

Oracle Application Object Library provides several configuration files for the Generic Loader that you can use with your setup data.

These configuration files operate on the following data:

- Concurrent program definitions
- Request groups
- Lookup types and lookup values
- Profile options and profile option values
- Flexfields setup data
- Attachments definitions
- Messages

- Security information

Attachments Setup Data Configuration File

Use the file afattach.lct for loading attachments setup data.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
FND_ATTACHMENT_ FUNCTIONS	FND_ATTACHMENT_BLOCKS FND_ATTACHMENT_BLOCK_ ENTITIES FND_DOC_CATEGORY_USAGES	APPLICATION_ SHORT_NAME FUNCTION_NAME FUNCTION_TYPE

The entity definitions are:

```
DEFINE FND_ATTACHMENT_FUNCTIONS
  KEY   FUNCTION_NAME    VARCHAR2(30)
  KEY   FUNCTION_TYPE    VARCHAR2(1)
  KEY   APP_SHORT_NAME    VARCHAR2(50)
  CTX   OWNER       VARCHAR2(7)
  BASE  SESSION_CONTEXT_FIELD  VARCHAR2(61)
  BASE  ENABLED_FLAG      VARCHAR2(1)

  DEFINE FND_ATTACHMENT_BLOCKS
    KEY   BLOCK_NAME    VARCHAR2(30)
    CTX   OWNER     VARCHAR2(7)
    BASE  QUERY_FLAG    VARCHAR2(1)
    BASE  SECURITY_TYPE   VARCHAR2(50)
    BASE  ORG_CONTEXT_FIELD VARCHAR2(61)
    BASE  SET_OF_BOOKS_CONTEXT_FIELD  VARCHAR2(61)
    BASE  BUSINESS_UNIT_CONTEXT_FIELD  VARCHAR2(61)
    BASE  CONTEXT1_FIELD   VARCHAR2(61)
    BASE  CONTEXT2_FIELD   VARCHAR2(61)
    BASE  CONTEXT3_FIELD   VARCHAR2(61)

    DEFINE FND_ATTACHMENT_BLK_ENTITIES
  KEY   BLK_ENTITY  REFERENCES
FND_DOCUMENT_ENTITIES
  BASE  DISPLAY_METHOD     VARCHAR2(1)
  BASE  INCLUDE_IN_INDICATOR_FLAG  VARCHAR2(1)
  CTX   OWNER      VARCHAR2(7)
  BASE  PK1_FIELD     VARCHAR2(61)
  BASE  PK2_FIELD     VARCHAR2(61)
  BASE  PK3_FIELD     VARCHAR2(61)
  BASE  PK4_FIELD     VARCHAR2(61)
  BASE  PK5_FIELD     VARCHAR2(61)
  BASE  SQL_STATEMENT   VARCHAR2(2000)
  BASE  INDICATOR_IN_VIEW_FLAG  VARCHAR2(1)
  BASE  QUERY_PERMISSION_TYPE   VARCHAR2(1)
  BASE  INSERT_PERMISSION_TYPE  VARCHAR2(1)
  BASE  UPDATE_PERMISSION_TYPE  VARCHAR2(1)
  BASE  DELETE_PERMISSION_TYPE  VARCHAR2(1)
  BASE  CONDITION_FIELD    VARCHAR2(61)
        BASE  CONDITION_OPERATOR VARCHAR2(50)
  BASE  CONDITION_VALUE1   VARCHAR2(100)
  BASE  CONDITION_VALUE2   VARCHAR2(100)
    END FND_ATTACHMENT_BLK_ENTITIES
  END FND_ATTACHMENT_BLOCKS

  DEFINE FND_DOC_CATEGORY_USAGES
    KEY CATEGORY_USAGE REFERENCES
    FND_DOCUMENT_CATEGORIES
    BASE ENABLED_FLAG VARCHAR2(1)
    CTX  OWNER   VARCHAR2(7)
 END FND_DOC_CATEGORY_USAGES
END FND_ATTACHMENT_FUNCTIONS

DEFINE FND_DOCUMENT_ENTITIES
  KEY   DATA_OBJECT_CODE  VARCHAR2(30)
  BASE  APP_SHORT_NAME   VARCHAR2(50)
  BASE  TABLE_NAME     VARCHAR2(30)
  BASE  ENTITY_NAME   VARCHAR2(40)
  CTX   OWNER     VARCHAR2(7)
  BASE  PK1_COLUMN   VARCHAR2(30)
  BASE  PK2_COLUMN   VARCHAR2(30)
  BASE  PK3_COLUMN   VARCHAR2(30)
```

```
BASE  PK4_COLUMN   VARCHAR2(30)
  BASE  PK5_COLUMN   VARCHAR2(30)
  TRANS USER_ENTITY_NAME  VARCHAR2(240)
  TRANS USER_ENTITY_PROMPT VARCHAR2(40)
END FND_DOCUMENT_ENTITIES

DEFINE FND_DOCUMENT_CATEGORIES
  KEY   CATEGORY_NAME   VARCHAR2(30)
  BASE  APP_SHORT_NAME   VARCHAR2(50)
  CTX   OWNER      VARCHAR2(7)
  BASE  START_DATE_ACTIVE  VARCHAR2(11)
  BASE  END_DATE_ACTIVE   VARCHAR2(11)
  BASE  ATTRIBUTE_CATEGORY VARCHAR2(30)
  BASE  ATTRIBUTE1   VARCHAR2(150)
  BASE  ATTRIBUTE2   VARCHAR2(150)
  BASE  ATTRIBUTE3   VARCHAR2(150)
  BASE  ATTRIBUTE4   VARCHAR2(150)
  BASE  ATTRIBUTE5   VARCHAR2(150)
  BASE  ATTRIBUTE6   VARCHAR2(150)
  BASE  ATTRIBUTE7   VARCHAR2(150)
  BASE  ATTRIBUTE8   VARCHAR2(150)
  BASE  ATTRIBUTE9   VARCHAR2(150)
  BASE  ATTRIBUTE10   VARCHAR2(150)
  BASE  ATTRIBUTE11   VARCHAR2(150)
  BASE  ATTRIBUTE12   VARCHAR2(150)
  BASE  ATTRIBUTE13   VARCHAR2(150)
  BASE  ATTRIBUTE14   VARCHAR2(150)
  BASE  ATTRIBUTE15   VARCHAR2(150)
  BASE  DEFAULT_DATATYPE_ID VARCHAR2(50)
  BASE  APP_SOURCE_VERSION VARCHAR2(255)
  TRANS USER_NAME     VARCHAR2(255)
END FND_DOCUMENT_CATEGORIES

DEFINE FND_DOCUMENT_DATATYPES
  KEY   DATATYPE_ID   VARCHAR2(50)
  KEY   NAME     VARCHAR2(30)
  CTX   OWNER     VARCHAR2(7)
  BASE  START_DATE_ACTIVE VARCHAR2(11)
  BASE  END_DATE_ACTIVE  VARCHAR2(11)
  TRANS USER_NAME    VARCHAR2(30)
END FND_DOCUMENT_DATATYPES
```

Concurrent Program Configuration File

The concurrent program configuration file afcpprog.lct downloads and uploads concurrent program definitions. It takes as parameters program name and application name.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
PROGRAM	INCOMPATIBILITY	CONCURRENT_PROGRAM_NAME APPLICATION_SHORT_NAME
EXECUTABLE		EXECUTABLE_NAME

The entity definition is:

```
DEFINE PROGRAM
  KEY   CONCURRENT_PROGRAM_NAME   VARCHAR2(30)
  KEY   APPLICATION       VARCHAR2(50)
  CTX   OWNER          VARCHAR2(7)
  TRANS USER_CONCURRENT_PROGRAM_NAME  VARCHAR2(240)
  BASE  EXEC_APPLICATION     VARCHAR2(50)
  BASE  EXECUTABLE_NAME      VARCHAR2(30)
  BASE  EXECUTION_METHOD_CODE   VARCHAR2(1)
  BASE  ARGUMENT_METHOD_CODE    VARCHAR2(1)
  BASE  QUEUE_CONTROL_FLAG            VARCHAR2(1)
  BASE  QUEUE_METHOD_CODE             VARCHAR2(1)
  BASE  REQUEST_SET_FLAG              VARCHAR2(1)
  BASE  ENABLED_FLAG                  VARCHAR2(1)
  BASE  PRINT_FLAG                    VARCHAR2(1)
  BASE  RUN_ALONE_FLAG                VARCHAR2(1)
  BASE  SRS_FLAG                      VARCHAR2(1)
  TRANS DESCRIPTION                   VARCHAR2(240)
  BASE  CLASS_APPLICATION         VARCHAR2(50)
  BASE  CONCURRENT_CLASS_NAME         VARCHAR2(30)
  BASE  EXECUTION_OPTIONS             VARCHAR2(250)
  BASE  SAVE_OUTPUT_FLAG              VARCHAR2(1)
  BASE  REQUIRED_STYLE                VARCHAR2(1)
  BASE  OUTPUT_PRINT_STYLE            VARCHAR2(30)
  BASE  PRINTER_NAME                  VARCHAR2(30)
  BASE  MINIMUM_WIDTH                 VARCHAR2(50)
  BASE  MINIMUM_LENGTH                VARCHAR2(50)
  BASE  REQUEST_PRIORITY              VARCHAR2(50)
  BASE  ATTRIBUTE_CATEGORY      VARCHAR2(30)
  BASE  ATTRIBUTE1                VARCHAR2(150)
  BASE  ATTRIBUTE2                VARCHAR2(150)
  BASE  ATTRIBUTE3                VARCHAR2(150)
  BASE  ATTRIBUTE4                VARCHAR2(150)
  BASE  ATTRIBUTE5                VARCHAR2(150)
  BASE  ATTRIBUTE6                VARCHAR2(150)
  BASE  ATTRIBUTE7                VARCHAR2(150)
  BASE  ATTRIBUTE8               VARCHAR2(150)
  BASE  ATTRIBUTE9            VARCHAR2(150)
  BASE  ATTRIBUTE10             VARCHAR2(150)
  BASE  ATTRIBUTE11               VARCHAR2(150)
  BASE  ATTRIBUTE12               VARCHAR2(150)
  BASE  ATTRIBUTE13              VARCHAR2(150)
  BASE  ATTRIBUTE14               VARCHAR2(150)
  BASE  ATTRIBUTE15               VARCHAR2(150)
  BASE  OUTPUT_FILE_TYPE          VARCHAR2(4)
  BASE  RESTART                   VARCHAR2(1)
  BASE  NLS_COMPLIANT             VARCHAR2(1)
  BASE  CD_PARAMETER           VARCHAR2(240)
  BASE  INCREMENT_PROC            VARCHAR2(61)
  BASE  MLS_EXECUTABLE_APPLICATION  VARCHAR2(50)
  BASE  MLS_EXECUTABLE_NAME          VARCHAR2(50)
  BASE  ENABLE_TIME_STATISTICS        VARCHAR2(1)
  BASE  SECURITY_GROUP_NAME           NUMBER
  BASE  RESOURCE_CONSUMER_GROUP       VARCHAR2(30)

  BASE  ROLLBACK_SEGMENT              VARCHAR2(30)
  BASE  OPTIMIZER_MODE              VARCHAR2(30)

END PROGRAM
```

Flexfields Setup Data Configuration File

Use the file afffload.lct for loading flexfields data.

> **Warning:** Do not modify the data files you download using the flexfields configuration file. You risk corrupting your flexfields data. Oracle Applications does not support any changes you make to the data files.

The configuration file includes the following entities:

- Value sets
- Descriptive flexfields
- Key flexfields

Flexfield Value Sets

The entity VALUE_SET includes the following table details of table validated value sets, and user exit details of special/pair validated value sets. It includes the values, the normalized value hierarchy, value qualifier values, security rules, security rule lines, security rule usage details, rollup groups, or value hierarchies for the value set.

The key for this entity is FLEX_VALUE_SET_NAME.

Example

```
>FNDLOAD apps/apps 0 Y DOWNLOAD @FND:admin/import/afffload.lct out.ldt \
 VALUE_SET FLEX_VALUE_SET_NAME="Loader_Test"

>FNDLOAD apps/apps 0 Y UPLOAD @FND:admin/import/afffload.lct out.ldt -
```

Descriptive Flexfields

The entity DESC_FLEX includes context column, attribute columns, context, and segment details. This entity references the VALUE_SET for the value set used by a given SEGMENT.

The key is composed of APPLICATION_SHORT_NAME and DESCRIPTIVE_FLEXFIELD_NAME.

Example

```
>FNDLOAD apps/apps 0 Y DOWNLOAD @FND:admin/import/afffload.lct out.ldt \
DESC_FLEX APPLICATION_SHORT_NAME="FND"
DESCRIPTIVE_FLEXFIELD_NAME="FND_FLEX_TEST"

>FNDLOAD apps/apps 0 Y UPLOAD @FND:admin/import/afffload.lct out.ldt -
```

Key Flexfields

The entity KEY_FLEX includes the unique ID column, structure column, segment columns, flexfield qualifier, segment qualifier, structure, Account Generator workflow process, shorthand alias, cross-validation rule, cross-validation rule line, segment, flexfield qualifier assignment, and segment qualifier assignment details.

References VALUE_SET for the value set used by the given segment.

The key is composed of APPLICATION_SHORT_NAME and ID_FLEX_CODE.

Example

```
>FNDLOAD apps/apps 0 Y DOWNLOAD @FND:admin/import/afffload.lct out.ldt \
KEY_FLEX APPLICATION_SHORT_NAME="SQLGL" ID_FLEX_CODE="GL#"

>FNDLOAD apps/apps 0 Y UPLOAD @FND:admin/import/afffload.lct out.ldt -
```

Folders Configuration File

The folder configuration file fndfold.lct downloads and uploads folder definitions.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
FND_FOLDERS	FND_FOLDER_COLUMNS	NAME
	FND_DEFAULT_FOLDERS	OBJECT

The entity definition is:

```
DEFINE FND_FOLDERS

   KEY   FOLDER_ID             NUMBER
   KEY   OBJECT                VARCHAR2(30)
   KEY   FOLDER_NAME           VARCHAR2(80)
   KEY   LANGUAGE              VARCHAR2(4)
   KEY   CREATED_BY            VARCHAR2(4000)
   KEY   CREATOR               VARCHAR2(4000)
   BASE  WINDOW_WIDTH          VARCHAR2(4000)
   BASE  PUBLIC_FLAG           VARCHAR2(1)
   BASE  AUTOQUERY_FLAG        VARCHAR2(1)
   CTX   LAST_UPDATE_DATE      VARCHAR2(11)
   CTX   OWNER                 varchar2(4000)
   BASE  WHERE_CLAUSE          VARCHAR2(2000)
   BASE  ORDER_BY              VARCHAR2(30)

  DEFINE FND_FOLDER_COLUMNS

     KEY   FOLDER_ID            NUMBER
     KEY   ITEM_NAME            VARCHAR2(30)
     BASE  DISPLAY_MODE         VARCHAR2(1)
     BASE  SEQUENCE             VARCHAR2(4000)
     CTX   LAST_UPDATE_DATE     VARCHAR2(11)
     CTX   OWNER                varchar2(4000)
     BASE  ITEM_WIDTH           VARCHAR2(4000)
     TRANS ITEM_PROMPT          VARCHAR2(80)
     BASE  X_POSITION           VARCHAR2(4000)
     BASE  Y_POSITION           VARCHAR2(4000)

  END FND_FOLDER_COLUMNS

  DEFINE FND_DEFAULT_FOLDERS
     KEY   FOLDER_ID              NUMBER
     KEY   OBJECT                 VARCHAR2(30)
     KEY   USER_RESP_TYPE         varchar2(10)
     KEY   USER_ID                varchar2(240)
     CTX   LAST_UPDATE_DATE       VARCHAR2(11)
     CTX   OWNER                  varchar2(4000)
     BASE  APPLICATION_SHORT_NAME VARCHAR2(4000)
     BASE  BEHAVIOR_MODE          VARCHAR2(1)

  END FND_DEFAULT_FOLDERS
```

Additional Considerations

Please note the following when working with folder definitions:

- To change the language you are downloading, set the environment variable NLS_LANG before running the loader.

  Language is a striping column. Folder data is never seeded or translated. As users operate the screen and make customizations, their changes are saved in their current language.

  Do not, for example, create folders in the English language, download the folder definitions and then upload them into another language because all the prompts would remain in English and the WHERE clause saved with the folder may be language sensitive.

- The user who is logged in and creates a given folder is by default the owner of that folder.
- A system administrator can change an owner of a folder using the Administer Folders form under the System Administrator responsibility. A default folder can be assigned to users and responsibilities. The default folder assignments are also handled by the folder loader.
- Make sure the owner and responsibility that is assigned to a folder is in the destination database. If the owner or assigned responsibility of a folder is only on the source database, it will not be uploaded to the destination database.

Download Examples

To download all folders

```
FNDLOAD username/password@database 0 Y DOWNLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt FND_FOLDERS
```

To download folders by "friendly" names

```
FNDLOAD username/password@database 0 Y DOWNLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt FND_FOLDERS
NAME="<name of folder>"
```

Example:

```
FNDLOAD username/password@database 0 Y DOWNLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt FND_FOLDERS
NAME="Receipts Summary Basic"
```

Note: The name of the folder is case-sensitive.

To download folders by internal object names

```
FNDLOAD username/password@database 0 Y DOWNLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt FND_FOLDERS
OBJECT="<internal object name>"
```

Example:

```
FNDLOAD username/password@database 0 Y DOWNLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt FND_FOLDERS
OBJECT="ARXRWRCT"
```

Upload Example

To upload folders:

```
FNDLOAD username/password@database 0 Y UPLOAD
$FND_TOP/patch/120/import/fndfold.lct <name of file>.ldt
```

Lookups Configuration File

Use the file aflvmlu.lct for loading Lookup types and Lookups values.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
FND_LOOKUP_TYPE	FND_LOOKUP_VALUE	VIEW_APPSNAME LOOKUP_TYPE SECURITY_GROUP

The entity definition is:

```
DEFINE FND_LOOKUP_TYPE
  KEY   VIEW_APPSNAME     VARCHAR2(50)
  KEY   LOOKUP_TYPE       VARCHAR2(30)
  CTX   APPLICATION_SHORT_NAME  VARCHAR2(50)
  BASE  CUSTOMIZATION_LEVEL   VARCHAR2(1)
  CTX   OWNER         VARCHAR2(6)
  TRANS MEANING        VARCHAR2(80)
  TRANS DESCRIPTION       VARCHAR2(240)

  DEFINE FND_LOOKUP_VALUE
    KEY   LOOKUP_CODE    VARCHAR2(30)
    BASE  ENABLED_FLAG    VARCHAR2(1)
    BASE  START_DATE_ACTIVE  VARCHAR2(10)
    BASE  END_DATE_ACTIVE   VARCHAR2(10)
    BASE  TERRITORY_CODE   VARCHAR2(2)
    BASE  TAG                    VARCHAR2(30)
    BASE  ATTRIBUTE_CATEGORY  VARCHAR2(30)
    BASE  ATTRIBUTE1     VARCHAR2(150)
    BASE  ATTRIBUTE2        VARCHAR2(150)
    BASE  ATTRIBUTE3       VARCHAR2(150)
    BASE  ATTRIBUTE4       VARCHAR2(150)
    BASE  ATTRIBUTE5       VARCHAR2(150)
    BASE  ATTRIBUTE6       VARCHAR2(150)
    BASE  ATTRIBUTE7       VARCHAR2(150)
    BASE  ATTRIBUTE8        VARCHAR2(150)
    BASE  ATTRIBUTE9        VARCHAR2(150)
    BASE  ATTRIBUTE10       VARCHAR2(150)
    BASE  ATTRIBUTE11       VARCHAR2(150)
    BASE  ATTRIBUTE12        VARCHAR2(150)
    BASE  ATTRIBUTE13      VARCHAR2(150)
    BASE  ATTRIBUTE14       VARCHAR2(150)
    BASE  ATTRIBUTE15       VARCHAR2(150)
    CTX   OWNER             VARCHAR2(6)
    TRANS MEANING          VARCHAR2(80)
    TRANS DESCRIPTION      VARCHAR2(240)
  END FND_LOOKUP_VALUE
END FND_LOOKUP_TYPE
```

Messages Configuration File

Use the file afmdmsg.lct for uploading and downloading messages in a database.

Use the Generic Loader and afmdmsg.lct for transferring messages between databases only. Use the Message Dictionary Generator for moving messages into binary runtime files and readable text files. See: Message Dictionary Generator, page B-22.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
FND_NEW_MESSAGES		APPLICATION_SHORT_NAME MESSAGE_NAME

The entity definition is:

> **Note:** to change the language you are downloading, set the environment variable NLS_LANG before running the loader.

```
DEFINE FND_NEW_MESSAGES
  KEY   APPLICATION_SHORT_NAME   VARCHAR2(50)
  KEY   MESSAGE_NAME             VARCHAR2(30)
  CTX   OWNER                    VARCHAR2(7)
  CTX   MESSAGE_NUMBER           VARCHAR2(50)
  TRANS MESSAGE_TEXT             VARCHAR2(2000)
  CTX   DESCRIPTION              VARCHAR2(240)
  CTX   TYPE                     VARCHAR2(30)
  CTX   MAX_LENGTH               NUMBER
END FND_NEW_MESSAGES
```

Profile Options and Profile Values Configuration File

Use the file afscprof.lct for loading profile options and profile values.

> **Note:** For downloading in previous releases, a NULL profile option value in the database downloaded as a value of NULL; now, if a NULL value exists in the database, nothing is downloaded for it. For uploading in previous releases, if a NULL profile option value exists in the loader data file (indicated by the absence of a value row or the value was ""), then a NULL value was inserted into the database. Now, if a NULL value exists in the loader data file, the corresponding value is deleted from the database upon uploading.

The following table lists the entities, sub-entities (if any), and download parameters for

this configuration file.

Entity	Sub-entities, if any	Download Parameters
PROFILE	FND_PROFILE_ OPTION_VALUES	PROFILE_NAME APPLICATION_SHORT_NA ME

The entity definition is:

```
DEFINE PROFILE
  KEY   PROFILE_NAME       VARCHAR2(80)
  CTX   OWNER          VARCHAR2(7)
  CTX   APPLICATION_SHORT_NAME   VARCHAR2(50)
  TRANS USER_PROFILE_OPTION_NAME  VARCHAR2(240)
  TRANS DESCRIPTION        VARCHAR2(240)
  BASE  USER_CHANGEABLE_FLAG   VARCHAR2(1)
  BASE  USER_VISIBLE_FLAG    VARCHAR2(1)
  BASE  READ_ALLOWED_FLAG    VARCHAR2(1)
  BASE  WRITE_ALLOWED_FLAG    VARCHAR2(1)
  BASE  SITE_ENABLED_FLAG    VARCHAR2(1)
  BASE  SITE_UPDATE_ALLOWED_FLAG  VARCHAR2(1)
  BASE  APP_ENABLED_FLAG     VARCHAR2(1)
  BASE  APP_UPDATE_ALLOWED_FLAG  VARCHAR2(1)
  BASE  RESP_ENABLED_FLAG    VARCHAR2(1)
  BASE  RESP_UPDATE_ALLOWED_FLAG  VARCHAR2(1)
  BASE  USER_ENABLED_FLAG    VARCHAR2(1)
  BASE  USER_UPDATE_ALLOWED_FLAG  VARCHAR2(1)
  BASE  START_DATE_ACTIVE    VARCHAR2(11)
  BASE  END_DATE_ACTIVE     VARCHAR2(11)
  BASE  SQL_VALIDATION      VARCHAR2(2000)

  DEFINE FND_PROFILE_OPTION_VALUES
    KEY   LEVEL        VARCHAR2(50)
    KEY   LEVEL_VALUE     VARCHAR2(100)
    KEY   LEVEL_VALUE_APP    VARCHAR2(50)
    CTX   OWNER       VARCHAR2(7)
    BASE  PROFILE_OPTION_VALUE  VARCHAR2(240)
  END FND_PROFILE_OPTION_VALUES
END PROFILE
```

Request Groups Configuration File

Use the file afcpreqg.lct for loading request group data.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
REQUEST_GROUP	REQUEST_GROUP_ UNIT	REQUEST_GROUP_NAME APPLICATION_SHORT_NAME

The entity definition is:

```
DEFINE REQUEST_GROUP
  KEY   REQUEST_GROUP_NAME    VARCHAR2(30)
  KEY   APPLICATION_SHORT_NAME  VARCHAR2(50)
  CTX   OWNER               VARCHAR2(7)
  TRANS DESCRIPTION          VARCHAR2(800)
  BASE  REQUEST_GROUP_CODE    VARCHAR2(30)
END REQUEST_GROUP
```

Security Information Configuration File

Use the file afsload.lct for downloading and uploading forms, functions, menus, and menu entries.

The following table lists the entities, sub-entities (if any), and download parameters for this configuration file.

Entity	Sub-entities, if any	Download Parameters
FORM		FORM_APP_SHORT_NAME, FORM_NAME
FUNCTION		FUNC_APP_SHORT_NAME FUNCTION_NAME
MENU	ENTRY	MENU
ENTRY		[None]

The entity definition is:

```
DEFINE FORM
  KEY   APPLICATION_SHORT_NAME VARCHAR2(50)
  KEY   FORM_NAME VARCHAR2(30)
  TRANS USER_FORM_NAME VARCHAR2(80)
  TRANS DESCRIPTION VARCHAR2(240)
  CTX   OWNER VARCHAR2(7)
END FORM

DEFINE FUNCTION
  KEY   FUNCTION_NAME VARCHAR2(30)
  BASE  FORM REFERENCES FORM
  BASE  TYPE VARCHAR2(30)
  BASE  PARAMETERS VARCHAR2(2000)
  BASE  WEB_HOST_NAME VARCHAR2(80)
  BASE  WEB_AGENT_NAME VARCHAR2(80)
  BASE  WEB_HTML_CALL VARCHAR2(240)
  BASE  WEB_ENCRYPT_PARAMETERS VARCHAR2(1)
  BASE  WEB_SECURED VARCHAR2(1)
  BASE  WEB_ICON VARCHAR2(30)
  TRANS USER_FUNCTION_NAME VARCHAR2(80)
  TRANS DESCRIPTION VARCHAR2(240)
  CTX   OWNER VARCHAR2(7)
END FUNCTION

DEFINE MENU
  KEY   MENU_NAME VARCHAR2(30)
  TRANS USER_MENU_NAME VARCHAR2(80)
  TRANS DESCRIPTION VARCHAR2(240)
  CTX   OWNER VARCHAR2(7)

 DEFINE ENTRY
    TRANS PROMPT VARCHAR2(60)
    TRANS DESCRIPTION VARCHAR2(240)
    CTX   SUBMENU REFERENCES MENU
    CTX   FUNCTION REFERENCES FUNCTION
    CTX   OWNER VARCHAR2(7)
  END ENTRY
END MENU
```

Message Dictionary Generator

The Message Dictionary Generator (FNDMDGEN) is a concurrent program that generates binary runtime files from the database for Oracle Applications Message Dictionary messages. The following sections describe the operation of the Message Dictionary Generator.

For more information on using the Message Dictionary and creating messages, see the *Oracle Applications Developer's Guide*.

> **Note:** Use the Generic Loader and corresponding configuration file for uploading and downloading message text files into a database.

Message Repositories

Message information is stored in two different repositories, each of which has its own

format and serves a specific need. Following is a description for each of the message repositories, including the message attributes they store.

Database

The FND_NEW_MESSAGES table in the database stores all Oracle Applications messages for all languages. Database messages are directly used only by the stored procedure Message Dictionary API. Database message data can be edited using the Messages form.

Database Attributes are: APPLICATION, LANGUAGE, NAME, NUMBER, TEXT, DESCRIPTION

Runtime

A runtime binary file stores the messages for a single application and a single language. The file is optimized for rapid lookup of individual messages by message NAME.

A runtime file is located in:

```
<APPL_TOP>/$APPLMSG/<LANGUAGE>.msb
```

where <APPL_TOP> is the application basepath, APPLMSG is an environment variable whose usual value is "mesg", and <LANGUAGE> is the NLS language code (for example: 'US', or 'F'.). A typical message file would be $FND_TOP/mesg/US.msb.

Runtime Attributes are: NAME, NUMBER, TEXT

Usage

The help that you get when you invoke the Message Dictionary Generator without any program arguments (i.e., FNDMDGEN dbuser/dbpassword 0 Y) is:

```
FNDMDGEN <Oracle ID/password> 0 Y <language codename> [application
shortname] [mode] [filename] \
```

where mode is:

DB_TO_ RUNTIME From Database to Runtime file (.msb)

Wildcards

Either <language codename> or [application shortname] can be wildcarded by passing the value "ALL". The following describes how wildcards are used:

From DB Messages come from the FND_NEW_MESSAGES table. Wildcards match all the messages in the database.

To RUNTIME In the case of wildcards, separate runtime files are created for each combination of language and application.

Generic File Manager Access Utility (FNDGFU)

The Generic File Manager (GFM) is a set of PL/SQL procedures that leverages Oracle HTTP Server functionality to provide generic upload and download capabilities of unstructured data between a client (usually a web browser) and a database.

FNDGFU is an access utility that allows the upload of files from the local file system to the GFM database file system. It supports simple uploads of single files as well as bulk uploads of many files. FNDGFU also offers a download option that provides a convenient and quick means of retrieving the contents of large objects (LOBs) if the file identifier is known.

This utility is used in uploading help files for the Oracle Applications online help system. For information on downloading and uploading Oracle Applications help files, see: Downloading and Uploading Help Files, page 10-2.

To delete files loaded to the database run the Purge Obsolete Generic File Manager Data concurrent program.

Usage

FNDGFU is located in the $FND_TOP/bin directory. Putting this directory on your path will allow you to invoke FNDGFU easily.

Upload files to the GFM

To upload files using FNDGFU use the following syntax:

```
FNDGFU <logon> [param] <filenames>
```

where

<logon> Specifies a standard Oracle logon string of the form username/password. To specify a particular database, append an @ sign and the database SID (@database).

[param] Includes the following parameters (in any order) as appropriate:

PROGRAM_NAME=<name> specifies the name of the program on whose behalf the LOB is to be maintained.

PROGRAM_TAG=<name> specifies the program tag, which is a string used by the GFM client program to further categorize the LOB.

LANGUAGE=<language_code> specifies the language of the file.

PLS_CALLBACK=<plsql procedure> specifies the procedure to execute once for each uploaded file. The

procedure must accept file_id as its only parameter. FNDGFU will call the specified procedure after each uploaded file, passing in the new file identifier, for example: PLS_CALLBACK=mypackage.myprocedure.

CONTENT_TYPE=<mime_type> specifies the default mime type to use for uploaded files not qualified by a content map.

CONTENT_MAP=<contentmapfile> specifies a text file that maps filename suffixes onto content types. The text file consists of lines of the form <suffix>=<mime_type> where suffix is any string matched against the end of the filename. For example: ".txt = text/plain", ".html = text/html", and ".ps = application/postscript".

<filenames> Specifies the files to upload. Any number of files may be uploaded.

Download files from the GFM

To download a file using the FNDGFU utility, use the following syntax:

```
FNDGFU <logon> DOWNLOAD=<fileid> [LINE_BREAKS=<mode>] [filename]
```

where

<logon> Specifies a standard Oracle logon string of the form username/password. To specify a particular database, append an @ sign and the database SID (@database).

<fileid> Specifies the identifier of the large object (LOB) to download.

<mode> Specifies how to treat line breaks for a text document. This parameter is ignored for nontext content. The following values are valid:

LF - Line breaks will be represented using "/n" in the downloaded output. This is the default mode if the LINE_BREAK parameter is omitted.

CRLF - Line breaks will be left in the canonical format.

[filename] Specifies the file into which to download. If omitted, downloaded contents are streamed to the standard output.

Example of FNDGFU Upload

The FNDGFU utility can be used to upload new or changed help files. Use the

following arguments to upload help files:

```
FNDGFU <apps/pwd> 0 Y PROGRAM_NAME=FND_HELP
PROGRAM_TAG=<application>:<custom_level> CONTENT_TYPE=<mime_type>
LANGUAGE=<language_code> <filenames>
```

where

<apps/pwd>	is the APPS schema username/password. To specify a particular database, append an @ sign and the database SID (@database).
<application>	is the Application short name.
<custom_level>	is the files'customization level. Use the number 100 or above for customized help files. To replace previously uploaded files, use the same customization level when uploading the new files. To override previously uploaded files without deleting them from the database, use a higher customization level.
<mime_type>	is the files' MIME type.
<language_code>	is the files' language code.
<filenames>	is a space-separated list of files to upload, or a filename glob in the current directory.

Enter all arguments on a single command line. They may appear on separate lines here and in the examples that follow depending on the display medium.

Example 1

FNDGFU apps/apps@devdb 0 Y PROGRAM_NAME=FND_HELP PROGRAM_TAG=GL:100 CONTENT_TYPE=text/html LANGUAGE=US file1.htm file2.htm

- connects to apps/apps@devdb
- identifies uploaded files as part of Oracle General Ledger (GL) help
- identifies the uploaded files' customization level as 100
- identifies their MIME type as text/html
- identifies their language as US English (US)
- uploads the two specified .htm files in the current directory (in UNIX)

Example 2

FNDGFU apps/apps@custdb 0 Y PROGRAM_NAME=FND_HELP PROGRAM_TAG=FND:100 CONTENT_TYPE=image/gif *.gif

- connects to apps/apps@custdb
- identifies uploaded files as part of Application Object Library (FND) help
- identifies the uploaded files' customization level as 100
- identifies their MIME type as image/gif
- does not identify their language, which defaults to userenv('LANG')
- uploads all .gif files in the current directory (in UNIX)

Purging Generic File Manager Data

To purge uploaded files from the Generic File Manager, run the concurrent program, Purge Obsolete Generic File Manager Data.

This concurrent program should also be used to periodically expunge expired data. It is recommended that you schedule this program to run every day or so, using the default parameter values.

Purge Obsolete Generic File Manager Data

To purge uploaded files from the Generic File Manager, run the concurrent program, Purge Obsolete Generic File Manager Data.

This concurrent program should also be used to periodically delete obsolete data. It is recommended that you schedule this program to run every day or so, using the default parameter values.

Program Parameters

Expired

Enter "Y" if you want to purge expired data only. Enter "N" if you want the purge to include all data. The default is "Y."

Program Name

Enter the program name(s) to process. Leave blank to process all programs.

Program Tag

Enter the program tag(s) to process. Leave blank to process all program tags.

C

Functional Administrator and Functional Developer Tasks

Overview of Functional Administrator and Functional Developer Responsibilities

Oracle Applications ships two responsibilities that provide access to a subset of system administrator tasks. These tasks are primarily those setup tasks using Oracle Applications HTML-based pages.

Functional Administrator Responsibility

From the Functional Administrator responsibility you can create and/or manage the following features.

From the Security tab:

- Grants
- Permissions and Permission Sets

For more information on using grants and permissions, see: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*.

From the Core Services tab:

- Lookups
- Messages
- Profiles and Profile Categories
- Functions
- Menus

- Caching Framework

For more information on using the Lookups and Messages windows, refer to the respective online help as well as the *Oracle Applications Developer's Guide.*For more information on Profiles, see Overview of Setting User Profiles, *Oracle Applications System Administrator's Guide - Maintenance*. For more information on functions and menus, see: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*. For more information on the Caching Framework, see the section Caching Framework, page C-5 and the *Oracle Applications Java Caching Framework Developer's Guide* available from the "Oracle Application Framework Documentation Resources, Release 12", Oracle*Metalink* Note 391554.1.

From the Personalization tab:

- Application Catalog
- Import/Export

For more information on Personalization, see the section Oracle Application Personalization Framework, page C-3.

Functional Developer Responsibility

From the Functional Developer responsibility you can create and/or manage the following features.

From the Security tab:

- Objects
- Permissions and Permission Sets

For more information on objects and permissions, see: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security* and Overview of Data Security, *Oracle Applications System Administrator's Guide - Security*.

From the Core Services tab:

- Lookups
- Messages
- Profiles
- Functions
- Menus
- Cache Components

For more information on using the Lookups and Messages windows, refer to the online

help as well as the *Oracle Applications Developer's Guide*. For more information on Profiles, see Overview of Setting User Profiles, *Oracle Applications System Administrator's Guide - Maintenance*. For more information on functions and menus, see: Overview of Oracle Applications Security, *Oracle Applications System Administrator's Guide - Security*. For more information on the Cache Components, see the *Oracle Applications Java Caching Framework Developer's Guide* available from the "Oracle Application Framework Documentation Resources, Release 12", Oracle*Metalink* Note 391554.1.

Oracle Application Personalization Framework

Personalization allows you to declaratively tailor the UI look-and-feel, layout or visibility of Oracle Application Framework-based (HTML-based) pages to suit business needs or user preferences.

Durability of OA Framework personalization is largely attributed to the declarative architecture and the object-oriented approach underlying the implementation of the page. Declarative UI component definitions are stored in the form of meta-data in a database repository. Personalizations are translated into offsets from the base meta-data definition and stored separately. At runtime, the applicable personalizations meta-data is uploaded from the repository and layered over the base meta-data definition to produce the net effect. Product upgrades and patches affect only the base meta-data definition, so customer personalizations continue to function properly as applicable.

For more information on Oracle Application Personalization Framework, see the Oracle Application Framework Personalization Guide, *Oracle Application Framework Personalization Guide*, as well as the "Oracle Application Framework Documentation Resources, Release 12", Note 391554.1, on Oracle*MetaLink*.

OA Personalization Framework comes with an administration user interface, which is available under the Functional Administrator responsibility. This interface contains the following two pages that can be used to personalize the pages of OA Framework-based applications at various personalization levels without modifying any code:

- Application Catalog
- Import/Export

The Application Catalog page is useful for managing several personalizations across pages and applications, especially where the administrator does not have a responsibility that can access the page directly.

You can change the layout of a page by adding rows and columns to the customizable regions. You can also change the layout direction and order of contents inside these regions. You can update different elements and you can also add, create, or remove the contents from different regions.

> **Note:** To rearrange contents across different regions, you must first remove them from their current location and then add them inside the

new destination region.

To activate, inactive, or delete specific personalizations, or manage the translation of the personalizations made for the page in question, navigate to the *Manage Personalization Levels* page.

Depending on the type of page you selected to personalize, (configurable or non-configurable), you are automatically directed to one of the following two personalization launch pages:

Page Layout Personalization: (Configurable page). This launch page provides a boxed preview of the flexible layout structure within your page and displays controls that take you to different pages or flows where you specify and apply your actual personalizations.

> **Important:** Page Layout Personalization: (Configurable page). This launch page provides a boxed preview of the flexible layout structure within your page and displays controls that take you to different pages or flows where you specify and apply your actual personalizations.

- Mandatory user-entered parameters
- Flow/business-logic
- Limited access to specific users
- Multi-organization access control

These parameters might not be available and the page might fail with unexpected errors. You should instead access the Personalization UI for your configurable page using the global *Personalize Page* link on the page itself, when the **Personalize Self-service Defn** profile option is enabled.

Page Hierarchy Personalization: (Non-configurable page). This launch page displays the entire structure of the selected page in a hierarchy table (HGrid), rather than as a visual boxed layout.

The Import/Export page allows you to both export meta-data to XML files, and import XML files into a MDS repository.

Both administration-level and user-level personalizations may be extracted from one database and loaded into another. This allows you the freedom to create and test personalizations in a test database before deploying the personalizations to a production instance.

Use the **FND:Personalization Document Root Path** (FND_PERZ_DOC_ROOT_PATH) profile option to define the root path of the current deployed environment where personalizations are exported to and imported from. We recommend you set this profile to the $APPL_TOP staging area and at the site level.

```
$APPL_TOP/<CompanyIdentifier>/<CustProductShortName>/<ProductVersion>/md
s/webui
```

See Deploying Personalizations, *Oracle Application Framework Personalization Guide* for more information.

Caching Framework

Caching provides a powerful, flexible, easy-to-use mechanism for storing database results and other Java objects in memory for repeated usage. This mechanism minimizes expensive object initializations and database round trips, thereby improving application performance.

Application data is cached using component caches. Each component cache is identified by a name. The objects contained in a component cache are generally of the same type and share the same caching attributes. Each component cache has an associated cache loader class. The loader class has the logic for loading the cached object in case of a cache miss. When an object is requested from a component cache, if the object is found, it is returned from the cache. Otherwise, the loader is used to load the object place it in the cache.

For additional information on the Caching Framework, see the "Oracle Applications Java Caching Framework Developer's Guide", available from the "Oracle Application Framework Documentation Resources, Release 12", Note 391554.1, on Oracle*MetaLink.*

Caching Framework comes with an administration user interface, which is available under the Functional Administrator responsibility. This interface contains the following three pages that can be used to implement tuning of the memory management policies and perform administrative operations:

- Overview page
- Tuning page
- Global Configuration page

As a general rule, cache administration should not be required unless there are some performance problems.

The Caching Framework Overview page provides a Cache Usage Summary, listing the following:

- Total Cache Components - All registered cache components in the system. This includes the cache components that have statistics enabled.
- Global Idle Time - A global setting for the elapsed time since an object was accessed last. This value applies to cache components that rely on the default Idle Time and will not override the Idle Time setting of the individual cache components.
- Cache Components with Statistics Enabled - All registered cache components in the system that have statistics enabled.

On the Tuning page, you can search for cache components and then measure the Caching Framework performance by enabling statistics for frequently-used components. The statistics provided include hits, misses, the hit/misses ratio and invalidation count for each cache component. You can also clear collected statistics, and clear the cache.

> **Tip:** A cache 'miss' is when a requested object from a cache component is not found in the cache. To reduce the 'misses' value for a particular cache component, update the **Time Out Type** and **Time Out After** values of the cache component definition. An object is marked 'invalid' when the object has been *idle* beyond the idle timeout period or the object was updated, making the copy in the cache invalid. When an object is 'invalid', any subsequent get() operations on the object gets a new copy of the object from the database.

- **Time Out Type**: Choose either *Idle Time* (recommended) or *Time to Live*. Both values refer to the duration after which the object is marked invalid.
  - *Idle Time*: Starts from the last time the object was requested from the cache. Choose this value when the primary consideration is the memory. This option prevents infrequently used objects from being cleaned up from the cache.
  - *Time to Live*: Starts from the time the object is loaded into the cache. Choose this value when the primary consideration is data consistency. This option guarantees that the values are refreshed after the specified time interval regardless of the usage.
- **Time Out After**: This refers to the Time Out Type. We recommend choosing *Global Idle Time*.
  - *Global Idle Time*: The component cache gets a timeout value that is equal to the global idle time specified. The current default is 15 minutes. This value can be changed on the Global Configuration page.

    > **Note:** Changes to the cache components definition's **Time Out Type** and **Time Out After** values will not be put into effect until after the middle tier is bounced.

Through the Global Cache Configuration page you can update the cache statistics and cache policy for all the cache components.

- **Cache Statistics**: You can choose to enable statistics for all the cache components. However, doing so may affect the performance of the system. You can also clear statistics for all the cache components.
- **Cache Policy**: You can set the Global Idle Time profile option, which refers to the

duration after which any object is marked invalid. You can also clear all cache components, which removes all the cache components from the middle tier. Changing the cache policy can affect performance.

Note: Enabling or disabling the statistics collection of the cache components only affects the current Java Virtual Machine (JVM). To enable/disable statistics collection in other JVMs, bounce those JVMs. The same is true for changes to the Global Idle Time profile option.

D

Oracle Self-Service Web Applications

Overview of Oracle Self-Service Web Applications (HTML-based Applications)

Important: This information is provided for backward compatibility only.

The Oracle Self-Service Web Applications, including Self-Service Expenses, Self-Service Human Resources, Internet Procurement, Internet Receivables, Self-Service Time, Web Suppliers, iStore, iPayment, iSupport, iMarketing, and eTravel from Oracle, extend the functionality of Oracle Applications by adding a browser-based, walk up and use functionality that supplements Oracle Applications.

The Oracle Self-Service Web Applications can be either inquiry or transactional. Inquiry modules read but do not update the Oracle Applications database; transactional modules update the database.

Oracle Self-Service Web Applications Architecture

The architecture consists of the following components:

- A web browser
- Oracle HTTP Server, powered by Apache
- HTML documents
- Java Server Pages, JavaBeans and Servlets

Definitions

The following definitions will help you to understand the big picture of Oracle Self-Service Web Applications.

Oracle HTTP Server

The Oracle HTTP Server (powered by Apache) is based on the open source HTTP server created by the Apache Software Foundation. Information on the Apache Server can be found at http://www.apache.org. This provides the communication services of Oracle Internet Application Server (iAS). The Apache Server is modular. In addition to the standard Apache modules (often referred to as mods) the Oracle HTTP Server adds a number of Oracle specific modules, along with an extension to the functionality of several of the standard mods. Theses include mod_cgi, mod_ssl, mod_jserv and mod_perl.

Common Gateway Interface (CGI)

The industry standard technique for running applications on a web server. Oracle HTTP Server supports this standard.

HTML (HyperText Markup Language)

A format for encoding hypertext documents that may contain text, graphics, and references to programs, and references to other hypertext documents. HTML is a subset of Standard Generalized Markup Language (SGML).

HTTP (HyperText Transfer Protocol)

A protocol used to request documents from the web server.

JavaBeans

A reusable Java class which has specific naming conventions for its methods and variables. JavaBean components can be used to perform well-defined tasks, such as connecting to a database, maintaining client information, or rendering a screen page.

Javascript

Javascript is a scripting language that adds significant power to HTML files without the need for server-based CGI programs.

Java Server Pages

JSPs allow for the embedding of servlet code within HTML pages. The operation of JSPs is similar to that of server-side includes.

Java Servlets

A small, "pluggable" extension to a server that will enhance the server's functionality. Java servlets are a key component of server-side Java devlopment.

mod_cgi

An Apache module that provides for the execution of Common Gateway Interface (CGI) applications through the invocation of an operating system shell that runs the application and uses the CGI to deliver data to the application..

mod_jserv

An Apache module that routes all servlet requests to the Apache JServ Servlet engine. The servlet engine provides the runtime environment to execute servlets. The servlet engine executes from within a Java Virtual Machine (JWM) running on the same node, or a different node, to the Apache HTTP Server. Each JVM has one servlet engine but the number of servlet engines is not proportional to the number of JServ processes. As the mod_jserv and Apache JServ servlet engines are different processes, potentially running on different machines, a protocol called Apache JServ Protocol (AJP) is used for communication.

For more information on the AJP Protocol refer to
http://java.apache.org/jserv/protocol/AJPv11.html

Web Applications Dictionary

An active data dictionary that employs the Oracle Forms-based interface. The data dictionary stores specific information about Self-Service Web Applications data, including prompts, language, navigation, and security.

Web Browser

The client user interface component. The browser you use must support tables and frames and be Javascript enabled. The embedded Javascript coding provides a mechanism for client side caching of user-entered data during a transaction, and simple client side validation of user-entered data. Execution of simple Javascript code logic at the client side results in reduced network traffic between the web browser client and the web server.

Oracle Workflow

Workflows can be defined for business flows so users can be sent automatically all the information they need to make a decision and have other business processes run automatically based upon their responses.

Workflows are defined using the Workflow Builder, a Windows GUI interface that enables users to design the business process, the activities, items, messages and lookup

lists, and roles (the approval chain). This workflow is then integrated into the business transaction process. For Web Employees, it is integrated with the requisition approval process.

Notifications generated in the workflow chain can be viewed with the Oracle Self-Service Web Applications or a Workflow-supported email system.

Oracle Self-Service Human Resources includes a predefined workflow process to generate offer letters.

All workflow processes are customizable.

For more information, see the Oracle Workflow documentation.

Web Applications Dictionary

This is an Oracle Forms-based data dictionary used to define flow content. When users query for data, information is displayed on a web page, complete with hypertext links that enable the user to drill down to more detailed information. The pages that are linked in this way constitute a flow, alternatively referred to as an inquiry. Using the Web Applications Dictionary, you specify the content of, and links between the pages that make up a flow. Specifically, you can specify:

- HTML page format (headers, text, tables)
- Object content by associating with Applications Business Views or PL/SQL
- Business Flows among Objects (hypertext links)
- Page Content (fields, selection criteria)

Web Applications Dictionary also serves as a real time execution engine to retrieve information from the database. Oracle Self Service applications reference the data dictionary at run time to retrieve data from the database and generate dynamic HTML pages.

The Web Applications Dictionary provides a means of defining business flows which can then be web-enabled. All inquiry flows were built using Web Applications Dictionary. These can be customized as needed.

The Web Applications Dictionary is part of Oracle Applications, Release 12 and is part of the "AK Common Modules". Once installed, it is accessed in the same manner as all of the core Oracle Applications.

See: Web Applications Dictionary Overview, page D-5.

Displaying Information Accessed from Servlets and Java Server Pages

When you invoke an OSSWA function that displays information as dynamically generated web pages, the following sequence of events takes place:

1. The user clicks the hyperlink of a function on an OSSWA menu. A URL embedded in the HTML source code is accessed from the browser that calls for a Java servlet.

2. The Oracle HTTP Server, powered by Apache, routes the request to mod_jserv.

3. mod_jserv takes requests and forwards them to Apache Jserv, the servlet engine.

4. The servlet engine generates the response, communicating with the database as required. If the servlet needs to execute any Java Server Pages (JSP) it will contact Oracle JSP. Oracle JSP is a translator and runtime environment for JSPs. Oracle JSP can run as a standalone translator or as part of a servlet engine, to dynamically compile JSPs as required.

5. The response is returned to mod_jserv.

6. The HTTP Server returns the response to the client.

Web Applications Dictionary Overview

The Web Application Dictionary is an active data dictionary that enables you to define applications for the web, and generate many of the application's characteristics at runtime. The data dictionary stores key information about your application, including appearance, language, security requirements, navigation, and data. Because this information is stored in an active data dictionary, you can create an inquiry application for the web specifically designed to meet your business needs.

An Oracle Forms user-interface is provided for you to enter your application's characteristics in the active data dictionary. Through this user-interface, you can customize existing inquiry applications for the web, or create new ones without programming effort. You can create applications that are customizable, extensible, and multilingual.

With Oracle Web Application Dictionary you can:

- Develop applications for the web without programming
- Generate the application web pages at runtime
- Register your application definition in an active data dictionary
- Customize and extend existing applications, and maintain your customizations
- Seamlessly integrate Oracle Applications data and company intranet content
- Completely reconcile company transactions through a web inquiry interface
- Graphically illustrate your application data relationships using Object Navigator

Definitions

Below are some terms used with the Web Applications Dictionary.

Object

A database view.

Attribute

A reusable field used in a web inquiry application. For example, customer name and customer number are both attributes. An attribute is not associated with data. For example, the customer name attribute can be reused anytime a customer name field is displayed on a web inquiry screen.

Object Attribute

A reusable field that results when you associate an attribute with an object.

Region

A logical grouping of data. For example, customer information can be grouped in one region and shipping information can be grouped in another region. A region also represents a section of a web page.

Region Item

A reusable field that results when you associate an attribute or object attribute with a region.

Optional Web Application Dictionary Windows

The steps above show one way of creating a flow using the Web Application Dictionary. However, there are additional optional screens that are provided as well, specifically the Assign Regions screen and its related screen. You can use the Assign Regions screen to assign an object attribute to many regions at once. You can optionally navigate to this screen from the Object Attributes window using the Multiple Assignments button.

> **Important:** The Attribute Values window is not applicable to the Web Application Dictionary. The Attribute Navigation button in the Regions window causes the Attribute Values window to be displayed

Setting the Folder Mode

Three Common Modules folder windows display different fields, depending if you use Oracle Product Configurator, Oracle Self-Service Web Applications, or both. Use the

MODE parameter in the Form Functions window to set which mode to use, Product Configurator or Applications for the Web (Self-Service Web Applications).

If your site uses only one mode, set the MODE parameter at the site level. If your site uses both modes, set the MODE parameter at the user level.

To set the folder mode:

1. Log in to Oracle Applications, choose the System Administrator responsibility, and Open the Form Functions window.
2. For the Object Workbench, Define Regions, and Define Attributes window, set the MODE parameter to one of the following:
3. WEBAPPS if you are using Oracle Self-Service Web Applications.
4. CONFIGURATOR if you are using Oracle Product Configurator.

   For example, if you are using the Product Configurator, set:

   ```
   MODE="CONFIGURATOR"
   ```

> **Note:** The MODE parameter can be set at either the site or user level. If your site uses both Product Configurator and Self-Service Web Applications, set MODE at the user level according to how each user uses these windows.

Web Applications Dictionary Tasks

This section contains tasks using the Web Application Dictionary.

Defining Objects

You must define an object for each database view to be used in your flow. This function registers the view in the Web Application Dictionary.

> **Note:** You can only define one object per database view.

Before defining an object, create views to use in your web inquiry.

To define an object:

1. In the Web Application Dictionary, navigate to the Objects folder window.
2. Enter an object name.
3. Select an application.

4. Select a database object, i.e., a database view.

5. Choose the Object Attributes button to define the attributes for the database object.

6. Choose the Unique Keys button and define the primary and unique keys for the database object.

7. Enter the primary key.

8. Save your work.

9. Choose the Foreign Keys button to define foreign keys for the database object.

Assigning Attributes to Objects

Associate defined attributes with one or more objects (database views) to create object attributes.

> **Note:** Uniform Resource Locator (URL) attributes must be object attributes.

> **Note:** If you are updating existing assignments and you change a long label, you are prompted if you want to change the label for all related object attributes and region items. If you choose OK, all related labels are changed.

Prerequisites to this task are:

- Define objects.
- Define the attributes to assign to objects.

To assign attributes to objects:

1. Navigate to the Object Attributes folder window.

2. Select an existing attribute name to assign to an object.

3. Optionally, select a database view column name corresponding to the object attribute.

4. Enter a long label for the object attribute. The default is the label used when the attribute was defined, but can be overridden.

Note: The remaining data in the Object Attributes folder window defaults from when you defined the attribute. You may override these defaults.

5. Choose the Create Attributes button to create additional attributes. When you close the Attributes window, you are prompted to add the attributes you just created to your object attributes.

To assign multiple regions:

1. Choose the Multiple Assignments button.
2. Enter all the regions that you want the current object attribute assigned to.

Defining Attributes

Attributes can be defined and then assigned to one or more objects.

Note: If you are updating existing attributes and you change a long label, you are prompted if you want to change the label for all related object attributes and region items. If you choose OK, all related labels are changed.

To define attributes:

1. Navigate to the Attributes folder window. Do this by choosing the Create Attributes button from the Object Attributes folder window.
2. Enter an attribute ID, the internal name for this attribute.
3. Enter an application to associate with the attribute.
4. Enter a user–friendly attribute name to be used in lists of values.
5. Enter a long label for the attribute. The default is the attribute name. This is the attribute prompt in your web inquiry application.
6. Optionally, indicate how the text should appear on the browser: bold, italic, and so on.
7. Select a vertical alignment: Top, Center, or Bottom.
8. Select a horizontal alignment: Left, Center, or Right.

9. Enter the datatype for the attribute.

10. Enter the display length for the attribute value.

11. Optionally, enter a free–form description for the attribute.

Defining Unique Keys

For each object, a unique primary key must be defined. A primary key ensures that each row of data can be uniquely identified and cannot be duplicated.

Prerequisites for this task are:

- Define objects.
- Define attributes.
- Define object attributes.

To define a unique key:

1. In the Web Application Dictionary, navigate to the Unique Keys window by choosing the Unique Keys button from the Objects folder window.

2. Enter a name for the unique (primary) key.

3. Enter at least one unique key column sequence. The sequence determines the order the specified columns are evaluated.

Defining Foreign Keys

The combination of primary keys and foreign key relationships determine the navigation through your web flow. That is, if your flow must have navigation from one region to another based upon the same object, a foreign key must be defined for that object.

The prerequisites of this task are:

- Define objects.
- Define attributes.
- Define object attributes.
- Define Unique Key(s).

To define a foreign key:

1. In the Web Application Dictionary, navigate to the Foreign Keys window by choosing the Foreign Keys button from the Objects folder window.
2. Enter the foreign key.
3. Select the parent object (database view).
4. Enter the referenced key. This is the unique (or primary) key of the parent object.
5. Optionally, enter a description for the relationship.
6. Optionally, enter the inverse relationship.
7. Optionally, enter a description for the inverse relationship.
8. Enter the foreign key column.
9. Enter the referenced key column.
10. Repeat the last two steps until all referenced key columns have been assigned.

Defining Regions

Regions can be defined and then assigned to one or more pages.

You can define regions that do not display. Such regions serve as a way of navigating from one object to another.

Before performing this task, define your objects.

To define a region:

1. Navigate to the Regions folder window.
2. If you want to copy an existing region to then modify and save as a new region, choose the Copy button. Enter a new application name, region ID and region name.
3. If you are creating a new region from scratch, enter the ID for the region.
4. Enter a user–friendly region name.
5. Enter the application associated with the region.
6. Select the object name associated with the region.
7. Select a region style: Single–row, Multi–row.

8. If you selected Single–row in the previous step, enter the number of columns (a field and its label) to display in the region before the line wraps.

9. Optionally, enter a free–form description for the region.

10. Use the Region Items button to navigate to the Region Items window.

Creating Region Items

Region items are attributes or object attributes that are placed within a region. These are typically the attributes that you want to display in the region. However, there are exceptions to this.

Prerequisites to this task are:

- Define attributes to associate with regions.
- Define objects.
- Define object attributes to associate with regions.
- Define regions.

To create region items:

1. In the Web Applications Dictionary, navigate to the Region Items folder window. Do this by selecting a region in the Regions window and choosing the Region Items button.

2. Select the attribute type, either attribute or object attribute, to associate to a region. An attribute is usually reserved for use with a button.

3. Select the name of an existing attribute or object attribute.

4. Enter a display sequence for the region item.

   This determines the order of the region items, whether they display or not. If you do not want a region item displayed, select the Hidden item style in the next step.

5. Select an item style, either Button or Text.

   Note: The Checkbox, Hidden, and Poplist item styles are not supported.

6. Optionally, indicate whether the region item can be queried.

7. Optionally, indicate whether the underlying column of the region item should

determine the order in which data is displayed, and whether that order is ascending or descending.

This generates a web query form.

E

Timezone Support

User-Preferred Timezones

Release 12 of Oracle E-Business Suite includes as standard a feature called *User-Preferred Time Zone Support*. In most existing E-Business Suite implementations, all users interact with the system in the "corporate time zone", which will normally be the time zone of the headquarters of the implementing company, and the time zone in which the database runs. This means that remote users have to be aware of the time difference between their location and that of the corporate headquarters.

Employing the user-preferred time zone feature enables users to specify their local time zone for both display and entry of date-with-time fields. Key consequences of this are:

- Users see date-with-time fields in their preferred (local) time zone, and can enter dates with time in this time zone
- Date fields without a time component are not affected by this feature
- The data in the database continues to be stored in the standard corporate time zone

This appendix discusses the capabilities, limitations, and implementation details of the user-preferred time zone feature.

Time Zone Concepts

Conceptually, there are two types of date fields:

- Dates *with* a time component – used to indicate a specific point in time within a particular day
- Dates *without* a time component – used to denote a particular day, but not a specific point in time within that day

Date fields with a time component can be represented in any time zone, and thus displayed in whichever time zone is most meaningful to the end user. Generally, users

prefer to view dates in their own (local) time zone. With the user-preferred time zone feature enabled, date with time fields will be converted to the user's preferred time zone for display.

Date fields without a time component cannot be represented in different time zones, because no meaningful conversion is possible for a date that does not include a specific time. Such a date is entered with respect to one time zone, and in general must be viewed as a day in that time zone, regardless of the location (and possibly different time zone) in which it is being viewed. Oracle E-Business Suite typically uses the corporate time zone for these day definitions: dates without a time component represent the day with respect to the corporate headquarters (corporate days).

There are some exceptions to the above rule. For example, dates without a time component may be held as *ANSI dates*, to represent dates independently of the time zone in which they are being viewed. In such a case, a benefit that starts on 1st January will start on that date wherever in the world it is made available; that is, it will apply to anyone who is in a time zone where it is 1st January.

Many dates without a time component represent pointers to a financial period. These dates are not meant to indicate the exact hour and minute that a transaction occurred, but rather the financial period into which the transaction is accounted. This is a financial bucketing from the perspective of the implementing company. For example, the invoice dates on Payables or Receivables invoices never change based on who is looking at them: they are classified as invoices for that day (and thus that period), regardless of the viewer or his local time zone.

For further details of setting up Oracle Applications to use multiple time zones, see Oracle*MetaLink* Note 402650.1, *User Preferred Time Zone Support in Oracle E-Business Suite Release 12*. For a discussion of time zones in the context of other globalization-related topics, see Globalization Support in *Oracle Applications Concepts*.

Upgrade Considerations

There are no upgrade considerations for E-Business Suite customers upgrading from 11.5.10 CU2 or higher who have already enabled user-preferred time zone support. The upgrade will be transparent and the relevant functionality will not change.

For E-Business Suite customers who are upgrading to Release 12 from a release in which the user-preferred time zone feature was not enabled or not used, existing time zone practices must be taken into account when enabling user-preferred time zone support.

Prior to Release 11.5.10 (with CU2), users could deal with time zone differences in one of two ways:

- By entering data in the standard corporate time zone (as recommended by Oracle).

  The choices in this case are between:

  1. Enabling the user-preferred time zone feature and allowing users to continue

working in the standard corporate time zone (as before the upgrade)

2. Enabling the user-preferred time zone feature, setting a preferred time zone, and starting to interact with the system in this time zone. Failing to set a user-level time zone preference at user level will result in the standard corporate time zone continuing to be used.

- By entering data in the local time zone, and using a custom solution to resolve any issues that arise when comparing data across transactions initiated in different time zones.

  In this case, enabling the user-preferred time zone feature could result in unwanted date-with-time conversions, whereby local dates with time are converted to the user-preferred time zone: this is because the system assumes that all dates with time are stored in the corporate time zone.

For further details, refer to *Oracle Applications Upgrade Guide: Release 11i to Release 12*.

Implementation Details

This section provides the technical considerations involved in implementing the user-preferred time zone feature.

3.1 Technology Stack Requirements

For user-preferred time zone support to operate correctly, all of the following must be true:

- The operating system time zone setting for the database server must be set to the standard corporate time zone.
- The database must be configured to use the time zone file `timezlrg.dat` rather than the `timezone.dat` file.
- The database must be started in the standard corporate time zone.
- Every application tier JVM (Java Virtual Machine) must be started in the standard corporate time zone. This can be achieved by setting the application tier operating system time zone to match the standard corporate time zone.
- Profile 'Server Timezone' (SERVER_TIMEZONE_ID) must be set at site level, and must be set to the same standard corporate time zone as the database.
- Profile 'Client Timezone' (CLIENT_TIMEZONE_ID) must be set at user level. This is applicable to Oracle Forms based UIs.
- Preference 'Timezone' must be set at user level. This is applicable to HTML based UIs.

- Profile 'Enable Timezone Conversions' (ENABLE_TIMEZONE_CONVERSIONS) must be set to 'Yes' (or 'Y') at site level.
- Profile 'Concurrent: Multiple Time Zones' (CONC_MULTI_TZ) must be set to 'No' (or 'N') at the site level.
- Environment variable FORMS_APPSLIBS must be set in the Forms tier
- Forms must be launched through the Personal Home Page or Navigator portlet.

These requirements are discussed in more detail below.

Time Zone File

The database must be started using the `timezlrg.dat` file, which contains the time zone definitions that are used within Oracle E-Business Suite. To do this in a UNIX environment, issue a command such as:

```
setenv ORA_TZFILE $ORACLE_HOME/oracore/zoneinfo/timezlrg.dat
```

before starting the database.

The database must also be started in the standard corporate time zone. To set this in a UNIX environment, issue a command such as:

```
setenv TZ <Timezone Code>  [For example, 'America/Los_Angeles']
```

You can verify your setup by running the following command in SQL*Plus:

```
select to_char(SYSDATE, 'DD-MON-RRRR HH24:MI:SS')
from dual;
```

to ensure the date with time returned are correct for the corporate time zone.

3.1.2 Applications Profiles and Preferences

The profile option Server Timezone (SERVER_TIMEZONE_ID) should be set at site level to the standard corporate time zone (the time zone in which the server has been set to run).

> **Caution:** This profile option should not be changed once set, as existing data will not be updated.

Users may specify their preferred time zone at user level. This is done by setting the 'Timezone' preference in HTML-based applications, or by setting the 'Client Timezone' profile option in Forms-based applications. As with most profile options, the user will need to log out and log back in for the change to take effect. The preferred time zone may be changed as often as needed.

The profile option 'Enable Timezone Conversions' (ENABLE_TIMEZONE_CONVERSIONS) has a default value of 'No' at site level. This will cause the applications to continue showing all dates in the corporate time zone.

Setting this value to 'Yes' will enable the automatic conversion of all date with time fields to the user-preferred time zone.

> **Important:** Unless users are notified of this change, they may think that they are still operating in the corporate time rather than local time (or vice versa), and consequently enter or interpret data erroneously.

Note the behavior of the existing profile 'Concurrent: Multiple Time Zones' (CONC_MULTI_TZ). This was an older feature to handle batch processing. Setting this profile to 'Yes' alters the default value that appears for the Scheduled Start Date in the Submit Requests screen to SYSDATE-1. With the new user preferred time zone feature enabled, this profile is no longer needed, and should have a value of 'No'.

3.1.3 Environment Variable FORMS_APPSLIBS

This environment variable controls multiple aspects of Oracle Forms in the Oracle E-Business Suite environment, and must be left unchanged from the installed setting.

Launching Forms-based Applications

The time zone feature is only available in Oracle Forms based user interfaces within Oracle E-Business Suite when the user logs in through the Personal Home Page or the Navigator portlet. Direct launching of Forms, for example by typing a URL into the browser address line, is supported only for bootstrap purposes, and will not enable the time zone feature, or other features such as language settings and date formats.

Index

A

B

C

D

E

F

P

R

S

T

U

V

W