Oracle® Applications System Administrator's Guide

RELEASE 11

March 1998

ORACLE®

Enabling the Information Age™

Oracle® Applications System Administrator's Guide Release 11

The part number for this volume is A58194–01.

Major Contributors: Steve Carter

Contributors: Tia Alexander, Ram Bhoopalam, Anne Carlson, Steve Carter, Siu Chang, Angela Hsieh, Elizabeth Mitcham, John Mitcham, Tony Onisko, Georges Smine, Sara Woodhull

Contents

Preface

Welcome to Release 11 of the *Oracle System Administrator's Guide.*

This user's guide includes the information you need to work with Oracle Applications System Administrator's duties effectively. It contains detailed information about the following:

- Overview and reference information
- Oracle Applications System Administrator's implementation suggestions
- Specific tasks you can accomplish as a Applications System Administrator
- How to use Oracle Applications System Administrator's windows
- Oracle Applications System Administrator's programs and reports
- Oracle Applications System Administrator's functions and features
- Oracle Applications System Administrator's system setup

This preface explains how this user's guide is organized and introduces other sources of information that can help you.

About This User's Guide

This guide is the primary source of information about Oracle Applications System Administration. It contains overviews as well as task and reference information. This guide includes the following chapters:

- Chapter 1 describes the job of an Oracle Applications System Administrator, and contrasts it with the job of an Oracle Database Administrator.
- Chapter 2 explains how to secure access to the data and functionality within your Oracle Applications.
- Chapter 3 explains how to audit your application users and the changes they effect on your application's data.
- Chapter 4 describes the help architecture for HTML as well as explaining how to customize Oracle Applications help.
- Chapter 5 explains the role of user profiles in Oracle Applications.
- Chapters 6 and 7 explain concurrent processing in Oracle Applications, including how you can manage programs running concurrently in the background while your users continue to perform online tasks and how to manage your concurrent programs and organize those programs into groups and sets.
- Chapter 8 explains using printers with Oracle Applications.
- Chapter 9 explains Oracle Applications security tasks that require a database administrator to either explicitly perform, or assist by performing prerequisite tasks.
- Chapter 10 explains how you can use document sequences and dynamic currency with your Oracle Applications.
- Chapter 11 describes the architecture and implementation details for the CUSTOM library.

The Appendixes provide a reference source about the default menus, user profile options, and runtime Alerts included with Oracle System Administration.

This user's guide is available online

All Oracle Applications user's guides are available online, in both HTML and Adobe Acrobat format. (Most other Oracle Applications documentation is available in Adobe Acrobat format.)

The paper and online versions of this manual have identical content; use whichever format is most convenient.

The HTML version of this book is optimized for on–screen reading, and lets you follow hypertext links for easy access to books across our entire library; you can also search for words and phrases if your national language is supported by Oracle's Information Navigator. The HTML documentation is available from the Oracle Applications toolbar, or from a URL provided by your system administrator.

You can order an Oracle Applications Documentation Library CD containing Adobe Acrobat versions of each manual in the Oracle Applications documentation set. Using this CD, you can search for information, read it on–screen, and print individual pages, sections, or entire books. When you print from Adobe Acrobat, the resulting printouts look just like pages from an Oracle Applications hardcopy manual.

> **Note:** There may be additional material that was not available when this user's guide was printed. To learn if there is a documentation update for this product, look at the main menu on this product's HTML help.

Assumptions

This guide assumes you have a working knowledge of the principles and customary practices of your business area. If you have never used Applications we suggest you attend one or more of the Oracle Applications System Administrator's training classes available through Oracle Education. (See Other Information Sources for more information about Oracle training.)

This guide also assumes that you are familiar with the Oracle Applications graphical user interface. To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide.*

Do Not Use Database Tools to Modify Oracle Applications Data

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 like 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 forms, 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.

Warning: Oracle Applications does not support *any* customization of Oracle Application Object Library tables or modules, even by Oracle consultants. Oracle Application Object Library table names begin with "FND_".

Do not write data directly into or change data in FND_ tables through any custom program or using any tool, including SQL*Plus, Oracle Data Browser, database triggers or other programming tools. You risk corrupting your database and damaging all your applications.

When you use Oracle Applications forms to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications also keeps track of who changes information. But, 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.

Consequently, we STRONGLY RECOMMEND that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications tables, unless we tell you to do so in our manuals.

Other Information Sources

You can choose from many sources of information, including documentation, training, and support services, to increase your knowledge and understanding of Oracle Applications System Administrator's.

Most Oracle Applications documentation is available in Adobe Acrobat format on the *Oracle Applications Documentation Library* CD. We supply this CD with every software shipment.

If this manual refers you to other Oracle Applications documentation, use only the Release 11 versions of those manuals unless we specify otherwise.

Oracle Applications User's Guide

This guide explains how to navigate, enter data, query, run reports, and introduces other basic features of the graphical user interface (GUI). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent requests.

You can also access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

Related User's Guides

If you do not have the hardcopy versions of these manuals, you can read them by choosing Library from the Help menu, or by reading from the Oracle Applications Document Library CD, or by using a web browser with a URL that your system administrator provides.

Oracle Applications Flexfields Guide

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

Oracle Workflow Guide

This manual explains how to define new workflow business processes as well as customize existing Oracle Applications–embedded workflow processes. You also use this guide to complete the setup steps necessary for any Oracle Applications product that includes workflow–enabled processes.

Oracle Alert User's Guide

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

Country–Specific Manuals

Use these manuals to meet statutory requirements and common business practices in your country or region. They also describe additional features added to Oracle Applications System Administrator's to meet those requirements. Look for a User's Guide appropriate to your country; for example, see the *Oracle Financials for the Czech Republic User's Guide* for more information about using this software in the Czech Republic.

Oracle Applications Character Mode to GUI Menu Path Changes

This is a quick reference guide for experienced Oracle Applications end users migrating from character mode to a graphical user interface (GUI). This guide lists each character mode form and describes which GUI windows or functions replace it.

Oracle Financials Open Interfaces Guide

This guide is a compilation of all open interface discussions in all Oracle Financial Applications user's guides. You can also read about the Oracle Applications System Administrator's open interface tables in the appendix of the *Oracle Applications System Administrator's User's Guide.*

Multiple Reporting Currencies in Oracle Applications

If you use Multiple Reporting Currencies feature to record transactions in more than one currency, use this manual before implementing Oracle Applications System Administrator's. The manual details additional steps and setup considerations for implementing Oracle Applications System Administrator's with this feature.

Multiple Organizations in Oracle Applications

If you use the Oracle Applications Multiple Organization Support feature to use multiple sets of books for one Oracle Applications System Administrator's installation, use this guide to learn about setting up and using Oracle Applications System Administrator's with this feature.

Oracle Applications Implementation Wizard User's Guide

If you are implementing more than one Oracle product, you can use the Oracle Applications Implementation Wizard to coordinate your setup activities. This guide describes how to use the wizard.

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*. It also provides information to help you build your custom Developer/2000 forms so that they integrate with Oracle Applications.

Oracle Applications User Interface Standards

This manual contains the user interface (UI) standards followed by the Oracle Applications development staff. It describes the UI for the Oracle Applications products and how to apply this UI to the design of an application built by using Oracle Forms 4.5.

Installation

Oracle Applications Installation Manual

This manual and the accompanying release notes provide information you need to successfully install Oracle Financials, Oracle Public Sector Financials, Oracle Manufacturing, or Oracle Human Resources in your specific hardware and operating system software environment.

Oracle Applications Upgrade Manual

This manual explains how to prepare your Oracle Applications products for an upgrade. It also contains information on finishing the upgrade procedure for each product. Refer to this manual and the *Oracle Applications Installation Manual* when you plan to upgrade your products.

Oracle Application Object Library Technical Reference Manual

The *Oracle Applications Object Library Technical Reference Manual* contains database diagrams and a detailed description of Oracle Applications System Administrator's and related applications database tables,

forms, reports, and programs. This information helps you convert data from your existing applications, integrate Oracle Applications System Administrator's with non–Oracle applications, and write custom reports for Application Object Library.

You can order a technical reference manual for any product you have licensed. Technical reference manuals are available in paper format only.

Other Information

Training

Oracle Education offers a complete set of training courses to help you and your staff master Oracle Applications. We can help you develop a training plan that provides thorough training for both your project team and your end users. We will work with you to organize courses appropriate to your job or area of responsibility.

Training professionals can show you how to plan your training throughout the implementation process so that the right amount of information is delivered to key people when they need it the most. You can attend courses at any one of our many Educational Centers, or you can arrange for our trainers to teach at your facility. In addition, we can tailor standard courses or develop custom courses to meet your needs.

Support

From on–site support to central support, our team of experienced professionals provides the help and information you need to keep Applications working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle server, and your hardware and software environment.

About Oracle

Oracle develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as a complete family of financial, manufacturing, and human resource applications.

Oracle products are available for mainframes, minicomputers, personal computers, network computers**,** and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle offers its products, along with related consulting, education, and support services, in over 140 countries around the world. Oracle Corporation is the world's leading supplier of software for information management, and is the world's second largest software company.

Thank You

Thank you for using Oracle Applications and this user's guide.

We value your comments and feedback. At the end of this manual is a Reader's Comment Form you can use to explain what you like or dislike about this user's guide. Mail your comments to the following address or call us directly at (650) 506–7000.

Oracle Applications Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Or, send electronic mail to **appsdoc@us.oracle.com**.

CHAPTER

1 What Is System Administration?

This chapter briefly describes the job of an Oracle Applications System Administrator, and contrasts it with the job of an Oracle Database Administrator.

What Is System Administration?

A System Administrator is a person responsible for controlling access to Oracle Applications and assuring smooth ongoing operation. Each site where Oracle Applications is installed needs a system administrator to perform tasks such as:

- Managing and controlling security. Decide which users have access to each application, and within an application, which forms, functions, and reports a user can access.
- Setting up new users. Register new Oracle Applications users, and give them access to only those forms, functions, and reports they need to do their jobs.
- Auditing user activity. Monitor what users are doing and when they do it. Choose who to audit and what type of data to audit.
- Setting user profiles. A user profile is a set of changeable options that affects the way Oracle Applications look and behave. A System Administrator can set user profile values at the site, application, responsibility, and user levels.
- Managing concurrent processing. Concurrent Processing is an Oracle Applications facility that lets long–running, data–intensive tasks run simultaneously with online operations, taking full advantage of multitasking and parallel processing. A System Administrator can monitor and control concurrent processing using a few simple forms.

System vs. Database Administrator

You can think of Oracle Applications as having two sides: a front end that users see and work with, and a back end where data manipulation is performed.

The natural division between user applications and the underlying database structures results in two separate job functions: system administrator, and database administrator.

An Oracle Applications System Administrator administers the user interface or applications side of Oracle Applications.

An Oracle Database Administrator (DBA) administers the data that users enter, update, and delete while using Oracle Applications.

CHAPTER

2

Managing Oracle Applications Security

This chapter explains how to secure access to the data and functionality within your Oracle Applications. The key element in Oracle Applications security is the definition of a *responsibility*. A responsibility defines:

- Application database privileges
- An application's functionality that is accessible
- The concurrent programs and reports that are available

As System Administrator you define application users, and assign one or more responsibilities to each user.

This chapter begins with an essay explaining security in Oracle Applications. Afterwards, major topics are explained in greater detail, along with descriptions of the forms you use to implement security in Oracle Applications. Some topics will refer you to other chapters which contain more detailed information, as well as relevant form descriptions.

Overview of Oracle Applications Security

As System Administrator, you define Oracle Applications users, and assign one or more responsibilities to each user.

Defining Application Users

You allow a new user to sign–on to Oracle Applications by defining an *application user.* An application user has a username and a password. You define an initial password, then the first time the application user signs on, they must enter a new (secret) password.

When you define an application user, you assign to the user one or more responsibilities. If you assign only one responsibility, the user, after signing on, immediately enters an application.

If you assign two or more responsibilities, the user, after signing on, sees a window listing available responsibilities.

Responsibilities define Application Privileges

A *responsibility* is a level of authority in Oracle Applications that lets users access only those Oracle Applications functions and data appropriate to their roles in an organization. Each responsibility allows access to:

- A specific application or applications, such as Oracle General Ledger or Oracle Planning.
- A set of books, such as U.S. Operations or German Sales or an organization, such as New York Manufacturing or New York Distribution.
- A restricted list of windows that a user can navigate to; for example, a responsibility may allow certain Oracle Planning users to enter forecast items, but not enter master demand schedule items.
- A restricted list of functions a user can perform. For example, two responsibilities may have access to the same window, but one responsibility's window may have additional function buttons that the other responsibility's window does not have.
- Reports in a specific application; as system administrator, you can assign groups of reports to one or more responsibilities, so the responsibility a user choose determines the reports that can be submitted.

Each user has at least one or more responsibilities and several users can share the same responsibility. A system administrator can assign users

any of the standard responsibilities provided with Oracle Applications, or create new custom responsibilities.

Defining a Responsibility

When you define a responsibility, you assign to it some or all of the components described below:

Data Group (required)

A Data Group defines the mapping between Oracle Applications products and ORACLE IDs. A Data Group determines which Oracle database accounts a responsibility's forms, concurrent programs, and reports connect to. See: Defining Data Groups: page 6 – 27.

Request Security Group (optional)

A request security group defines the concurrent programs, including requests and request sets, that may be run by an application user under a particular responsibility. See: Defining a Request Security Group: page 2 – 6. See: Organizing Programs into Request Groups: page 6 – 17.

Menu (required)

A menu is a hierarchical arrangement of application functions (forms) that displays in the Navigate window. Menus can also point to non–form functions (subfunctions) that do not display in the Navigate window, but that define the range of application functionality available for a responsibility. Each responsibility is associated with a menu. See: Overview of Function Security: page 2 – 22.

Function and Menu Exclusions (optional)

A responsibility may optionally have function and menu exclusion rules associated with it to restrict the application functionality enabled for that responsibility. See: Overview of Function Security: page 2 – 22.

Additional Notes About Responsibilities

Predefined Responsibilities

All Oracle Applications products are installed with predefined responsibilities. Consult the reference guide for your Oracle Applications product for the names of those predefined responsibilities.

Additionally, instances of the major components that help define a responsibility (data groups, request security groups, menus, and functions) are predefined for Oracle Applications.

Responsibilities and Request Security Groups

When a request group is assigned to a responsibility, it becomes a *request security group*.

From a standard submission form, such as the Submit Requests form, users can run only the reports, concurrent programs, and request sets that are in their responsibility's request security group.

- If you do not include the Submit Requests form on the menu for a responsibility, then you do not need to assign a request security group to the responsibility.
- If a request security group is not assigned to a responsibility, then users working under that responsibility cannot run any reports, request sets, or other concurrent programs from a standard submission form.

Responsibilities and Function Security

Oracle Applications GUI–based architecture aggregates several related business functions into a single form. Parts of an application's functionality may be identified as individual Oracle Applications functions, which can then be secured (i.e., included or excluded from a responsibility).

See: Overview of Function Security: page 2 – 22

Defining a Request Security Group

Beyond this short introduction, request groups and request security groups are discussed in greater detail, as part of a broader range of topics not necessarily limited to application security, in Chapter 7 – Managing Concurrent Programs and Reports.

See:

Organizing Programs into Request Groups: page 6 – 17

Request Groups: page 6 – 46

Using Request Security

You use request security to specify the reports, request sets, and concurrent programs that your users can run from a standard submission form, such as the Submit Requests form.

To set up request security, you define a request group using the Request Groups form. Using the Responsibilities form, you assign the request group to a responsibility. The request group is then referred to as a *request security group*. See: Request Security Groups: page 6 – 17.

You can define a request group to contain single requests, request sets, or *all* the requests and request sets in an application.

If you choose to include all the requests and requests sets in an application, the user has automatic access to any new requests and request sets (without owners) in the future.

A request security group can contain requests and request sets from different applications. If you want to define request security groups that own requests from different applications, please refer to the discussion on Data Groups. See: Defining Data Groups: page 6 – 27.

Individual Requests and Request Sets

Reports or concurrent programs that are not included in a request security group on an individual basis, but that do belong to a request set included in a request security group, have the following privileges:

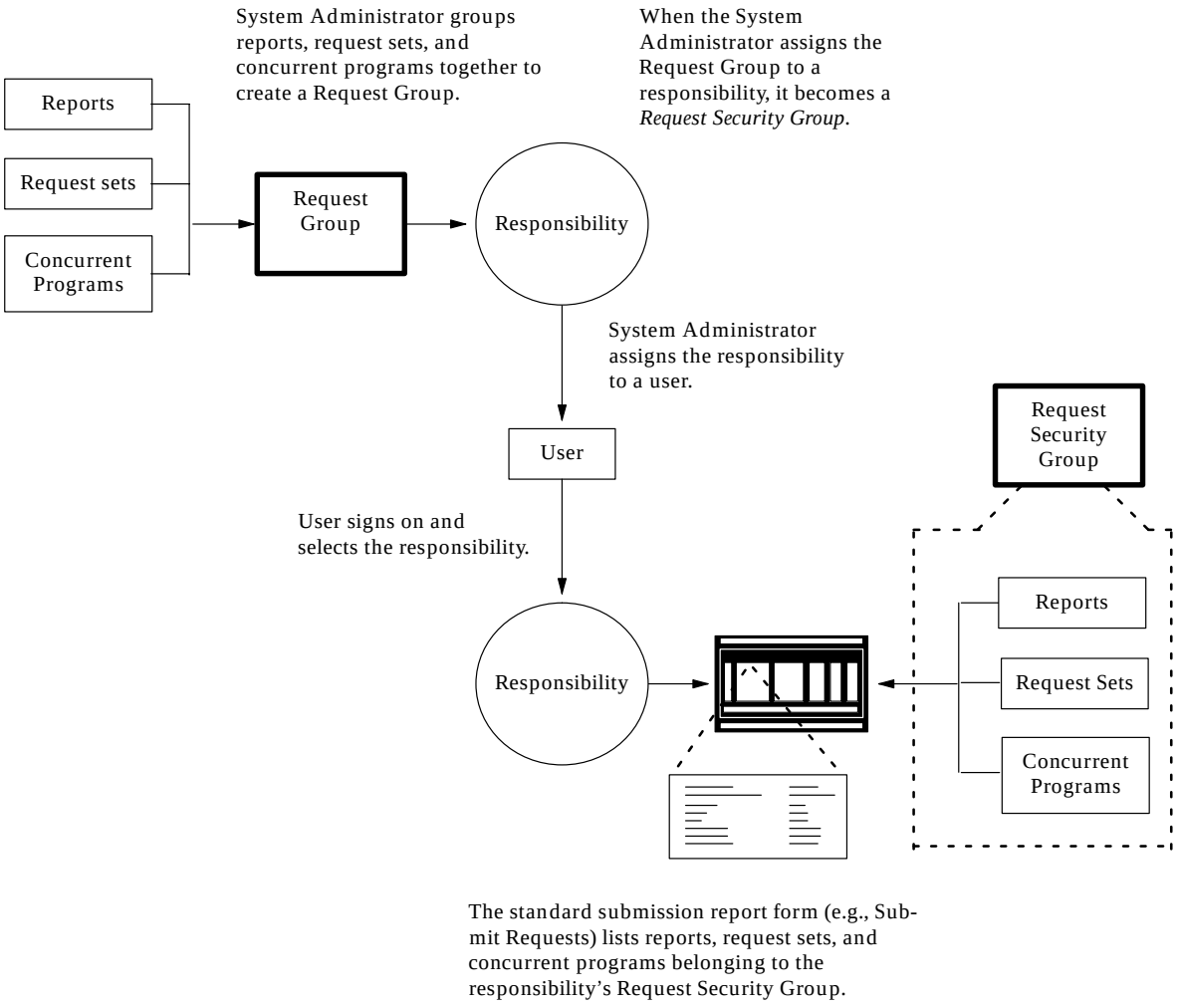
- Users cannot use the Submit Requests form to run single requests and request sets that are not in their responsibility's request security group.
- Users can, however, run request sets that contain requests that are not in their request security group, if the request set is in their request security group.

If you assign a request set, but not the requests in the set, to a request security group, the user:

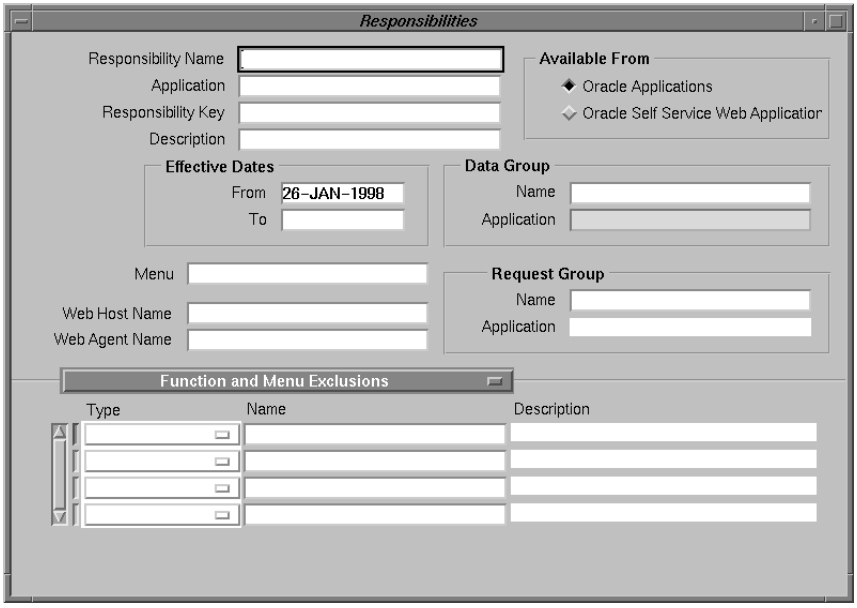
- cannot edit request information in the request set definition
- cannot stop specific requests in the set from running
- can edit the request set by deleting requests from it or adding other requests to it, only if the user is the assigned owner of the request set

The Request Security Groups figure illustrates the relationship between a request security group, application user, and a responsibility.

Request Security Groups



Responsibilities Window



Use this window to define a responsibility. Each application user is assigned at least one responsibility. A responsibility determines if the user accesses Oracle Applications or Oracle Self–Service Web Applications, which applications functions a user can use, which reports and concurrent programs the user can run, and which data those reports and concurrent programs can access.

> **Note:** Responsibilities cannot be deleted. To remove a responsibility from use, set the Effective Date's To field to a past date. You must restart Oracle Applications to see the effect of your change.

See: Overview of Function Security: page 2 – 22

Prerequisites

- Use the Data Groups window to list the ORACLE username your responsibility's concurrent programs reference on an application–by–application basis.
- Use the Request Groups window to define the Request Group you wish to make available with this responsibility.
- Use the Menus window to view the predefined Menu you could choose to assign to this responsibility.

Responsibilities Block

An application name and a responsibility name uniquely identify a responsibility.

Responsibility Name

If you have multiple responsibilities, a pop–up window includes this name after you sign on.

Application

This application name does not prevent the user of this responsibility from accessing other applications' forms and functions if you define the menu to access other applications.

Responsibility Key

This is a unique name for a responsibility that is used by loader programs. Loaders are concurrent programs used to "load" such information as messages, user profiles and user profile values into your Oracle Applications tables. To help ensure that your responsibility key is unique throughout your system, begin each Responsibility Key name with the application short name associated with this responsibility.

Menu

The menu whose name you enter must already be defined with Oracle Applications. See: Menus: page 2 – 36.

Web Host Name

If your Web Server resides on a different machine from your database, you must designate the host name (URL) here. Otherwise, the Web Host Name defaults to the current database host server.

Web Agent Name

Enter the PL/SQL Agent Name for the database used by this responsibility. If you do not specify an Agent Name, the responsibility defaults to the agent name current at log–on.

Available From

A responsibility may be associated with only one applications system. Select between Oracle Self–Service Web Applications or Oracle Applications.

Effective Dates

From/To

Enter the start/end dates on which the responsibility becomes active/inactive. The default value for the start date is the current date, and if you do not enter an end date, the responsibility is valid indefinitely.

You cannot delete a responsibility because its information helps to provide an audit trail. You can deactivate a responsibility at any time by setting the end date to the current date. If you wish to reactivate the responsibility, change the end date to a date after the current date, or clear the end date.

Data Group

Name/Application

The data group defines the pairing of application and ORACLE username.

Select the application whose ORACLE username forms connect to when you choose this responsibility. The ORACLE username determines the database tables and table privileges accessible by your responsibility. Transaction managers can only process requests from responsibilities assigned the same data group as the transaction manager.

Request Group

Name/Application

If you do not assign a request security group to this responsibility, a user with this responsibility cannot run requests, request sets, or concurrent programs from the Submit Requests window, except for request sets owned by the user. The user can access requests from a Submit Requests window you customize with a request group code through menu parameters.

See:

Overview of Oracle Applications Security: page 2 – 2

Customizing the Submit Requests Window Using Codes: page 6 – 18

Function and Menu Exclusions Block

Define function and menu exclusion rules to restrict the application functionality accessible to a responsibility.

Type

Select either Function or Menu as the type of exclusion rule to apply against this responsibility.

- When you exclude a function from a responsibility, all occurrences of that function throughout the responsibility's menu structure are excluded.
- When you exclude a menu, all of its menu entries, that is, all the functions and menus of functions that it selects, are excluded.

Name

Select the name of the function or menu you wish to exclude from this responsibility. The function or menu you specify must already be defined in Oracle Applications.

Self–Service Applications Security

Oracle Self–Service Web Applications uses columns, rows and values in database tables to define what information users can access. Table columns represent ”attributes” that can be assigned to a responsibility as Securing Attributes or Excluded Attributes. These attributes are defined in the Web Application Dictionary.

See Also

Data Security: *(Oracle Self–Service Applications for the Web User’s Guide)*

Defining Attributes: *(Oracle Self–Service Applications for the Web User’s Guide)*

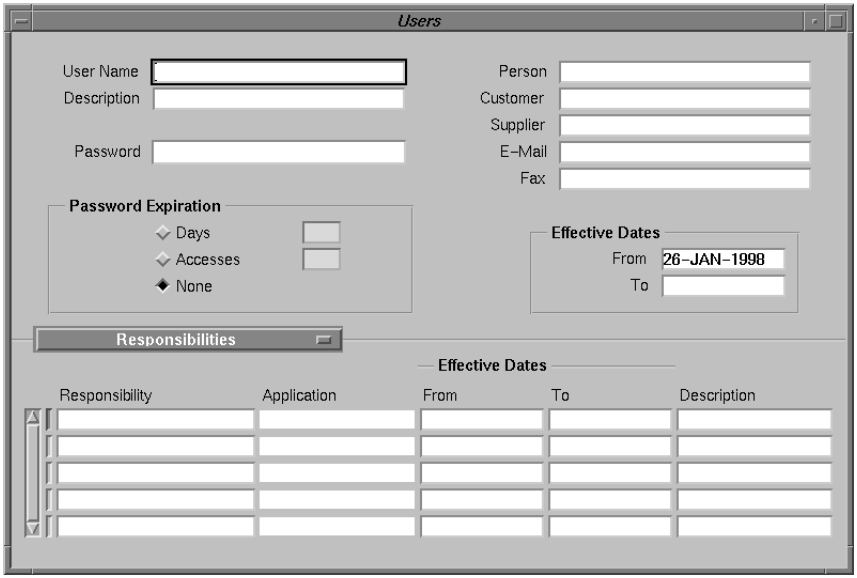
Securing Attributes

Use the List of Values to select valid attributes. You may assign any number of securing attributes to the responsibility.

Excluded Attributes

Use the List of Values to select valid attributes. You can assign any number of Excluded Attributes to a responsibility.

Users Window



Use this window to define an application user. An application user is an authorized user of Oracle Applications and / or Oracle Self–Service Applications who is uniquely identified by an application username. Once defined, a new application user can sign on to Oracle Applications and access data through Oracle Applications windows.
See: Overview of Oracle Applications Security: page 2 – 2.

Users Block

User Name

Enter the name of an application user. An application user enters this username to sign on to Oracle Applications.

- The username must not contain more than one word.
- You should use only alphanumeric characters ('A' through 'Z', and '0' through '9') in the username.

Please note that you must limit your username to the set of characters that your operating system supports for filenames.

Suggestion: We recommend that you define meaningful usernames, such as the employee's first initial followed by their

last name. Or, for a group account, you can define the application username so as to indicate the purpose or nature of the group account.

Password

Enter the initial password of an application user. An application user enters this password along with her or his username to sign on to Oracle Applications.

- A password must be at least five characters and can extend up to 100 characters.
- You should use alphanumeric characters ('A' through 'Z', and '0' through '9') in a password. All other characters are invalid.

This window does not display the password you enter. After you enter a password, you must re–enter it to ensure you did not make a typing error.

If the application user already exists and the two entries do not match, the original password is NOT changed, and you navigate automatically to the next field.

If you are defining a new application user and the two entries do not match, you are required to enter the password again. For a new user, you cannot navigate to the next field until the two entries match.

The first time an application user signs on, they must change his or her password. If a user forgets their password, you can reassign a new password in this field.

As System Administrator, you can set an initial password or change an existing password, but you cannot access the user's chosen password.

Person, Customer, and Supplier

Use these fields to enter the name of an employee (person), customer, or supplier contact. Enter the last name and first name, separated by a comma, of the employee, customer, or supplier who is using this application username and password. Use the List of Values to select a valid name.

E–Mail/Fax

Enter the E–mail address and / or fax number for this user.

Password Expiration

Days

Enter the maximum number of days between password changes. A pop–up window prompts an application user to change her or his password after the maximum number of days you specify has elapsed.

Accesses

Enter the maximum allowed number of sign–ons to Oracle Applications allowed between password changes. A pop–up window prompts an application user to change her or his password after the maximum number of accesses you specify has elapsed.

Suggestion: We recommend that you require application users to make frequent password changes. This reduces the likelihood of unauthorized access to Oracle Applications.

Effective Dates

From/To

The user cannot sign onto Oracle Applications before the start date and after the end date. The default for the start date is the current date. If you do not enter an end date, the username is valid indefinitely.

You cannot delete an application user from Oracle Applications because this information helps to provide an audit trail. You can deactivate an Oracle Applications user at any time by setting the End Date to the current date.

If you wish to reactivate a user, change the End Date to a date after the current date, or clear the End Date field.

Responsibilities Block

Responsibility

Select the name of a responsibility you wish to assign to this application user. A responsibility is uniquely identified by application name and responsibility name.

From/To

You cannot delete a responsibility because this information helps to provide an audit trail. You can deactivate a user's responsibility at any time by setting the End Date to the current date.

If you wish to reactivate the responsibility for the user, change the End Date to a date after the current date, or clear the End Date.

Securing Attributes

Securing attributes are used by Oracle Self–Service Web Applications to allow rows (records) of data to be visible to specified users or

responsibilities based on the specific data (attribute values) contained in the row.

You may assign one or more values for any of the securing attributes assigned to the user. If a securing attribute is assigned to both a responsibility and to a user, but the user does not have a value for that securing attribute, no information is returned for that attribute.

For example, to allow a user in the ADMIN responsibility to see rows containing a CUSTOMER_ID value of 1000, assign the securing attribute of CUSTOMER_ID to the ADMIN responsibility. Then give the user a security attribute CUSTOMER_ID value of 1000.

When the user logs into the Admin responsibility, the only customer data they have access to has a CUSTOMER_ID value of 1000.

Attribute

Select an attribute you want used to determine which records this user can access. You can select from any of the attributes assigned to the user's responsibility.

Value

Enter the value for the attribute you want used to determine which records this user can access.

Users of a Responsibility Report

This report documents who is using a given responsibility. Use this report when defining or editing application users.

Report Parameters

Application Name

Choose the name of the application to which the responsibility you want in your report belongs.

Responsibility Name

Choose the name of the responsibility you want in your report.

Report Heading

The report heading indicates the application name and responsibility for which you requested a report.

Column Headings

User Name

The name of the user who is assigned to the responsibility.

Start Date

The date the responsibility became active for the user.

End Date

The date the responsibility either becomes inactive or became inactive for the user. If no end date appears for a user, then this responsibility is always enabled for the user.

Description

The description of the user who is assigned to the responsibility.

Active Responsibilities Report

This report shows all the responsibilities that are currently active, the users who can currently access each responsibility, and the start and end dates when they can access the responsibility.

Report Parameters

None.

Report Heading

This displays the name of the report, the date and time the report was run, and the page number.

Column Headings

Application Name

The name of the application associated with the responsibility.

Responsibility Name

The name of the currently active responsibility.

User Name

The name of the user who can currently access the responsibility.

Start Date

The date when the user can begin accessing the responsibility.

End Date

The date when the user can no longer access the responsibility. See: Overview of Oracle Applications Security: page 2 – 2.

Active Users Report

This report shows all the usernames that are both currently active and have at least one active responsibility. It also displays all the responsibilities that users can access, and the start and end dates when they can access each responsibility.

Report Parameters

None.

Report Heading

The report heading displays the name of the report, the date that the report was run, and the page number.

Column Headings

User Name

The Oracle Applications name of the currently active user. The start and end dates that you specify in the Users window determine whether a username is currently active.

Application Name

The name of the application associated with the responsibility.

Responsibility Name

The name of the currently active responsibility.

Start Date

The date when the user can begin accessing the responsibility. You can specify a start date when you assign the responsibility to the user in the Responsibilities block of the Users window.

End Date

The date when the user can no longer access the responsibility. You specify an end date when you assign the responsibility to the user in Responsibilities block of the Users window.

Reports and Sets by Responsibility Report

This report identifies which reports (and other concurrent programs) and report sets are available to any given responsibility. Use this report when defining or editing responsibilities.

Report Parameters

If you enter no parameters, the report documents all reports and report sets accessible from each responsibility.

Application Short Name

Choose the application name associated with the responsibility whose available reports and report sets you wish to report on.

If you do not choose an application name, the report documents all reports and report sets accessible from each responsibility.

Responsibility Name

Choose the name of a responsibility whose available reports and report sets you wish to report on. You must enter a value for Application Short Name before entering a value for Responsibility Name.

Report Headings

The report headings list the report parameters you specify, and provide you with general information about the contents of the report.

Overview of Function Security

Function security is the mechanism by which user access to applications functionality is controlled.

Oracle Applications GUI–based architecture aggregates several related business functions into a single form. Because all users should not have access to every business function in a form, Oracle Applications provides the ability to identify pieces of applications logic as *functions*. When part of an application's functionality is identified as a function, it can be secured (i.e., included or excluded from a responsibility).

Application developers register functions when they develop forms. A System Administrator administers function security by creating responsibilities that include or exclude particular functions.

Terms

Function

A function is a part of an application's functionality that is registered under a unique name for the purpose of assigning it to, or excluding it from, a responsibility.

There are two types of functions: form functions, and non–form functions. For clarity, we refer to a form function as a *form*, and a non–form function as a *subfunction*, even though both are just instances of functions in the database.

Form (Form Function)

A form function (*form*) invokes an Oracle Forms form. Form functions have the unique property that you may navigate to them using the Navigate window.

Subfunction (Non–Form Function)

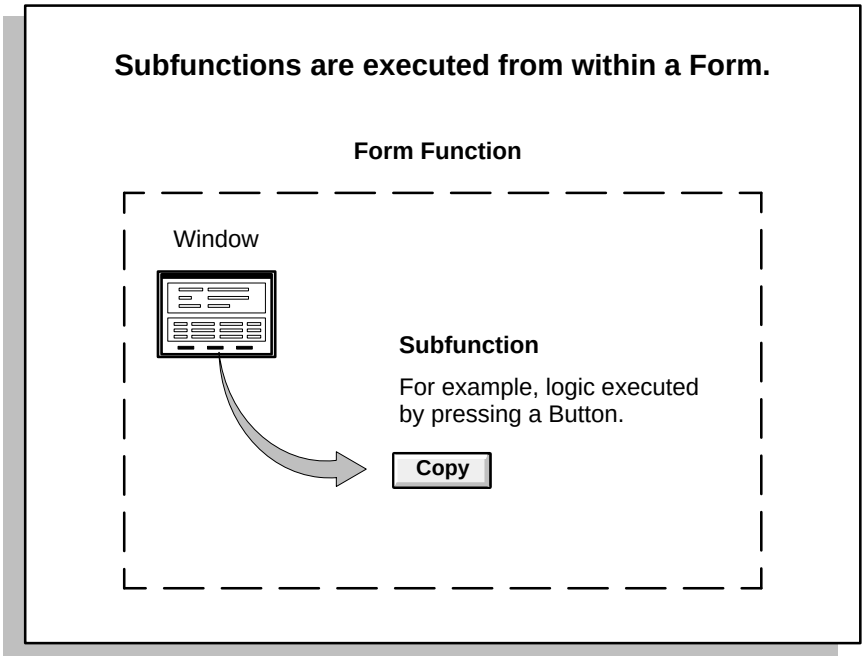
A non–form function (*subfunction*) is a securable subset of a form's functionality: in other words, a function executed from within a form.

A developer can write a form to test the availability of a particular subfunction, and then take some action based on whether the subfunction is available in the current responsibility.

Subfunctions are frequently associated with buttons or other graphical elements on forms. For example, when a subfunction is enabled, the corresponding button is enabled.

However, a subfunction may be tested and executed at any time during a form's operation, and it need not have an explicit user interface impact. For example, if a subfunction corresponds to a form procedure not associated with a graphical element, its availability is not obvious to the form's user.

Figure 2 – 1



Menu

A menu is a hierarchical arrangement of functions and menus of functions. Each responsibility has a menu assigned to it.

Menu Entry

A menu entry is a menu component that identifies a function or a menu of functions. In some cases, both a function and a menu of functions correspond to the same menu entry. For example, both a form and its menu of subfunctions can occupy the same menu entry.

Responsibility

A responsibility defines an application user's current privileges while working with Oracle Applications. When an application user signs on, they select a responsibility that grants certain privileges, specifically:

- The functions that the user may access. Functions are determined by the menu assigned to the responsibility.
- The concurrent programs, such as reports, that the user may run.
- The application database accounts that forms, concurrent programs, and reports connect to.

Forms and Subfunctions

A form is a special class of function that differs from a subfunction in two ways:

- Forms appear in the Navigate window and can be navigated to. Subfunctions do not appear in the Navigate window and cannot be navigated to.
- Forms can exist on their own. Subfunctions can only be called by logic embodied within a form; they cannot exist on their own.

A form as a whole, including all of its program logic, is always designated as a function. Subsets of a form's program logic can optionally be designated as subfunctions if there is a need to secure those subsets.

For example, suppose that a form contains three windows. The entire form is designated as a function that can be secured (included or excluded from a responsibility.) Each of the form's three windows can be also be designated as functions (subfunctions), which means they can be individually secured. Thus, while different responsibilities may include this form, certain of the form's windows may not be accessible from each of those responsibilities, depending on how function security rules are applied.

Functions, Menus, and the Navigate Window

Form functions or *forms* are selected using the Navigate window. The arrangement of form names in the Navigate window is defined by the menu structure assigned to the current responsibility.

The following types of menu entries are not displayed by the Navigate window:

- Subfunctions
- Menus without Entries

- Menu Entries without a Prompt

If none of the entries on a menu are displayed by the Navigate window, the menu itself is not displayed.

How Function Security Works

Developers Register Functions

- Developers can require parts of their Oracle Forms code to look up a unique *function name*, and then take some action based on whether the function is available in the current responsibility.
- Developers register functions. They can also register parameters that pass values to a function. For example, a form may support data entry only when a function parameter is passed to it.

Warning: In general, System Administrators should not modify parameters passed to functions that are predefined as part of the Oracle Applications products. The few cases where function parameters may be modified by a System Administrator are documented in the relevant technical reference manual or product update notes.

- Typically, developers define a menu including all the functions available in an application (i.e., all the forms and their securable subfunctions). For some applications, developers may define additional menus that restrict the application's functionality by omitting specific forms and subfunctions.
- When developers define menus of functions, they typically group the subfunctions of a form on a subfunction menu they associate with the form.

System Administrators Exclude Functions

- Each Oracle Applications product is delivered with one or more predefined menu hierarchies. System Administrators can assign a predefined menu hierarchy to a responsibility. To tailor a responsibility, System Administrators exclude functions or menus of functions from that responsibility using exclusion rules.
- If a System Administrators cannot create the desired menu by applying exclusion rules to a predefined menu, they can define a new menu hierarchy. In this case, we recommend that they construct their menu hierarchy using forms and their associated menus of subfunctions. In other words, System Administrators

should leave the developer–defined associations between forms and their menus in tact.

Available Functions Depend on the Current Responsibility

- When a user first selects or changes their responsibility, a list of functions obtained from the responsibility's menu structure is cached in memory.
- Functions a System Administrator has excluded from the current responsibility are marked as unavailable.
- Form functions in the function hierarchy (i.e., menu hierarchy) are displayed in the Navigate window. Available subfunctions are accessed by working with the application's forms.

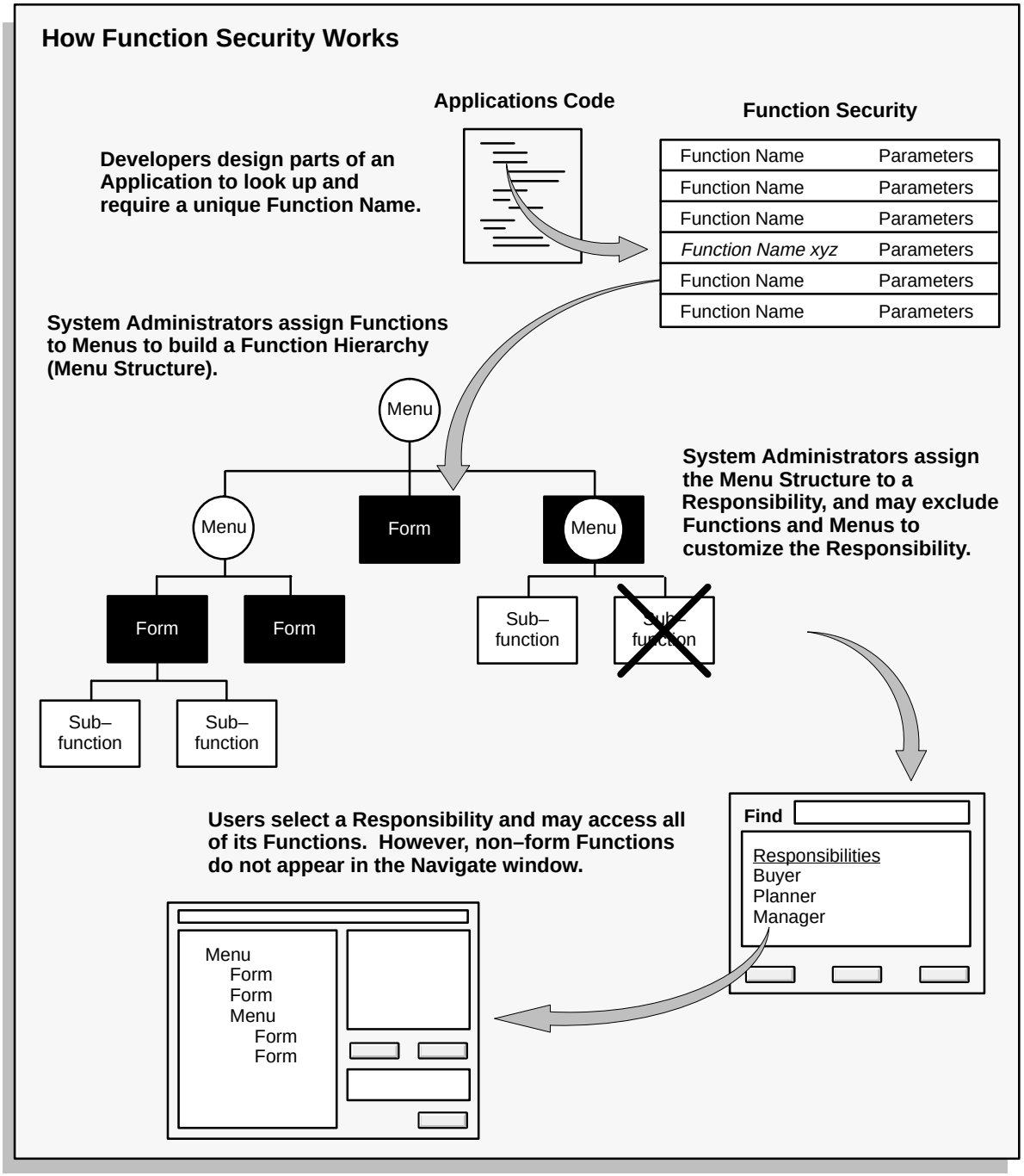
Visibility of Excluded Functions

Some subfunctions are associa ted with a graphical element, for example, a button, and their exclusion may result in:

- the dimming of the button
- the absence of the button.

Other subfunctions may not correspond to a graphical element, and their exclusion may not be obvious to an end user.

Figure 2 – 2


How Function Security Works
Applications Code
Function Security
Developers design parts of an Application to look up and require a unique Function Name.
Function Name
Parameters
Function Name
Parameters
Function Name
Parameters
Function Name xyz
Parameters
Function Name
Parameters
Function Name
Parameters
System Administrators assign Functions to Menus to build a Function Hierarchy (Menu Structure).
Menu
Menu
Form
Menu
System Administrators assign the Menu Structure to a Responsibility, and may exclude Functions and Menus to customize the Responsibility.
Form
Form
Sub–function
Sub–function
Sub–function
Sub–function
Find
Responsibilities
Buyer
Planner
Manager
Users select a Responsibility and may access all of its Functions. However, non–form Functions do not appear in the Navigate window.
Menu
Form
Form
Menu
Form
Form

A ”full access” responsibility with a menu that includes all the functions in an application is predefined for each Oracle Applications product. Some applications may provide additional predefined responsibilities that include a smaller set of functions (i.e., fewer forms and subfunctions).

As a System Administrator, you can restrict the functionality a responsibility provides by defining rules to exclude specific functions or menus of functions. In fact, we recommend that you use exclusion rules to customize a responsibility in preference to constructing a new menu hierarchy for that responsibility.

For example, suppose you want to customize a responsibility to restrict the functionality of a form included in that responsibility. First, you examine the predefined menus that group the subfunctions associated with that form. Then, using exclusion rules, you can restrict the form’s functionality by excluding certain of the form’s subfunctions from the responsibility.

If you cannot create the responsibility you need by applying exclusion rules, you may build a custom menu for that responsibility using predefined forms (i.e., form functions) and their associated menus of subfunctions. However, we recommend that you do not disassociate a form from its developer–defined menus of subfunctions.

Securing Functions Using Predefined Menus

Use the Responsibilities form to:

- Limit a predefined responsibility’s functionality by excluding menus and functions from it.
- Define a new responsibility and assign a predefined menu to it. Customize the new responsibility’s functionality by excluding menus and functions.
- By assigning the same menu hierarchy to different responsibilities and excluding different functions and menus, you can easily customize an application’s functionality.

Securing Functions Using New Menus

Use the Menus form to define menus pointing to functions that you want to make available to a new responsibility.

- Use forms and their associated menus of subfunctions to define new menus.

Assign the menu structure to a new responsibility using the Responsibilities form.

- For that responsibility, tailor a form's functionality by excluding particular subfunctions.
- By excluding a subfunction executed from within a form, the functionality of that form can be varied from one responsibility to another.
- By applying exclusion rules to the predefined menus of subfunctions associated with a form, you can easily customize a form's functionality.

Excluding Functions from a Responsibility

A system administrator may exclude functions or menus from the menu structure assigned to a responsibility.

- When a menu is excluded, all of its menu entries, that is, all the functions and menus of functions that it selects, are excluded.
- When you exclude a function from a responsibility, all occurrences of that function throughout the responsibility's menu structure are excluded.

Defining a New Menu Structure

When defining a new menu structure:

- Create a logical, hierarchical listing of functions. This allows for easy exclusion of functions when customizing the menu structure for different responsibilities.
- Create a logical, hierarchical menu that guides users to their application forms.

Tasks for Defining a Custom Menu Structure

- Determine the application functionality required for different job responsibilities.
- Identify predefined menus, forms, and form subfunctions to use as entries when defining a new menu. Understand predefined menus by printing Menu Reports using the Submit Requests window.

Suggestion: To simplify your work, use predefined menus for your menu entries. You can exclude individual functions after a menu structure is assigned to a responsibility.

- Plan your menu structure. Sketch out your menu designs.
- Define the lowest–level menus first. A menu must be defined before it can be selected as an entry on another menu.
- Assign menus and functions to higher–level menus.
- Assign menus and functions to a top–level menu (root menu).
- Document your menu structure by printing a Menu Report.

Warning: Start with a blank Menus form (blank screen). Menus cannot be copied. A menu saved under a different name overwrites the original menu (there is no “Save As” feature).

Notes About Defining Menus

Build Menus From Scratch

- Menus cannot be copied. Menu definitions cannot be saved under a different name (i.e., there is no “Save As” capability).
- When a menu name displays in the Menus form, be sure you are in Query mode before overwriting the menu’s name.

Define Menus for Fast and Easy Keyboard Use

- Design menu prompts with unique first letters, so typing the first letter automatically selects the form or menu
- Design the sequence of menu prompts with the most frequently used functions first (i.e., lower sequence numbers).
- Entries cannot be copied from one menu definition to another.

Note when Changing Menu Names or Modifying Entries

- When you change a menu’s name, the menu entries are not affected. The menu’s definition exists under the new name.
- Other menus calling the menu by its old menu name, automatically call the same menu by its new (revised) name.
- When defining menus or selecting a ”root” menu to assign to a responsibility, the old menu name is not in a list of values.
- When modifying a predefined menu, all other menus that call that menu display the menu’s modifications.
- For example, if you modify GL_TOP by adding another prompt that calls a form function, all menus that call GL_TOP will display the additional prompt when GL_TOP displays.

Preserving Custom Menus Across Upgrades

Preserve custom menus during upgrades of Oracle Applications by using unique names for your custom menus. For example, you can start the menu's name with the application short name of a custom application. Define a custom application named *Custom General Ledger*, whose application short name is CGL. Define your custom menu names to start with CGL, for example, CGL_MY_MENU.

Remember that the Oracle Applications standard menus may be overwritten with upgrade versions. Therefore, if you attached your custom menu as a submenu to one of the preseeded Oracle Applications menus, recreate the attachment to it following an upgrade. An alternative is to attach a standard Oracle Applications menu as a submenu to your custom mnu; the link from your custom menu to the standard menu should survive the upgrade.

Function Security Reports: page 2 – 39

Special Function for Oracle HRMS, Oracle Sales and Marketing

In most Oracle Applications products, you can open multiple forms from the Navigator window without closing the form you already have open. However, when you define a new responsibility whose custom menu accesses Oracle Sales and Marketing forms, or Oracle HRMS task flows, you must include the function *Disable Multiform, Multisession* as an entry on the responsibility's top–level menu.

You can identify an Oracle Sales and Marketing form by the OSM prefix contained in the form's function name.

In Oracle HRMS, a task flow is a method of linking windows so that you carry information from one window to the next, in sequence, to complete a task. You can identify an Oracle HRMS form that may be part of a task flow by the PER or PAY prefix in the form's function name. For details on administering Oracle HRMS task flows, and on determining whether a form is part of a task flow, see: Linking Windows in Task Flows, *Oracle Human Resources User's Guide*.

Attention: You should not include the *Disable Multiform, Multisession* function on menus that do not include either Oracle Sales and Marketing or Oracle HRMS forms.

To include the *Disable Multiform, Multisession* function on a menu:

- Add a Function menu entry to the top–level menu (i.e., the menu referenced by your new responsibility).

- Select the function whose User Function Name and Function Name are:
  - Disable Multiform, Multisession
  - FND_FNDSCSGN_DISABLE_MULTIFORM
- Save your changes.

Summary of Function Security

Functions:

- A function is a set of code in Oracle Applications that is executed only if the name of the function is present in a list maintained on a responsibility–by–responsibility basis.
- Functions can be excluded from a responsibility by a System Administrator.
- There are two types of function: a form function or *form*, and a non–form function or *subfunction*. A subfunction represents a securable subset of a form's functionality.

Form Functions:

- A function that invokes a form.
- Form functions appear in the Navigate window and can be navigated to.

Subfunctions:

- A function that is executed from within a form. Subfunctions can only be called by logic embodied within a Form Function.
- Subfunctions do not appear in the Navigate window and cannot be navigated to.

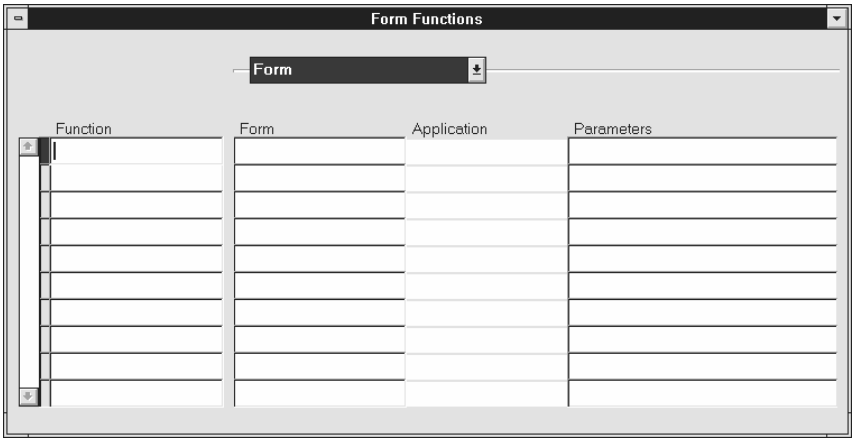
Menus:

- Menus contain menu entries which point to a function, another menu, or a function *and* another menu.
- Menus appear in the Navigate window.
- Menus can be excluded from a responsibility by a System Administrator.

See Also

Function Security Reports: page 2 – 39

Form Functions Window



Define new functions. A function is a part of an application's functionality that is registered under a unique name for the purpose of assigning it to, or excluding it from, a responsibility.

There are two types of functions: form functions, and non–form functions.

For clarity, we refer to a form function as a *form*, and a non–form function as a *subfunction*, even though both are just instances of functions in the database.

Form Functions Block

Function

Users do not see this unique function name. However, you may use this name when calling your function programmatically. You should follow the naming conventions for functions.

User Function Name

Enter a unique name that describes your function. You see this name when assigning functions to menus. This name appears in the Top Ten List of the Navigator window.

Type

Type is a free-form description of the function's use. A function's type is passed back when a developer tests the availability of a function.

The developer can write code that takes an action based on the function's type.

By convention, Oracle Applications form functions are registered with a type of FORM. A few, specialized functions that determine common form behaviors are registered with a type of UTIL.

Even if you do not register a form function with a type of FORM, Oracle Applications treats it as a form if you specify a valid Form Name / Application.

Form

Form /Application

If you are defining a form function, select the name and application of your form.

Parameters

Enter the parameters you wish to pass to your function. Separate parameters with a space.

For a form function, if you specify the parameter QUERY_ONLY=YES, the form opens in query–only mode. Oracle Application Object Library removes this parameter from the list of form parameters before opening the form in query–only mode.

You can also specify a differnt form name to use when searching for help for a form in the appropriate help file. The syntax to use is:

HELP_TARGET = "*alternative_form_name*"

Your form name overrides the name of the form. See: Help Targets in Oracle Applications: page 4 – 5.

Some Oracle Applications forms are coded to accept particular form parameters. For example, the Run Requests form accepts a TITLE parameter you can use to change the Run Requests window title. The syntax you should use is:

TITLE="*appl_short_name*:*message_name*"

where *appl_shortname*:*message_name* is the name of a Message Dictionary message. See: Customizing the Submit Requests Window using Codes: page 6 – 18.

Warning: In general, System Administrators should not modify parameters passed to functions that are predefined as part of the Oracle Applications products. The few cases where

function parameters may be modified by a System Administrator are documented in the relevant technical reference manual or product update notes.

Web Regions

The fields in the Web regions are only required if your function will be accessed from Oracle Self–Service Web Applications. You do not need to enter any of these fields for SmartClient and Web–deployed Applications functions.

Host Name

The URL (universal resource locator) or address required for your function consists of three sections: the Host Name, Agent Name, and the HTML Call. The Host name is the IP address or alias of the machine where the Webserver is running.

Agent Name

The second section of your functions URL is the Oracle Web Agent. The Oracle Web Agent determines which database is used when running your function. Defaults to the last agent used.

HTML Call

The last section of your functions URL is the HTML Call. The HTML Call is used to activate your function. The function may be either a static web page or a procedure.

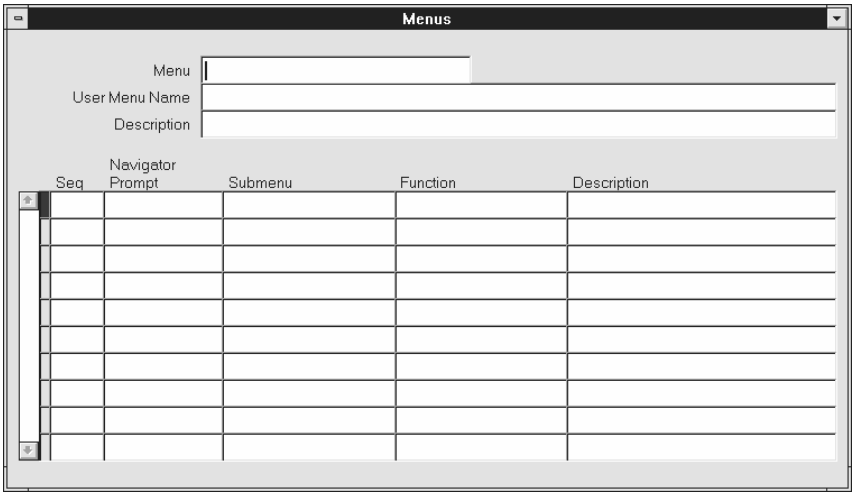
Secured

Secured is only required when your function is accessed by Oracle Workflow. Checking Secured enables recipients of a workflow E–Mail notification to respond using E–Mail.

Encrypt Parameters

Checking Encrypt Parameters adds a layer of security to your function to ensure that a user cannot access your function by altering the URL in their browser window. You must define Encrypt Parameters when you define your function to take advantage of this feature.

Menus Window



Define a new menu or modify an existing menu.

A menu is a hierarchical arrangement of functions and menus of functions. Each responsibility has a menu assigned to it.

A ”full access” responsibility with a menu that includes all the functions in an application is predefined for each Oracle Applications product. As a System Administrator, you can restrict the functionality a responsibility provides by defining rules to exclude specific functions or menus of functions. In fact, we recommend that you use exclusion rules to customize a responsibility in preference to constructing a new menu hierarchy for that responsibility.

If you cannot create the responsibility you need by applying exclusion rules, you may build a custom menu for that responsibility using predefined forms (i.e., form functions) and their associated menus of subfunctions. However, we recommend that you do not disassociate a form from its developer–defined menus of subfunctions.

See:

Overview of Function Security: page 2 – 22

Implementing Function Security: page 2 – 28

Prerequisites

- Register your application with Oracle Application Object Library using the Applications window.

- Define any menus that you intend to call from your menu. Define the lowest–level submenus first. A submenu must be defined before it can be called by another menu.

Suggestion: By calling submenus from your menu, you can group related windows together under a single heading on your menu. You can reuse your menu on other menus.

Menus Block

Menu entries detail the options available from your menu.

Menu

Choose a name that describes the purpose of the menu. Users do not see this menu name.

User Menu Name

You use the user menu name when a responsibility calls a menu or when one menu calls another.

Menu Entries Block

Sequence

Enter a sequence number to specify where a menu entry appears relative to other menu entries in a menu. The default value for this field is the next whole sequence number.

A menu entry with a lower sequence number appears before a menu entry with a higher sequence number.

Attention: If you change sequence numbers or frequently insert and delete menu entries, carefully check the default value. This value may be a duplicate sequence number or an out of sequence number.

Suggestion: You cannot replace a menu entry sequence number with another sequence number that already exists. If you want to add menu entries to a menu entry sequence that uses only integers, carefully renumber your menu entries to a sequence range well outside the sequence range you want, ensuring that you do not use existing sequence numbers

Once you save this work, you can go back and renumber each entry to have the final sequence number you want.

Navigator Prompt

Enter a user–friendly, intuitive prompt your menu displays for this menu entry. You see this menu prompt in the hierarchy list of the Navigator window.

Suggestion: Enter menu prompts that have unique first letters so that power users can type the first letter of the menu prompt to choose a menu entry.

Submenu

Call another menu and allow your user to select menu entries from that menu.

Function

Call a function you wish to include in the menu. A form function (form) appears in the Navigate window and allows access to that form. Other non–form functions (subfunctions) allow access to a particular subset of form functionality from this menu.

Description

Descriptions appear in a field at the top of the Navigate window when a menu entry is highlighted.

Function Security Reports: page 2 – 39

Function Security Reports

Use the function security reports to document the structure of your 10SC menus. You can use these reports as hardcopy to document your customized menu structures before upgrading your Oracle Applications software.

The function security reports consist of the Function Security Functions Report, the Function Security Menu Report, and the Function Security Navigator Report.

These reports are available through the Function Security Menu Reports request set. For each report, specify the responsibility whose function security you want to review.

Function Security Function Report

Specify a responsibility when submitting the report. The report output lists the functions accessible by the specified responsibility.

The report does not include items excluded by function security rules.

Function Security Menu Report

Specify a responsibility when submitting the report. The report output lists the complete menu of the responsibility, including all submenus and functions.

The report indicates any excluded menu items with the rule that excluded it.

Function Security Navigator Report

Specify a responsibility when submitting the report. The report output lists the menu as it appears in the navigator for the responsibility specified.

This reports does not include items excluded by function security rules, or non–form functions that do not appear in the navigator.

Menu Report

This report documents the structure of your charcter mode menus. Use this report when defining new menus or editing existing menus. You can also use this report as hardcopy to document your customized menu structures before upgrading your Oracle Applications software.

Attention: The Menu Report only details menus created and used with the Oracle Applications running in character mode. Use the Function Security menu reports to detail menus created with Oracle Applications 10SC.

The Menu Report prints:

- a hierarchical listing of the menu entries for the menu you request
- the action each menu entry invokes
- the name of the application that contains the menu
- the name of the action the menu calls

Report Parameters

Main Menus Only

Select Yes or No to indicate whether the menu you want in your report is a main menu. If you choose Yes, only main menus appear as selections in the window for the Menu Name parameter.

Application Name

Choose the name of the application to which the menu you want in your report belongs.

Menu Name

Choose the name of the menu you want in your report.

Report Heading

The report heading contains the menu's application and name.

Column Headings

Prompt

The prompt your menu displays for this menu entry.

Description

The description of this menu entry.

Action Type

The type of action this menu entry invokes when selected. Valid values are:

Menu	Navigate to a submenu.
Function	Navigate to a function

Application

The name of the application associated with the menu or function that your menu calls.

Action

The name of the menu or function.

CHAPTER

3 User and Data Auditing

This chapter explains how to audit your application users and the changes they effect on your application's data.

Overview of User and Data Auditing

There are two types of auditing in Oracle Applications: auditing users, and auditing database row changes.

Auditing User Activity

Auditing users is supported by:

- Sign–On:Audit Level profile option setting
- Audit Reports

See:

Signon Audit Concurrent Requests: page 3 – 11

Signon Audit Forms: page 3 – 13

Signon Audit Responsibilities: page 3 – 16

Signon Audit Unsuccessful Logins: page 3 – 18

Signon Audit Users: page 3 – 20

Sign–On Audit lets you track who signs on to your application and what they do.

Based on the audit level you choose, Sign–On Audit records usernames, dates and times of users accessing the system, as well as what responsibilities, forms, and terminals users are using.

Auditing Database Row Changes

Auditing database row changes is supported by:

- From the **Help** menu, **About This Record** ...
- AuditTrail:Activate profile option setting
- Audit forms – see below.

See:

Reporting on AuditTrail Data: page 3 – 22

Audit Installations: page 3 – 34

Auditing User Activity

Oracle Applications provides a Sign–On Audit feature that allows you to:

- Track what your users are doing and when they do it.
- Choose who to audit and what type of information to audit.
- View quickly online what your users are doing.
- Check the security of your application.

With Sign–On Audit, you can record usernames, terminals, and the dates and times your users access Oracle Applications. Sign–On Audit can also track the responsibilities and forms your users use, as well as the concurrent processes they run.

Major Features

Selective Auditing

Sign–On Audit lets you choose who to audit and what type of user information to track. You can selectively determine what audit information you need, to match your organization's needs.

Monitor Application Users

The Monitor Users form gives you online, real–time information about who is using Oracle Applications and what they are doing.

You can see what users are signed on (application username and operating system login name), what responsibilities, forms, and terminals they are using, how long they have been working on forms, and what ORACLE processes they are using.

Sign–On Audit Reports

Sign–On Audit Reports give you historical, detailed information on what your users do in your application.

You can give search criteria to narrow your search for information.

You can also sort your Sign–On Audit information to create easy–to–read reports.

Setting Up Sign–On Audit

You use the Sign–On:Audit Level user profile option to control who Sign–On Audit tracks and the level at which they are audited.

Use the Monitor Users form to view online what your users are doing.

Use the Submit Reports form to submit Sign–On Audit Reports that give you detailed audit information.

Enabling Sign–On Audit

Use the System Profile Values form to enable Sign–On Audit. Choose the scope of your audit and who to audit by setting the user profile level at the user, responsibility, application, or site profile levels.

Users cannot see nor change this profile option.

After you set or change audit levels, the new audit levels for a user take effect the next time the user signs onto Oracle Applications from the operating system.

Selecting Audit Levels

The Sign–On:Audit Level profile option allows you to select a level at which to audit users who sign on to Oracle Applications. Four audit levels increase in functionality: None, User, Responsibility, and Form.

None is the default value, and means do not audit any users who sign on to Oracle Applications.

Auditing at the User level tracks:

- who signs on to your system
- the times users log on and off
- the terminals in use

Auditing at the Responsibility level performs the User level audit functions and tracks:

- the responsibilities users choose
- how much time users spend using each responsibility

Auditing at the Form level performs the Responsibility level audit functions and tracks:

- the forms users choose
- how long users spend using each form

Factoring in System Overhead

In planning your organization's Sign–On Audit implementation, you should consider the additional system overhead required to precisely

monitor and audit your users as they access Oracle Applications. The more users you audit and the higher the level of auditing, the greater the likelihood of incurring additional system overhead.

Example – Audit Users, Responsibilities, & Forms

One example implementation of Sign–On Audit is to audit all of your users' sign–ons, the responsibilities they select, and the forms they access.

To set up this implementation, set "Sign–On:Audit Level" to:

- Form audit
- At the Site profile level

Example – Audit a specific responsibility, excepting one user

Another example of using Sign–On Audit is for an organization to audit all users of the Personnel Manager responsibility, except for MJONES.

In this example, you do not care to audit the forms the users access or the responsibilities they select.

To set up this implementation, set "Sign–On:Audit Level" to:

- User audit
- At the responsibility profile level for the Personnel Manager responsibility

You also set "Sign–On:Audit Level" to:

- None
- At the user profile level for the application user MJONES

Using the Application Monitor

Use the Monitor Users form to monitor who is using Oracle Applications and what they are doing. You can monitor your users at any time.

The Application Monitor lets you see what users are signed on, what responsibilities, forms, and terminals they are using, how long they have been working on forms, and what ORACLE processes they are using.

Attention: You can only monitor those users that are being audited by Sign–On Audit. The Application Monitor also reflects the level of auditing you define for your users.

About This Record Window

You can display Sign–On Audit data by choosing from the **Help** menu, **About This Record**...

Sign–On Audit can automatically tie in ”About This Record” information for records that are inserted or updated by audited users. This additional information appears in the ”About This Record” window when you set the Who:Display Type profile option to Extended.

Extended information shows the Oracle Applications session number, the operating system login name, and the terminal that a user you are tracking with Sign–On Audit used to insert or update a row.

As System Administrator, you can use the System Profile Values form to set “Who:Display Type” to let any user, responsibility, application, or site view Extended ”About This Record” information.

Who: Display Type Profile Option

The Who: Display Type profile option allows you to choose between two different displays in the About This Record window:

”Normal” displays the:

- name of the user who created the row
- date the user created the row
- name of the table containing the row
- name of the user who last updated the row

”Extended” displays Normal information, plus:

- the user’s operating system logon
- the user’s terminal identification

Users cannot see nor change this profile option.

This profile option is visible and updatable at all four levels.

Notifying of Unsuccessful Logins

Sign–On Audit can track user logins and provide users with a warning message if anyone has made an unsuccessful attempt to sign on with their application username since their last sign–on. This warning message appears after a user signs on.

You or your users can activate this feature using the Personal Profile Values form by setting the ”Sign–On:Notification” user profile option to Yes.

You do not have to audit the user with Sign–On Audit to use this notification feature.

Sign–On Audit Reports

Use the Submit Requests form to print standard audit reports.

You can generate reports detailing what users are signing on, what responsibilities they are accessing, what forms they are using, what concurrent requests they are submitting, and who is attempting to log on to other users' accounts.

Oracle Applications provide the following Sign–On Audit reports:

Signon Audit Concurrent Requests: page 3 – 11 (shows who submitted what requests)

Signon Audit Forms: page 3 – 13 (shows who accessed what forms)

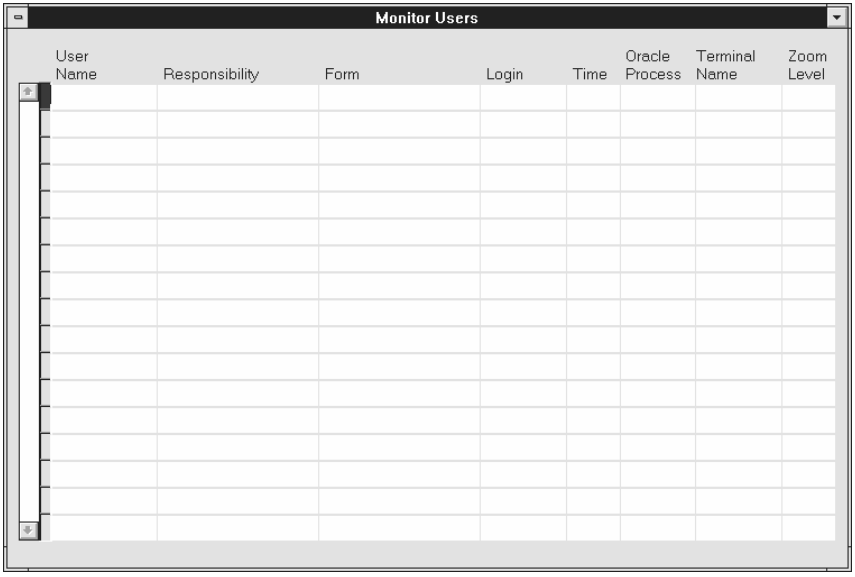
Signon Audit Responsibilities: page 3 – 16 (shows who accessed what responsibilities)

Signon Audit Unsuccessful Logins: page 3 – 18 (shows who unsuccessfully attempted to sign on as another user)

Signon Audit Users: page 3 – 20 (shows who signed on to Oracle Applications)

For each report, you can also specify search criteria that makes your report as brief as you need.

Monitor Users Window



Use this window to monitor what your application users are currently doing. You can see which users are signed on and what responsibilities, forms (windows), and terminals they are using.

You can also see how long they have been logged in and what ORACLE processes they are using.

In addition, you can monitor all users at a site, all users accessing a specific application or a specific responsibility, or you can monitor individual users. You can only monitor those users for whom you have activated Sign–On Audit. See: Overview of User and Data Auditing: page 3 – 2.

Prerequisites

- Select a value for the Sign–On:Audit Level profile option, using the Update System Profile Options window.

Monitor Users Block

Responsibility

The user's responsibility only appears if you have enabled Sign–On Audit at either the Responsibility or Form audit level.

Form

The user's form only appears if you have enabled Sign–On Audit at the Form audit level.

Zoom Level

The number of zoom levels the user used to get to the form shown.

Terminal

The name of the terminal that the user is working on.

ORACLE Process

The ORACLE process of the user.

Time

The length of time the user has been logged on to this application.

Login

The user's login name.

Signon Audit Concurrent Requests Report

Use this report to view information about who is requesting what concurrent requests and from which responsibilities and forms.

Attention: You can only generate Signon Audit Concurrent Requests Reports for those users you are auditing.

Report Parameters

Sort By

Sort the information in your report by operating system login name, the requested start date, and / or application username.

Login Name

Search for a specific login name that meets your other search criteria. If you leave this parameter blank, your report contains all login names that meet your other search criteria.

User Name

Search for a specific application username that meets your other search criteria. If you leave this parameter blank, your report contains all application usernames that meet your other search criteria.

From Request Start Time/To Request Start Time

Search for concurrent requests that meet your other search criteria and have requested start times in a specific time period. Use these parameters to specify the start and end of your time period with the format DD–MON–YY. If you leave these parameters blank, your report contains concurrent requests from any date that also meet your other search criteria to the current date for this parameter.

Report Heading

The report heading displays the search criteria you entered as parameter values.

Column Headings

Login Name

The operating system login name of the user who submitted the concurrent request.

Request ID

The concurrent request ID of the submitted concurrent request. Use the Concurrent Requests form to view completion information for a concurrent request ID.

Concurrent Program Name

The name of the concurrent program the user submitted. Use the Concurrent Programs form to view detail information about a concurrent program.

User Name

The Oracle Applications username of the user who submitted the concurrent request. Use the Users form to view detail information about an application user. See: Users: page 2 – 14.

Responsibility Name

The name of the responsibility from which the user submitted the concurrent request. The responsibility displays only if you audited the user at the responsibility or form Sign–on Audit level. Use the Responsibilities form to view detailed information about a responsibility. See: Responsibilities: page 2 – 9.

Form Name

The name of the form from which the user submitted the concurrent request. The form name displays only if you audited the user at the form Sign–On Audit level.

Requested Start Time

The date and time the concurrent request started running. The format is DD–MON–YY HH24:MM.

Signon Audit Forms Report

Use this report to view who is navigating to what form and when they do it.

Attention: You can only generate a Signon Audit Forms Report for those users you are auditing.

Report Parameters

Sort By

Sort the information in your report by the time users entered or left a form, the name of the form that users access, the operating system login name of the user, the responsibility users access, the terminal that users are on, and / or the application username.

Login Name

Search for information about a specific login name that meets your other search criteria. If you leave this parameter blank, your report contains all login names that meet your other search criteria.

User Name

Search for information about a specific application username that meets your other search criteria. If you leave this parameter blank, your report contains all application usernames that meet your other search criteria.

Terminal Name

Search for information about a specific terminal that meets your other search criteria. If you leave this parameter blank, your report contains all terminal names that meet your other search criteria.

Responsibility Name

Search for information about a specific responsibility that meets your other search criteria. If you leave this parameter blank, your report contains all responsibilities that meet your other search criteria.

Form Name

Search for information about a specific form that meets your other search criteria. If you leave this parameter blank, your report contains all forms that also meet your other search criteria.

From Active Date/To Active Date

Search for information about forms accessed by users within a specific time period and that meet your other search criteria. Use these parameters to specify the start and end of your time period with the format DD–MON–YY. If you leave these parameters blank, your report contains forms accessed from any date that also meet your other search criteria to the current date for this parameter.

Report Heading

The report heading displays the search criteria you entered as parameter values.

Column Headings

Username

The Oracle Applications username of the user who accessed the form. Use the Users form to view detailed information about an application user. See: Users: page 2 – 14.

Login Name

The operating system login name of the user who accessed the form.

Terminal Name

The operating system ID of the terminal from which the user accessed the form.

Responsibility Name

The name of the responsibility from which the user accessed the form. The responsibility displays only if you audited the user at the responsibility or form Sign–on Audit level. Use the Responsibilities form to view detailed information about a responsibility. See: Responsibilities: page 2 – 9.

Start Active Time/End Active Time

The dates and times when the user accessed / exited the form. The start active time and end active time display only if you audited the user at the form Sign–on Audit level.

Form Name

The name of the form that the user accessed. The form name displays only if you audited the user at the form Sign–on Audit level.

Signon Audit Responsibilities Report

Use this report to view who is selecting what responsibility and when they do it.

Attention: You can only generate Signon Audit Responsibilities Reports for those users you are auditing.

Report Parameters

Sort By

Sort the information in your report by the time users entered or left a responsibility, the operating system login name of the user, the responsibility name, the terminal that users are on, and/or the application username.

Login Name

Search for information about a specific login name that meets your other search criteria. If you leave this parameter blank, your report contains all login names that meet your other search criteria.

User Name

Search for information about a specific application username that meets your other search criteria. If you leave this parameter blank, your report contains all application usernames that meet your other search criteria.

Terminal Name

Search for information about a specific terminal that meets your other search criteria. If you leave this parameter blank, your report contains all terminal names that meet your other search criteria.

Responsibility Name

Search for information about a specific responsibility that meets your other search criteria. If you leave this parameter blank, your report contains all responsibilities that meet your other search criteria.

From Active Date/To Active Date

Search for information about responsibilities accessed by users within a specific time period and that meet your other search criteria. Use these parameters to specify the start and end of your time period with the format DD–MON–YY. If you leave these parameters blank, your report contains responsibilities accessed from any date that also meet your other search criteria to the current date for this parameter.

Report Heading

The report heading displays the search criteria you entered as parameter values.

Column Headings

Username

The Oracle Applications username of the user who selected the form. Use the Users form to view detail information about an application user. See: Users: page 2 – 14.

Login Name

The operating system login name of the user who selected the responsibility.

Terminal Name

The operating system ID of the terminal from which the user selected the responsibility.

Responsibility Name

The name of the responsibility the user used. The responsibility displays only if you audited the user at the responsibility or form Sign–on Audit level. Use the Responsibilities form to view detailed information about a responsibility. See: Responsibilities: page 2 – 9.

Start Active Time/End Active Time

The dates and times when the user selected / exited the responsibility. The start active time and end active time display only if you audited the user at the responsibility or form Sign–On Audit level.

Signon Audit Unsuccessful Logins Report

Use this report to view who unsuccessfully attempted to sign on to Oracle Applications as another user. An unsuccessful login occurs when a user enters a correct username but an incorrect password.

You can generate Signon Audit Unsuccessful Logins Reports for any users, regardless of whom you are auditing.

Report Parameters

Sort By

Sort the information in your report by the time users attempt to login, operating system login name of the user, the terminal that users are on, and / or the application username.

Login Name

Search for information about a specific login name that meets your other search criteria. If you leave this parameter blank, your report contains all login names that meet your other search criteria.

User Name

Search for information about a specific application username that meets your other search criteria. If you leave this parameter blank, your report contains all application usernames that meet your other search criteria.

Terminal Name

Search for information about a specific terminal that meets your other search criteria to make your report as brief as you need. If you leave this parameter blank, your report contains all terminal names that meet your other search criteria.

From Attempt Date/To Attempt Date

Search for information about unsuccessful logins within a specific time period and that meet your other search criteria. Use these parameters to specify the start and end of your time period with the format DD–MON–YY. If you leave these parameters blank, your report contains unsuccessful logins from any date that also meet your other search criteria to the current date for this parameter.

Report Heading

The report heading displays the search criteria you entered as parameter values.

Column Headings

Username

The Oracle Applications username of the user who unsuccessfully signed on. Use the Users form to view detail information about an application user. See: Users: page 2 – 14.

Login Name

The operating system login name of the user who unsuccessfully tried to sign on.

Terminal

The operating system ID of the terminal from which the user unsuccessfully tried to sign on.

Attempt Time

The date and time when the user unsuccessfully tried to sign on. See: Monitor Users field help: page 3 – 9.

Signon Audit Users Report

Use this report to view who signs on and for how long.

Attention: You can only generate Signon Audit Users Reports for those users you are auditing.

Report Parameters

Sort By

Sort the information in your report by the time users start or finish using an application username, the operating system login name of the user, the terminal that users are on, and/or the application username.

Login Name

Search for information about a specific login name that meets your other search criteria to make your report as brief as you need. If you leave this parameter blank, your report contains all login names that meet your other search criteria.

User Name

Search for information about a specific application username that meets your other search criteria to make your report as brief as you need. If you leave this parameter blank, your report contains all application usernames that meet your other search criteria.

Terminal Name

Search for information about a specific terminal that meets your other search criteria to make your report as brief as you need. If you leave this parameter blank, your report contains all terminal names that meet your other search criteria.

From Active Date/To Active Date

You can search for information about users logged into Oracle Applications within a specific time period and that meet your other search criteria. Use these parameters to specify the start and end of your time period with the format DD–MON–YY. If you leave these parameters blank, your report contains user information from the first date that also meets your other search criteria to the current date.

Report Heading

The report heading displays the search criteria you entered as parameter values.

Column Headings

Session Number

The Oracle Applications session number which uniquely identifies each application user sign–on.

User Name

The Oracle Applications username of the user who signed on. Use the Users form to view detailed information about an application user.
See: Users: page 2 – 14.

Login Name

The operating system login name of the user who signed on.

Terminal Name

The operating system ID of the terminal from which the user signed on.

Start Active Time/End Active Time

The dates and times when the user signed / exited onto Oracle Applications. The start active time and end active time display only if you audited the user at the user Sign–On Audit level.

ORACLE Process

The ORACLE Process ID used during the user's sign–on. Consult your Database Administrator for more information concerning ORACLE Processes.

System Process

The operating system process ID used during the user's sign–on. Consult your operating system administrator for more information concerning your operating system process ID.

Reporting On AuditTrail Data

AuditTrail lets you keep a history of changes to your important data: what changed, who changed it, and when. With AuditTrail, you can easily determine how any data row or element obtained its current value. You can track information on most types of fields, including character, number and date fields.

When you enter or update data in your forms, you change the database tables underlying those forms. AuditTrail tracks which rows in the database were updated at what time, and which user was logged in using the associated form(s).

AuditTrail

Oracle Applications Releases 10.4 and above provide a mechanism based on Oracle database triggers. AuditTrail stores change information in a ”shadow table” of the audited table. This mechanism saves audit data in an uncompressed but ”sparse” format, and you enable auditing for particular tables and groups of tables (”audit groups”).

Setting Up Release 11 AuditTrail

You can choose to store and retrieve a history of all changes users make on a given table. Auditing is accomplished using *audit groups*, which functionally group tables to be audited. For a table to be audited, it must be included in an enabled audit group.

The steps for setting up AuditTrail include:

Verify Select Privileges on SYS.DBA_TABLES

Have your database administrator grant SELECT privileges on SYS.DBA_TABLES to the APPLSYS account. Normally, this step would already have been done as part of your installation or upgrade.

Define Audit Groups

These are groups of tables and columns, where you do not necessarily need to include all the columns in a given table. You enable auditing for audit groups rather than for individual tables. You would typically group together those tables that belong to the same business process (for example, purchase order tables).

A given table can belong to more than one audit group. If so, the table is audited according to the highest "state" of enabling for any of its groups, where Enabled is the highest, followed by Disable Dump Data, Disable No Growth, and Disable Purge Table, in that order.

You can enable auditing for a maximum of 240 columns for a given table, and you can enable auditing for all types of table columns except LONG, RAW, or LONG RAW. Your audit group must include all columns that make up the primary key for a table; these columns are added to your audit group automatically. Once you have added a column to an audit group, you cannot remove it. See: Audit Groups: page 3 – 35.

Define Audit Installations

You choose the registered Oracle IDs at your site that you want to audit. This allows you to audit across multiple application installations. When a table is added to an audit group, auditing will automatically be enabled for all installations of the table for which audit is enabled. See: Audit Installations: page 3 – 34.

Run the Audit Trail Update Tables Report to Enable Auditing

Your AuditTrail definitions (and auditing) do not take effect until you run the Audit Trail Update Tables Report. If you change any of your definitions later, you must rerun this program. You run the Audit Trail Update Tables Report from the standard submission (Submit Reports) form.

Attention: AuditTrail requires two database connections. If your operating platform does not automatically support two database connections (e.g., VMS or MPE/XL), then add to your environment file the environment variable FDATDB=<database connect string>.

Audit Trail Update Tables Report

This program creates database triggers on the tables in your audit groups for your installations. It also creates shadow tables, one for each audited table, to contain the audit information. If you have changed your audit definitions or disabled auditing for an audit group, the program drops or modifies the auditing triggers and shadow tables appropriately.

The program also builds special views you can use to retrieve your audit data for reporting.

When auditing is enabled for the first time, a shadow table to the audited table is automatically created in the same Oracle ID as the audited table. The shadow table contains only the columns to be audited, and all columns in the shadow table are unconstrained, regardless of their status in the table to be audited.

For example, NULLs are always permitted in the shadow table. All columns in the shadow table have the same data types and sizes as their counterparts in the audited table.

The name of the shadow table is the first 26 characters of the original table name plus the suffix "_A" (Audit).

Shadow Table Columns

All AuditTrail shadow tables contain certain special auditing columns. These columns include:

- AUDIT_USER_NAME (the Application User ID, except when changes are applied using SQL*Plus, in which case it is the Oracle ID)
- AUDIT_TIMESTAMP (the date/time when the insertion occurred)
- AUDIT_TRANSACTION_TYPE (I for Insert, U for Update, D for Delete, L for Last, and C for Current)
- AUDIT_TRUE_NULLS (VARCHAR2(250) column containing a delimited list of column names that have changed from NULL)
- The Primary Key for the table. This is not a special column, but rather all the columns composing the primary key of the audited table. Note that, by convention, all audited columns are stored when a row is deleted. Likewise, an insert results in a row of NULL values in the shadow table. Changes to the primary key are marked as deletes, but new primary key values are inserted also.

For example, suppose you have the following table:

```
SQL> DESCRIBE AUDIT_DEMO

 NAME                           NULL?    TYPE
 ------------------------------ -------- ----
 PRIMARY_KEY                             NUMBER(5)
 VALUE_ONE                               VARCHAR2(5)
 VALUE_TWO                               VARCHAR2(5)
```

```
VALUE_THREE                                       VARCHAR2(5)
```

Its shadow table is as the following (assuming you audit all of your table columns):

```
SQL> DESCRIBE AUDIT_DEMO_A

 NAME                              NULL?    TYPE
 --------------------------------- -------- ----
 AUDIT_TIMESTAMP                   NOT NULL DATE
 AUDIT_TRANSACTION_TYPE            NOT NULL VARCHAR2(1)
 AUDIT_USER_NAME                   NOT NULL VARCHAR2(100)
 AUDIT_TRUE_NULLS                           VARCHAR2(250)
 PRIMARY_KEY                                NUMBER
 VALUE_ONE                                  VARCHAR2(5)
 VALUE_TWO                                  VARCHAR2(5)
 VALUE_THREE                                VARCHAR2(5)
```

Auditing Triggers and Procedures

When auditing is enabled, the automatically–generated database trigger in the ”After” event on the audited table performs the auditing.

This trigger calls a stored procedure to compare each column being audited to see if its value is changing. If so, the procedure saves the previous (old) value to the shadow table.

Auditing creates one row in the shadow table for each audited transaction against the table; thus, a single row in the shadow table represents all old values for all changed columns on that transaction.

The data is not compressed, since a table uses only one byte for a NULL, and AuditTrail represents all unchanged values as NULLs in the shadow table (”sparse” format).

The audit trigger names contain the first 26 characters of the audited table name plus ”_AI”, ”_AU” or ”_AD”, where one of I, U or D indicates Insert, Update or Delete, respectively. Likewise, the audit procedure names use the first 26 characters of the table name plus ”_AIP”, ”_AUP” or ”_ADP”. Your table names must be unique within the first 26 characters.

Views

After a shadow table is created, views onto the shadow table are created to allow easier access to the data in the ”sparse” rows. These

views simplify tasks such as querying a row/column's value on a given date and tracking changes to a row/column over time.

The view name contains the first 26 characters of the audited table name plus "_AC#" or "_AV#" where C or V indicates the type of view and # indicates a number. Due to limitations in creation size, the shadow table columns may need to be broken into multiple views, which are numbered sequentially.

Each view allows slightly different access to the data. One allows the user to reconstruct the value for a row at a given time (_AC), while the other provides simple access to when a value was changed (_AV).

For our example table, the _AV1 and _AC1 views are created as follows:

```
SQL> DESCRIBE AUDIT_DEMO_AV1
 NAME                           NULL?    TYPE
 ------------------------------ -------- ----
 AUDIT_TIMESTAMP                         DATE
 AUDIT_TRANSACTION_TYPE                  VARCHAR2(1)
 AUDIT_USER_NAME                         VARCHAR2(100)
 AUDIT_TRUE_NULLS                        VARCHAR2(250)
 PRIMARY_KEY                             NUMBER
 VALUE_ONE                               VARCHAR2(5)
 VALUE_TWO                               VARCHAR2(5)
 VALUE_THREE                             VARCHAR2(5)

SQL> DESCRIBE AUDIT_DEMO_AC1
 NAME                           NULL?    TYPE
 ------------------------------ -------- ----
 AUDIT_TIMESTAMP                         DATE
 AUDIT_TRANSACTION_TYPE                  VARCHAR2(1)
 AUDIT_USER_NAME                         VARCHAR2(100)
 PRIMARY_KEY                             NUMBER
 VALUE_ONE                               VARCHAR2(5)
 VALUE_TWO                               VARCHAR2(5)
 VALUE_THREE                             VARCHAR2(5)
```

How Data Appears in Tables and Views

Here is an example of how data appears in your original table, your shadow table, and your audit views after a series of changes (starting with an empty AUDIT_DEMO table).

```
SQL> INSERT INTO AUDIT_DEMO VALUES (1,'A','A','A');
SQL> INSERT INTO AUDIT_DEMO VALUES (2,'X','X','X');
```

```
SQL> SELECT PRIMARY_KEY KEY, VALUE_ONE VAL_1,
     VALUE_TWO VAL_2, VALUE_THREE VAL_3 FROM AUDIT_DEMO;

 KEY VAL_1 VAL_2 VAL_3
---- ----- ----- -----
   1 A     A     A
   2 X     X     X

SQL> UPDATE AUDIT_DEMO SET VALUE_ONE ='B'
     WHERE PRIMARY_KEY = 1;

 KEY VAL_1 VAL_2 VAL_3
---- ----- ----- -----
   1 B     A     A
   2 X     X     X

SQL> UPDATE AUDIT_DEMO SET VALUE_TWO ='B'
          WHERE PRIMARY_KEY = 1;

 KEY VAL_1 VAL_2 VAL_3
---- ----- ----- -----
   1 B     B     A
   2 X     X     X

SQL> UPDATE AUDIT_DEMO SET VALUE_THREE ='B'
     WHERE PRIMARY_KEY = 1;
SQL> UPDATE AUDIT_DEMO SET VALUE_ONE ='Y'
     WHERE PRIMARY_KEY = 2;
SQL> UPDATE AUDIT_DEMO SET VALUE_ONE = NULL
     WHERE PRIMARY_KEY = 1;
SQL> UPDATE AUDIT_DEMO SET VALUE_ONE ='C'
     WHERE PRIMARY_KEY = 1;
```

After our two inserts and six updates, the final values in the audited table are:

```
 KEY VAL_1 VAL_2 VAL_3
---- ----- ----- -----
   1 C     B     B
   2 Y     X     X
```

The final values in the corresponding shadow table are as follows. A row in the shadow table represents the state of the audited row *before* the audited row was changed. Note that if a value in a row doesn't change during the transaction, the shadow table records a null for that value in that transaction.

In our example, the first two rows in the shadow table represent the state where there was no data for our two audited rows before they were inserted. The "prior values" are null values for the two insert transaction (type I) rows. Similarly, when we update the first value of row 1 to be the value B instead of A, the shadow table records the value A in its third row:

```
SQL> SELECT TO_CHAR(AUDIT_TIMESTAMP, 'HH24:MI:SS') TIME,
     AUDIT_TRANSACTION_TYPE TYPE, AUDIT_USER_NAME NAME,
     PRIMARY_KEY KEY, VALUE_ONE VAL_1, VALUE_TWO VAL_2,
     VALUE_THREE VAL_3, AUDIT_TRUE_NULLS FROM AUDIT_DEMO_A;

TIME     TYPE NAME    KEY VAL_1 VAL_2 VAL_3 AUDIT_TRUE_NULLS
-------- ---- ------ ---- ----- ----- ----- ---------------
11:08:16 I    FND60     1
11:08:40 I    FND60     2
11:18:40 U    FND60     1 A
11:20:12 U    FND60     1       A
11:21:54 U    FND60     1             A
11:22:15 U    FND60     2 X
14:20:50 U    FND60     1 B
14:21:15 U    FND60     1                   NYNN

8 rows selected.
```

Given the current values of the row in the audited table, you can trace the changes made to the row by backing up through the corresponding rows in the shadow table.

In our example table, we made two insert and six update transactions, so we see those eight transactions in our shadow table. In the last row, the NYNN indicates that the value in the second table column (VALUE_ONE) has changed from an actual null value (the Y) rather

than being an unchanged value (represented by null in the shadow table).

The following two views provide further ways of examining your audited data.

The rows with a transaction type of C in the view indicate the current value of the row when the data was selected (the view is a join between the shadow table and the audited table, so the current value row reflects the current state of the audited table).

The _AC view provides a ”filled–in” version of the data, where unchanged values appear instead of being represented by null values. You can order this view by the primary key (rather than by timestamp), so all rows in the shadow table that correspond to a single audited row appear together, with a secondary ordering by timestamp.

```
SQL> SELECT TO_CHAR(AUDIT_TIMESTAMP, 'HH24:MI:SS') TIME,
     AUDIT_TRANSACTION_TYPE TYPE, AUDIT_USER_NAME NAME,
     PRIMARY_KEY KEY, VALUE_ONE VAL_1, VALUE_TWO VAL_2,
     VALUE_THREE VAL_3 FROM AUDIT_DEMO_AC1
     ORDER BY PRIMARY_KEY, AUDIT_TIMESTAMP;

TIME     TYPE NAME         KEY VAL_1 VAL_2 VAL_3
-------- ---- ---------- ---- ----- ----- -----
11:08:16 I    FND60         1 A     A     A
11:18:40 U    FND60         1 B     A     A
11:20:12 U    FND60         1 B     B     A
11:21:54 U    FND60         1 B     B     B
14:20:50 U    FND60         1       B     B
14:21:15 U    FND60         1 C     B     B
17:53:34 C                  1 C     B     B
11:08:40 I    FND60         2 X     X     X
11:22:15 U    FND60         2 Y     X     X
17:53:34 C                  2 Y     X     X

10 rows selected.
```

Attention: If the changes to your audited table occur faster than one change per second (that is, more frequently than the one–second granularity provided by SYSDATE), you may see ”blurring” of records — more than one record per transaction — in the _AC view because of joins used in this view. However, the shadow table itself remains correct for your transactions, and you can resolve those transactions using the shadow table directly.

The _AV1 view provides a more sparse view of the audit data, ordered by timestamp:

```
SQL> SELECT TO_CHAR(AUDIT_TIMESTAMP, 'HH24:MI:SS') TIME,
     AUDIT_TRANSACTION_TYPE TYPE, AUDIT_USER_NAME NAME,
     PRIMARY_KEY KEY, VALUE_ONE VAL_1, VALUE_TWO VAL_2,
     VALUE_THREE VAL_3, AUDIT_TRUE_NULLS
     FROM AUDIT_DEMO_AV1;

TIME     TYPE NAME    KEY VAL_1 VAL_2 VAL_3 AUDIT_TRUE_NULLS
-------- ---- ------ ---- ----- ----- ----- ----------------
11:08:16 I    FND60     1
11:08:40 I    FND60     2
11:18:40 U    FND60     1 A
11:20:12 U    FND60     1       A
11:21:54 U    FND60     1             A
11:22:15 U    FND60     2 X
14:20:50 U    FND60     1 B
14:21:15 U    FND60     1                   NYNN
17:58:31 C              1 C     B     B
17:58:31 C              2 Y     X     X

10 rows selected.
```

Here is an example of how you might use a view to determine who changed a particular value and when:

```
SQL> SELECT TO_CHAR(AUDIT_TIMESTAMP, 'HH24:MI:SS') TIME,
     AUDIT_TRANSACTION_TYPE TYPE, AUDIT_USER_NAME NAME
     FROM AUDIT_DEMO_AV1
     WHERE PRIMARY_KEY = 1
     AND VALUE_ONE = 'B';

TIME     TYPE NAME
-------- ---- ------
14:20:50 U    FND60
```

Similarly, you might want to determine who changed a value to null and when:

```
SQL> SELECT TO_CHAR(AUDIT_TIMESTAMP, 'HH24:MI:SS') TIME,
     AUDIT_TRANSACTION_TYPE TYPE, AUDIT_USER_NAME NAME
     FROM AUDIT_DEMO_AV1
     WHERE PRIMARY_KEY = 1
     AND VALUE_ONE  IS NULL
     AND SUBSTR(AUDIT_TRUE_NULLS,2,1) = 'Y';

TIME     TYPE NAME
-------- ---- ------
14:21:15 U    FND60
```

Changing Your Audit Tables

You may add columns to the shadow table after auditing has begun on a table. However, the shadow table does not track the column changes that occurred before the column was added. If you add must rerun the Audit Trail Update Tables Report to:

- add the necessary column to the shadow table
- regenerate the audit triggers and procedures for the table so that they now audit the additional column

Reporting on Audit Information

Report on Your Audit Data

You should write audit reports as needed. AuditTrail provides the views of your shadow tables to make audit reporting easier; you can write your reports to use these views.

You may want to create one or more indexes to your shadow table to speed up your reporting. However, such indexes decrease performance during actual auditing of transactions, so you should drop your indexes from the shadow table when you have finished reporting.

Attention: Because the structure of the audited table may change between product versions, AuditTrail does not support upgrading existing shadow tables or audited data. Before an upgrade, you should archive the shadow tables and perform all necessary reporting on the audited data.

Disabling AuditTrail and Archiving Audit Data

You may report on your audits or disable auditing at any time. When you disable auditing, you should do the following procedure:

Stop Auditing New Transactions

Disable auditing using *either* "Disable – Prepare for Archive" or "Disable – Interrupt Audit" and running the Audit Trail Update Tables report.

Disable – Prepare for Archive	Copies the current values of all rows in the audited table into the shadow table, and then disables the auditing triggers. There is no longer any recording of any changes. You should archive the shadow table before you purge it.
Disable – Interrupt Audit	Modifies the triggers to store one "final" row in the shadow table for each row that is modified in the audit table (remember that a given row in the shadow table represents the data in the audited row *before* an update). If a row in the table being audited is changed again (a second time), that change is not recorded. The shadow table grows slowly, until it contains one row for each row in the table being audited. Then there is no longer any recording of any changes.

Archive Your Audit Data

You should archive the information in the shadow tables according to your business needs.

Clean Out the Shadow Table

Before you restart auditing, you should clean out the shadow table. If there were transactions during the time auditing was disabled, and you did not clean out the shadow table, the data in the shadow table would be invalid because it would have a gap where transactions were not recorded. You purge the shadow table(s) by setting the audit group to Disable – Purge Table and running the Audit Trail Update Tables report.

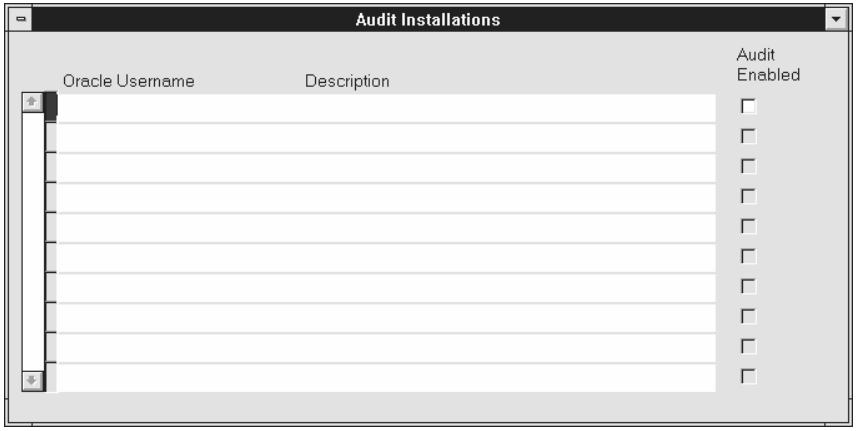
Disable – Purge Table	Drops the auditing triggers and views and deletes all data from the shadow table.

Restart Auditing (If Desired)

You restart auditing by setting the audit group to Enable Requested and running the Audit Trail Update Tables report again.

Attention: If you disable using Disable Purge Table and then reenable auditing for a table, AuditTrail flushes the contents of the shadow table when auditing is reenabled. You should archive any shadow table data that you want to keep before you reenable auditing.

Audit Installations Window



Use this window to enable AuditTrail for an ORACLE username at your installation. An ORACLE username grants access privileges to an application's tables and database objects.

For auditing to take effect, you must also define one or more audit groups and run the Audit Trail Update Tables report. See: Reporting on AuditTrail Data: page 3 – 22.

Prerequisites

- Register your ORACLE username. See: ORACLE Users: page 9 – 5.

Audit Installations Block

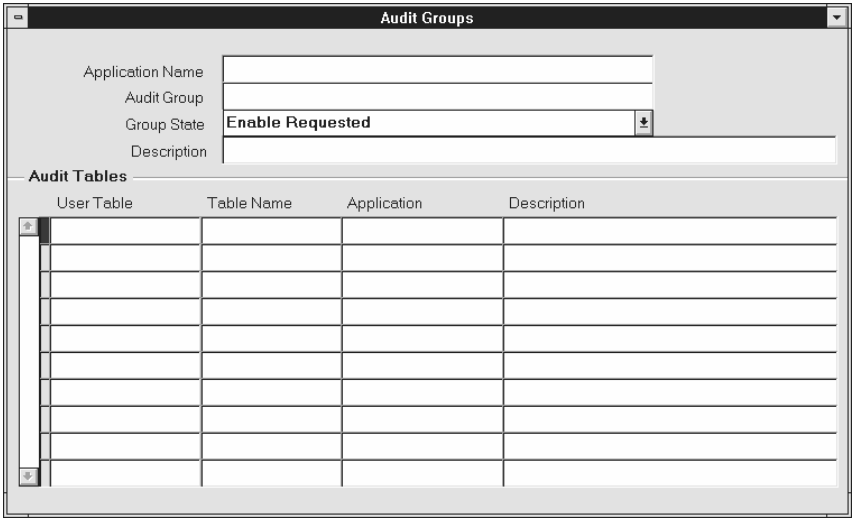
Oracle Username

Select the Oracle username that owns the tables you wish to audit.

Audit Enabled

Check the Audit Enabled check box to enable AuditTrail for an Oracle username. Before auditing takes effect you must define one or more audit groups and run the Audit Trail Update Tables report.

Audit Groups Window



Use this window to select the tables that you wish to audit. You audit a table by defining an audit group, which may consist of one or more tables.

First identify the tables you want to audit, then, using the Audit Tables window, select which columns in each table you wish to audit. Or, select which columns in a particular table you wish to audit (using the Audit Tables window), then define your audit group (using this window).

To enable or disable auditing for the tables in your audit group, run the Audit Trail Update Tables program using the Submit Requests window. If you change the definition or audit state of your group later, you must rerun this program.

Prerequisites

- Define an audit installation using the Audit Installations window.

Attention: Your tables and their primary key information must already be registered and defined for successful auditing. If the table you want to audit is a custom table (not shipped as part of Oracle Applications), you should also perform the following two steps:

- Register your table *and* its primary key columns using Oracle Application Object Library's Tables window (Application Developer Responsibility).

- Run the Register Tables concurrent program from the Submit Requests window.

Audit Groups Block

Identify your audit group and enable or disable auditing for this group.

Application Name

Select the name of an application to associate with your audit group. The combination of application name and group name uniquely identifies your audit group. An audit group may be used to audit tables in additional applications.

Group Name

Enter the name of the audit group.

State

Choose Enable Requested if you are defining a new audit group. When you run the Audit Trail Update Tables report, the concurrent program creates database triggers for the tables in your audit group. Once you have run the program, this field displays Enabled for audit groups where AuditTrail is active.

Attention: All primary key columns in each table in an audit group are automatically selected for auditing, whether or not you use the Audit Tables window to select which columns you wish to audit.

To disable auditing for a group, choose one of the following options and then run the Audit Trail Update Tables report to have your changes take effect.

Disable – Prepare for Archive	Copies the current values of all rows in the audited table into the shadow table, and then disables the auditing triggers. This option requires the most space, since there is at least one row in the shadow table for every row in the audited table (and another row in the shadow table for each transaction on the original row in the audited table). You should then archive the table before you empty the shadow table.

Disable – Interrupt Audit	Modifies the triggers to store one final row in the shadow table as the audited row is modified in the audit table (remember that a given row in the shadow table represents the data in the audited row *before* an update). Inserts or further changes are no longer audited. The shadow table then grows slowly, and the data may be accessed by the existing audit views.
Disable – Purge Table	Drops the auditing triggers and views and deletes all data from the shadow table.

Audit Tables Block

Identify the application tables you want to audit in your audit group**.**

User Table

Select the end user table name (frequently the same name as the table name) for your database table. Once you choose a table, you see its table name and associated application.

Table Name

This field displays the actual name for the table you have selected to include in your audit group.

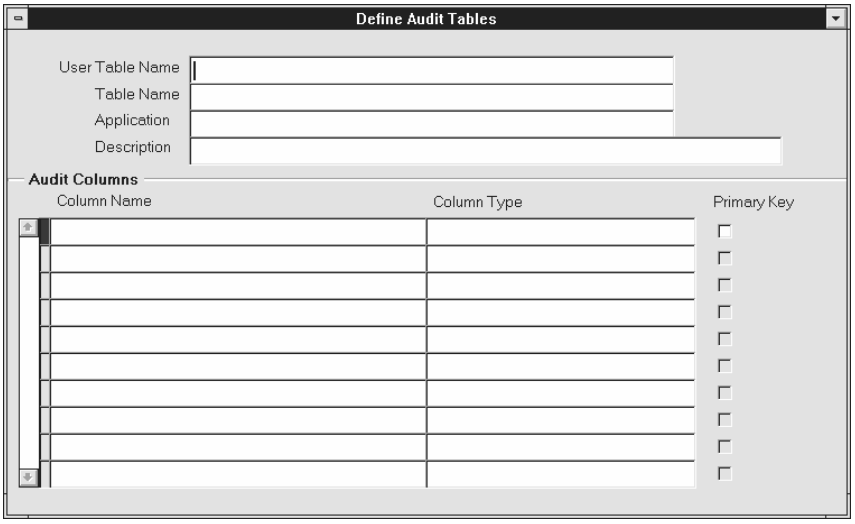
Application

This field displays the application name for the table you have selected to include in your audit group.

Description

This field displays the description for the table you have selected to include in your audit group.

Audit Tables Window



Use this window to select which columns in a table you wish to audit.

First identify the columns in a table you want to audit. Then, using the Audit Groups window, include the table as part of an audit group. Or, you may define your audit group first (using the Audit Groups window), and then select which columns in the table you want to audit (using this window).

To enable or disable auditing for the tables in your audit group (i.e., the columns you have selected here), you must run the Audit Trail Update Tables program using the Submit Requests window. If you select additional columns to audit, or change the definition or audit state of your group later, you must rerun this program.

Prerequisites

- Define an audit installation using the Audit Installations window.

Attention: Your tables and their primary key information must already be registered and defined for successful auditing. If the table you want to audit is a custom table (not shipped as part of Oracle Applications), you should also perform the following two steps:

- Register your table *and* its primary key columns using Oracle Application Object Library's Tables window (Application Developer Responsibility).

- Run the Register Tables concurrent program from the Submit Requests window.

Define Audit Tables Block

Identify the application table you want to audit. Successively selecting *Go – Next Record* from the menu or toolbar displays, in alphabetical order, the name of each application table registered at your installation site.

User Table Name

Select the end user table name (frequently the same name as the table name) for your database table. Once you choose a table, you see its table name and associated application.

Table Name

This field displays the actual name for the table you have selected to include in your audit group.

Application

This field displays the application name for the table you have selected to include in your audit group.

Audit Columns Block

Select the columns you want to audit. Successively selecting *Go – Next Record* from the menu or toolbar displays, in alphabetical order, the name of each application table registered at your installation site.

- You cannot delete a column from auditing once it has been selected.
- You may add additional columns to be audited.
- Each time you select a column to be audited, that change affects every audit group that includes the table which owns the column.

Column Name

Enter the name of the database column you want to audit. You should not explicitly enter the names of your table's primary key columns,

since they are entered automatically, and you will get an error message if you try to save a duplicate column name. You can query to see which columns appear automatically.

Note that once you have chosen a column, you cannot delete it from the audit set, though you may add other columns to the set later.

Once you choose a column, you see its column type and whether it is part of the primary key for this table.

Column Type

This field describes the type of data the column stores, for example, varchar2.

Primary Key

This field displays Yes or No indicating whether the column you are auditing is a primary key column.

Any primary key columns you do not select to audit are automatically included when you save your column selections. For example, if the table you are auditing has two primary key columns, and you choose to audit one of them, the second primary key column is automatically selected when you save your column selections.

CHAPTER

4

Oracle Applications Help

This chapter describes the Oracle Applications help architecture for HTML as well as explaining how to customize Oracle Applications help.

Overview of Oracle Applications Help for HTML

The Web–enabled Oracle Applications use a Web browser such as Netscape Navigator V3.0 or Internet Explorer V3.0 to display online help. When you choose an item from the Help menu, you view the help you requested in an independent browser window. You can use the buttons provided by the browser to navigate to help topics you have already viewed in your current help session, or use the next and previous buttons within the help window to navigate through the documentation following a predetermined path. You can exit from the help window at any time.

See: Getting Help,
(Oracle Applications User's Guide)

This chapter explains how you can customize Oracle Applications help using an editing tool to either create new HTML files or modify the existing HTML files. It assumes you are already familiar with HTML authoring techniques.

Help File Architecture: page 4 – 2

Customizing Oracle Applications Help: page 4 – 4

Help Targets in Oracle Applications: page 4 – 5

Help File Architecture

Help Directories

In general, there are three Help directories for each Oracle Applications product. The first of these directories, called a *product help directory,* contains the vast bulk of online help for a single Oracle Applications product. For example, a product help directory contains concept modules to help you understand the concepts underlying a particular Oracle Applications product. It also contains task modules that describe the usage of a product's forms and reports. To help you find the information you need, each product help directory also contains a contents page with links to all the concept and task modules contained in that directory.

The second help directory provided for each Oracle Applications product is a *release notes help directory* that describes what is new in the current release of the product. You link to release notes help from the contents page of the product help.

The third directory is a *custom help directory* for for each product. A predefined link to a custom help file is encoded on the contents page of each set of product help files.

Each Oracle Application is delivered with a "dummy" custom help file located in the custom help directory. You can replace this dummy file with a file containing your own custom help. Then you can link from the standard product help file to your own custom help.

Help Directory Names

The naming convention for Oracle Applications help directories is based on the application short name of the product. For example, the application short name for Oracle General Ledger is GL, and so all of Oracle General Ledger's help directory names begin with GL. The following table describes the naming convention for the three types of help directories.

Directory Name	Naming Convention	Example based on GL
Help	<application_short_name>	GL
Release Notes	<application_short_name>NEW	GLNEW
Custom	<application_short_name>CUST	GLCUST

Table 4 – 1 (Page 1 of 1)

Library Help File

There is a single help file, LIBRARY.HTM, that serves as the master table of contents for all the product help files. You can view this library help file from the Help menu, or by pressing the Library button from within any Oracle Applications help window. From the library help file, you can link to the contents page for any Oracle Applications product.

Help File Directories

If you are installing help on a PC, all .HTM and .GIF files, including product help files, release note help files, custom help files, and the library help file, are located in the subdirectory C:\APPS10\AU10\DOCS\<language>.

Overview of Oracle Applications Help: page 4 – 2

Customizing Oracle Applications Help: page 4 – 4

Customizing Oracle Applications Help

To customize Oracle Applications help you can:

- Create a custom help file to supplement an Oracle Applications product help file

and / or

- Modify an Oracle Applications product help file

Create a custom help file

To supplement an Oracle Applications product help file with your own custom help file, create a correspondingly named <application_short_name>CUST.HTM file containing the anchor <A Name = "<application_short_name>CUST">, along with your custom help text. Overwrite the dummy custom help file that is delivered with your Oracle Applications product.

Upgrade Consequences:

Each Oracle Applications product help system will always contain a link to a custom help file. Thus, your custom help file should continue to work after you upgrade to a newer client release. Note, you may need to re–propagate your custom help file to your help directories after an upgrade, since it may be replaced by a dummy custom help file during the upgrade.

Modify an Oracle Applications product help file

The Oracle Applications help files are in the HTML format and can readily be modified to add your own information.

The .HTM files that comprise the bulk of your help source files contain help text as well as markup information that is used by the browser. It is very important that you not modify the .HTM markup information when you modify help text. In fact, you should back up the help source files that Oracle delivers before you customize them in case you need to revert to the originals.

Upgrade Consequences:

Oracle does not provide any mechanism for identifying changes between releases of Oracle Applications help files. With each new release of Oracle Applications, you will need to either reapply your changes to the new help files, or else continue to use your old customized help files and forgo using the most current Oracle Applications help.

Creating or Modifying a HTML Help File

If you choose to create a HTML Help file, or to modify a help file that Oracle Applications provides, you need a file editor, such as Microsoft Word or any other simple text editor. However, to modify the existing .HTM files we recommend using an HTML editor like Netscape Gold3.0.

Overview of Oracle Applications Help: page 4 – 2

Help File Architecture: page 4 – 2

Help Targets in Oracle Applications

When you select Help in Oracle Applications, the topic for your current window opens. If you select help from a reports parameter window, your help file opens to a discussion of that report. If you change responsibilities, you access the correct help file for your new application.

Oracle Applications uses special targets in order to locate the correct help file for a window, and then the correct text within the help file. Oracle Applications looks in the *docs* directory for your language to find a help directory with the name of the current window's application short name. Once in the product directory, the product's registry file is scanned to determine the correct .HTM file within the directory. The URL to this file is then passed to the browser with a text anchor to enable the help file to open at the pertinent section of the file.

When calling help from a window, Oracle Applications looks for a help tag based on the form name and the window name in the syntax *form_name.window_name*. You can override the *form_name* portion of the target by specify a HELP_TARGET parameter in the parameter field of the Form Functions window. Use the syntax HELP_TARGET = *"alternative_form_name"*. See: Form Functions: page 2 – 33.

When calling help from a report parameter window, the help target is based on SRS.<concurrent_program_shortname>. The help file searched is the one that owns the report or program.

CHAPTER

5

User Profiles

This chapter tells you about the role of user profiles in Oracle Applications, including an overview of user profiles and a detailed description of the form you use to set user profile values for your user community.

The Overview of User Profiles includes definitions of key concepts, and an explanation of how to set site, application, responsibility, and user profile options in Oracle Applications.

Overview of Setting User Profiles

A user profile is a set of changeable options that affect the way your application looks and behaves. As System Administrator, you control how Oracle Applications operate by setting user profile options to the values you want. You can set user profile options at four different levels: site, application, responsibility, and user.

Major Features

Set of Books

You can further control security by assigning a set of books to a responsibility, application or site. A set of books is a company or group of companies within Oracle Applications that share a common account code, calendar, and functional currency.

Setting User Profile Options

As System Administrator, you use the System Profile Values window to set profile options for your user community. If you change a user profile option value, your change takes effect as soon as your users log on again or change responsibilities. See: System Profile Values Window: page 5 – 6.

When you set a user profile, you provide Oracle Applications with standard information (such as printer) that describes a user, responsibility, application, or site. You can set values for user profile options at each profile level.

Site Option settings pertain to all users at an installation site.

Application Option settings pertain to all users of any responsibility associated with the application.

Responsibility Option settings pertain to all users currently signed on under the responsibility.

User Option settings pertain to an individual user, identified by their application username.

The values you set at each level provide run–time values for each user's profile options. An option's run–time value becomes the highest–level setting for that option.

When a profile option may be set at more than one level, site has the lowest priority, superseded by application, then responsibility, with user having the highest priority. For example, a value entered at the site level may be overridden by values entered at any other level. A value entered at the user level has the highest priority, and overrides values entered at any other level.

For example, for a given user, assume the printer option is set only at the site and responsibility levels. When the user logs on, the printer option assumes the value set at the responsibility level, since it is the highest–level setting for the option.

Suggestion: As System Administrator, you should set site–level option values before specifying profile options at the other three levels after the installation of Oracle Applications. The options specified at the site–level work as defaults until the same options are specified at the other levels.

Application users may use the Personal Profile Values window to set their own personal profile options at the user level. Not all profile options are visible to users, and some profile options, while visible, may not be updated by end users.

See Also

Setting Your Personal User Profile, *Oracle Applications User's Guide*

Using Profile Options as a Parameter or Segment Default Value

Profile option settings may be used as a default value for a concurrent program's parameter or flexfield's segment. Table 5 – 1 lists the forms you may use to enter a profile option whose setting serves as a default value.

To use a profile option's setting as a default value, navigate to the form's Default Type field and select *Profile*. Then, enter the profile option's internal name in the Default Value field.

Forms for entering Profile Option as a Default

Form	Window	Field
Concurrent Programs	Parameters	Parameter Detail region – Default Type / Default Value
Request Set	Report Parameters	Default Type / Default Value
Key Flexfield Segments	Segment	Validation Information region – Default Type / Default Value
Descriptive Flexfield Segments	Segment	Validation Information region – Default Type / Default Value

Table 5 – 1 (Page 1 of 1)

Examples of User Profile Options

Example #1

Your Accounts Payable department recently purchased a printer, and you want all the reports from that department to print on that new printer. You simply change the "Printer" profile option for Oracle Payables to reflect the purchase of the new printer.

Suggestion: Example #2 highlights the importance of default profile options. If an application user of Oracle Payables or a responsibility associated with Oracle Payables already has a value specified for the printer profile option, that value will override the value you set at the application level. We suggest you first set user profile options at the site level, and then work your way up the hierarchy to other levels when appropriate. User profile options not set at one level default to the user profile options set at the next lower level.

Example #2

You can further control security within Oracle Applications by assigning a set of books to a responsibility, application or site. By assigning a set of books to a responsibility, you control not only the forms and functions that the responsibility can access, but the specific set of books as well.

See your Oracle Applications product reference guide for information on how to define a set of books.

User Profile Option Values Report

This report documents user profile option settings. Use this report when defining different profile option values for several responsibilities, or users, or for different applications.

Report Parameters

Profile Option Name

Choose the profile option name whose values you wish to report on. If you do not select a profile option name, then this report will document all profile options.

User Name

Choose the name of a user whose profile option values you wish to report on.

Application Short Name

Choose the name of an application whose profile option values you wish to report on.

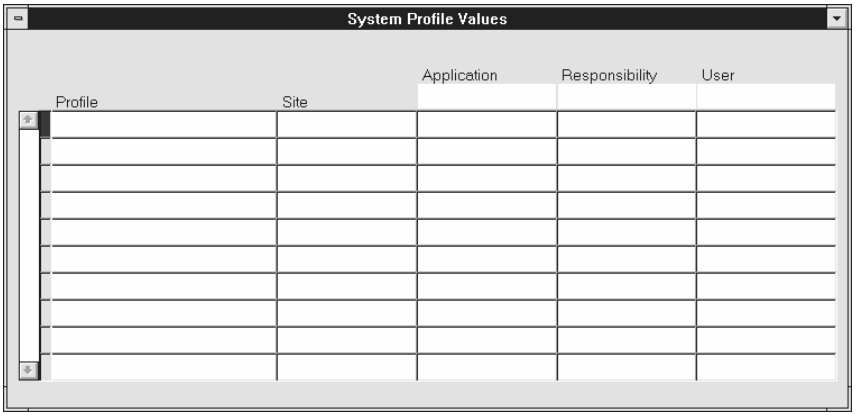
Responsibility Name

Choose the name of a responsibility whose profile option values you wish to report on.

Report Headings

The report headings display the specified report parameters and provide you with general information about the contents of the report.

System Profile Values Window



Use this window to view and set profile option values.

You can view and set profile options at the site, application, responsibility, and user levels. Your settings affect users as soon as they sign on or change responsibility. See: Overview of Setting User Profiles: page 5 – 2.

Profile Values Block

Set values for profile options at one or more levels. Each value overrides those set to its left. For example, a User Value setting overrides a Responsibility Value setting, which overrides an Application Value setting, which overrides a Site Value setting.

Profile

This field displays the name of a profile option.

Site

This field displays the current value, if set, for all users at the installation site.

Application

This field displays the current value, if set, for all users working under responsibilities owned by the application identified in the Find Profile Values block.

Responsibility

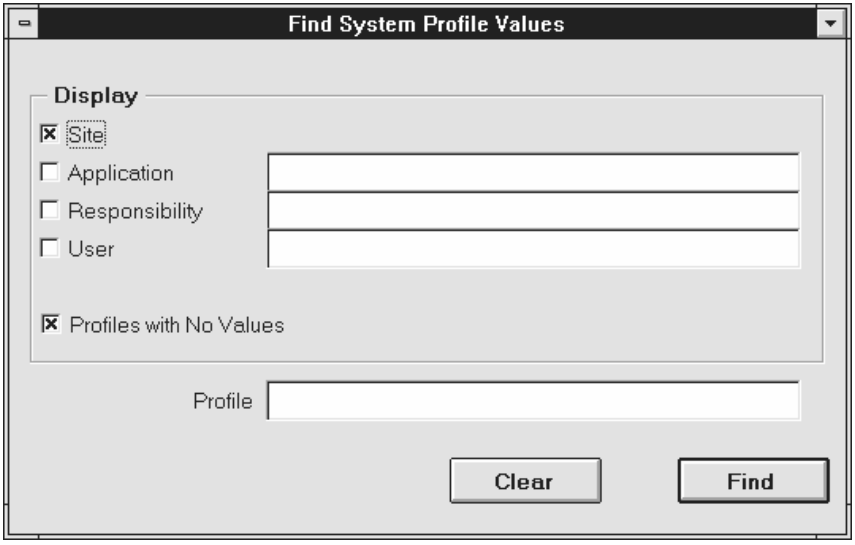
This field displays the current value, if set, for all users working under the responsibility identified in the Find Profile Values block.

User

This field displays the current value, if set, for the application user identified in the Find Profile Values block.

Suggestion: You should set site–level default values for any required options after installation of an application. If you do not assign a particular profile option at any of the four levels, that option does not have a default value and may cause errors when you use forms, run reports, or run concurrent requests.

Find System Profile Values Block



Specify the level or levels at which you wish to view and set profile option values.

Display

You can view the values set for your installed profile options at each of four levels:

- Site, which affects all users at an installation site.
- Application, which affects all users working under responsibilities owned by a particular application.

- Responsibility, which affects all users working under a specific responsibility.
- User, which affects a unique application user.

You can find the values for all profile options that include a specific character string, such as “OE:” for Oracle Order Entry. You can also display only profile options whose values are currently set.

Site

Check the Site check box if you wish to display the values for profile options at an installation site.

Application

Select an application if you wish to display profile option values for responsibilities owned by that application.

Responsibility

Select a responsibility if you wish to display profile option values for a specific responsibility.

User

Select an application user if you wish to display profile option values for a specific user.

Profile

Enter the name of the profile option whose values you wish to display. You may search for profile options using character strings and the wildcard symbol (%). For example, to find all the profile options prefixed by “Concurrent:”, you could enter “Conc%” and press the Find button.

Profiles with No Values

Select whether to display all profiles, including those without currently set values. If this check box is unselected, only profiles with current values are retrieved.

Find

Choose the Find button to display all profile options, or the profile options you are searching for, at the level or levels you specified.

CHAPTER

6

Managing Concurrent Programs and Reports

This chapter explains how to manage 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.

The essays in this chapter are organized under the following topics:

- Overview of concurrent programs and requests
- Organizing programs into Request Sets
- Organizing programs into Request Groups
- Defining program incompatibility rules
- Defining application database connections or Data Groups
- Copying and modifying program definitions

Form descriptions follow at the end of the chapter.

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.

Request Groups and Request Sets

Reports and concurrent programs can be assembled into request groups and request sets.

- A *request group* is a collection of reports or concurrent programs. A System Administrator defines report groups in order to control user access to reports and concurrent programs. Only a System Administrator can create a request group.
- *Request sets* define run and print options, and possibly, parameter values, for a collection of reports or concurrent program. End users and System Administrators can define request sets. A System Administrator has request set privileges beyond those of an end user.

Standard Request Submission and Request Groups

Standard Request Submission is an Oracle Applications feature that allows you to select and run all your reports and other concurrent programs from a single, standard form. The standard submission form is called *Submit Request*, although it can be customized to display a different title.

- The reports and concurrent programs that may be selected from the Submit Requests form belong to a *request security group*, which is a request group assigned to a responsibility.
- The reports and concurrent programs that may be selected from a customized Submit Request form belong to a request group that uses a code.

In summary, request groups can be used to control access to reports and concurrent programs in two ways; according to a user's responsibility, or according to a customized standard submission (Submit Request) form. See: Customizing the Submit Request Window using Codes: page 6 – 18.

Additional Information: Running Oracle Applications Reports and Programs in the *Oracle Applications User's Guide*

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.

Organizing Programs into Request Sets

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.

As System Administrator, you have privileges beyond those of your application users, including a privileged version of the Request Set window. See: Request Set Windows, *Oracle Applications User's Guide*.

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.

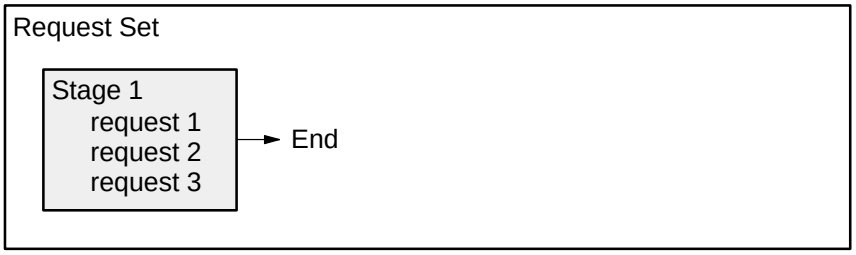
Use the Request Set form to create and edit request sets. See: Request Set Windows, *Oracle Applications User's Guide.*

Request Set Stages

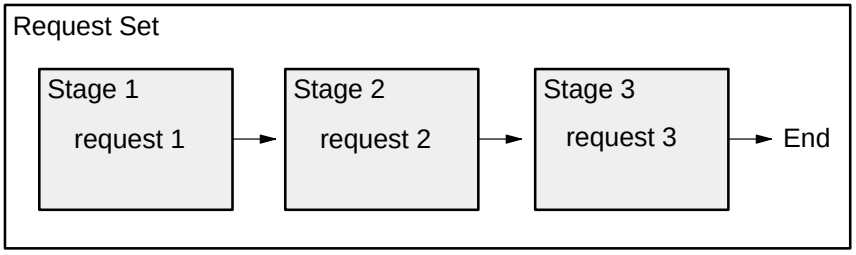
Organizing Request Sets into Stages

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.

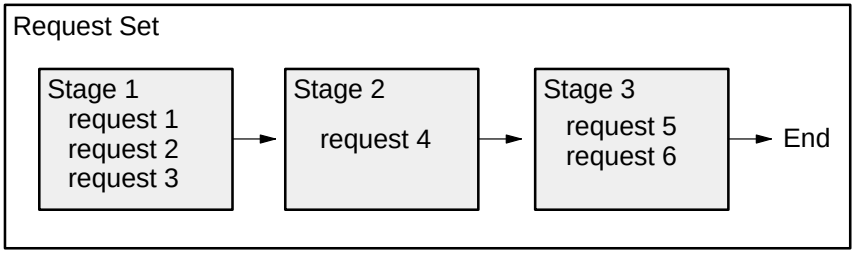


To run requests in sequence, you assign requests to different stages, and then link the stages in the order you want the requests to run.



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.

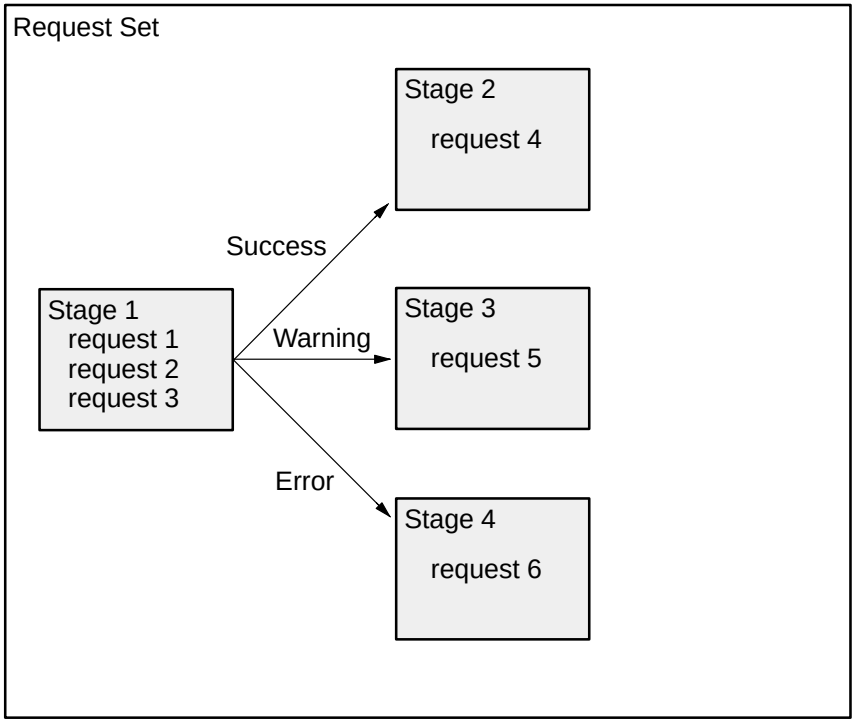


Using Stage Status

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, the request set in Figure 6 – 1 always begins with Stage 1. If Stage 1 complete with the status Warning, then the Warning link is followed, and Stage 3 is submitted. After Stage 3 completes, the set ends, since there are no links that may be followed.

Figure 6 – 1

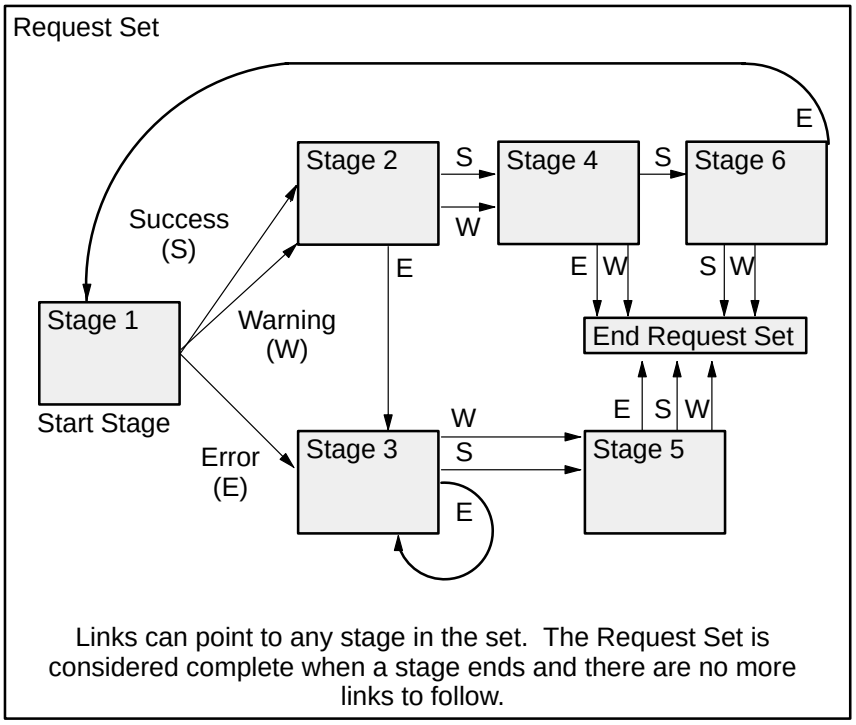


In this example, the stage status is determined using the Standard stage function. The Standard stage function uses the statuses of the requests within the stage to calculate the status for the stage. If all of the requests in a stage complete with a status of Success, then the status for the stage is 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.

Figure 6 – 2



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.

Stage Functions

The completion status of a stage is determined by a predefined function. A function can use information about the requests in a stage when determining the completion status for the stage. The Standard Oracle Applications function uses the completion status of the requests it contains, but almost any information from the request may be used to determine the completion status of a stage. Stage functions are defined by your Oracle application or can be customized at your site. For information on creating custom stage functions see the *Oracle Applications Developer's Guide*.

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 runs a critical stage, then the completion status of the set will be the same as the completion status of the most recently run 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.

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.

Attention: 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.

Request Sets as Concurrent Programs

When you define a request set or a stage within a request set that allows incompatabilities, 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 incompatabilities, 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. Table 6 – 1 displays request phase and status information that pertains to request sets.

Request Set Request Phase and Status

Phase	Status	Description
RUNNING	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.

Table 6 – 1 (Page 1 of 1)

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.

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 – 12.

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.

User does not own request set

All users can submit request sets that are added to a their request security group even if they contain requests that are not in the request security group. If the user does not own the request set, they:

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

User owns request set

If the user owns the request set, they:

- 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 Adminstrators 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 – 51.

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 – 23 .

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

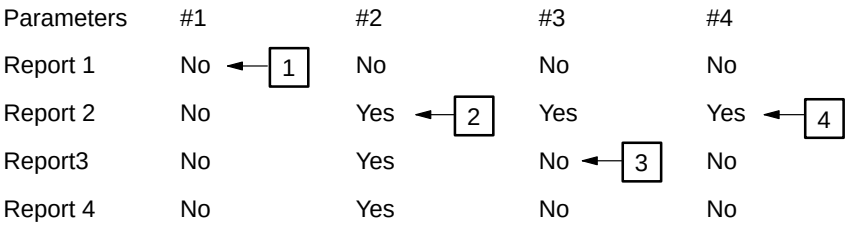
Selecting a value for a shared parameter provides a default for subsequent occurrences of the parameter. Changing a shared parameter's value provides a new default for subsequent occurrences of the parameter, but does not affect prior requests in the set.

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

Attention: 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.

Figure 6 – 3

Example Settings for a Shared Parameter



1 Selecting “No” for the first report defaults “No” in subsequent requests.

2 Selecting “Yes” for the second report, after selecting “No” for the first report, defaults “Yes” in subsequent reports, but does not change the first report (prior reports).

3 Selecting “No” for the third report, after selecting “Yes” for the second report, after selecting “No” for the first report, defaults “No” in subsequent reports, but does not change the first or second reports (prior reports).

4 Once all the reports parameters contain values, updating a shared parameter does not update the values in either subsequent or prior reports. For example, selecting “No” for the first report and navigating through all the parameter pop–up windows provides the “No” value for all of the shared parameters. Selecting “Yes” afterwards for the second report does not update the first, third, or fourth reports.

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 below:

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

Table 6 – 2 (Page 1 of 1)

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.

Figure 6 – 4

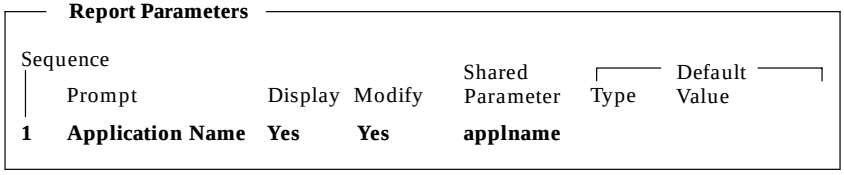
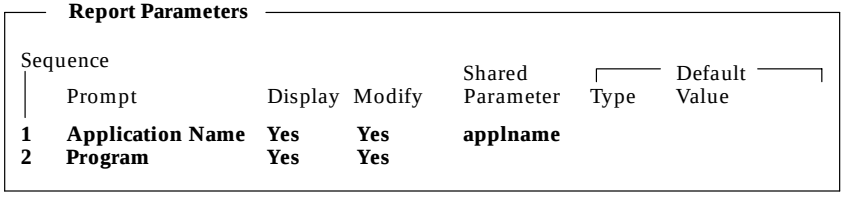


Figure 6 – 5



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.

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 Group

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.

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.

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. See: Menus: page 2 – 36.

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.

Passing Parameters to a Submit Requests Window

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)

Table 6 – 3 (Page 1 of 1)

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 – 18.

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 – 18.

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, GLOBA L.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.
```

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.

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.

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 – 51.

Program incompatibility and run–alone program definitions are enforced using 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

Concurrent Conflicts 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.

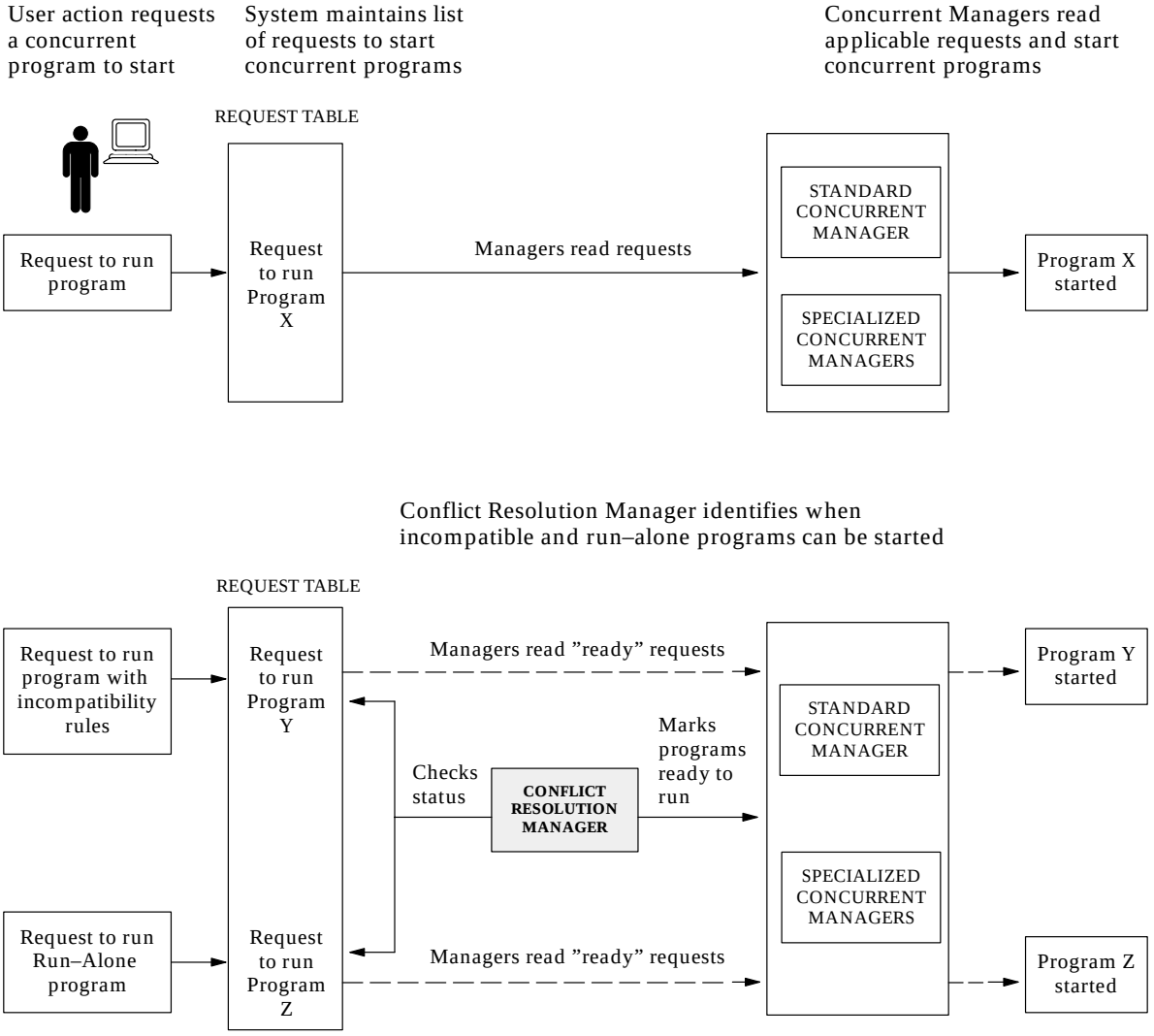
If a program is identified as Run Alone, then the Conflict Resolution Manager prevents the concurrent managers from starting other programs in the same conflict domain.

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 figure below illustrates the role of the Conflict Resolution Manager when enforcing program incompatibility rules.

Figure 6 – 6

Conflict Resolution Manager enforces incompatibility rules



Defining Data Groups

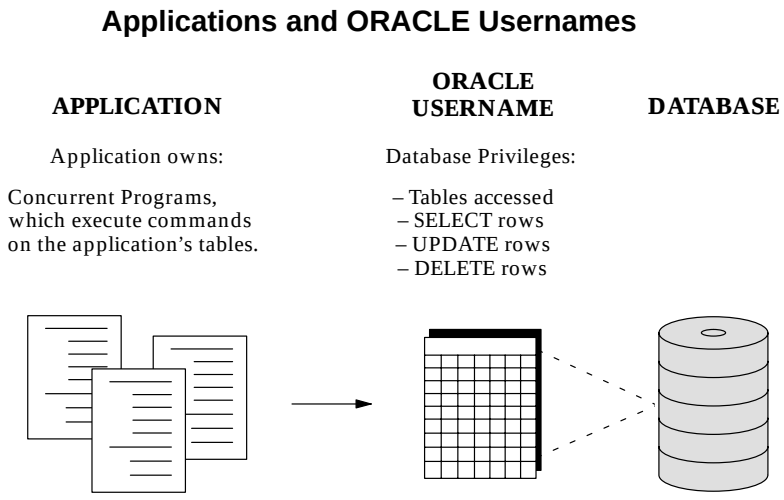
This essay explains how you can define data groups, which specify your applications database connections.

Introduction to Data Groups

A data group is a list of Oracle Applications and the Oracle username assigned to each application. Each application in a data group must have a Oracle username assigned to it. An application may be listed only once in a data group.

An Oracle username and password allow access to an application's tables in an Oracle database. Each Oracle username in a data group determines the database tables and table privileges accessible by the corresponding application or applications.

Figure 6 – 7



Data Group's Purpose

Each responsibility has a data group associated with it. A data group serves two purposes:

1. It identifies the Oracle username that forms connect to when you select the responsibility.
2. Concurrent managers use a data group to match the application that owns a report or concurrent program (submitted by a user of the responsibility) with a Oracle username.

Remarks on Data Group's structure

There are four points concerning the makeup of any Data Group.

1. Within each data group, an application can be listed only one time.
2. Within each data group, an Oracle username can be paired with more than one application.
3. Within each data group, the application *Application Object Library* is automatically included.
   - Application Object Library's Oracle username cannot be updated or deleted.
   - Application Object Library owns the database tables that oversee concurrent processing and the standard submission of reports by any Oracle Application.
4. Within each data group, a Tool Oracle username can be associated with each application–Oracle username pair. If that application–Oracle username pair is chosen as the one for forms to connect to when the responsibility is selected, then the Tool Oracle username is relevant, and appears by default.

Attention: To prevent users of a responsibility from connecting to the database using a Tool Oracle username, you can backspace over (erase) the default entry. You cannot select a Tool Oracle username associated with another application in the data group.

The Tool Oracle username identifies the database tables and privileges a user of the responsibility has when connecting to the database using an Oracle Tool, for example, SQL*Plus.

- When users of the responsibility select Oracle Tools window to run a tool that requires a database connection, the Tool Oracle Username appears as the default tool access Oracle Username.
- Users do not need to enter an Oracle password to use the Oracle Username.

Attention: The Oracle Tools window may not be available on your desktop platform.

Warning: If the Oracle Tool is query–only, for example, Oracle Browser, you may assign an unrestricted Tool Oracle Username. If the Oracle Tool allows write privileges, you should assign a Tool Oracle Username that is **restricted** (read–only).

Warning: Modifying data or table structures using an Oracle Tool, for example, SQL*Plus or SQL*Forms, may damage your data's integrity and is **not supported** by Oracle.

Using Data Groups with multiple Sets of Books

Use data groups to support multiple installations of an Oracle Applications product (for example, Oracle Payables) that supports multiple sets of books, where a different application is associated with each set of books. See: Using Data Groups with multiple product installations: page 6 – 30.

For example, with two installations of Oracle Payables supporting two Sets of Books, use data groups to indicate which Oracle Payables Oracle username to access from a certain General Ledger responsibility.

Define a data group for each application installation (set of books).

Define a responsibility for each application installation (set of books), and assign the appropriate data group to each responsibility.

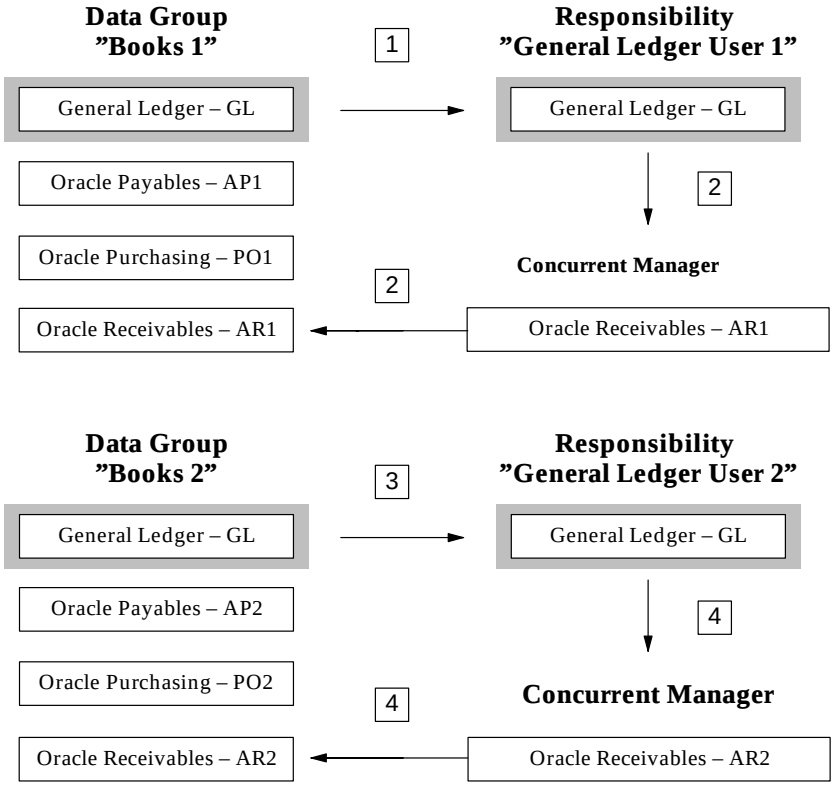
Using Data Groups to include custom applications

Use data groups to include custom applications you develop using Oracle's Application Object Library. To integrate a custom application with Oracle Applications, you must register the application using the Applications window. See: Applications: page 9 – 10.

An example of using two Data Groups to support two installations of Oracle Payables, Oracle Purchasing, and Oracle Receivables is illustrated below.

Figure 6 – 8

Using Data Groups with multiple product installations.



1 Data Group Books 1 is assigned to Responsibility General Ledger User 1.

2 A user of responsibility General Ledger User 1 submits an Oracle Receivables report, which runs accessing data with ORACLE ID AR1 privileges.

3 Data Group Books 2 is assigned to Responsibility General Ledger User 2. Note: both responsibilities attach to the same set of forms (same ORACLE ID).

4 A user of responsibility General Ledger User 2 submits an Oracle Receivables report, which runs accessing data with ORACLE ID AR2 privileges.

See:

Defining Data Groups: page 6 – 27

Modifying Data Groups: page 6 – 31

Data Groups: page 6 – 63
ORACLE Users field help: page 9 – 5

Modifying Data Groups

Predefined Standard Data Groups

During installation or upgrade of Oracle Applications, a standard data group is defined that pairs each installed application with an ORACLE username (note: this occurs 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, or add new Application–ORACLE username pairs to the group.

Defining new Data Groups

Since the installation process automatically defines Data Groups for Oracle Applications, you only need to define any additional data groups you wish to utilize.

You can copy a data group and give it a new name, creating a new data group. Each application, its assigned Oracle username, and, if present, its Tool Oracle username and description, are copied to the new data group.

Suggestion: Make a backup copy of your standard Data Group, and do not assign it to a responsibility. That way, if you ever inadvertently connect the wrong Oracle username to an application, or lose track of your applications' configuration, you have an initial configuration you can revert to.

Adding a custom application to a Data Group

If a custom application is developed with Oracle Application Object Library, to include it in a Data Group, you:

- Register the application with Oracle Applications using the Applications form
- Assign an Oracle username to the application using the ORACLE Usernames form

Registering an Oracle Username

Registering an Oracle username with Oracle Applications sets up the privileges to the Oracle Application Object Library database tables

(such as flexfield tables, menu tables, and so on) that are necessary to successfully use Oracle Applications. See: Overview of Applications DBA Duties: page 9 – 2.

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.

Warnings for Modifying Program Definitions: page 6 – 41

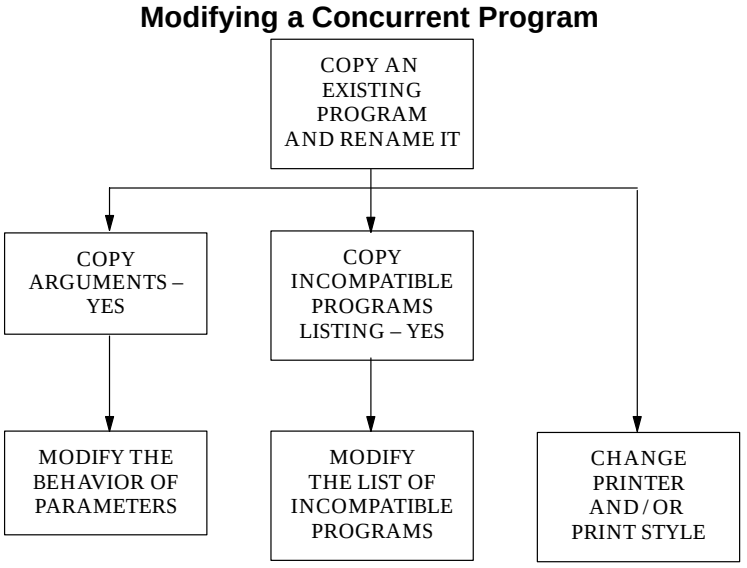
Copying and Renaming a concurrent program

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.

Figure 6 – 9



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.

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 – 23.

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.

Attention: 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.

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.

Attention: 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.

See: Planning and Defining Values and Value Sets (*Oracle Applications Flexfields Guide.*)

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.

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.

See:

Behavior of Program Parameters: page 6 – 39

Behavior of Parameters in Request Set: page 6 – 40

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 – 13.

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 – 41.

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.

Suggestion: 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

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.
	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.
	No default entered.	No default value is displayed.

Table 6 – 4 (Page 1 of 1)

Behavior of Parameters in Request Set

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.
		Parameter is displayed. – Display set to No.	Parameter is not displayed.
	No	Parameter not displayed.	Parameter not displayed.
Modify	n / a	Yes	Value can be modified.
	n / a	No	Value cannot be modified.

Table 6 – 5 (Page 1 of 2)

Parameter Details	Concurrent Programs form	Request Set form	Run Requests form
Default Type & Value	Yes – Default Type and Value entered.	Default Type and Value cannot be modified.	Default values can be changed by the user.
	No default entered.	Yes – a Default Type and Value can be entered.	Default values can be changed by the user.
		No – Default Type and Value are not entered.	No default value is displayed.

Table 6 – 5 (Page 2 of 2)

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.

Table 6 – 6 (Page 1 of 2)

Action	Form Used	Warning
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.

Table 6 – 6 (Page 2 of 2)

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: page 7 – 16.

The table below summarizes the steps to follow in our example.

Example – Modifying a Program's Parameters

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.
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

Table 6 – 7 (Page 1 of 2)

Form Used	Task
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.

Table 6 – 7 (Page 2 of 2)

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

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

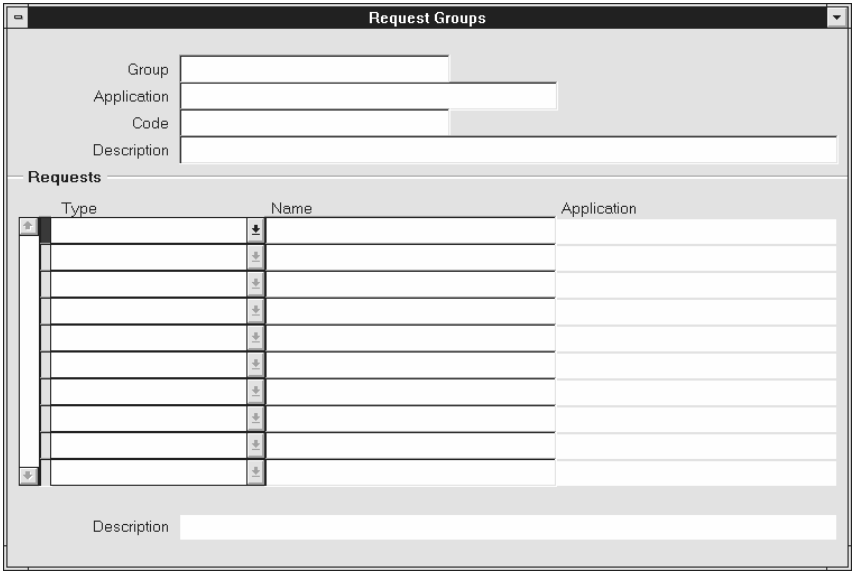
Choose the application name associated with the concurrent programs whose program information you wish to report on.

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.

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 – 18.

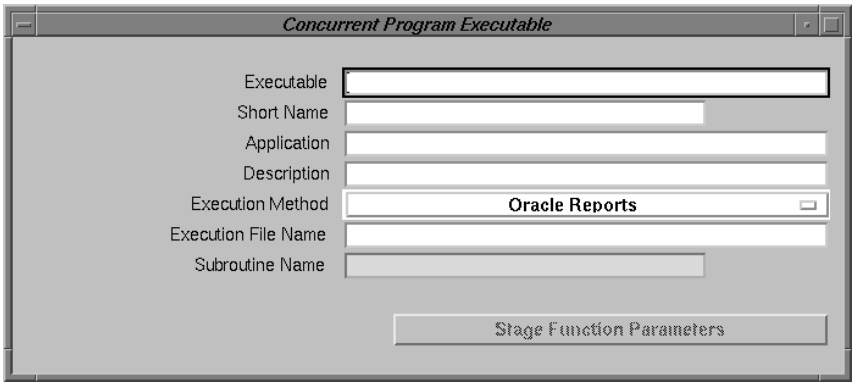
Requests Block

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

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.

The Installation Guide for your operating system details where to place the execution files for each execution method.

Attention: You cannot add new immediate programs to a concurrent manager program library. We recommend that you use spawned concurrent programs instead.

See: Concurrent Processing
(*Oracle Applications Installation Guide)*

Concurrent Program Executable Block

The combination of application name plus program name uniquely identifies your concurrent program executable.

See: Concurrent Programs Window: page 6 – 51

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.

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:

FlexRpt	The execution file is wrnitten using the FlexReport API.
FlexSql	The execution file is written using the FlexSql API.
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 stored procedure.
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. See the *Oracle Applications Installation Guide* for details on the path Oracle Applications uses to find each executable file.

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.

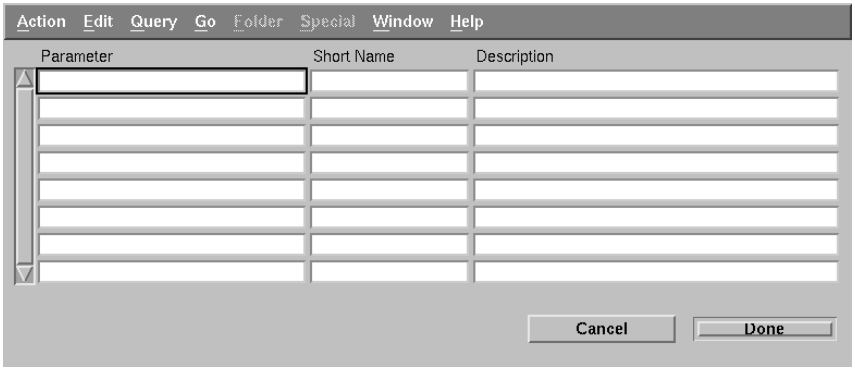
Only immediate programs or spawned programs using the Unified C API use the subroutine field.

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 Executions Method.

Stage Function Parameters Window



List the Parameters that your custom Stage Function uses.

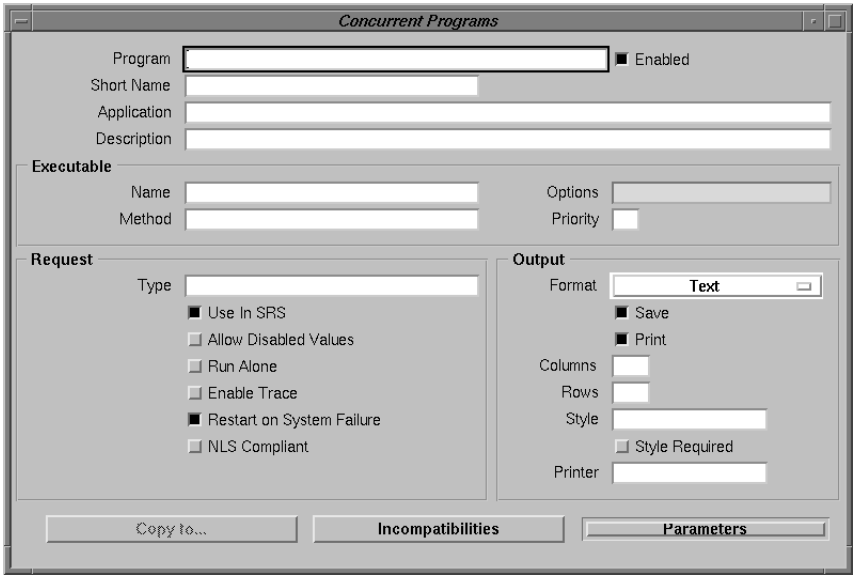
Parameter

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.

Concurrent Programs Window



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

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 – 48.

If you define a concurrent program with the bitmapped version of Oracle Reports, you must set the execution option field to "VERSION=2.0b".

You can also pass ORIENTATION=<*value*> and PAGESIZE=<*width*>x<*height*> parameters to your bitmapped Oracle Reports program. Units of width and height are set in your Oracle Reports program.

There should be no spaces before or after the execution options values. The parameters should be separated by only a *single* space.

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.
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 uesd 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 – 48.

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.

You can define a concurrent manager to run only certain types of concurrent requests. See: Concurrent Request Types, *Oracle Applications System Administrator's Guide*.

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.

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.

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.

Enable Trace

Turns on SQL tracing when program runs.

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.

Output

Format

Select the output format for your concurrent program from the following:

- Text
- HTML
- PDF
- PS (Post Script)

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

Enter Yes or No to indicate whether to allow the concurrent managers to print your program's output to a printer. If you enter No, your concurrent program's output is never sent to the printer.

The default for this field is Yes.

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.

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

Use these buttons to open detail windows for program incompatibilities your program parameters.

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.

Incompatibilities Choose this button to open the Incompatible Programs window.

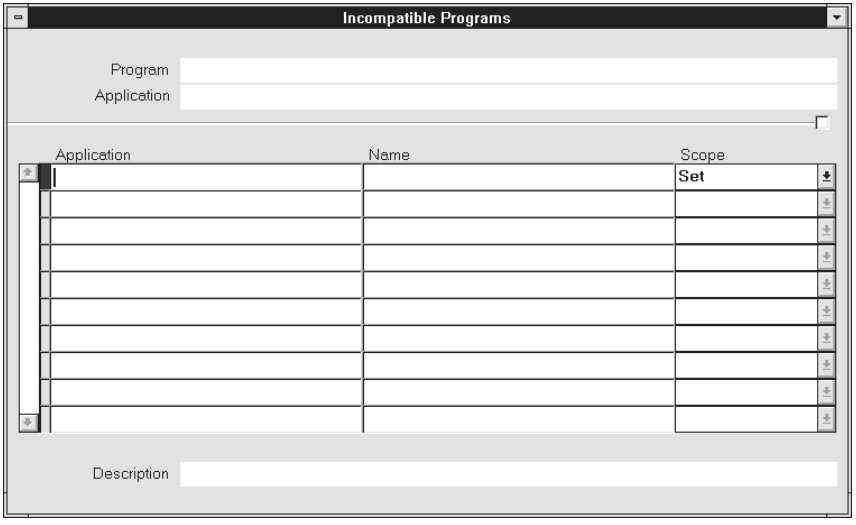
Parameters Choose this button to open the Concurrent Program Parameters window.

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.

See: Incompatible Programs Window: page 6 – 56

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. See: Administer Concurrent Managers: page 7 – 72.

Application

Although the default for this field is the application of your concurrent program, you can enter any valid application name.

Name

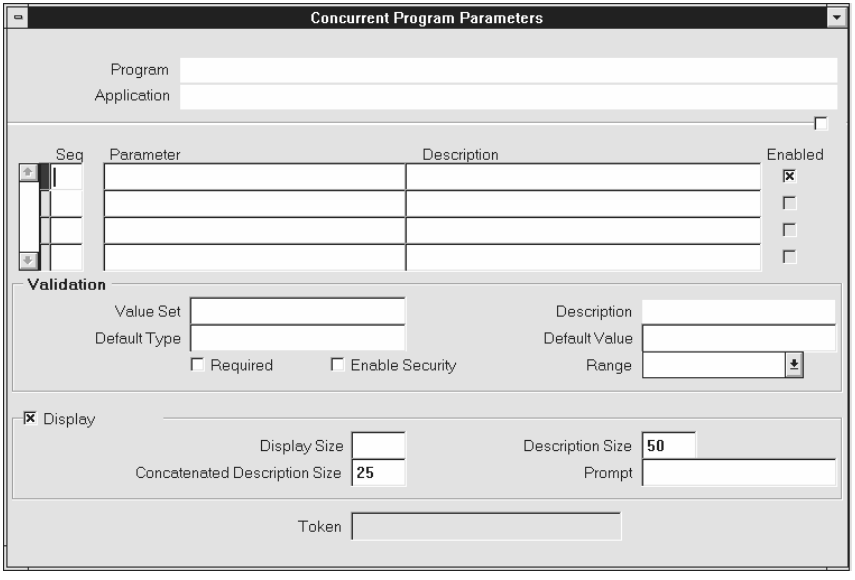
The program name and application you specify must uniquely identify a concurrent program.

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).

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.

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.

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.

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.

Current Date The default value is the current date in the format DD–MON–YY or DD–MON–YYYY, depending on the length of the segment.

Maximum Size	Date Format
9	DD–MON–YY
11	DD–MON–YYYY

Current Time The default value is the current time or the current date and time, depending on the length of the segment.

Maximum Size	Time Format
5	HH24:MI
8	HH24:MI:SS
15	DD–MON–YY HH24:MI
17	DD–MON–YYYY HH24:MI
18	DD–MON–YY HH24:MI:SS
20	DD–MON–YYYY HH24:MI:SS

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.

If you choose Current Date or Current Time, you skip the next field.

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. You do not need to include $PROFILES$ (the Release 9 format).

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.

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.

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.

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.

Prompt

A user sees the prompt instead of the parameter name in the Parameters window of the Submit Requests window.

The default is the name of the parameter.

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.

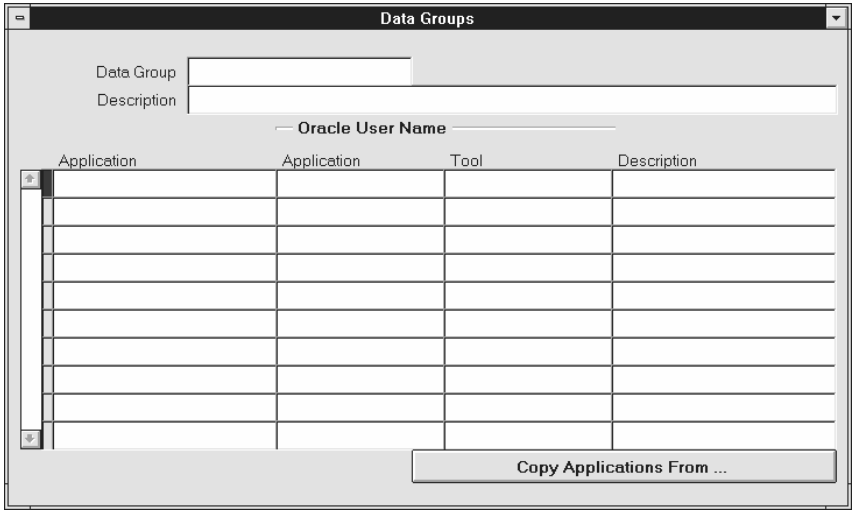
Suggestion: We recommend that you set the Concatenated Description Size for each of your parameters so that the total Concatenated Description Size for your program is 80 or less, since most video screens are 80 characters wide.

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.

See: Incompatible Programs Window: page 6 – 56

Data Groups Window



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. All data 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 Username 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 User Name

Application

Select the ORACLE username you want to assign to an application. An application uses an ORACLE username to access tables in the database. Each ORACLE username allows access to a predefined set of tables in the database.

Each responsibility within Oracle Applications is assigned to a data group. When you sign on to Oracle Applications under a given responsibility:

- Each application's programs and reports access application tables in the database using the ORACLE username assigned to it in the responsibility's data group.

Tool

A Tool ORACLE username can be added to each application–ORACLE username pair within a data group. A Tool ORACLE username identifies an ORACLE account that users can automatically connect to when using an Oracle Tool, for example, SQL*Plus. Users select the Oracle Tools window from the standard Oracle Applications menu.

Attention: The Oracle Tools window may not be available on your desktop platform. The Oracle Tools window is primary for the character mode environment.

When a responsibility is defined, a data group is chosen, and an application within the data group is selected for the responsibility's forms to connect to.

- When a Tool ORACLE username is associated with that application, users of an Oracle Tool can connect to that ORACLE account without entering a username and password.
- If a Tool ORACLE username is not associated with that application, then users of an Oracle Tool must enter a username and password to obtain access to an ORACLE account.

Attention: You cannot select a Tool ORACLE username to associate with Application Object Library. The Application Object Library–ORACLE username pair is inserted into each data group, and cannot be updated or deleted.

Suggestion: You should assign a Tool ORACLE username that is **restricted** (read–only).

Warning: Modifying data or table structures using an Oracle Tool, for example, SQL*Plus, may damage your data's integrity and is **not supported** by Oracle.

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.

CHAPTER

7

Managing Concurrent Processing

This chapter explains concurrent processing in Oracle Applications and how you can manage programs running concurrently in the background while your users continue to perform online tasks.

The essays in this chapter are organized under the following topics:

- Overview of Concurrent Processing
- Reviewing Requests and Log Files
- Changing the Status of Concurrent Requests
- Managing Concurrent Processing Files and Tables
- Concurrent Processing User Profile Settings
- Defining Managers and their Work Shifts
- Specializing Managers to run only certain programs
- Grouping Programs as a Request Type
- Controlling Concurrent Managers
- Overview of Parallel Concurrent Processing
- Managing Parallel Concurrent Processing

Form descriptions follow at the end of the chapter.

Overview of Concurrent Processing

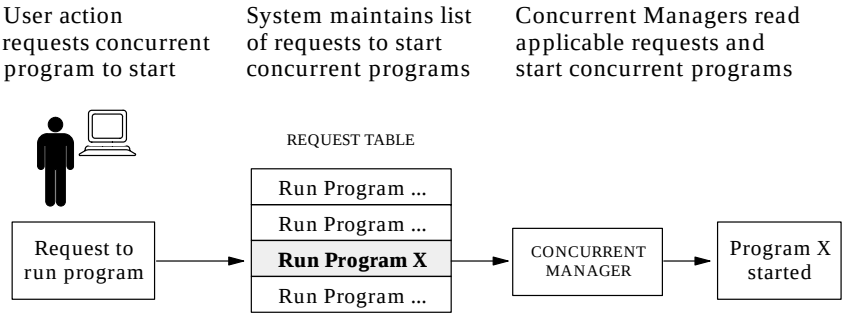
This section explains how a request to run a concurrent program is handled by Oracle Applications, and what the life cycle of a concurrent request is.

In Oracle Applications, concurrent processing simultaneously executes programs running in the background with online operations. As System Administrator, you can manage when programs are run and how many operating system processes Oracle Applications devotes to running programs in the background.

Concurrent Requests, Programs, and Processes

When a user runs a report, a request to run the report is generated. The command to run the report is a *concurrent request*. The program that generates the report is a *concurrent program*. Concurrent programs are started by a *concurrent manager.*

Figure 7 – 1



Concurrent Managers start concurrent programs

Every time your users request a concurrent program to be run, their request is inserted into a database table, and is uniquely identified by a request ID. Concurrent managers read requests from this table.

Part of a manager's definition is how many operating system processes it can devote to running requests. This number is referred to as the manager's number of *target processes.*

Running concurrent programs

A concurrent program actually starts running based on:

- When it is scheduled to start
- Whether it is placed on hold,
- Whether it is incompatible (cannot run) with other programs
- Its request priority

Concurrent Request Priorities

The priority of a concurrent request is determined by application username, and is set by the System Administrator using the Concurrent:Priority user profile option.

The first available concurrent manager compares the request's priority to other requests it is eligible to process, and runs the request with the highest priority.

When choosing between requests of equal priority, the concurrent manager runs the oldest request first.

Parent requests and Child requests

Often, several programs may be grouped together, as in a request set. Submitting the request set as a whole generates a request ID, and as each member of the set is submitted it receives its own request ID. The set's request ID identifies the *Parent* request, and each of the individual programs' request ID identifies a *Child* request.

Life cycle of a concurrent request

A concurrent request proceeds through three, possibly four, life cycle stages or *phases:*

Pending	Request is waiting to be run
Running	Request is running
Completed	Request has finished
Inactive	Request cannot be run

Within each phase, a request's condition or *status* may change. Below appears a listing of each phase and the various states that a concurrent request can go through.

Concurrent Request Phase and Status

Phase	Status	Description
PENDING	Normal	Request is waiting for the next available manager.
	Standby	Program to run request is incompatible with other program(s) currently running.
	Scheduled	Request is scheduled to start at a future time or date.
	Waiting	A child request is waiting for its Parent request to mark it ready to run. For example, a report in a report set that runs sequentially must wait for a prior report to complete.
RUNNING	Normal	Request is running normally.
	Paused	Parent request pauses for all its child requests to complete. For example, a report set pauses for all reports in the set to complete.
	Resuming	All requests submitted by the same parent request have completed running. The Parent request is waiting to be restarted.
	Terminating	Running request is terminated, by selecting *Terminate* in the Status field of the Request Details zone.
COMPLETED	Normal	Request completes normally.
	Error	Request failed to complete successfully.
	Warning	Request completes with warnings. For example, a report is generated successfully but fails to print.
	Cancelled	Pending or Inactive request is cancelled, by selecting *Cancel* in the Status field of the Request Details zone.
	Terminated	Running request is terminated, by selecting *Terminate* in the Status field of the Request Details zone.
INACTIVE	Disabled	Program to run request is not enabled. Contact your system administrator.
	On Hold	Pending request is placed on hold, by selecting *Hold* in the Status field of the Request Details zone.
	No Manager	No manager is defined to run the request. Check with your system administrator.

Table 7 – 1 (Page 1 of 1)

Reviewing Requests, Request Log Files, and Report Output Files

This essay explains how you, as System Administrator, can view and change the status of concurrent requests, and how to view request log and report output files.

How To View Request Status and Output

Use any of the following methods to view the status and output of concurrent requests.

Use the Requests Window

Use the Requests window to view the status of concurrent requests, and to view request log and report output files.

The System Administrator and Oracle Alert Manager have a privileged version of the Requests window that provides you with more capabilities than your end users. For example, using the Requests window, you can view the status of and log files for *all* concurrent requests (not just your own), including requests that completed unsuccessfully. On some platforms, you can even view the log files of running requests.

Using the same window, you can view your own report output online. You cannot, however, view report output from other users' requests.

From the Requests window, you can also:

- place and remove holds from any pending or inactive request
- cancel a pending request, or terminate a running request
- change the priority of any pending request
- view the manager log file
- determine where *any* pending request stands in the queue for each manager defined to accept the request
- determine when the Internal Concurrent Manager is inactive and needs to be restarted.

Run the Completed Concurrent Requests Report

You can run a report that lists parameters and any error messages associated with concurrent requests that have completed running. See: Completed Concurrent Requests Report: page 7– 31.

How to Modify Request Diagnostic Output

The Request Diagnostics window provides the user with request status information. This information consists of messages that explain the request's current status. You can enhance the information provided to include runtime statistics (CPU time) collected from previous runs of the request.

Collect Runtime Data

Set the profile option Concurrent:Collect Request Statistics to "Yes" to collect runtime statistics.

A concurrent request may be comprised of one or two processes: a sql*net shadow which consumes database server resources, and a front–end process such as a C executable. The time used by the CPU is collected for both of these types of processes.

Summarize and View Runtime Statistics

To review the statistics you must run the Purge Concurrent Request and/or Manager Data program to process the raw data and have it write the computed statistics to the FND_CONC_STAT_SUMMARY table. You can review the statistics on a request by request basis using the Diagnostics window from the Requests window.

Setting End User Report and Log File Access Privileges

The user profile option *Concurrent:Report Access Level* determines report output file and log file access privileges for your end users. As System Administrator, you can set this profile option to either "User" or "Responsibility".

All users can can review the log and report output files from requests that they submitted.

If you set the *Concurrent:Report Access Level* option to "Responsibility" at the User level, that user can also review the log and report output files from all requests submitted from the current responsibility.

If you set the *Concurrent:Report Access Level* option to "Responsibility" at the Responsibility level, any user of a responsibility can also view the log and report output files from all requests submitted by any other user of that responsibility.

Enabling the Report Review Agent

Using the Report Review Agent, you can copy an entire report or log file or simply one page at a time to your PC, subject to restrictions you, as the System Administrator, impose on file transfer size.

Attention: Some of the steps below require that you perform some actions on your Oracle Applications server as well as on the client. Ensure that you have run the Release 11 Server update patch on your server before proceeding.

1. Your database administrator can enable RPC capability (the Remote Procedure Calls used by the Report Review Agent) on your applications server by completing the following steps in order:

   1. Install the RPC libraries into your current ORACLE server installation. Follow the instructions in the `readme.txt` file found on the Oracle Server and Applications Server Update CD.

   2. For Applications Server Releases 10.5 and 10.6 (with RPC capability already enabled as described above), run both AutoInstall patch #350831 and concurrent manager / file server patch #351575.

2. To set up the Report Review Agent, a database or computer administrator must modify the SQL*Net configuration using Oracle Network Manager. See the 10SC Installation Guide for details on modifying your SQL*Net configuration.

3. Set the File Server:Enabled profile option to ”Yes” to invoke the Report Review Agent to access files on concurrent processing nodes. See: System Profile Values Window: page 5– 6.

4. When using a custom editor to view a concurrent output or log file, the Report Review Agent will make a ”temporary” copy of the file on the client. Set the File Server:Delete Temporary Files profile to ”Yes” to automatically delete these files when the user exits Oracle Applications. See: Defining the Reports Editor: page 7– 8, Profile Options in Oracle Application Object Library: page A – 2.

5. Set the File Server:Maximum Transfer Size profile option to specify, in bytes, the maximum allowable size of files transferred by the Report Review Agent, including those downloaded by a user with the ”Copy File...” menu option in the Oracle Applications Report File Viewer and those ”temporary” files which are automatically downloaded by custom editors.

   If this profile is null, there is no size limit. See: System Profile Values Window: page 5– 6.

6. Set up your directory tree so that the concurrent managers place all log and out file directories in the same parent directory. See the *Oracle Applications Installation Manual* for your server platform for details on setting the necessary environment variables.

   Briefly, you direct the concurrent managers to place log and report output files for all products in the same parent directory by setting

the environment variable APPLCSF on the server to the directory where all log and output files should reside. You must use names acceptable in DOS for all paths.

Note: The APPLCSF environment variable is specific to your server. The APPLCSF variable is not used on the client.

Defining the Reports Viewer

The Oracle Applications Report File Viewer is used by default for viewing your text report files. You may see PostScript reports by setting a profile option to launch a PostScript–capable viewer. Specify custom viewerss by setting the viewer profile options. Profile options can be changed using the Personal Profile Values form or the System Profile Values form (System Administrator responsibility).

Custom viewers will automatically copy the entire file to a temporary directory on the client.

See:

Setting Your Personal User Profile
(*Oracle Applications User's Guide)*

System Profile Values Window: page 5– 6

Set the Viewer:Text Profile Option

The Viewer:Text profile option must be set to a program that is on the user's PATH, or a full pathname to the executable must be used.

Set the Viewer profile option to the pathname of the executable of the viewer. This profile option can be set at the site, application, responsibility, or user level. Appending the string $$FILE$$ to the Viewer profile option causes the filename to be substituted in the Windows command. Appending $$TITLE$$ causes the window title to be substituted into the command (for viewing tools that support setting the window title).

An example of providing the full pathname:

```
Viewer:Text    c:\windows\write.exe $$FILE$$
```

Set the Viewer:Postscript Profile Option

To review PostScript reports online, set the Viewer:Postscript profile option to a PostScript previewer. You append $$FILE$$ and $$TITLE$$ just as with the Viewer: Text profile option.

```
Viewer:Postscript    c:\gs\gsview $$FILE$$
```

Set the Viewer:PDF Profile Option

To review PDF reports online, set the Viewer:PDF profile option to a PDF viewer. You append $$FILE$$ and $$TITLE$$ just as with the Viewer: Text profile option.

```
Viewer:PDF    c:\acrobat3\reader\AcroRd32.exe $$FILE$$
```

Set the Viewer:HTML Profile Option

To review HTML reports online, set the Viewer:HTML profile option to a browser. You append $$FILE$$ and $$TITLE$$ just as with the Viewer: Text profile option.

```
Viewer:PDF    c:\netscape\netscape.exe $$FILE$$
```

See: Viewing Request Output and Log Files
(*Oracle Application's User's Guide)*

Types of Log Files

Log files contain information about a concurrent program's execution, or a concurrent manager's activities. Log files are helpful when reviewing a problem request.

Log files are generated for all Completed concurrent requests.

There are three types of log files:

1. Request log files that document the execution of a concurrent program running as the result of a concurrent request. Every concurrent request generates a log file.
2. Manager Log files that document the performance of a concurrent manager that is running a request. The Manager Log file lists requests processed by a concurrent manager.
3. The Internal Concurrent Manager Log file that documents the performance of the Internal Concurrent Manager. It displays parameter values that are loaded when the Internal Concurrent Manager is started.

If a concurrent process ends in an error, you should review the log files to help diagnose the problem. You may also want to review the log files if a program's performance is questionable. For example, if a report runs very slowly or if it prints out data that you didn't expect.

The Internal Concurrent Manager Log file also records the time that each concurrent manager is started, and when each process monitor session or *pmon* cycle is initiated. During each pmon cycle, the Internal Concurrent Manager verifies the correct operation of each defined concurrent manager.

System Administrator Log File Privileges

Both you and your end users can review request log files and manager log files online. Only the System Administrator can display the Internal Concurrent Manager log file.

As System Administrator, you can use the Concurrent Requests and Administer Concurrent Program windows to view request and manager log files.

Operating System Access to Log Files

Log files are stored as standard operating system files in directories defined during the installation of Oracle Applications.

For example, Oracle General Ledger files are located using a path variable called $GL_TOP/$APPLLOG, or $APPLCSF/$APPLLOG, if the APPLCSF variable is set.

The complete path name to access an Oracle Applications log file depends on the operating system you are using. However, there are a number of file name conventions that are standard across all platforms.

For example:

Example – Request Log File name

For example, the log file naming convention in VMS and Unix is the letter *L* (*l*), followed by the *concurrent request ID*, followed by the extension *.REQ* (*.req*). In the example below, the concurrent request ID is 64225.

VMS	L64225.REQ
Unix	l64225.req

See: Concurrent Manager File Conventions
(Oracle Applications Installation Guide)

Operating System Access to Concurrent Manager Log Files

Concurrent manager log files are located in the log directory under FND_TOP, the variable that contains the path name to Application Object Library Files, or under $APPLTOP/$APPLLOG.

The concurrent manager log file naming convention in Unix is w*n*.mgr, where *n* is a number with up to 3 digits.

For most platforms, *n* is the Concurrent Process ID number assigned to the concurrent manager by the Internal Concurrent Manager, and is found in the Internal Concurrent Manager log file.

The log file name for the Internal Concurrent Manager is specified when you use the STARTMGR command from the operating system to start the concurrent managers.

See: Controlling the Internal Concurrent Manager from the Operating System: page 7– 55

System Reference Material
(Oracle Applications Installation Guide)

Operating System Access to Report Output Files

Report output files generated by concurrent programs are stored as standard operating system files in directories defined during the installation of Oracle Applications.

Path name to Output Files

The complete path name to access an Oracle Applications report output file depends on the operating system you are using. However, there are a number of file name conventions that are standard across all platforms.

- Each output file name includes the unique request ID assigned by the concurrent processing facility.

Example – Report Output File name

For example, the output file naming convention in VMS and Unix is *Application Username.Request ID*. Oracle Applications uses the first 8 characters of the application username in the output file name. In the example below, the application username is JSMITH and the concurrent request ID is 64225.

For example:

VMS and UNIX JSMITH.64225

See: Concurrent Manager File Conventions
(Oracle Applications Installation Guide)

Changing the Status of Concurrent Requests

This essay explains how to change a request's phase and status, and how to change the priority of a Pending or Inactive request.

Changing a Request's Phase and Status

A request is in one of four phases: Pending (waiting to be run), Running, Completed, or Inactive (unable to run). Within each phase, a request's condition is referred to as its status.

You can change the phase of a Pending, Running, or Inactive request by changing its status.

Pending and Inactive Requests

You may cancel Pending and Inactive requests. The request's phase and status becomes *Completed – Cancelled.*

You may place on hold Pending and Inactive requests. The request's phase and status becomes *Inactive – On Hold.* You can reverse this action by later selecting the request removing the hold.

Running Requests

You can terminate Running requests. The request's phase and status becomes *Completed – Terminated.*

Changing a Request's Status

You can change the status of a request, and its resulting phase, using the Requests window.

Changing the Priority of a Pending or Inactive request

Requests normally run according to start time, on "first–submitted, first–run" basis. However, a higher priority request starts before an earlier request.

As System Administrator, you can change the priority of any Pending or Inactive request using the Requests window.

Request Priority is associated with an application User

The priority of a user's requests defaults to the value you, as System Administrator, set for their *Concurrent:Priority* user profile option. Users cannot change the priority of their requests.

If a concurrent program has a defined priority, that priority overrides the user's profile option.

- Priorities range from 1 (highest) to 99 (lowest).
- The standard default is 50.
- Concurrent programs submitted by the Internal Concurrent Manager have a priority of zero (0), and override all other requests.

Suggestion: If you need to change the priority of a request frequently, you should consider assigning that concurrent program its own priority.

Managing Concurrent Processing Files and Tables

This section explains how to maintain the number of log and output files the operating system retains, and how to manage Application Object Library database tables that store information about concurrent requests and concurrent manager processes.

The database tables that are affected by running the Purge Concurrent Request and / or Manager Data program are:

FND_CONCURRENT_REQUESTS

This table contains a complete history of all concurrent requests.

FND_RUN_REQUESTS

When a user submits a report set, this table stores information about the reports in the report set and the parameter values for each report.

FND_CONC_REQUEST_ARGUMENTS

This table records arguments passed by the concurrent manager to each program it starts running.

FND_DUAL

This table records when requests do not update database tables.

FND_CONCURRENT_PROCESSES

This table records information about Oracle Applications and operating system processes.

FND_CONC_STAT_LIST

This table collects runtime performance statistics for concurrent requests.

FND_CONC_STAT_SUMMARY

This table contains the concurrent program performance statistics generated by the Purge Concurrent Request and / or Manager Data program. The Purge Concurrent Request and / or Manager Data program uses the data in FND_CONC_STAT_LIST to compute these statistics.

Maintenance Suggestions

Your MIS department and application users should agree on an archiving and file retention policy that is appropriate for your organization. To avoid running out of space on your disk drives, you should periodically delete Oracle Applications log files and output files.

Suggestion: You can run the program "Purge Concurrent Request and/or Manager Data" once and automatically resubmit the program for you at specific time intervals.

There are some sample guidelines for when to run the Purge Concurrent Requests and/or Manager Data program. Adopt these guidelines according to your user community's usage of Oracle Applications.

- every 30 days for normal usage
- every two weeks (14 days) for heavy usage
- if using the AGE mode, set the Mode Value to 5 to retain the five most recent days of concurrent request data, log files, and report output files.

Purging loses Audit data

When you purge concurrent request information, you lose audit details. The Signon Audit Concurrent Requests report uses this audit information.

Purge Concurrent Request and/or Manager Data Program

Use this program to delete:

- request log files, concurrent manager log files, and report output files from your product directories maintained by the operating system
- records (rows) from Application Object Library database tables that contain history information about concurrent requests and concurrent manager processes.

Use this program to compute performance statisics for each of the concurrent programs, if the Concurrent: Collect Request Statistics profile option is set to ”Yes”.

Report Options

Entity

All	Purges records from database tables that record history information for concurrent requests, history information for concurrent managers, and purges request log files, manager log files, and report output files from the operating system.
Manager	Purges records from database tables that record history information for concurrent managers, and purges manager log files from the operating system.
Request	Purges records from database tables that record history information for concurrent requests, and purges request log files and report output files from the operating system.

Mode

Age	Enter the number of days for which you want to save concurrent request history, log files, and report output files. The purge program deletes all records older (in days) than the number you enter. For example, if you enter ”5”, then all concurrent request history, log files, and report output files older than five days is purged.

Count

Enter the number of (most recent) records for which you want to save concurrent request history, log file, and report output files. The purge program starts from the most recent records, retains the number you enter, and purges all remaining records.

For example, if you enter ”5”, then the five most recent concurrent request history records, request log files, manager log files, report output files are saved, and all remaining records are purged.

Mode Value

Enter a value to define the number of days for Mode=Age or the number of records for Mode=Count. The valid values are 1 – 9999999.

User Name

Enter the application username whose concurrent request records and associated log files and report output files you wish to purge. Username has relevance when the Entity is either ”Request” or ”All”.

For example, if you enter JSMITH, then the program purges all request records, log files, and report output files associated with requests submitted by user JSMITH.

Oracle ID

Enter the Oracle ID that concurrent programs connect to for which you want to purge concurrent request records, and associated log files and report output files. Oracle ID has relevance when the Entity is either ”Request” or ”All”.

For example, if you enter AP1, then the program purges all request records, log files, and report output files associated with requests to run programs that connect to the AP1 Oracle ID.

Program Application

Select the application for which you want to purge concurrent request records, and associated log files and report output files. Program Application has relevance when the Entity is either ”Request” or ”All”.

For example, if you select Oracle Payables, then the program purges all request records, log files, and report output files associated with requests to run Oracle Payables programs.

Program

Select the program for which you want to purge concurrent request records, and associated log files and report output files. Program has relevance when the Entity is either ”Request” or ”All”.

For example, if you select Program X, then the purge program purges all request records, log files, and report output files associated with requests to run Program X.

Manager Application

Select the application associated with the concurrent manager for which you want to purge concurrent request records, and associated log files and report output files.

Manager Application is used with the *Manager* option, and has different effects when Entity is set to ”Request, and when Entity is set to ”Manager” or ”All”.

- When Entity is set to ”Request”, the program purges all request records, log files, and report output files associated with requests run by the concurrent manager named in the *Manager* option.
- When Entity is set to either ”Manager” or ”All”, in addition to the above, the program also purges all manager log files associated with the concurrent manager named in the *Manager* option.

Manager

Select the concurrent manager for which you want to purge concurrent request records, and associated log files and report output files.

Manager is used with the *Manager Application* option, and has different effects when Entity is set to ”Request,” and when Entity is set to ”Manager” or ”All”.

- When Entity is set to ”Request”, the program purges all request records, log files, and report output files associated with requests run by the concurrent manager named in the *Manager* option.
- When Entity is set to either ”Manager” or ”All”, in addition to the above, the program also purges all manager log files associated with the concurrent manager named in the *Manager* option.

Resp. Application

Select the application associated with the responsibility for which you want to purge concurrent request records, and associated log files and

report output files. Responsibility Application is used with the *Responsibility* option, and has relevance when the Entity is either ”Request” or ”All”.

Responsibility

Select the responsibility for which you want to purge concurrent request records, and associated log files and report output files. Responsibility has relevance when the Entity is either ”Request” or ”All”.

For example, if you select the System Administrator responsibility, then the program purges all request records, log files, and report output files associated with requests submitted by users operating under the System Administrator responsibility.

Report

Select whether you want a report listing the number of records purged by the Purge Concurrent Request and / or Manager Data program.

No	Run the program but do not generate a report.
Yes	Run the program and generate a report.

Concurrent Processing User Profile Settings

This essay explains the user profile option settings relevant to submitting concurrent requests.

Setting Concurrent Processing Options

End users can control certain runtime options for their concurrent requests. For example, you can choose a specific date on which to start a request.

If a user does not explicitly enter these options at the time of the request, concurrent processing options default to their user profile values.

As System Administrator, you set user profile values for your end users with the System Profile Values window. Both you and your end users can set some of your own profile values using the Personal Profile Values form.

Changing Concurrent Processing Options for submitted requests

You or your users can use the Requests window to change the concurrent processing options for a submitted request up until the time it starts running.

- As System Administrator you can change all concurrent options for any request.
- Your users can change most of their request's concurrent options.

  End users cannot change (nor set) the priority of their request, or the report access level for viewing request log files and report output files online.

See: Overview of Setting User Profiles: page 5– 2

Concurrent Processing User Profile Options

User Profile Option	Explanation
Concurrent: Hold Requests	"Yes" places concurrent requests on hold. "No" starts programs according to the request's priority and start time.
Concurrent: Multiple Time Zones	"Yes" ensures that requests are scheduled immediately regardless of the time zone your client is running in.

Table 7 – 2 (Page 1 of 2)

User Profile Option	Explanation
Concurrent: Report Access Level	Viewing a request's output/log files online and reprinting reports can be accessed according to: "Responsibility" – by anyone using the responsibility that submitted the request "User" – by only the user who submitted the request.
Concurrent: Report Copies	The number of output copies that print for each report.
Concurrent: Request Priority	Requests normally run according to start time, on a "first–submitted, first–run" basis. Priority overrides request start time. A higher priority request starts before an earlier request. Priorities range from 1 (highest) to 99 (lowest). The standard default is 50.
Concurrent: Request Start Time	The date and time requests are available to start running. If the start date and time is at or before the current date and time, requests may be run immediately.
Concurrent: Save Output	"Yes" saves concurrent program outputs in a standard file format. Some concurrent programs do not generate an output file.
Concurrent: Sequential Requests	"Yes" forces requests to run one at a time (sequentially) according to the requests' start dates and times. "No" means requests can run concurrently when their concurrent programs are compatible.
Concurrent: Wait for Available TM	You can specify the maximum number of seconds that the client will wait for a given transaction manager (TM) to become available before moving on to try a different TM.
Concurrent: URL Lifetime	This profile option determines the length of time in minutes a URL for a request ouput is retained before it is deleted from the system.
Concurrent: Use ICM	"Yes" enables the Internal Concurrent Manager to resolve request conflicts instead of the Conflict Resolution Manager.
Printer	The printer which prints your reports.

Table 7 – 2 (Page 2 of 2)

Updating Concurrent Request Profile Options

Most concurrent user profile options may be set by the System Administrator at all four levels: site, application, responsibility, and user. The user profile *Concurrent:Report Access Level* may not be set at the application level.

Your users can change the default values for most of the concurrent processing profile options. However, they cannot set Concurrent: Request Priority, Concurrent:Use ICM, or Concurrent: Report Access Level.

Defining Managers and their Work Shifts

This essay 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– 80.

In this essay, 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.

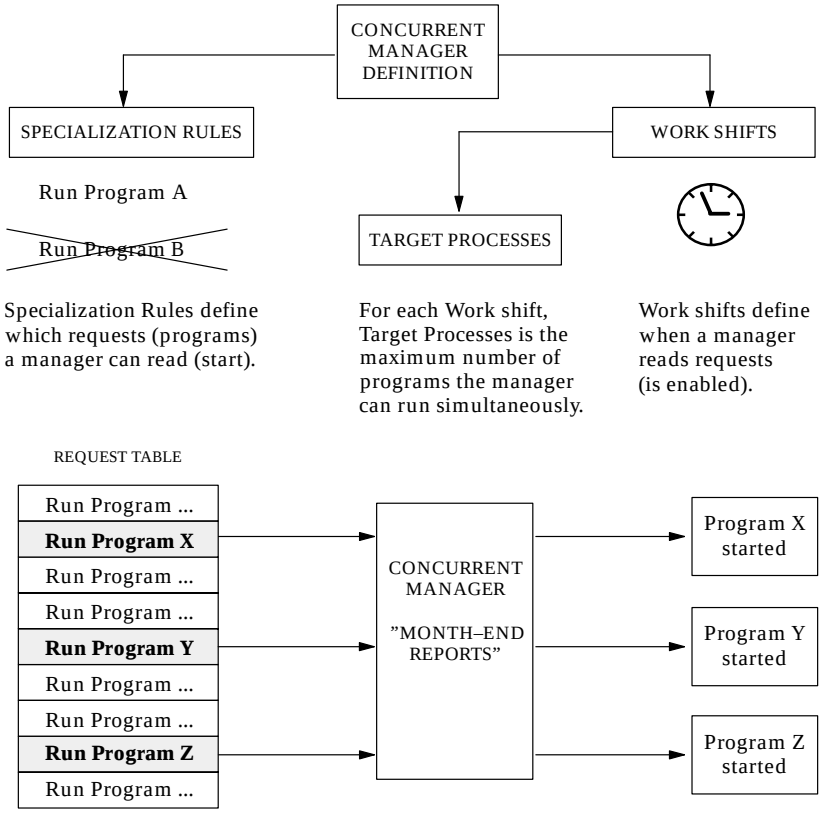
- For each work shift, you define the maximum number of operating system processes the manager can run concurrently to read requests (start programs) during the work shift.

- Specialize your manager to read only certain kinds of requests.

Figure 7 – 2 illustrates the details of defining a concurrent manager.

Figure 7 – 2

Defining a Concurrent Manager



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 Library can 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.

  The Internal Concurrent Manager also enforces program incompatibility rules by comparing program definitions for requested programs with those programs already running in an Oracle username designated as a logical database (i.e., an Oracle account where program incompatibility rules are enforced).

  You cannot alter the definition of the Internal Concurrent Manager.

See: Defining Program Incompatibility Rules: page 6– 23

- 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.

A transaction manager is associated with a particular data group, and uses that data group to connect to the database. Transaction managers can only process requests submitted from responsibilities associated with the same data group.

If you create custom data groups, you should define new transaction managers (using the predefined program libraries associated with the seeded transaction managers) for each application in your data group that uses transaction managers.

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.

See:

Work Shifts: page 7– 86

Work Shift by Manager Report: page 7– 32

Work Shifts Report: page 7– 33

For example, you can define work shifts such as:

- 8:00am–5:00pm, Monday–Friday.
- 11:00am–1:00pm, Wednesday(s).
- 6:00pm–11:59pm, April 15, 1994.

You can define a work shift to run during the night, when most or all of your employees are at home asleep, and are not using their terminals. For example, you can define a work shift as:

- 2:00am–6:00am, Monday–Friday.

You can define a work shift to run twenty–four hours a day on a certain day or days of the week, or on a specific date. For example, you can define a work shift as:

- Monday–Friday.
- Wednesday(s).
- April 15, 1994.

You can define work shifts to use only on special occasions. For example, you can define a work shift named ”Inventory” to use when your company is conducting an inventory.

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 on 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.

Table 7 – 3 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.

Overlapping Work Shift Priorities

Priority	Work Shift Definition	Example
1	Specific date and range of times	April 15, 1994 8:00am–5:00pm
2	Specific date and no range of times	April 15, 1994
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.

Table 7 – 3 (Page 1 of 1)

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.

Using Work Shifts to Balance Processing Workload

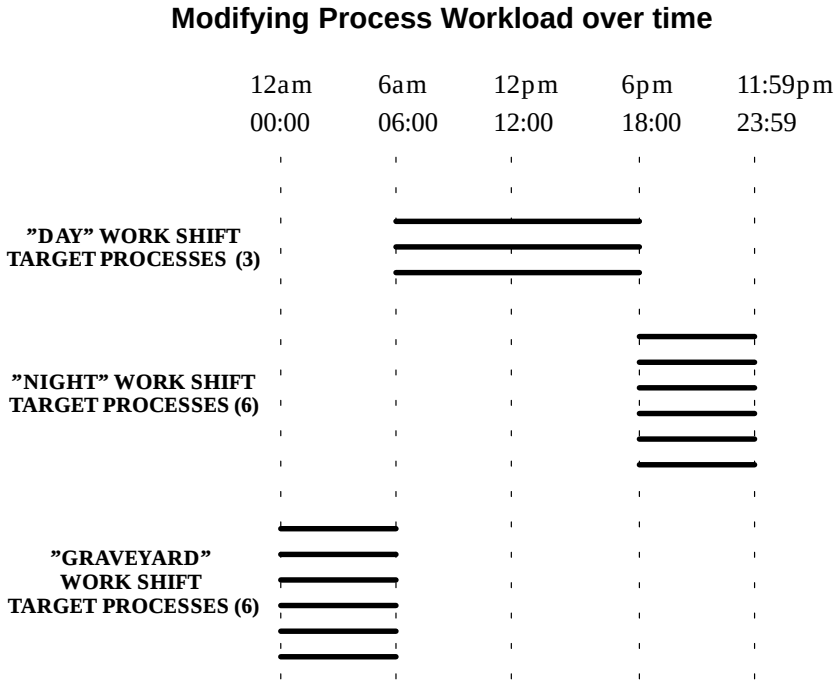
Part of a manager’s definition is how many operating system processes it can devote to reading requests. For each of these processes, referred to as a *target process*, a manager can start one concurrent program.

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.

Figure 7 – 3



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– 31.

Suggestion: 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.

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 dates in DD–MON–YY format.

Report Headings

The report headings list the specified parameters and provide you with general information about the contents of the report.

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.

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.

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– 80.

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– 88.

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– 36.

Examples are presented in on page 7– 36.

Specialization Rule Logic – Examples

Include Rules	Result
Include *X*	Run only program X
Include *X* OR Include *User Sam*	Run program X ...or Run requests by User Sam Net result: Run everyone's requests for program X, and run all of Sam's requests.

Table 7 – 4 (Page 1 of 1)

Exclude Rules	Result
Exclude *37*	Do not run program 37
Exclude *37* AND Exclude *User Sam*	Do not run program 37 ...and 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.

Table 7 – 5 (Page 1 of 1)

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
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.

Table 7 – 6 gives examples of the action types associated with specialization rules.

Examples – Specialization Rule Types

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".
	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.
	Program	Oracle Project Accounting – Sales Forecast	Manager *only* reads requests to start the concurrent program named "Sales Forecast".
	Request Type	Oracle Inventory – Overnight Reports	Manager *only* reads requests to start programs belonging to the request type "Overnight Reports".
	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".
	ORACLE ID	APPS2	Manager reads *all* requests to start programs *except* those that connect to the APPS2 Oracle ID.
	Program	Application Object Library – Purge Audit Tables	Manager reads *all* requests to start programs *except* requests for the program named "Purge Audit Tables".
	Request Type	Oracle Purchasing – Weekend Programs	Manager reads *all* requests to start programs *except* those belonging to the request type "Weekend Programs".
	User	Margaret	Manager reads *all* requests to start programs *except* those submitted by the application user "Margaret".

Table 7 – 6 (Page 1 of 1)

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”.

Simplify 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. This is useful when working with multiple installations of an application and data groups.

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.

Suggestion: 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).

Suggestion: 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”.

Suggestion: 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.

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– 88.

Combined specialization rule actions, their binding logic, and examples are presented in the following table.

Combined Specialization Rule Logic – Examples

Combination Rule Include Lines	Result
Include *Program X*	Run only program X
Include *Program X* AND Include *User Sam*	Run program X ...and Run requests by User Sam Net result: Run only Sam's requests for program X.

Table 7 – 8 (Page 1 of 1)

Combination Rule Exclude Lines	Result
Exclude *Program 37*	Do not run program 37
Exclude *Program 37* AND Exclude *User Sam*	Do not run program 37 ...and 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.

Table 7 – 7 (Page 1 of 1)

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
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.

Table 7 – 8 (Page 1 of 1)

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.

Suggestion: 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*.

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.

Using Multiple Actions

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...*
	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...*
	INCLUDE	With each additional Include line or additional require-ment, the manager reads FEWER REQUESTS.	Each line within a rule establishes an AND condition. *AND...INCLUDE...*

Table 7 – 9 (Page 1 of 1)

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.

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.

The Internal Concurrent Manager activates and deactivates individual managers and enforces run–alone program and program incompatibility rules. See: Controlling Managers: page 7– 54.

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).

Attention: 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– 72.

Controlling Managers

Manager	Control Function	Description
Internal Manager	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.
Any Other Manager	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".
	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".

Table 7 – 10 (Page 1 of 1)

Controlling the Internal Concurrent Manager from the Operating System

There are two 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.

Table 7 – 11 compares the Internal manager control states displayed by the Administer Concurrent Managers form with their corresponding operating system command. Not all arguments are shown.

Controlling the Internal Manager from the OS

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

Table 7 – 11 (Page 1 of 1)

Starting the Internal Concurrent Manager from the Operating System

To start the concurrent managers, you can invoke the STARTMGR command from your operating system prompt. This command starts the Internal Concurrent Manager, which in turn starts any concurrent managers you have defined.

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.

See: *Oracle Applications Installation Guide* for your operating system

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 is platform–dependent, so refer to your *Oracle Applications Installation Guide* for the specific syntax to use.

See: Concurrent Managers – Appendix: System Reference Material, *Oracle Applications Installation Guide* for your operating system

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="applsys/fnd"  mgrname="std"
 printer="hqseq1"  mailto="jsmith"  restart="N"
 logfile="mgrlog"  sleep="90"  pmon="5"  quesiz="10"
```

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 applsys/pwd 'Responsibility application shortname'
 'Responsibility name' 'Username' [WAIT={Y|N|n}] CONCURRENT
 'Program application shortname'  PROGRAM
```

Parameters

applsys/pwd	The ORACLE username and password that connects to Oracle Application Object Library data.
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.

Example Sequence

For example, you can write a shell script for your particular operating system that deactivates the Internal manager (and all the other managers) before shutting down, backing up, and restarting the database. You can also incorporate the STARTMGR command into the shell script to start up the Internal manager.

See: Controlling the Internal Concurrent Manager from the Operating System: page 7– 55

1. Shell script customized for specific operating system starts.
2. `CONCSUB` *`applsys/pwd`* `SYSADMIN 'System Administrator' SYSADMIN WAIT=Y CONCURRENT FND 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="applsys/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 username/password are still supplied, the CONCSUB utility will work as usual.

If username only is supplied (no '/pwd' in the first argument), it will prompt you for the password:

```
ORACLE Password:
```

The echo is turned off. For example, the command below does not include the ORACLE Password.

```
CONCSUB applsys SYSADMIN 'System Administrator' SYSADMIN
 CONCURRENT FND
FNDMNRMT Y 0 20221
ORACLE Password:
Submitted request 32157 for CONCURRENT FND FNDMNRMT Y 0
20221
```

Now, the first argument has to be the application username as usual (for example, SYSADMIN).

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 applsys 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 homogeneous 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 homogeneous networked environment.

Parallel Concurrent Processing Environments

Parallel concurrent processing runs in multi–node environments, such as cluster, massively parallel, and homogeneous 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. Typical cluster environments include IBM HACMP, VAX Cluster, or a cluster of Sequent servers.

With parallel concurrent processing in a cluster environment, a single ORACLE database resides in the common disk pool, while multiple instances of Oracle Parallel Server 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 Oracle Parallel Server instances run simultaneously on multiple nodes, with multiple concurrent managers also distributed across nodes.

Homogeneous Networked Environments

In homogeneous 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 homogeneous 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 Oracle Parallel Server.

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 SQL*Net to a node that is running ORACLE.

To each concurrent manager, you assign a primary and a secondary node. Initially, a concurrent manager is started on its primary node. In case of node or ORACLE instance failure, a concurrent manager migrates to its secondary node.

A concurrent manager 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 the same or other 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 fail over protection.

Internal Monitor Processes, like concurrent managers, can 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 on–line from any other node. Users need not log onto a node to view the log and output files from requests run on that node. See: Database Instances, Manager Location, and File Distribution: page 7– 66.

This capability relies on setup steps taken at install time. For more information, refer to the installation documentation for your platform.

Integration with Platform–Specific Queuing and Load–Balancing Systems

Some cluster or massively parallel systems have their own mechanisms for queuing batch processes or distributing process loads—for example, VMS batch queues or IBM LoadLeveler. Because users may wish to manage all processing, not just Oracle Applications processing, using these mechanisms, 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 and load–balancing systems, refer to the installation documentation for your platform.

Managing Parallel Concurrent Processing

This essay describes how to manage parallel concurrent processing from System Administration forms. It presents the following topics, each in the context of parallel concurrent processing:

Defining Concurrent Managers

You define concurrent managers using the Concurrent Managers window. When you define a manager, you specify the manager type, which may be either ”Concurrent Manager”, or ”Internal Monitor”. There is a third manager type, ”Internal Concurrent Manager”, that describes the Internal Concurrent Manager process that Oracle Applications predefines for you.

Attention: When using parallel concurrent processing, manager names cannot have embedded spaces in them. Name your managers using one word, or connect two words using a hyphen (manager–name) or underline (manager_name).

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– 80.

Administering Concurrent Managers

Target Nodes

Using 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.

When a manager’s 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 Administer Concurrent Managers form, you can start up, shut down, migrate, 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 Manger from any node in your parallel concurrent processing environment.

Starting Up Managers

You start up parallel concurrent processing by issuing an "Activate" command against the Internal Concurrent Manager from the Administer Concurrent Managers form, or by invoking the STARTMGR command from the operating system prompt. Regardless of the node from which you activate the Internal Concurrent Manager, it starts up on its assigned node (assuming that you operate from a node whose platform supports remote process startup.)

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 "Deactivate" command against the Internal Concurrent Manager from the Administer Concurrent Managers form. All concurrent managers and Internal Monitor processes are shut down before the Internal Concurrent Manager shuts down.

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 effect your changes, you issue a "Verify" command against the Internal Concurrent Manager from the Administer Concurrent Managers form.

Terminating a Concurrent Process

You can terminate a running concurrent process on the local node or on remote nodes by issuing a "Terminate" command from the Administer Concurrent Managers form.

Administer Concurrent Managers: page 7– 72

Controlling the Internal Manager from the Operating System: page 7– 55

Concurrent Managers: page 7– 80

Database Instances, Manager Location, and File Distribution

The following pages illustrate some example configurations for parallel concurrent processing.

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 SQL*Net to a node that is running ORACLE.

Concurrent program executable files, along with log and report output files, can be stored on a node's local disk. Alternatively, these files can be stored in one central location, and read remotely from other nodes.

Examples of Parallel Concurrent Processing

Parallel Concurrent Processing – Single Database Instance with Centralized and Shared Log / Output / Executable File System: page 7– 67

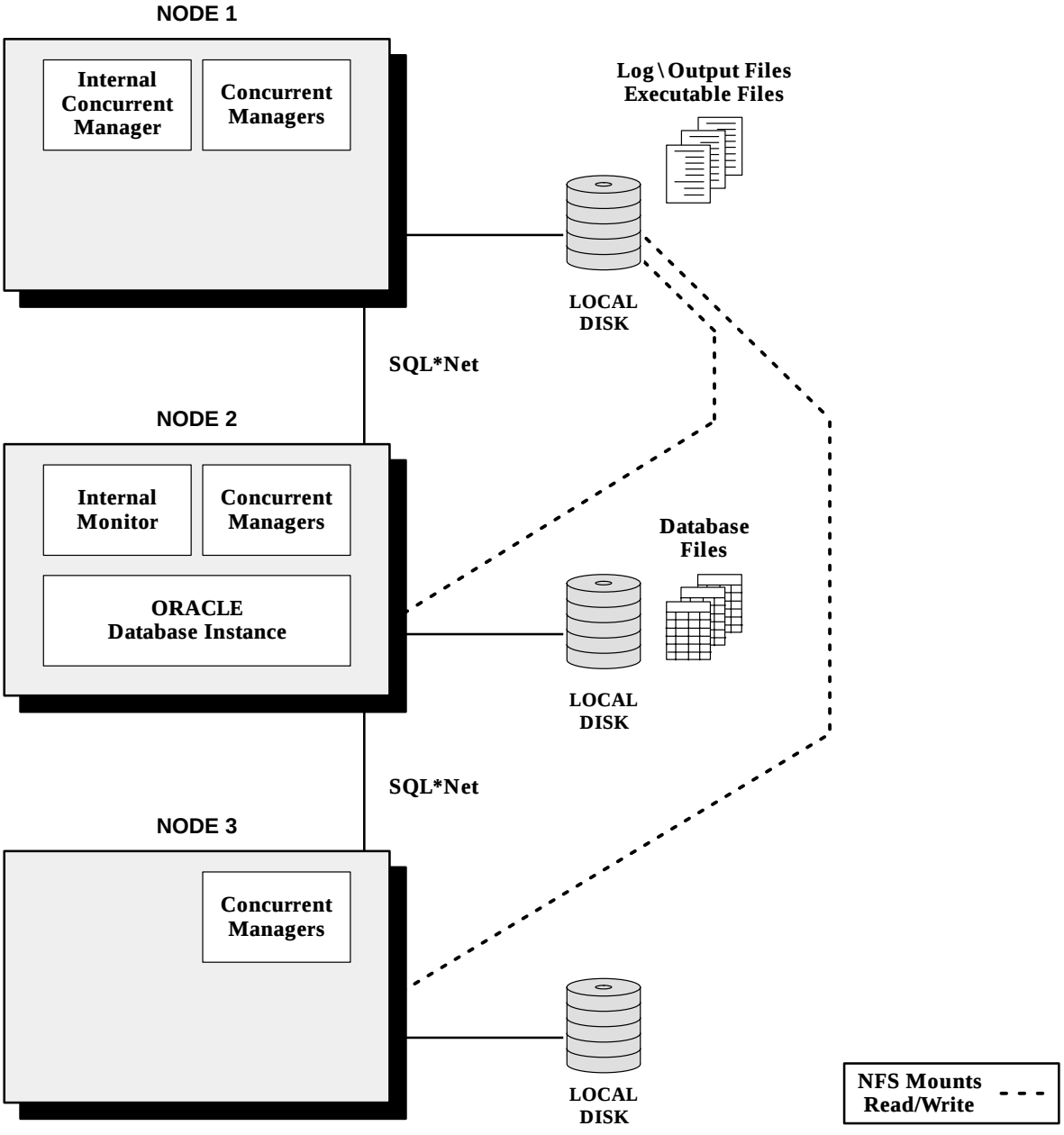
Parallel Concurrent Processing – Oracle Parallel Server with Centralized and Shared Log / Output / Executable File System: page 7– 68

Parallel Concurrent Processing – Single Database Instance with Distributed Log / Output / Executable Files: page 7– 69

Parallel Concurrent Processing – Node 1 acts as Database Server, Nodes 2–5 act as Concurrent Manager Servers: page 7– 70

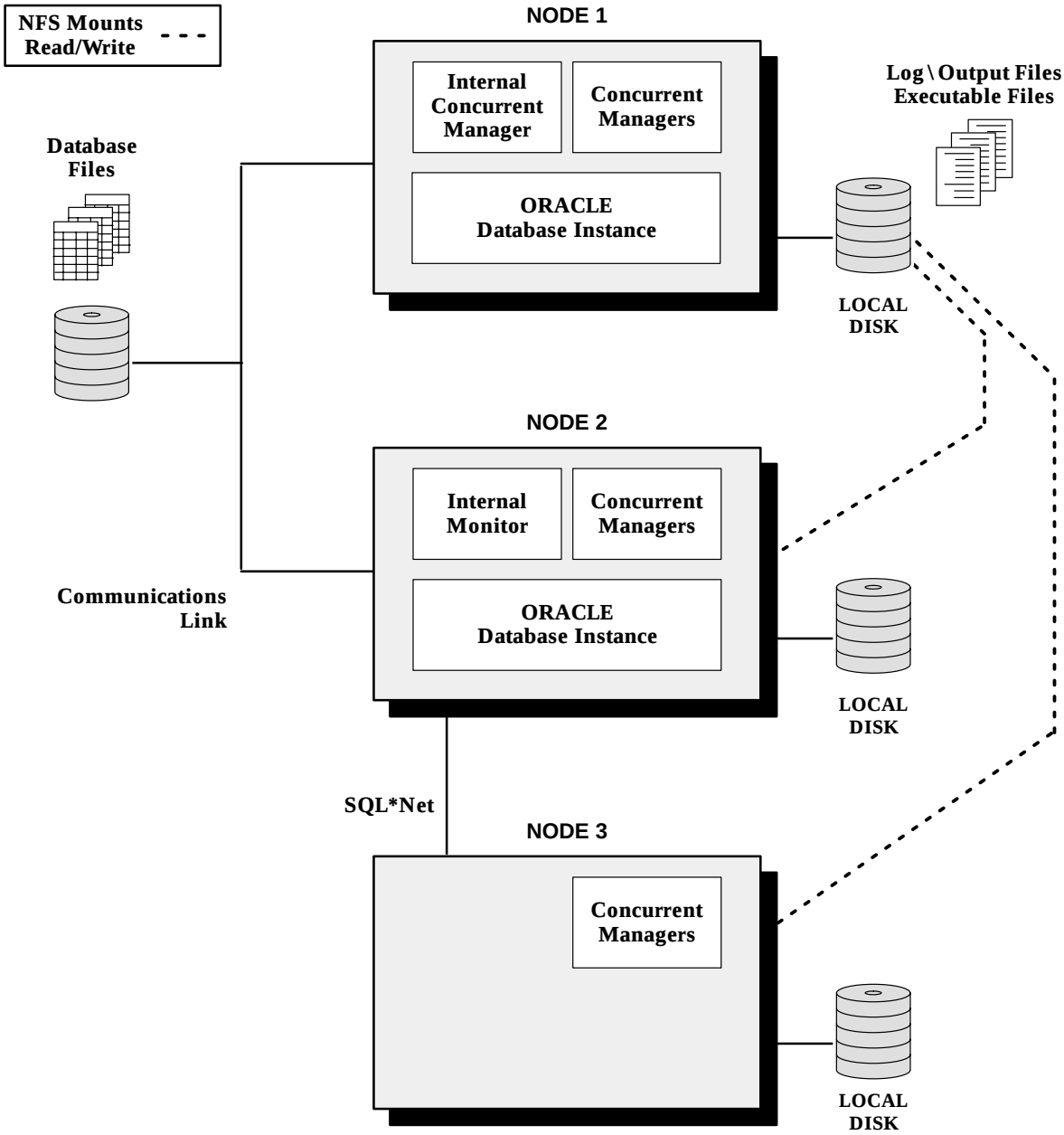
Parallel Concurrent Processing – Single Database Instance with Centralized and Shared Log/Output/Executable File System

Figure 7 – 4



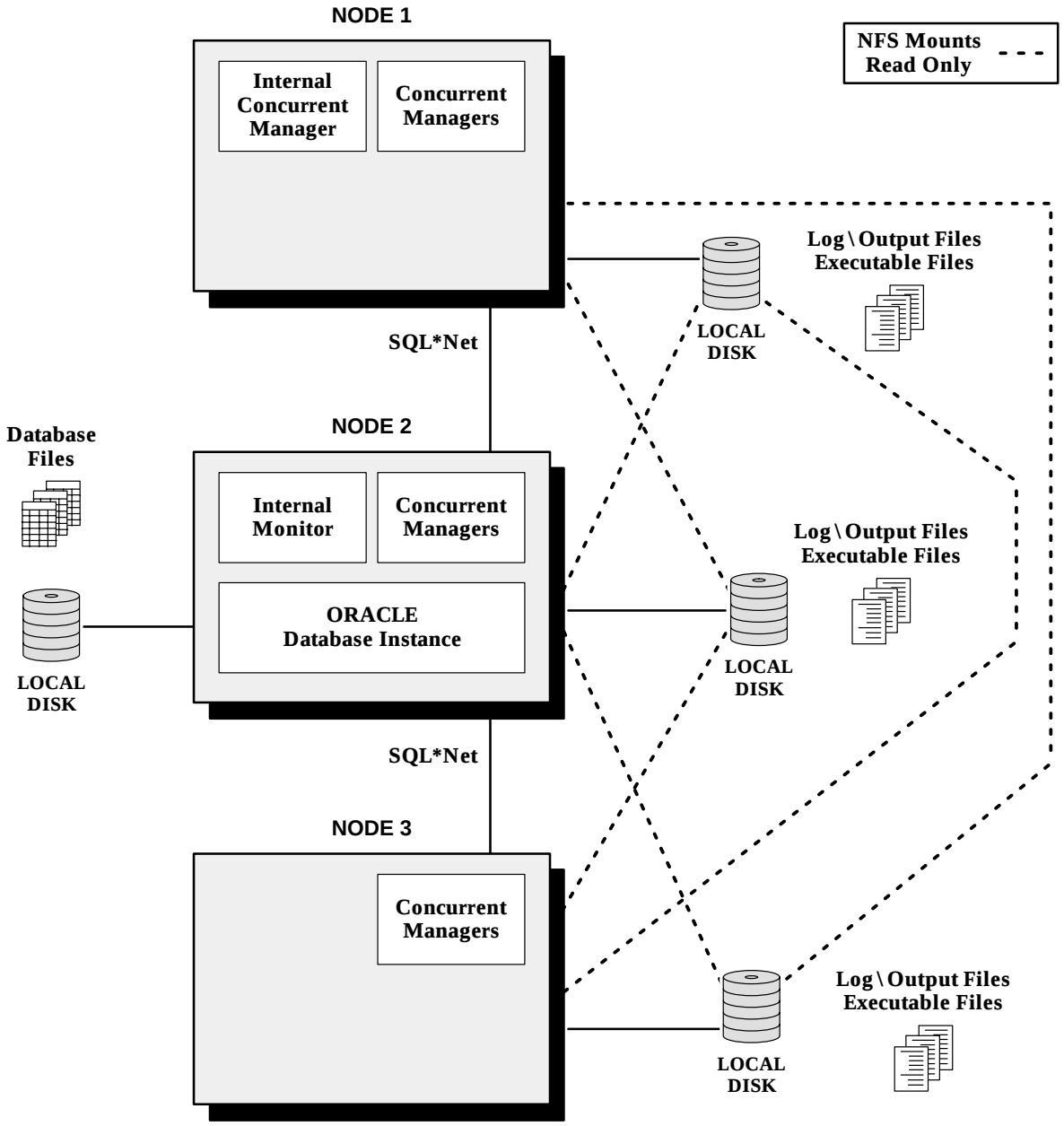
Parallel Concurrent Processing – Oracle Parallel Server with Centralized and Shared Log/Output/Executable File System

Figure 7 – 5



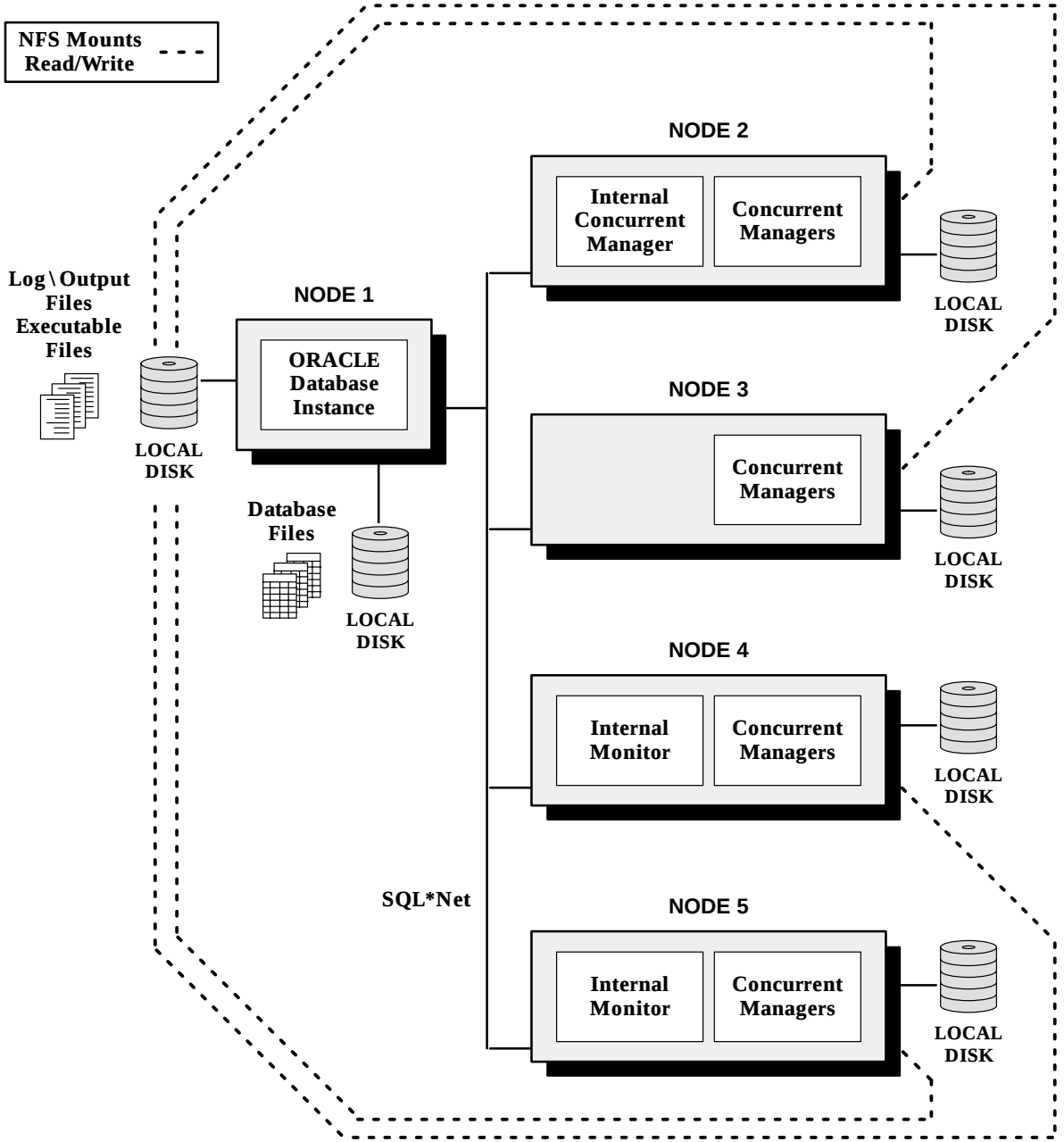
Parallel Concurrent Processing – Single Database Instance with Distributed Log/Output/Executable Files

Figure 7 – 6



Parallel Concurrent Processing – Node 1 acts as Database Server, Nodes 2–5 act as Concurrent Manager Servers

Figure 7 – 7



Installing Parallel Concurrent Processing

The following checklist summarizes the steps for installing parallel concurrent processing. Implementing the following steps may vary, depending on your operating platform. Please refer to the *Oracle Applications Installation Guide* for your operating system.

See: Parallel Concurrent Processing
(Oracle Applications Installation Guide)

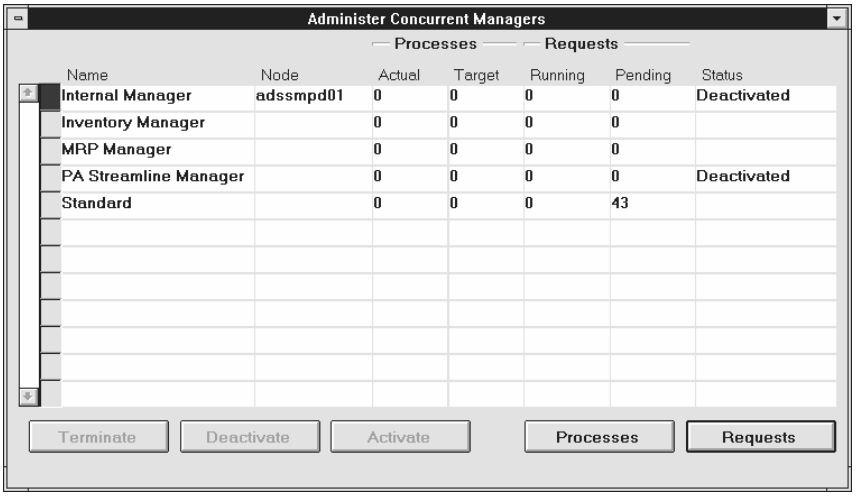
- ❑ Set up **applmgr** logins on each node.
- ❑ Create symbolic links between directories on each node.
- ❑ Create necessary grants and synonyms.
- ❑ Run AutoInstall.

  If you have already installed the current release of Oracle Applications software, you can set up parallel concurrent processing by regenerating environment files with the **adadmin** utility to define the appropriate environment variables.
- ❑ Install product files on additional nodes if necessary.
- ❑ Create environment files on parallel nodes.
- ❑ Run **afimpmon.sql** and edit **dcpstart** on each node.
- ❑ Define the concurrent managers.

Attention: When using parallel concurrent processing, manager names cannot have embedded spaces in them. Name your managers using one word, or connect two words using a hyphen (manager–name) or underline (manager_name).

See: Managing Parallel Concurrent Processing: page 7– 64

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.

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 which the manager submits its processes to.

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 lack of requests, 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 than the 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.

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.

Controlling a Specific Manager

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.

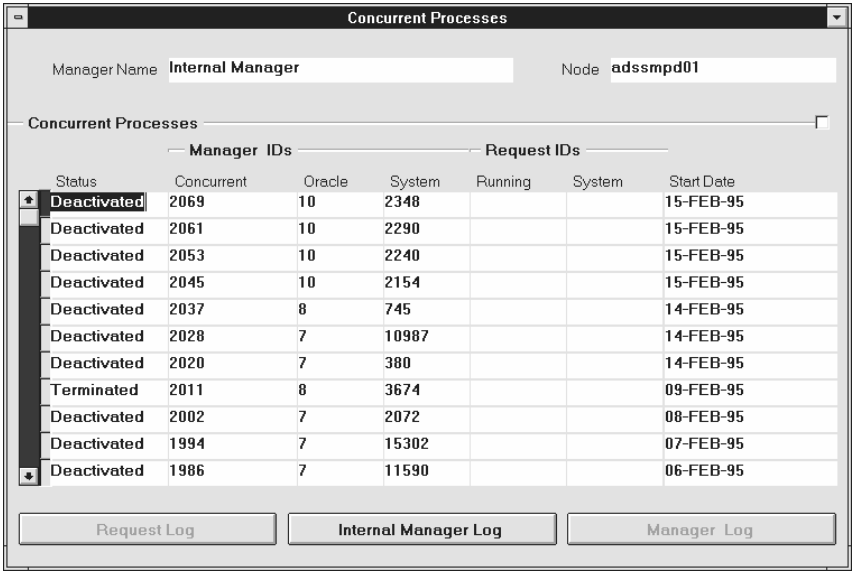
Reviewing a Specific Manager

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.

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

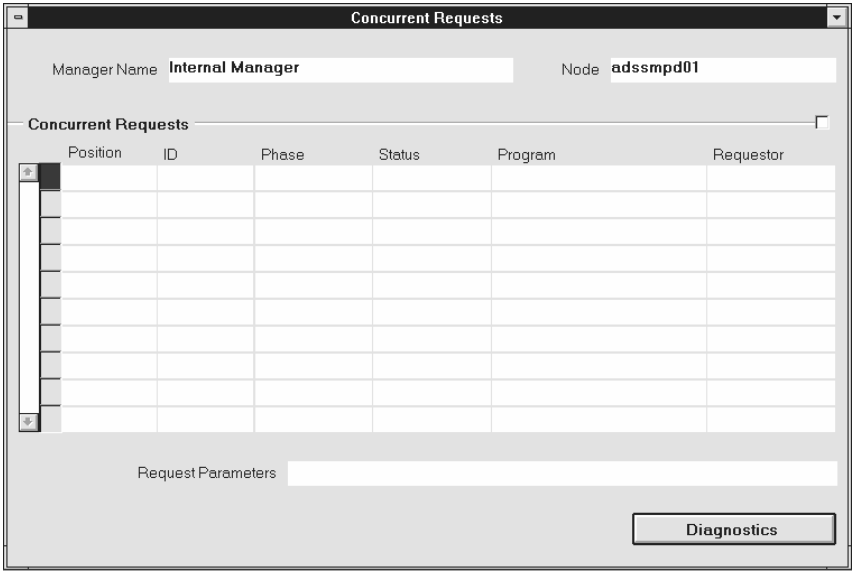
This field displays the operating system process ID for a spawned concurrent process.

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.

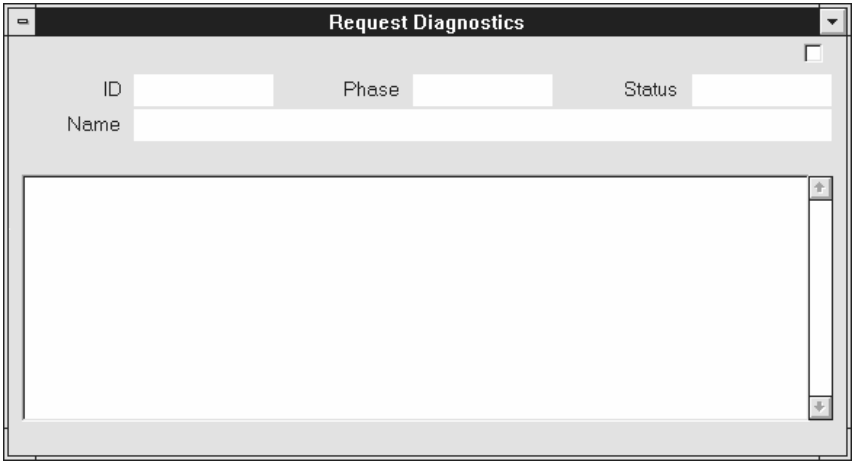
Request Log	Choose this button to view the log file of the process associated with the running request.
Internal Manager Log	Choose this button to view the Internal Concurrent Manager's log file.
Manager Log	Choose this button to view the log file of the concurrent manager who started running the request.

Concurrent Requests Window



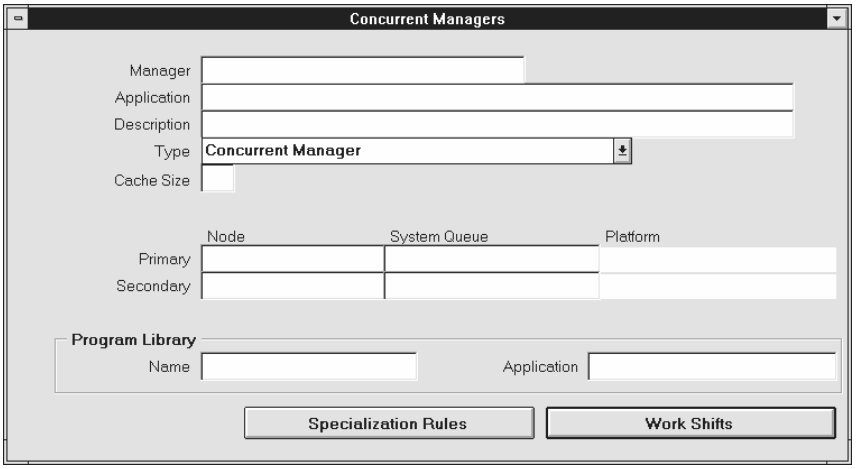
View all running and pending requests for a selected manager, whose name and node are identified near the top of the window.

Request Diagnostics Window



This window informs you when the request completed or if it did not complete, shows you a diagnostic message indicating why.

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 try to run those three requests before reading any new requests.

Suggestion: 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.

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– 92.

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.

Attention: To ensure that your manager uses your platform–specific queue management system, you should start the concurrent managers in the proper mode (set APPLDCP = OSQ). 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

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.

Immediate concurrent programs must be registered in a program library by an applications developer using Oracle Application Object Library.

See: *Developing Custom Extensions to Oracle Applications*

Program Library

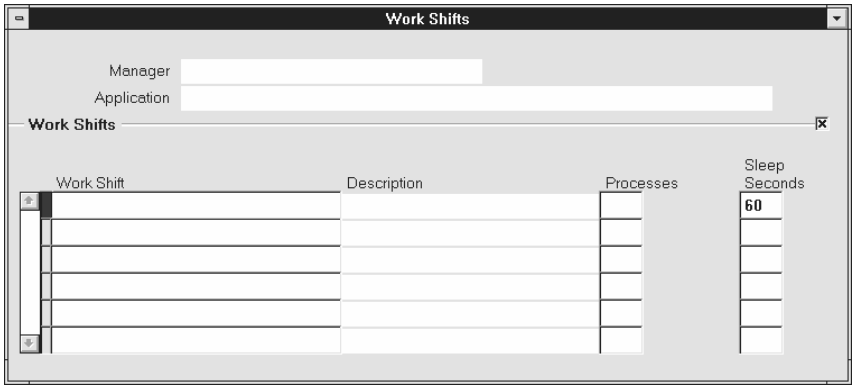
Concurrent managers can run only those immediate concurrent programs listed in their program library. They can also run concurrent programs that use only other type of concurrent program executable.

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.

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– 86.

Work Shift

Select the work shift(s) you want to assign to your manager.

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.

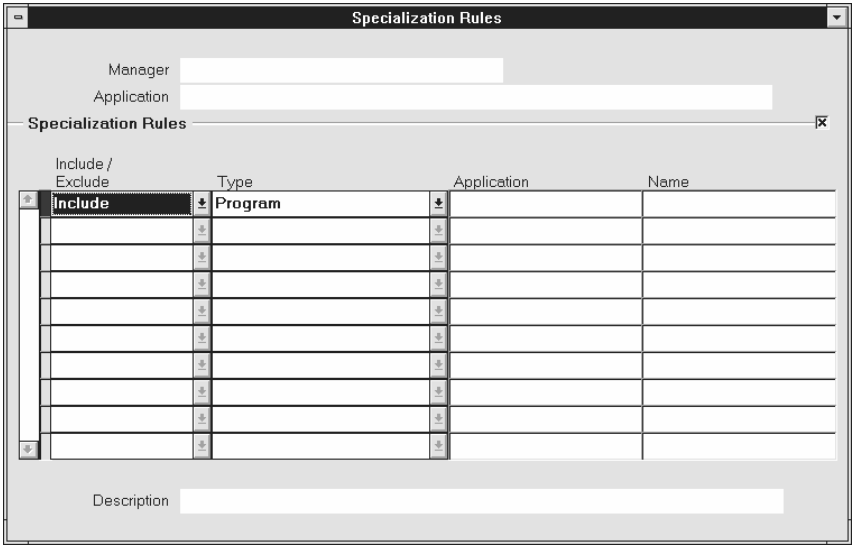
Sleep Seconds

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).

The default value is 60 (seconds).

Suggestion: Set the sleep time to be very brief during periods when the number of requests submitted is expected to be high. Describe your application–ORACLE username pair, if you wish.

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– 88.

- 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.

Application

Select the application associated with your:

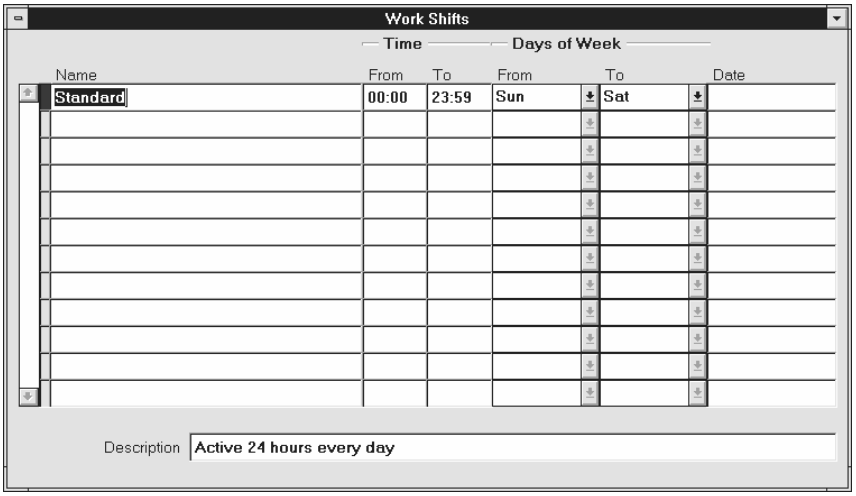
- Combined Rule
- Program
- Request Type

Name

Select the name of your:

- Combined Rule
- ORACLE ID
- Program
- Request Type
- User

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– 26.

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

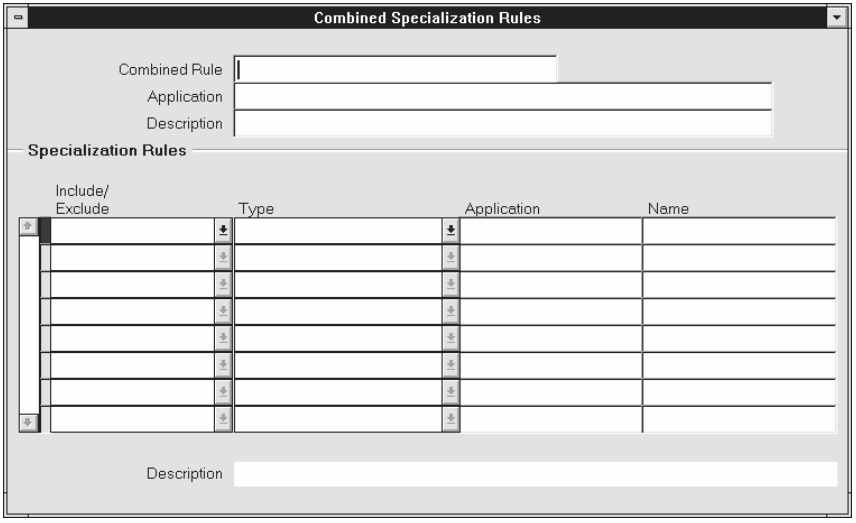
Enter the first and last days of this shift. For instance, if your shift name is ”Week Days”, you could enter ”Monday” in the ”Days of Week From” field and ”Friday” in the ”Days of Week To” field. If you enter a value in the ”Days of Week From” field, you must enter a value in the ”Days of Week To field”. 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 ”30–MAY–94” in this field to enable this workshift only on May 30, 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– 28.

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– 80

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.

Application

Select the application associated with your:

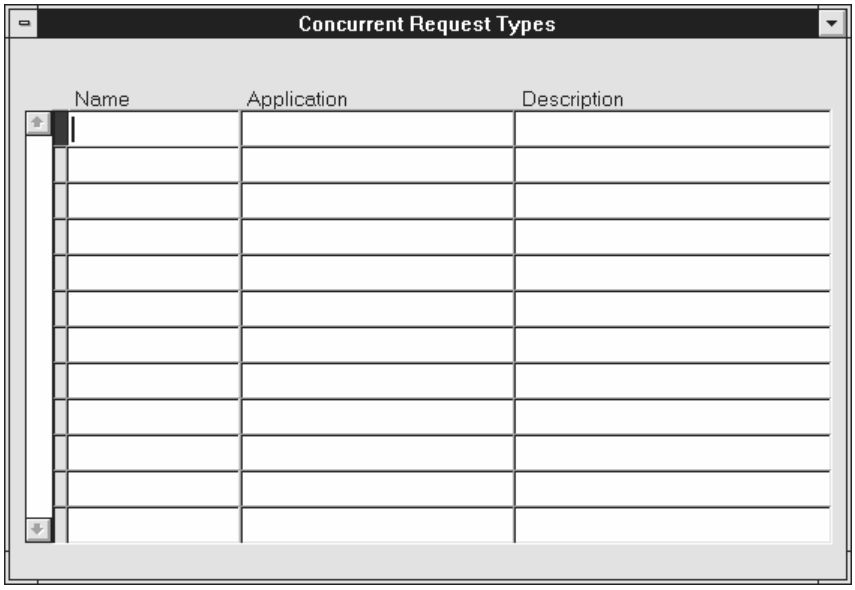
- Program
- Request Type

Name

Select the name of your:

- ORACLE ID
- Program
- Request Type
- User

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.

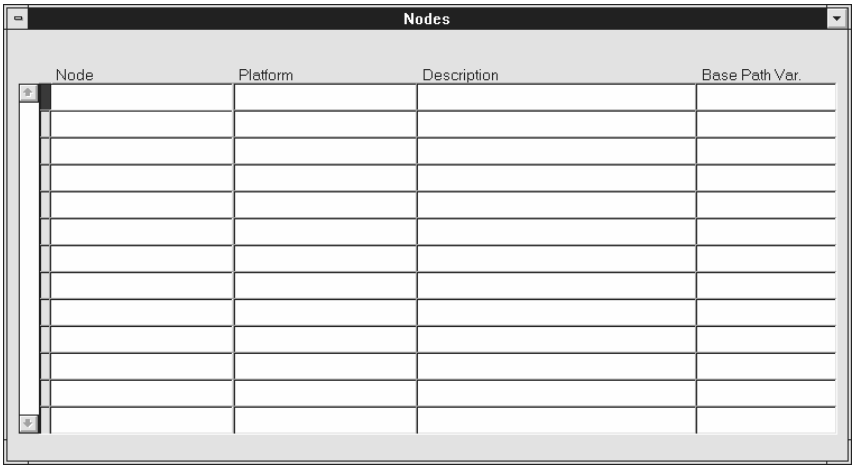
For example, you could define a request type as “end–of–month reports”, assign that request type to several concurrent programs, then define a concurrent manager to only run ”end–of–month” requests.

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.

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 Var.

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.

CHAPTER

8

Printers

This chapter tells you everything you need to know about using printers with Oracle Applications. The essays in this chapter are organized under the following topics:

- Overview of Printers and Printing
- Setting Up Your Printers
- Customizing Printing Support in Oracle Applications
- Postscript Printing in UNIX
- Hierarchy of Printer and Print Style Assignments
- Upgrading Printer Files

Form descriptions follow at the end of the chapter.

Overview of Printers and Printing

This essay explains how Oracle Applications handles printing instructions. The executive summary presents the major relationships between printing functions, related forms, and Oracle Applications. More detailed explanations follow the executive summary. See: Summary of Oracle Applications Printing: page 8 – 4.

Executive Summary

Oracle Applications reports are generated by Oracle Reports. 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 that issues an operating system print command.

Oracle Reports and report generation

Page break, carriage return, line feed, text bold on / off, and text underline on / off instructions *within the output file* are defined by values in an SRW driver file.

Page break, carriage return, and line feed instructions that are issued *before* the output file is to be printed or *after* the output file is printed must be entered in an Oracle Applications printer driver's initialization or reset strings, which are defined by the Printer Drivers form.

SRW Drivers and Oracle Applications Printer Drivers

When the report is not to be printed (number of copies = 0 and the target printer field is blank), Oracle Reports uses the SRW driver named by the print style in the Print Styles form.

When the report is to be printed (number of copies > 0) 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 is completed, the concurrent manager prepends an initialization string to the output file. The initialization string is defined using the Printer Drivers form.

The concurrent manager appends an reset string to the output file. The reset string is defined using the Printer Drivers form.

An Oracle Applications printer driver is typically executed in one of two methods, by issuing a print command or calling or print program.

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.

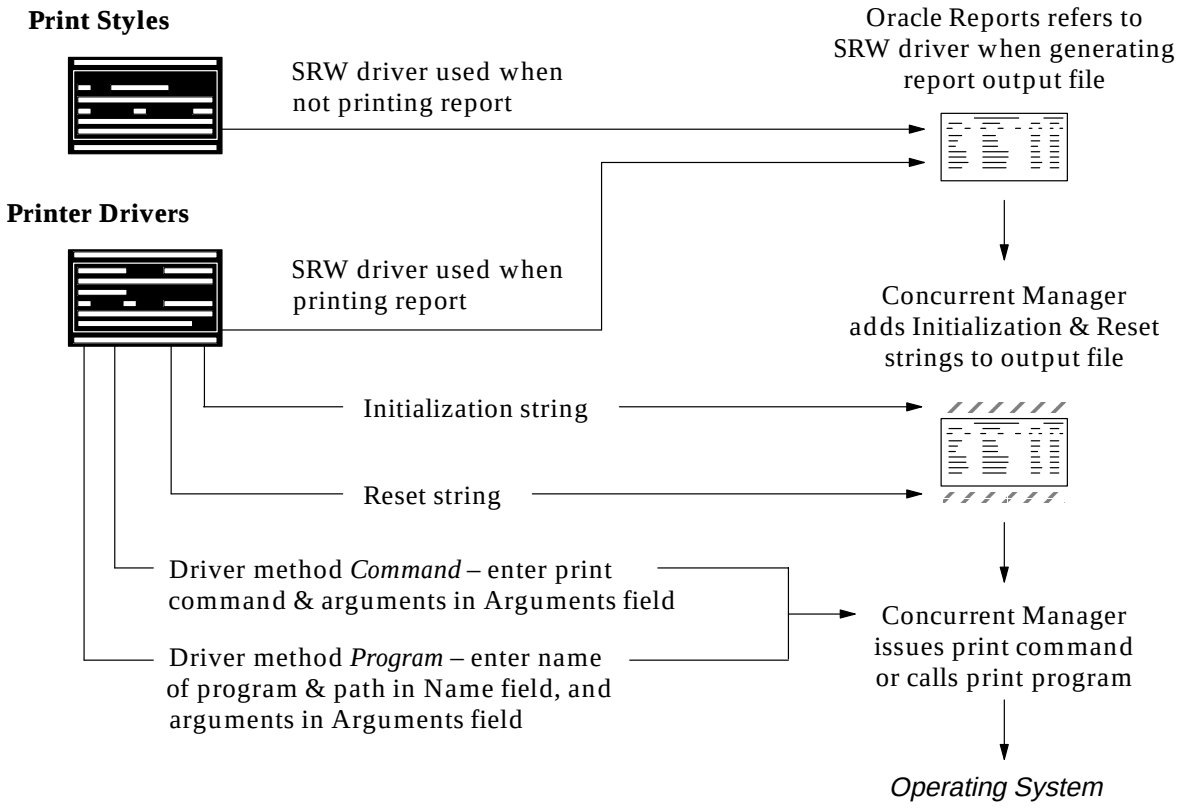
Concurent Manager can provide values for arguments

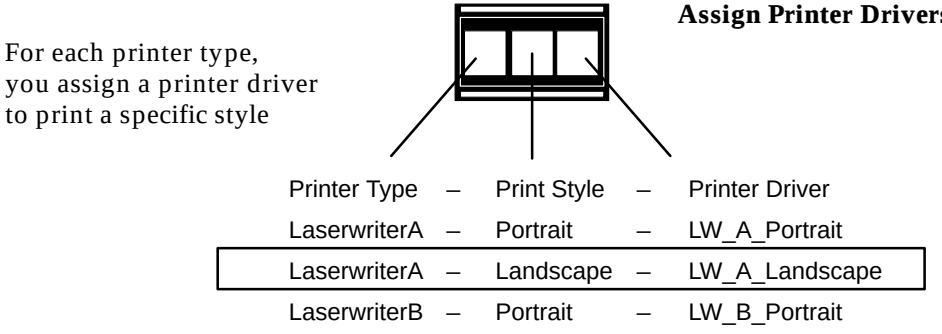
The concurrent manager may provide values for four arguments to an operating system print command or custom print program:

- 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

Figure 8 – 1

Summary of Oracle Applications Printing





Oracle Reports and Applications Printing

When you run an Oracle Applications report, Oracle Reports generates and formats the output.

Each report has a print style that defines its dimensions, that is, the number of columns and rows it contains.

Once a report is completed, an Oracle Applications printer driver attaches formatting instructions for the destination printer.

Text, document, and printer formatting instructions for printing a file generated by Oracle Reports are summarized in the table below.

Oracle Applications Formatting Instructions

Instruction	Explanation	Mechanism	Form
Format Text	Bold, underline, page breaks.	SRW driver control characters	Print Styles (printer not associated with concurrent request; i.e., copies = 0, printer field blank)
			Printer Drivers (printer associated with concurrent request)
Format Document	Width and height of report	Print Style number of columns and rows	Print Styles
Format Printer	Tell printer to print portrait or landscape, reset itself, etc.	Printer Driver initialization and reset strings	Printer Drivers

Table 8 – 1 (Page 1 of 1)

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.

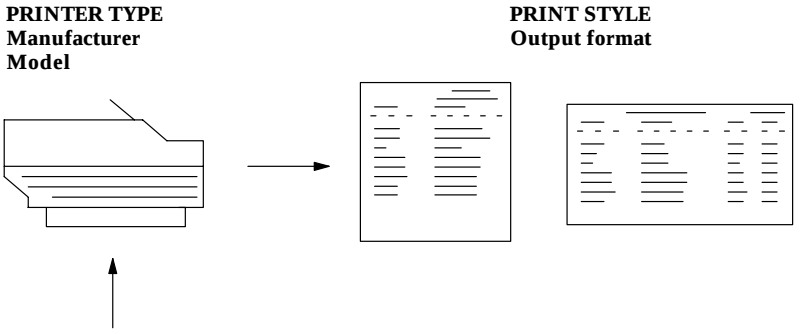
A *print style* tells the printer how a printed output should look. A *printer driver* delivers commands that tell the printer how to output the specified print style.

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.

Attention: You must assign both a print style, and a printer driver to print that style, to each printer type you wish to print reports from in that style.

Figure 8 – 2



Printer Type What kind of printer you have. This is the manufacturer and model. Two examples are a DEC LN03 printer and an HP Laserjet III printer.

Print Style A description of how your report should be printed. Print style determines the:

– Number of lines per page.

– Width of each line.

– Whether a header page should be printed.

- Number of lines per page.
- Width of each line.
- Whether a header page should be printed.

Printer Driver The set of commands that tell a printer how to print in the Print Style chosen.

- Initialization sets printing orientation.

- Reset clears printer's instructions for next print job.

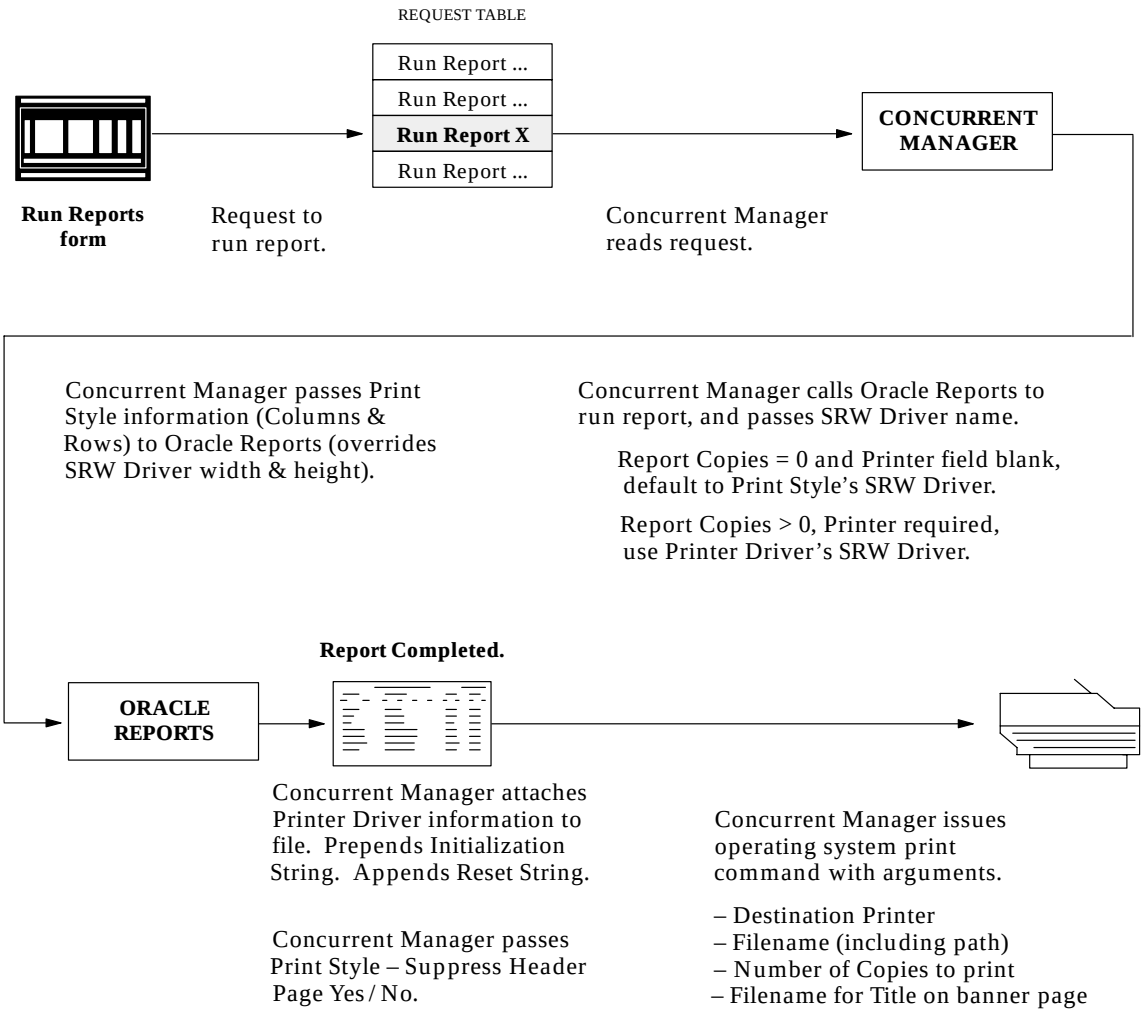
Sequence of Printing Events

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 Drivers form.

A printer driver formats the destination printer. An SRW Driver formats text and sets page breaks within an Oracle Reports file.

Figure 8 – 3

Sequence of Printing Events – Simplified Summary



Setting Character–Mode vs. Bitmap Printing

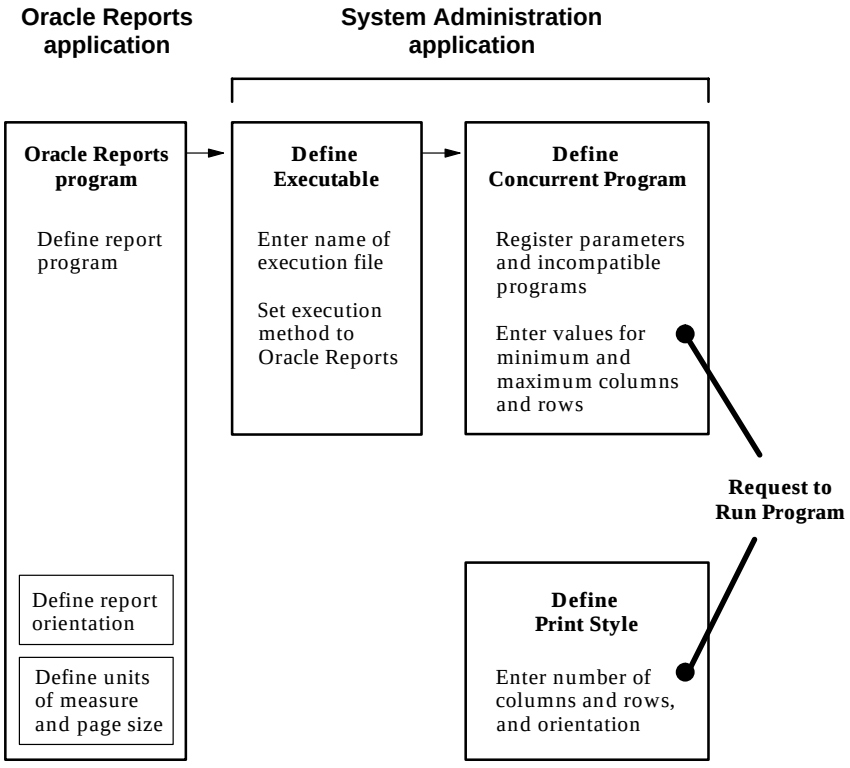
Running Character mode Oracle Reports Concurrent Programs

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.

Figure 8 – 4

Running Character Mode Oracle Reports Programs



Running Bitmap Oracle Reports Concurrent Programs

To run an Oracle Reports program in bitmap mode, query the concurrent program's definition in the Concurrent Programs form, and enter "VERSION=2.0b" in the Execution Options field. This defines the program to use the bitmap version of Oracle Reports.

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).

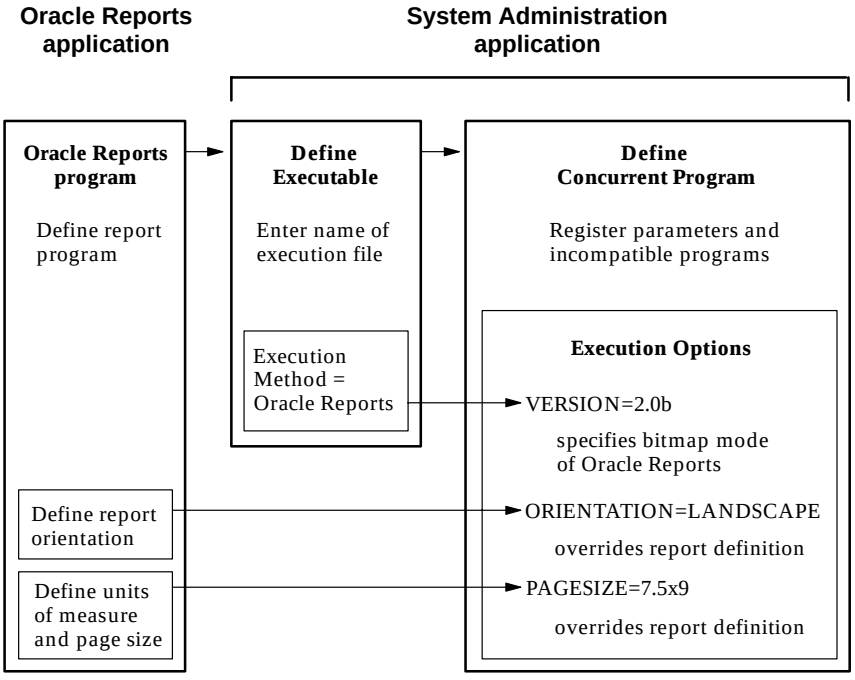
If you wish to override the program's definitions, you can enter values in the Execution Options field for ORIENTATION and PAGESIZE.

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:

```
VERSION=2.0b ORIENTATION=LANDSCAPE PAGESIZE=7.5x9
```

Figure 8 – 5

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:

```
VERSION=2.0b 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.

Defining Printer Types and Registering Printers

You register a printer so Oracle Applications recognizes the printer and can forward to it the output from a report program.

To register a printer with Oracle Applications, you must first specify what kind of printer it is by selecting a printer type. Because many printers can be registered as the same type of printer, you need only define each printer type once.

You register individual printers with Oracle Applications by specifying the printer’s operating system name, which uniquely identifies the printer, and selecting the type of printer it is.

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 (e.g., printer39), and select the printer type (e.g. LN03).
- If the correct printer type is not defined, you must define the new printer type (e.g., LN03) before you can register the printer.

Print Styles

A Print style defines the page format for a printer; the number of columns (page width), and the number of rows (page length).

Each printer type (i.e., each printer) can have one or more associated print styles.

Print styles allow you to setup 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 may not be changed.

Printer Drivers

To print in a particular style from a specific type of printer, 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.

Defining a printer driver allows you to enter information specific to a printer type which makes it print in the style you want.

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.

Printer Driver Definition

A printer driver definition consists of the following information:

Name	The name you give to your printer driver
Platform	The platform (if any) that this driver is specific to
SRW Driver	The name of the SQL*ReportWriter (SRW) driver (if any) that should be used for generating an SRW report.
Printer driver method	How your printer driver is invoked. Drivers can be invoked as operating system commands, programs, or subroutines.
Description	A description of your driver.
Program Name	The name of the program that invokes printing.
Arguments	Any standard arguments for your program (if method is Program) or the print command along with its arguments (if method is Command).
Initialization	Escape sequences to initialize your printer for your print style. The initialization string tells the printer how to orient the characters on the page, for example, whether to print portrait or landscape.
Reset	Escape sequences to reset your printer once printing completes. To use an analogy, the reset

string is similar to erasing a blackboard full of instructions, so the next set of commands will not be misinterpreted.

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 may have. Customize the predefined components as desired or if necessary. See: Customizing Printing Support in Oracle Applications: page 8 – 17.

Attention: 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 (i.e., manufacturer and model) used at your site that are not shipped with Oracle Applications. Also, for each print style you wish to output from a particular printer type, you need to assign to the printer type a combination of a print *style* and a printer *driver*.

Printers

When you register a printer with Oracle Applications, you identify the printer by its operating system name, and assign it a printer type.

You can only register a printer as a previously defined printer type.

Print Styles

To generate a report, the print style values for columns and rows are passed by the concurrent manager to Oracle Reports (i.e., values for the PAGESIZE token). A print style determines the dimensions of your report, or the:

- Number of lines per page (number of rows or page height).
- Width of each line (number of columns or page width).

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

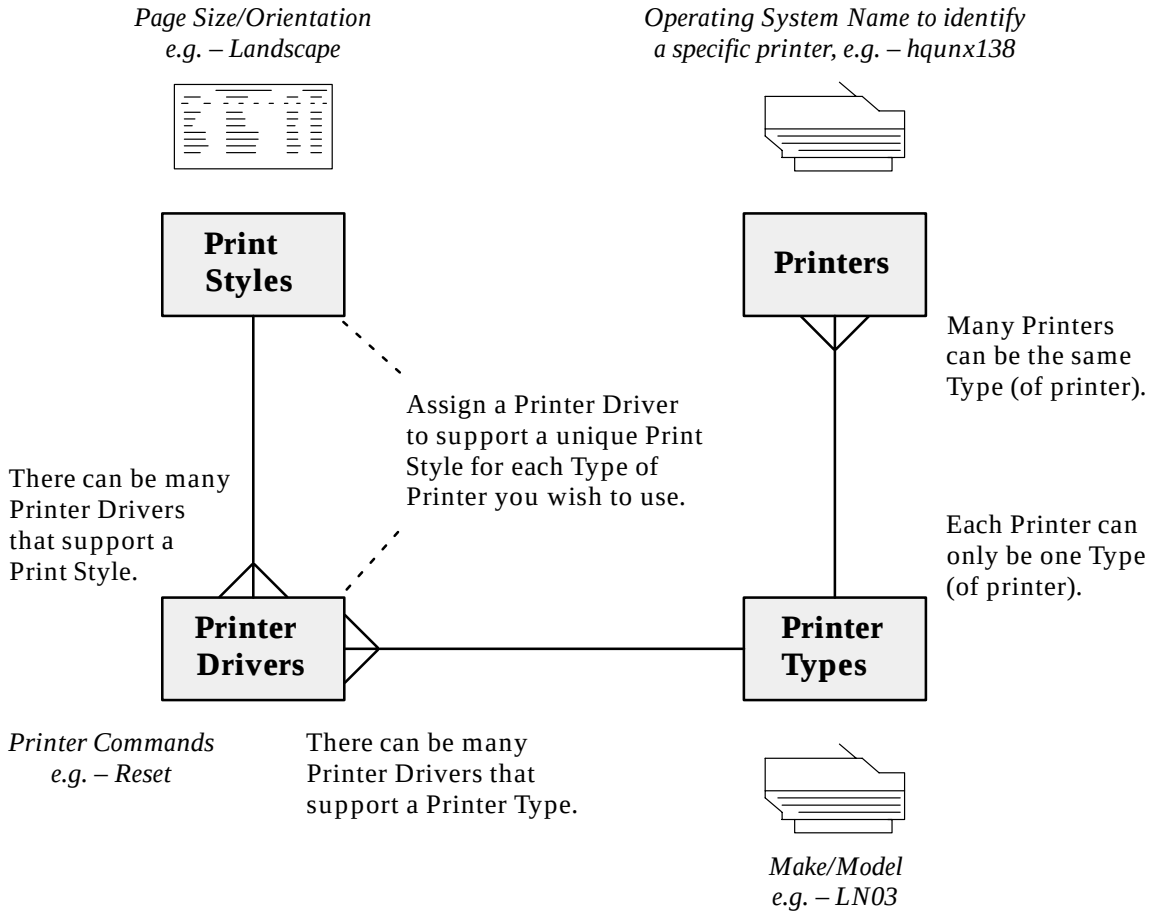
In the diagram below, a single line indicates “one” and three lines branching out indicate “many”. For example, a printer can only be one printer type, and a printer type can be assigned to many printers.

- Many printers can be registered as a particular printer type.
- A printer type can support multiple print styles.
- A printer driver must be assigned to a printer type for each print style you wish your output to be in.

See: Overview of Printers and Printing: page 8 – 2

Figure 8 – 6

Printers, Printer Types, Print Styles, and Printer Drivers



Printer Setup Information Is Cached On Demand

Printer setup information; 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 (read into memory).

Attention: 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 – 52

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 Print Styles form to query the combinations of print style and printer driver that support each type of printer you have.

Attention: Predefined printing components may have to be modified for different printer types and / or operating platforms.

For example, if a blank or extra 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 driver's:

- 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.

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 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 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 path to 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, for example, a UNIX shell or a VMS dcl, 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 a custom print program that it calls. An example of using these values as arguments is presented in Figure 8 – 7 and Table 8 – 2.

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.

Since print commands are operating system dependent, please refer to your *Oracle Applications Installation Guide.*

See: Printing
(Oracle Applications Installation Guide)

Figure 8 – 7

Example – Printer Drivers form's Arguments field.

```
lp –d$PROFILES$.PRINTER –n$PROFILES$.CONC_COPIES –t"$PROFILES$.TITLE"
$PROFILES$.FILENAME
```

Passing Arguments to 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.

Table 8 – 2 (Page 1 of 2)

Argument Syntax	Token and Value Retrieved
–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.

Table 8 – 2 (Page 2 of 2)

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.

Attention: 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 ".

Attention: 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 upgraded to release 11 printer drivers.

For non–printable 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 temp file. This option is recommended when using the Program driver method and the print program creates its own temp file.

This option helps to reduce the creation of temp files, since the concurrent manager will not create a temp 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 (i.e., fields are blank).

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.
- Oracle Reports will use the new driver if it is associated with a print style and/or printer driver definition.

Attention: 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 since 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 are used for slightly different reasons.

If you run an Oracle Reports program without printing the output file, that is, when the number of copies is zero (0) and the printer field is blank, the SRW driver associated with the report's print style is used.

If you run an Oracle Reports program and print the output file, that is, when the number of copies is greater than zero (0), the SRW driver that is correct for the type of printer the report is being printed from is chosen by selecting the SRW driver associated with the printer type.

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.

Attention: 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: Passing Arguments to UNIX lp Print Command: page 8 – 21.

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 8 – 27.

Enscript Arguments and Print Styles

Table 8 – 3 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.

Postscript Print Styles and Enscript Arguments

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

Table 8 – 3 (Page 1 of 2)

Print Style	Enscript Arguments	Explanation	Result
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)

Table 8 – 3 (Page 2 of 2)

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 would be set to *Command*.

Figure 8 – 8

Printer Drivers window Arguments field:

```
enscript –r –fCourier8 –B –P$PROFILES$.PRINTER $PROFILES$.FILENAME
```

Example – Using the UNIX 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.

Table 8 – 4 (Page 1 of 2)

Syntax	Explanation
–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.

Table 8 – 4 (Page 2 of 2)

In this example, the UNIX enscript command is entered along with two of the four 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.

Hierarchy of Printer and Print Style Assignments

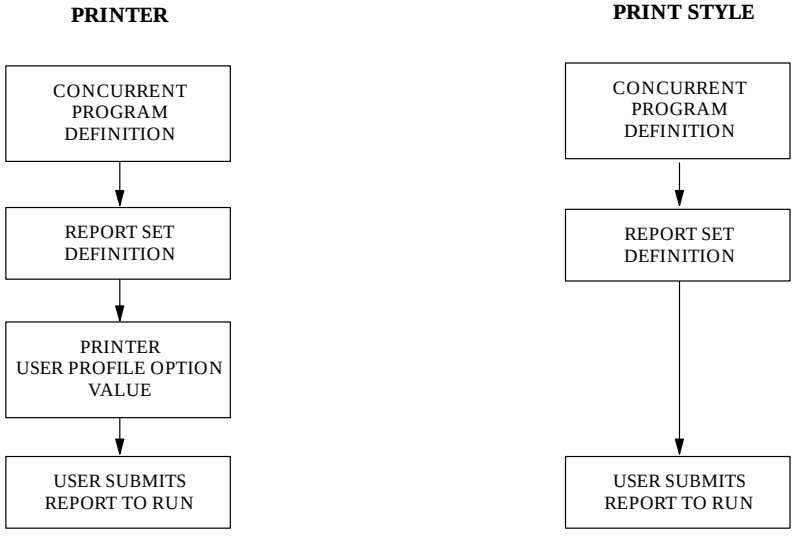
A printer and a print style can be chosen and their identities can be included in a concurrent program's definition. When a concurrent program is defined to send its output to a specific printer, or is required to generate its output in a specific print style, those values cannot be overridden by users, 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. Figure 8 – 9 illustrates the order of how printer or print style values are read by the concurrent manager when submitting a report program to run.

Attention: 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.

Figure 8 – 9

Order of Reading Printer or Print Style Settings



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.

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.
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.

Table 8 – 5 (Page 1 of 1)

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).

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.

Table 8 – 6 (Page 1 of 2)

Form	Explanation
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.

Table 8 – 6 (Page 2 of 2)

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

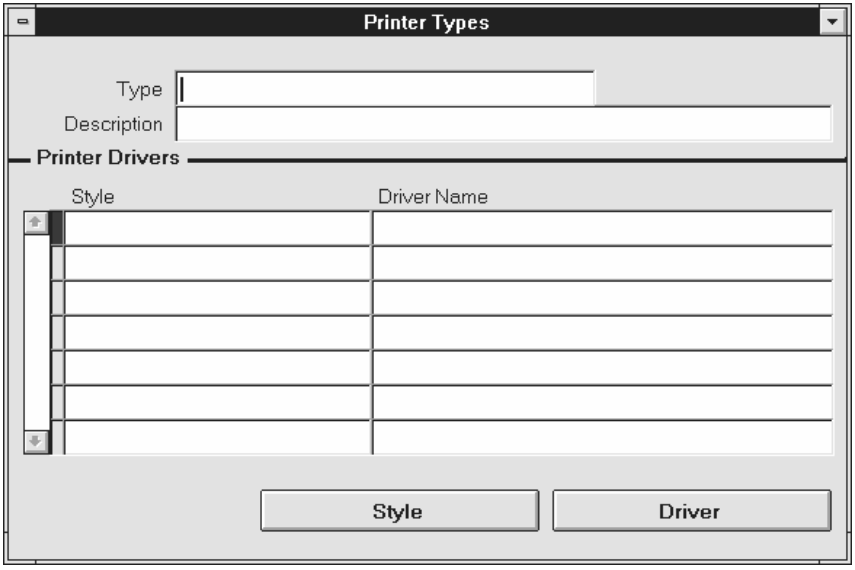
End users may:

- 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.

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.

Attention: 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 – 52

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.

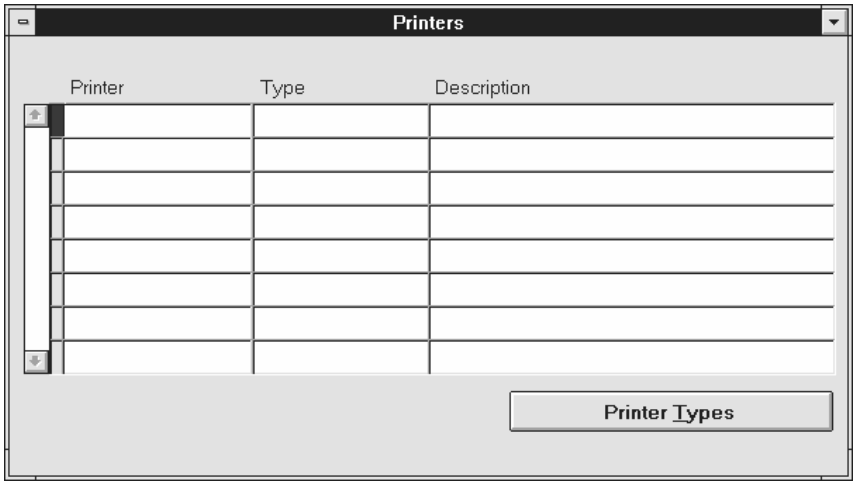
Printer Drivers Block

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.

Printers Window



Register printers with Oracle Applications by entering the operating system's name for the printer and assigning it a printer type (e.g., manufacturer and model).

You:

- Must register a printer before you can print reports from it, using Oracle Applications.
- 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

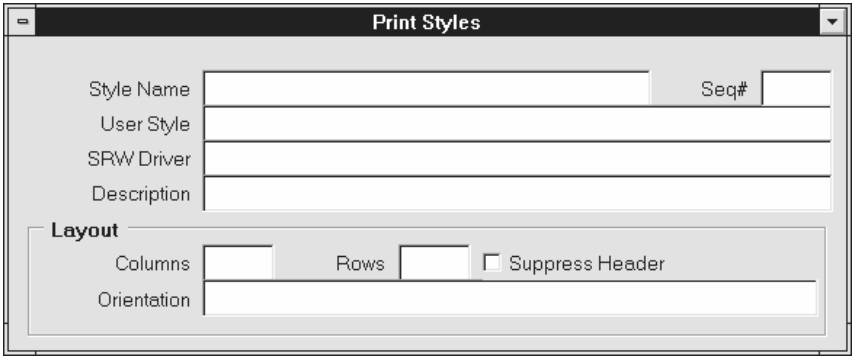
Enter the name your operating system specifies for the printer.

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.

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.

Attention: 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 – 52.

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.

Name

Multiple print styles display alphabetically in a list window according to their Name (not User Name).

You cannot update a print style's name.

User Name

This user name does not appear anywhere except this window.

Columns

Enter the number of columns your print style defines.

Rows

Enter the number of rows your print style defines.

Suppress Header Flag

Reports may print with a header page that indicates who requested the report and when. Check the Supress Header Flag 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.

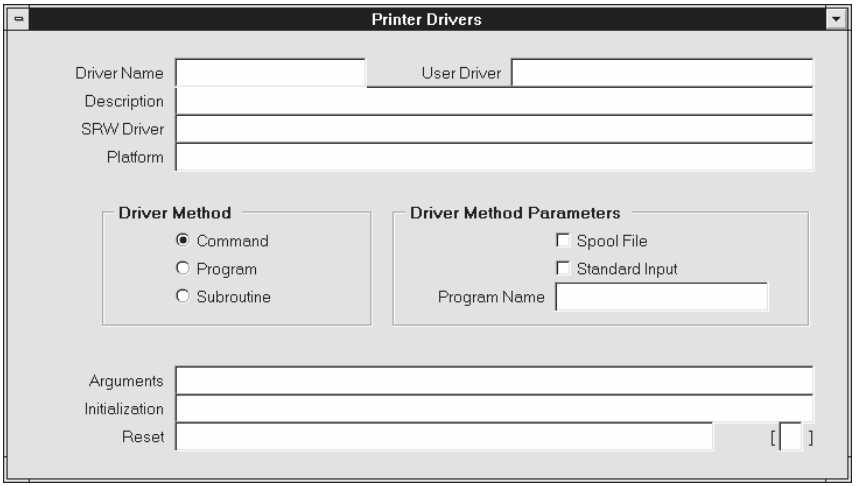
Orientation

Enter the orientation of your printed page, for example, portrait or landscape.

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.

Printer Drivers Window



Use this window to define your printer driver and printer commands.

Attention: 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

Printer drivers are supplied for the following printers:

- 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 Block

See: Controlling Concurrent Managers: page 7 – 52.

Name

The printer driver name must be unique for a given platform.

User Name

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

Select one of three methods by which your printer driver is invoked.

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.

- An example is the SYS$PRINT routine called on the VMS platform.
- 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

Select the name of a:

- Program the driver invokes if the driver method is Program.
- Subroutine the driver invokes if the driver method is Subroutine.

Arguments Region

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.

CHAPTER

9 Applications DBA Duties

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

Depending on the nature of the company and the installation site, these duties may sometimes belong to the System Administrator. As such, this “borderline” area of tasks, which encompasses forms from various locations on the System Administrator menu tree, is referred to as 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.

The database privileges of Oracle Applications products depend on their ORACLE usernames. ORACLE usernames are created by an ORACLE database administrator, and then are registered as ORACLE usernames by a System Administrator.

ORACLE Usernames

An ORACLE username identifies you as an authorized ORACLE database user.

- Each ORACLE username consists of a database username and password assigned by your database administrator.
- Each ORACLE username accesses a set of data within the ORACLE database.
- Usually each Oracle application has its own ORACLE username, in which application–specific data resides. That is, the tables and other database objects owned by the application are accessed by the ORACLE username.

Note that database usernames and passwords connect to the ORACLE database, while application usernames and passwords access Oracle Applications.

You access the ORACLE database through an Oracle Applications product, and the application's ORACLE username is what grants access privileges.

Registering an ORACLE username

The installation process automatically registers Oracle Applications ORACLE usernames, so you only need to register any additional ORACLE usernames that you need using the ORACLE Users window.

You must register an ORACLE username with Oracle Applications if:

- you create a custom application using Oracle Application Object Library
- you want to associate an additional ORACLE username with an Oracle Applications product

Attention: Before you can register an ORACLE username, your database administrator must first create an ORACLE username that connects to the ORACLE database. You then use the ORACLE Users window to register your ORACLE username.

Registering a new ORACLE username using the ORACLE Users window submits a concurrent request that sets up the necessary privileges to the Oracle Application Object Library database tables you need to run your application. These database tables contain information to allow your users access to Oracle Application Object Library features, such as menus and flexfields.

Reregistering ORACLE usernames

You should also reregister ORACLE usernames associated with custom applications built using Oracle Application Object Library each time you upgrade Oracle Application Object Library

When you change the privileges that an already registered ORACLE username has to the Oracle Application Object Library database tables:

- Oracle Applications then submits a concurrent request to create or recreate privileges to the Oracle Application Object Library database tables.
- The concurrent request must complete successfully in order for your changes to take effect.

Registering an ORACLE username as “Restricted”

Oracle Applications let you register ORACLE usernames as *Restricted* ORACLE usernames. A restricted ORACLE username prevents users from modifying data in Oracle Application Object Library tables.

Your database administrator can set up the ORACLE username to prevent users from modifying data in other Oracle Applications tables.

You can register an ORACLE username as restricted using the ORACLE Users window.

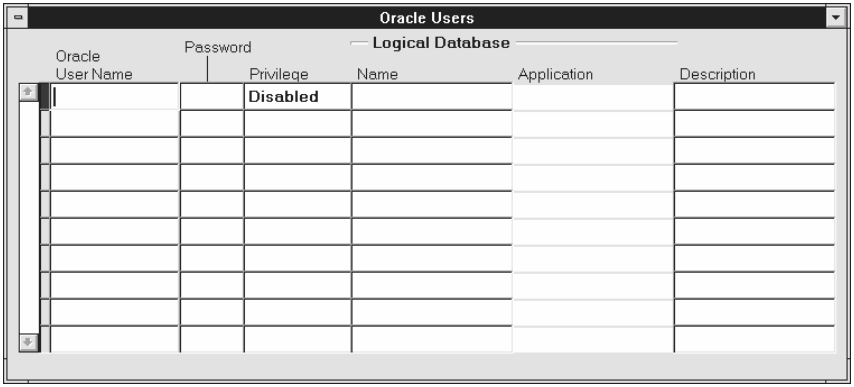
- When you register an ORACLE username as restricted, you submit a concurrent request that sets up read–only privileges to the Oracle Application Object Library database tables.
- Users with responsibilities that access restricted ORACLE usernames have read–only privileges to the Oracle Application Object Library database tables, which prevents them from inserting, updating, or deleting data related to such Oracle Application Object Library features as menus, flexfields, and so on.

Defining Data Groups

A data group assigns an ORACLE username to an Oracle Applications product, and includes a list of the valid Application–ORACLE username pairs.

The installation process automatically defines Data Groups for Oracle Applications, so you only need to define any additional data groups that you wish to utilize. See: Defining Data Groups: page 6 – 27. See: Data Groups: page 6 – 63.

ORACLE Users Window



Register an ORACLE username with Oracle Applications. An ORACLE username grants access privileges to the ORACLE database.

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.

When you register and enable an ORACLE username, you submit a concurrent request to set up the necessary database privileges between your ORACLE username and the Oracle Application Object Library database tables.

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 7 Server SQL Language Reference Manual.* 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.

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.

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.

Prerequisites

- Create an 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.
- Or, coordinate any change you intend to make to an existing ORACLE username password. You should register the password change in Oracle Applications and change the password in the database immediately afterwards.

Attention: Until you have both registered the changes in Oracle Applications and then implemented them in the database, responsibilities using your ORACLE username cannot connect to the database.

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.

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.

See:

Overview of Oracle Applications Security: page 2 – 2

Oracle Applications Installation Manual for your operating system

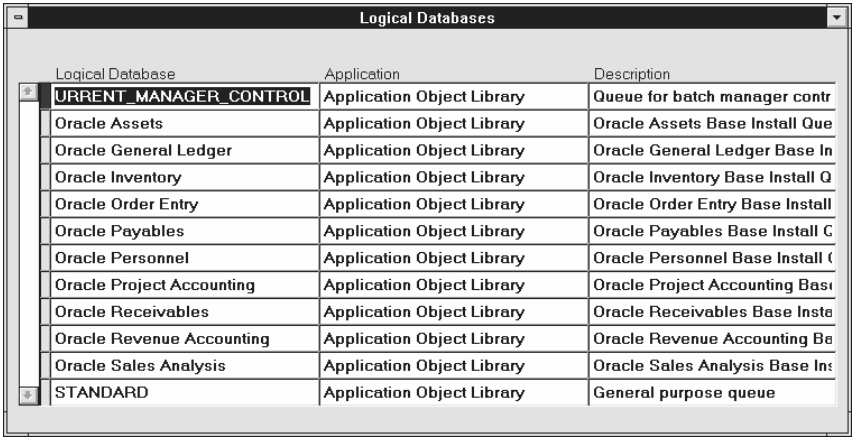
Install Group

Enter 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.

Attention: 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.

See: *Oracle Applications Installation Manual* for your operating system

Concurrent Conflicts Domains Window



Concurrent conflicts domains ensure that incompatible concurrent programs are not allowed to run simultaneously using related information.

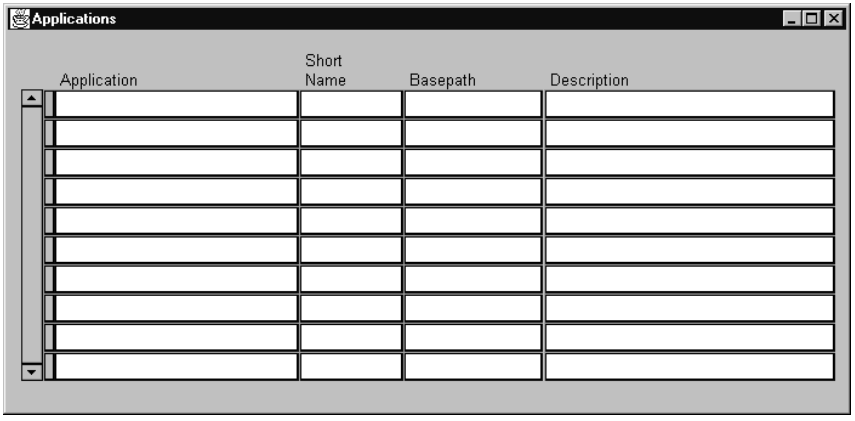
Concurrent managers use concurrent conflicts domains to determine which concurrent programs cannot run at the same time. For example:

- 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.

▶ Defining 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.

Applications Window



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

- ❑ Define an environment variable that translates to your application's basepath (see the *Oracle Applications Installation Guide* for your operating system).

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 name of an application, doing so may cause a change in the application code where you hardcode your application name. For example, if you pass program arguments through the menu that have application name hardcoded, you will also have to update them.

Attention: 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 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.

Suggestion: 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 search specific directories beneath the basepath for your application’s executable files and scripts when defining actions that reside in external files.

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 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.

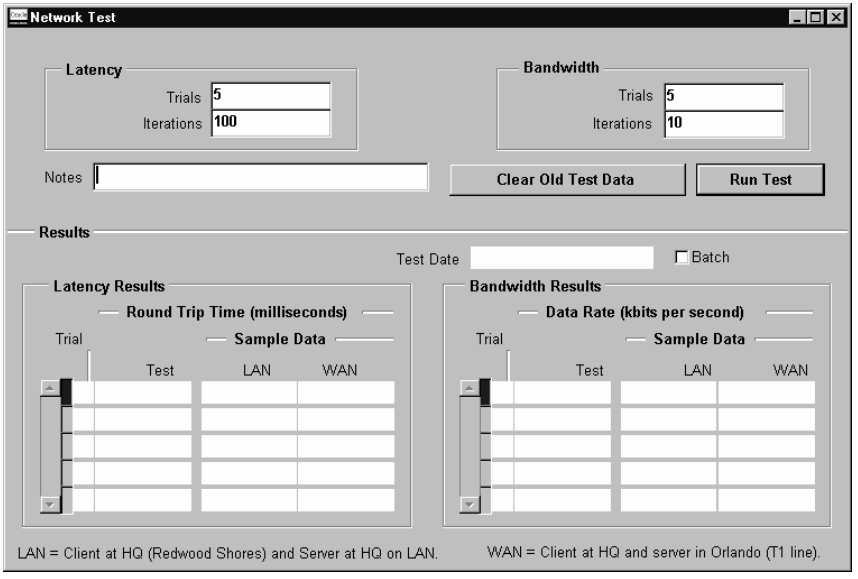
See: Development Environment
(*Oracle Applications Installation Manual)*

Network Test Window

Use the Network Test window to evaluate the performance of your network with Oracle Applications. Knowing the latency and bandwidth available lets you plan and modify your machine setup for the best performance.

The Network Test consists of a latency test and a bandwidth test. Latency is the time it takes for a single packet to make a round trip from your client side application to the server. The bandwidth test examines the data rate to see how many bytes per second your network can transfer from the server to the client.

You can provide notes to indicate the conditions for each test you run.



▶ To Test Latency On a Network

Specify the number of Trials and the Iterations for each trial.

For each iteration, a single packet is sent from the client application to the server and back. A trial consists of the specified number of iterations. The total time for all round trips in a trial is divided by the number of iterations to obtain the average latency that is that trial's result.

The default settings are 5 trials of 100 iterations each.

Select the Run Test button to preform the test.

▶ To Test Bandwidth On a Network

Specify the number of Trials and the Iterations for each trial. For each iteration, several kilobytes of data are sent from the client to the server and back. The form measures the average rate at which the data travels.

The default settings are 5 trials of 10 iterations each.

Select the Run Test button to perform the test.

Evaluating the Test Results

The results of both the latency and bandwidth tests display in the Results block.

Latency Results indicate the average round trip time for a single round trip from a PC client to the server.

Bandwidth results display the average data rate in kilobytes per second over each trial.

For comparison, the sample data fields show the results of tests completed at the development headquarters in Redwood Shores. These tests were conducted under ideal conditions; it is unlikely that your results can match them.

If one test result varies significantly from the other trials, discard that information.

Purging Your Data

Use the Clear Old Test Data button to purge previous test results from your database.

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.

Prerequisites

- ❑ Create default folders. See: Customizing the Presentation of Data in a Folder *(Oracle Applications User's Guide)*.

▶ **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:**

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:**

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

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.

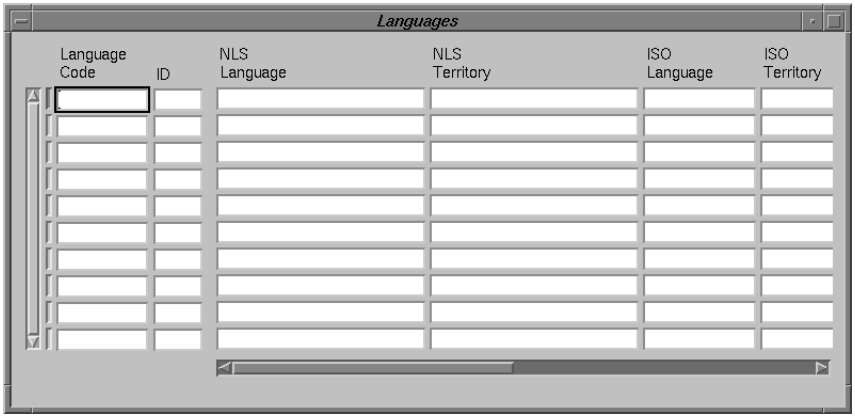
See Also

Customizing the Presentation of Data in a Folder *(Oracle Applications User's Guide)*

Querying Records in a Folder *(Oracle Applications User's Guide)*

Managing Folder Definitions *(Oracle Applications User's Guide)*

Languages Window



Use the Languages window to review and modify information about the languages available for use in Oracle Applications.

Languages Block

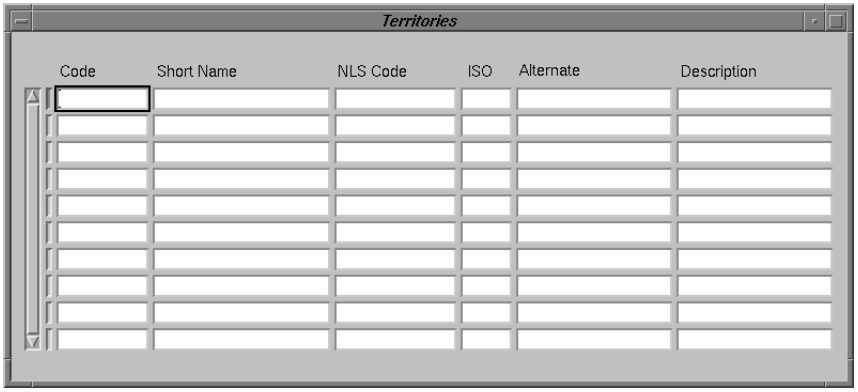
Each record includes the primary language, such as 'en' for English, the territory code where the dialect is spoken, such as 'US' for U.S.A., the short name for the dialect, such as 'usaeng', and the full name of the dialect, such as 'American English'. Each record also includes the internal language code and territory code, the ISO (International Standards Organization) language code and territory code, the code set for the dialect, and a status indicator for the dialect.

Normally you would not want to update the seeded data that comes with your products, but you may wish to modify the way the Language Description is represented in the Translations window.

Description

You can update the description of the Language to change the field name displayed in the Translations window.

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 upper case 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.

Description

You can update the description of the Territory to change the territory value displayed in List of Values used in Oracle Application products.

CHAPTER

10 Document Sequences

This chapter explains how to assign unique numbers to documents you create using Oracle Applications. Each time you enter a transaction, you create a document.

- For example, when you enter a payment, you create a payment document. Or when you enter an invoice, you create an invoice document.

By assigning unique numbers to documents, you can account for each transaction you enter and the document that accompanies it.

This chapter begins with an essay explaining what document sequences are and how they work in Oracle Applications.

Following the essay are descriptions of the forms you use to:

- Define sequences to number your documents.
- Define document categories to group documents together.
- Assign sequences to documents, defining which documents you will number using a particular sequence.

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.

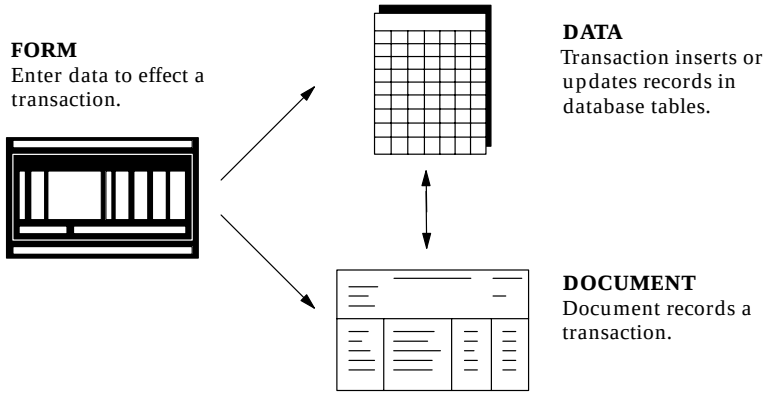
Document sequences can provide proof of completeness. For example, document sequences can be used to account for every transaction, even transactions that fail.

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.

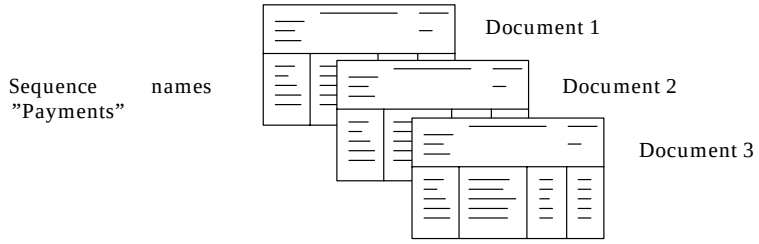
Document sequences generate audit data, so even if documents are deleted, their audit records remain.

Figure 10 – 1

Transactions generate documents. For example, recording payments generates payment documents.



A sequence defines how documents are numbered; what the first number in the sequence is, and whether numbers are generated automatically or entered manually.



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.

Attention: 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.

Attention: 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 below.

Examples – Automatic vs. Manual Sequences

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.**

Table 10 – 1 (Page 1 of 1)

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

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

Attention: 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.

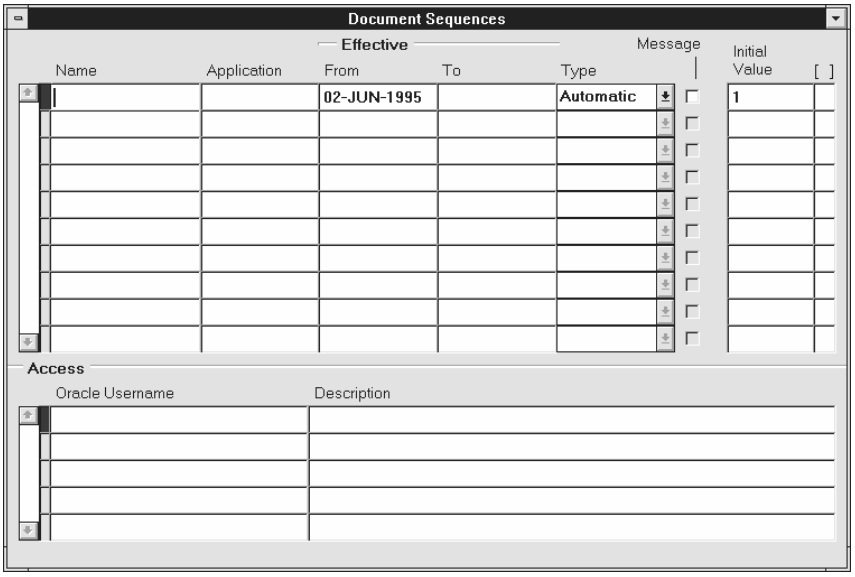
When you define a document sequence, you give the sequence a name, and define how the sequence numbers each document by:

- Choosing whether numbers are automatically generated in sequence, or entered manually by the user.
- Entering the initial value or first number in the sequence.

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.

Document sequences ensure that every document your users create can be accounted for. See: Sequences Assignments: page 10 – 14.

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.

You grant access to your document sequence from Oracle Applications products by selecting their ORACLE usernames in the Access block.

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.

Attention: 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 windows using the sequence).

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.

Attention: 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. The maximun sequence value is 1.0e+27.

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.

Access Block

Allow your document sequence to be accessed from your Oracle Applications products. You can extend access to your document sequence to additional applications by selecting each application's ORACLE username.

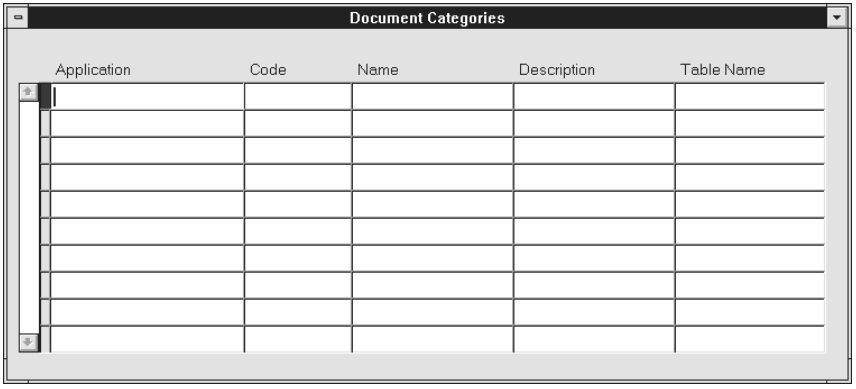
Extending access to your document sequence to more than one ORACLE username is especially useful when there is more than one installation of a given product, for example, when there are multiple sets of books.

Oracle Username

Select the Oracle Usernames that can access your sequence; more than one application may use a sequence to number their own documents.

Once your sequence is defined, you cannot delete access privileges to Oracle Usernames.

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

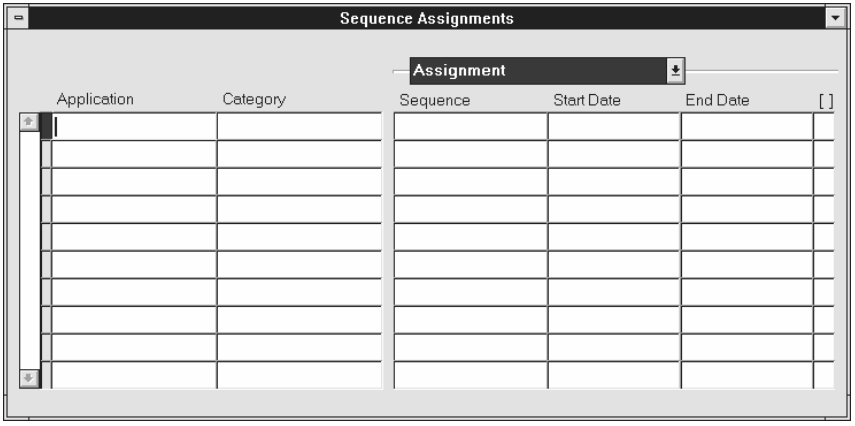
You can update the description, if you wish. For example, if the category description is predefined, you can change the description to a more familiar value.

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.

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 10 – 9.

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.

Attention: 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.

See: Defining Descriptive Flexfields Structures (*Oracle Applications Flexfields Guide*)

CHAPTER

11 Zoom and Customizing Oracle Applications

This chapter describes the architecture and implementation details for the CUSTOM library. The CUSTOM library allows you to write custom extensions, such as Zooms, to Oracle Applications. Zoom logic can be invoked by a user via the Applications toolbar or menu.

The following topics are covered:

- Customizing Oracle Applications with the CUSTOM Library
- Writing Code for the CUSTOM Library
- Events Passed to the CUSTOM Library
- When to Use the CUSTOM Library
- CUSTOM Library Package Procedures
- Support and Upgrading
- For Oracle Developers: Providing an Interface to the CUSTOM Library

Customizing Oracle Applications with the CUSTOM Library

The CUSTOM library allows extension of Oracle Applications without modification of Oracle Applications code. You can use the CUSTOM library for customizations such as Zoom (such as moving to another form and querying up specific records), enforcing business rules (for example, vendor name must be in uppercase letters), and disabling fields that do not apply for your site.

You write code in the CUSTOM library, within the procedure shells that are provided. All logic must branch based on the form and block for which you want it to run. Oracle Applications sends events to the CUSTOM library. Your custom code can take effect based on these events.

Attention: The CUSTOM library is provided for the exclusive use of Oracle Applications customers. The Oracle Applications products do not supply any predefined logic in the CUSTOM library other than the procedure shells described here.

Writing Code for the CUSTOM Library

The CUSTOM library is an Oracle Forms PL/SQL library. It allows you to take full advantage of all the capabilities of the Developer/2000 suite of products, and integrate your code directly with Oracle Applications without making changes to Oracle Applications code.

The CUSTOM library is located in the $AU_TOP/res/plsql directory (or platform equivalent). The CUSTOM library you modify must replace the default CUSTOM library in this directory in order for your code to take effect.

After you write code in the CUSTOM procedures, compile and generate the library using Oracle Forms. Then place this library into $AU_TOP/res/plsql directory (or platform equivalent). Subsequent invocations of Oracle Applications will then run this new code.

Warning: If there is a .plx (compiled code only) for a library, Oracle Forms always uses the .plx over the .pll. A .plx is only created when you generate a library using the Oracle Forms generator (using the parameter COMPILE_ALL set to Yes), not when you compile and save using the Oracle Forms Designer. Therefore, either delete the .plx file (so your code runs directly from the .pll file) or create your own .plx file using the Oracle Forms generator.

The specification of the CUSTOM package in the CUSTOM library cannot be changed in any way. You may add your own packages to the

CUSTOM library, but any packages you add to this library must be sequenced after the CUSTOM package. To ensure that your packages remain sequenced after the CUSTOM package even after a conversion from a .pld file, when program units are alphabetized, we recommend you name your packages with characters that come after C (for example, we recommend you name your own packages with names that begin with USER_).

Coding Considerations and Restrictions

Be aware of the open form environment in which Oracle Applications operate. Also, each running form has its own database connection.

The following considerations and restrictions apply to the CUSTOM library *and* any libraries you attach to CUSTOM:

- You cannot use any SQL in the library.
- Values of PL/SQL package variables (variables listed in the package specification) do not change between calls to the library, no matter which form calls the library.
- Startup code (PL/SQL code in a package but not contained in a function or procedure) executes only once, when the package is first called (CUSTOM is loaded only once in an Oracle Applications session, the first time it is called).
- Oracle Forms global variables in your code are visible to all running forms.

Altering Oracle Applications Code

Frequently you need to know the names of blocks and items within Oracle Applications forms for your CUSTOM logic. You should use the Examine feature available on the Help–>Tools menu while running the form of interest; it will give you easy access to all object names. You should not open Oracle Applications forms in the Oracle Forms Designer to learn this information.

You should exercise caution when changing any properties or values of items in the form from which CUSTOM logic is invoked. The CUSTOM library is intended to be a mechanism to augment Oracle code with your own. Using the CUSTOM library to alter Oracle code at runtime may bypass important validation logic and may jeopardize the integrity of your data. You should thoroughly test all logic you add to the CUSTOM library before using it in a production environment.

Following Coding Standards in the CUSTOM library

Within the CUSTOM library, you are free to write almost any code supported by the Developer/2000 toolset, so long as you follow all

Oracle Applications coding standards. However, there is an exception: you cannot call any APPCORE routines from CUSTOM. Almost all APPCORE routines start with the prefix "APP".

If you use Zoom or the CUSTOM library to invoke forms that you have developed, those forms must adhere completely to all of the Oracle Applications coding standards.

Attention: To invoke another form, use the function security routines. Do not use CALL_FORM since the Oracle Applications libraries do not support it.

Additional Information: *Oracle Applications Developer's Guide*

Events Passed to the CUSTOM Library

The CUSTOM library receives two different kind of events, generic and product–specific. Generic events are common to all the forms in Oracle Applications. These events are:

- WHEN–FORM–NAVIGATE
- WHEN–NEW–FORM–INSTANCE
- WHEN–NEW–BLOCK–INSTANCE
- WHEN–NEW–RECORD–INSTANCE
- WHEN–NEW–ITEM–INSTANCE
- WHEN–VALIDATE–RECORD
- SPECIALn (where n is a number)
- ZOOM
- EXPORT

The ZOOM event occurs when the user invokes Zoom from the menu (Action–>Zoom) or the toolbar. The EXPORT event occurs after an export operation is complete (Action–>Export).

The CUSTOM library also receives some product–specific events associated with the business rules of that product (for example, OE_LINES_PRICING). Please refer to the *Open Interfaces Manual* for your Oracle Applications product to see what product–specific events, if any, are passed to CUSTOM.

Additional Information: For further instructions on using events other than ZOOM in the CUSTOM library, please see the *Oracle Applications Developer's Guide*

When to Use the CUSTOM Library

There are several main cases for which you can code logic using the CUSTOM library. Each of these cases must be coded differently.

- **Zoom**—The addition of user–invoked logic on a per–block basis. A Zoom typically consists of opening another form and (optionally) passing parameter values to the opened form through the Zoom logic.
- **Logic for generic events**—Augment Oracle Applications logic for certain generic form events such as WHEN–NEW–FORM–INSTANCE or WHEN–VALIDATE–RECORD.
- **Logic for product–specific events**—Augment or replace Oracle Applications logic for certain product–specific events that enforce business rules.

  Additional Information: For further instructions on using events other than ZOOM in the CUSTOM library, please see the *Oracle Applications Developer's Guide*

Coding Zoom

Zoom allows the addition of user–invoked logic on a per–block basis. For example, you may want to allow access to the Vendors form from within the Enter Purchase Order form while the user is in the PO Header block of that form. You can enable Zoom for just that block, and when the user invokes it, you can open the Vendors form.

Only Oracle Applications customers use the Zoom feature; Oracle Applications products do not ship any predefined Zoom logic. Note that most Zooms that were predefined in the character–mode Oracle Applications Release 10 and earlier have been incorporated into Oracle Applications forms as buttons or windows for Release 11. In many cases, redesign of forms for Release 11 eliminated the need for predefined Zooms. Also, the native GUI environment allows users to cut–and–paste data between forms directly instead of relying on Release 10 Zooms to copy the data.

Zoom for Release 11 behaves as follows:

- Oracle Applications provides a menu entry and a button on the toolbar for the user to invoke Zoom when available. The button and the menu entry are disabled unless Zoom logic has been defined in the CUSTOM library for that form and block.

- Whenever the cursor changes blocks in the form, the form calls the ZOOM_AVAILABLE function in the CUSTOM library (via APPCORE). If this function returns TRUE, then the Zoom entries on the menu and toolbar are enabled; if it returns FALSE, then they are disabled.
- If the Zoom entries are enabled, then when the user invokes Zoom the form calls the Zoom event code in the CUSTOM library. You write code for this event that branches based on the current form and block.

▶ **To code Zooms into the CUSTOM library:**

1. Add a branch to the CUSTOM.ZOOM_AVAILABLE function that specifies the form and block where you want a user to be able to invoke Zoom. See: CUSTOM.ZOOM_AVAILABLE: page 11 –8
2. Add a branch to the CUSTOM.EVENT procedure for the ZOOM event.

   Inside that branch, specify the form and block where you want a user to be able to invoke Zoom. Add the logic you want to occur when the user invokes Zoom. See: CUSTOM.EVENT: page 11 –9

Support and Upgrading

To manage your customizations and handle upgrade considerations follow these guidelines:

Trouble with Forms Operating with the CUSTOM Library

If a form is operating incorrectly, and you have coded Zoom logic for it, use the menu to disable the CUSTOM library code temporarily (Help–>Tools–>Custom Code–>Off) so you can determine whether the problem comes from the customizations or Oracle Applications code. If you are using Release 10SC Production 12 or earlier, you should replace your CUSTOM library with the default CUSTOM library shipped with Oracle Applications before calling Oracle Support.

Upgrading

An Oracle Applications upgrade will overwrite any modifications you have made to the CUSTOM library, so you must keep a backup copy of CUSTOM with the changes you make. Replace the default CUSTOM library with your custom version after the upgrade.

Remember, form and block names may change after an upgrade to Oracle Applications. You should test any custom logic that you have defined to confirm that it still operates as intended before using it in a production environment.

CUSTOM Package

The CUSTOM package contains the following functions and procedure:

- CUSTOM.ZOOM_AVAILABLE
- CUSTOM.STYLE
- CUSTOM.EVENT

CUSTOM.ZOOM_AVAILABLE

Summary

```
function custom.zoom_available return BOOLEAN;
```

Description

If Zoom is available for this block, then return TRUE; otherwise return FALSE. Always test for the form and block name. Refer to the SYSTEM variables for form name and block name in your code and branch accordingly. The module name of your form must match the form file name.

By default this routine must return FALSE.

Example Code

The following example enables Zooms in the following places:

Form: FNDSCAUS, Block USER and

Form: FNDCPMCP, Block PROCESS

```
FUNCTION zoom_available RETURN BOOLEAN IS
  form_name  VARCHAR2(30) := NAME_IN('system.current_form');
  block_name VARCHAR2(30) := NAME_IN('system.cursor_block');
BEGIN
  IF (form_name = 'FNDSCAUS' AND block_name = 'USER') OR
     (form_name = 'FNDCPMCP' AND block_name = 'PROCESS')THEN
     RETURN TRUE;
  ELSE
     RETURN FALSE;
  END IF;
END zoom_available;
```

CUSTOM.STYLE

Summary

```
function custom.style(event_name varchar2) return integer;
```

Description

This function allows you to determine the execution style for some product–specific events. You can choose to have your code execute

before, after, or in place of the code provided in Oracle Applications. See the *Open Interface Manual* for your Oracle Applications product for a list of events that are available through this interface. Note that some product–specific events may not support all execution styles. CUSTOM.STYLE does not affect generic form events or Zoom.

Additional Information: *Oracle Applications Developer's Guide*

CUSTOM.EVENT

Summary

```
procedure custom.event(event_name varchar2);
```

Description

This procedure allows you to execute your code at specific events. Always test for event name, then for form and block name within that event. Refer to the SYSTEM variables for form name and block name in your code and branch accordingly. The module name of your form must match the form file name.

Always use FND_FUNCTION.EXECUTE to open a new session of a form. Do not use CALL_FORM or OPEN_FORM. The form function must already be defined with Oracle Application Object Library and added to the menu (without a prompt, if you do not want it to appear in the Navigator).

Additional Information: *Oracle Applications Developer's Guide*

By default, this routine must perform "null;".

Attention: Oracle Corporation reserves the right to pass additional values for event_name to this routine, so all code must be written to branch on the specific event_name passed.

Example Code

The following example contains logic for a Zoom, a product–specific event, and a generic form event.

The Zoom event opens a new session of a form and passes parameter values to the new session. The parameters already exist in the form being opened, and the form function has already been defined and added to the menu (without a prompt, so it does not appear in the Navigator).

```
procedure event(event_name varchar2) is

    form_name      varchar2(30) := name_in('system.current_form');
    block_name     varchar2(30) := name_in('system.cursor_block');
    param_to_pass1 varchar2(255);
    param_to_pass2 varchar2(255);
  begin
```

```
    if (event_name = 'ZOOM') then
      if (form_name = 'DEMXXEOR' and block_name = 'ORDERS') then
        /* The Zoom event opens a new session of a form and
        passes parameter values to the new session.  The
        parameters already exist in the form being opened:*/
        param_to_pass1 := name_in('ORDERS.order_id');
        param_to_pass2 := name_in('ORDERS.customer_name');
        fnd_function.execute(FUNCTION_NAME=>'DEM_DEMXXEOR',
                             OPEN_FLAG=>'Y',
                             SESSION_FLAG=>'Y',
                             OTHER_PARAMS=>'ORDER_ID="'||
                             param_to_pass1||
                             '" CUSTOMER_NAME="'||
                             param_to_pass2||'"');
         /* all the extra single and double quotes account for
         any spaces that might be in the passed values */
      end if;

    else
      null;
    end if;
  end event;
end custom;
```

Example of Implementing Zoom Using the CUSTOM Library

Here is an example of a simple customization you can do with the Zoom feature (in the CUSTOM library).

> **Note:** This Zoom demo/example is based on the training class form DEMXXEOR used in the class "Develop Extensions to Oracle Applications" (available through Oracle Education Services). The form files can be found on the Release 10SC Production 13 (and later) Development Kit CD–ROM, installable using the Oracle Installer file WINDOWS.PRD that you find under the ALPHAINS directory. The product name to install is "Training Class Forms (Develop Extensions to Oracle Applications)". See the Readme.txt file included for setting up the corresponding tables and views in your database.

The DEMXXEOR form is a very simple form for entering orders. In this example, we add two parameters (ORDER_ID and CUSTOMER_NAME) to the form that can be accepted from Zoom upon form startup. The form then fires an automatic query based on the parameters if one of the parameters has a value.

For the Zoom itself, we make Zoom available from the first block of the same form. The user can have a value in one or both of the Order Number and Customer Name fields. When the user clicks on the Zoom button, Zoom opens another session of the same form and automatically queries up any existing orders fitting the criteria.

Once you understand how this Zoom works and is implemented, you should be able to take the same approach with other forms (not necessarily zooming to another session of the same form).

Modify the Form

Using the Oracle Forms Designer, modify the Demo Orders form (DEMXXEOR.fmb) so it is able to receive parameter(s) from the Zoom code.

> **Note:** The DEMXXEOR form shipped on the Release 10SC Production 15 (and later) Development Kit CD–ROM already has these modifications.

1. Create parameter ORDER_ID (Char, size 255, no default value).
2. Create parameter CUSTOMER_NAME (Char, size 255, no default value).

   In theory, the parameter should match both the field the value comes from and the field it goes to, but it depends on the purpose

of the parameter. In this case, having Char instead of number allows wildcards for the automatic query, as does having the parameter longer than the actual field (standard query length for most items is 255, and these parameters will only be used for queries).

3. Modify the default WHERE clause of the target block (ORDERS) to use parameter values as query criteria if they are not null:

```
WHERE (:parameter.order_id is null or
  dem_orders_v.order_id like :parameter.order_id)
AND (:parameter.customer_name is null or
  dem_orders_v.customer_name like
:parameter.customer_name)
```

4. In the WHEN–NEW–FORM–INSTANCE trigger, add to the end of the existing code:

```
/* fire automatic query if a parameter has a value from
Zoom */
if (:parameter.order_id is not null) or
   (:parameter.customer_name is not null) then
  GO_BLOCK('ORDERS');
  do_key('EXECUTE_QUERY');
/* clear the parameters after the query so they don't remain
       criteria for future queries */
  :parameter.order_id := null;
  :parameter.customer_name := null;
end if;
```

5. Compile and generate the form, and put the .fmx file in the appropriate directory (assumes you have a custom application set up from the class).

6. Register the form, define a function for it (DEM_DEMXXEOR), and put the form function on a menu so you can access it. Verify that the form works properly from the Navigator before you modify the CUSTOM library.

Modify the CUSTOM Library

1. Open CUSTOM.PLL in the Oracle Forms Designer.

2. Attach the FNDSQF library (without hardcoded path name). Not necessary for Release Production 15 or later.

3. Modify the text of the CUSTOM package body as follows (the CUSTOM library comes with extensive comments that explain each section. You modify the actual code sections):

```
------------------------------------------------------------
PACKAGE BODY custom IS
  --
  -- Customize this package to provide specific responses to
  -- events within Oracle Applications forms.
  --
  -- Do not change the specification of the CUSTOM package
  -- in any way.
  --
  ------------------------------------------------------------
  function zoom_available return BOOLEAN is
  --
  -- This function allows you to specify if zooms exist for the
  -- current context. If zooms are available for this block, then
  -- return TRUE; else return FALSE.
  --
  -- This routine is called on a per-block basis within every
  -- Applications form. Therefore, any code that will enable
  -- Zoom must test the current
  -- form and block from which the call is being made.
  --
  -- By default this routine must return FALSE.

        form_name  varchar2(30) := name_in('system.current_form');
        block_name varchar2(30) := name_in('system.cursor_block');

  begin
        if (form_name = 'DEMXXEOR' and block_name = 'ORDERS') then
          return TRUE;
        else
          return FALSE;
        end if;

  end zoom_available;
  ------------------------------------------------------------------
  function style(event_name varchar2) return integer is
  --
  --  This Zoom example does not do anything to the STYLE function
  begin
    return custom.standard;
  end style;
  ------------------------------------------------------------------
  procedure event(event_name varchar2) is
  --
  -- This procedure allows you to execute your code at specific
  -- events. 'ZOOM' or product-specific events will be passed
  -- in event_name. See the Applications Technical Reference
  -- manuals for a list of events that are available through
  -- this interface.
```

```
        form_name  varchar2(30) := name_in('system.current_form');
        block_name varchar2(30) := name_in('system.cursor_block');

        param_to_pass1 varchar2(255);
        param_to_pass2 varchar2(255);
  BEGIN
        if (event_name = 'ZOOM') then
          if (form_name = 'DEMXXEOR' and block_name = 'ORDERS')
            then
            param_to_pass1 := name_in('ORDERS.order_id');
            param_to_pass2 := name_in('ORDERS.customer_name');
           /* use fnd_function.execute instead of open_form */
            FND_FUNCTION.EXECUTE(FUNCTION_NAME=>'DEM_DEMXXEOR',
                                 OPEN_FLAG=>'Y',
                                 SESSION_FLAG=>'Y',
                                 OTHER_PARAMS=>
                                'ORDER_ID="'||param_to_pass1||
                                '" CUSTOMER_NAME="'||
                                param_to_pass2||'"');
           /* all the extra single and double quotes account for
              any spaces that might be in the passed values */
          end if;
        else
          null;
        end if;

  end event;
END custom;
--------------------------------------------------------------------
```

4. Compile All and save your changes. Exit the Designer.

5. Use the Forms Generate program to generate a new .plx file for the CUSTOM library.

6. Verify that your file generated successfully. If you leave out the pathname for the output file, the file will be generated in c:\orawin\bin (or platform equivalent). Move it to c:\apps10\au10\.

7. Make sure that the function you call in your Zoom (DEM_DEMXXEOR) is somewhere on the menu in your responsibility. It does not need to be accessible from the menu in the Navigator (do this by adding it to the menu but leaving the prompt blank). Otherwise, the Zoom button will be enabled but the user will get a message saying that function is not available.

8. Try it out from the Oracle Applications Navigator.

APPENDIX

A User Profiles

This appendix lists the profile options that the system administrator can set for the site, application, responsibility, or user. The profile descriptions include the internal name of the profile option used when defaulting values from a profile option.

Profile Options in Oracle Application Object Library

This section lists each profile option in Oracle Application Object Library, which are available to every Oracle Application. For each profile option, we give a brief overview of how Oracle Application Object Library uses the profile's setting.

Account Generator:Purge Runtime Data

"Yes" ensures that the Oracle Workflow data used to generate accounting flexfield code combinations using the Account Generator is purged after the Account Generator has completed.

This profile option should always be set to "Yes" unless you are debugging the Account Generator. Running the Account Generator with this profile set to "No" fills up the workflow tables and slows performance.

Users can see and update this profile option.

This profile option is visible and updatable at all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is ACCOUNT_GENERATOR:PURGE_DATA.

Applications Web Agent

Provides the base URL for the Apps Schema's WebServer DAD. Your System Administrator sets this profile option during the install process.

Use the following syntax to enter your URL:

```
http://<WebServer.Machine_Name>/<DAD_name>/
```

Users can see but not update this profile option.

This profile option is visible and updatable at all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is APPS_WEB_AGENT.

Attachment File Upload Directory

Provides the directory path used to upload attachment files. Your System Administrator sets this profile option during the install process.

Users can see but not update this profile option.

This profile option is visible and updatable at all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is ATTACHMENT_FILE_DIRECTORY

AuditTrail:Activate

You can turn AuditTrail on or off (Yes or No). The default setting is No (Off).

When you enter or update data in your forms, you change the database tables underlying the forms you see and use.

AuditTrail tracks which rows in a database table(s) were updated at what time and which user was logged in using the form(s).

- Several updates can be tracked, establishing a trail of audit data that documents the database table changes.
- AuditTrail is a feature enabled on a form–by–form basis by a developer using Oracle's Application Object Library.
- All the forms that support AuditTrail are referred to as an *audit set*.
- Not all forms may be enabled to support AuditTrail.
- To enable or disable AuditTrail for a particular form, you need access to Oracle Application Object Library's *Application Developer* responsibility.

Users cannot see nor change this profile option.

This profile option is visible and updatable at the site and application levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	No	No
User	No	No

The internal name for this profile option is AUDITTRAIL:ACTIVATE.

Concurrent:Active Request Limit

You can limit the number of requests that may be run simultaneously by each user. or for every user at a site. If you do not specify a limit, no limit is imposed.

Users cannot see or update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	Yes

The internal name for this profile option is CONC_REQUEST_LIMIT.

Concurrent:Attach URL

Setting this option to ”Yes” causes a URL to be 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 on–line.

Only the System Administrator can update this profile option.

Users can see but not update this profile option.

This profile options is visible at all levels but can only updated at the Site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is CONC_ATTACH_URL.

Concurrent:Collect Request Statistics

Set this profile option to "Yes" to have statistics for your runtime concurrent processes collected.

To review the statistics you must run the Purge Concurrent Request and / or Manager Data program to process the raw data and have it write the computed statistics to the FND_CONC_STAT_SUMMARY table. You can then retrieve your data from this table using SQL*PLUS or on a report by report basis using the Diagnostics window from the Requests window.

Users cannot see nor change this profile option.

This profile option is visible at all levels bu can only be updated at the Site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is CONC_REQUEST_STAT.

Concurrent:Debug Flags

Your Oracle support representative may access this profile option to debug Transaction Managers.

Users cannot see nor change this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_DEBUG.

Concurrent:Hold Requests

You can automatically place your concurrent requests on hold when you submit them.

The default is “No”. The concurrent managers run your requests according to the priority and start time specified for each.

Changing this value does not affect requests you have already submitted.

“Yes” means your concurrent requests and reports are automatically placed on hold. To take requests off hold, you:

- Navigate to the Requests window to select a request
- Select the Request Control alternative region
- Uncheck the Hold check box

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_HOLD.

Concurrent:Multiple Time Zones

”Yes” sets the default value to ’Sysdate–1’ for the ’Schedules Start Date’ used by request submissions. Sysdate–1 ensures that you request is scheduled immediately regardless of which time zone your client session is running in. You should use this profile option when the client’s session is running in a different time zone than the concurrent manager’s session.

Users cannot see nor change this profile option.

This profile option is visible at all four levels and updatable at the Site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is CONC_MULTI_TZ.

Concurrent:PMON method

PMON refers to *process monitor*. The Internal Concurrent Manager monitors the individual concurrent managers' processes to verify the managers are running. The Internal Concurrent Manager uses one of two methods to monitor the individual managers's processes:

RDBMS (default) RDBMS refers to *relational database management system*. The Internal Concurrent Manager reads a database table holding values submitted by the individual managers. When a manager is no longer running, the table value for that manager changes.

OS OS refers to *operating system*. The Internal Concurrent Manager sends a signal to the operating system for each manager's system process ID to verify the process is intact.

Attention: The OS method should only be used when a hardware platform or operating system requires it.

To change this profile option setting, you must execute a SQL script from the applsys account. The script is titled afimpmon.sql and is located in the sql directory of the Application Object Library product directory. For example, in UNIX, the path may be $FND_TOP/sql/afimpmon.sql.

Additional Information: *Oracle Applications Installation Manual* for your operating system

Attention: Contact Oracle Worldwide Support before changing your Concurrent:PMON method profile option setting.

Users cannot see nor change this profile option.

This profile option is neither visible nor updatable from the System Profile Options form.

Level	Visible	Allow Update
Site	No	No
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is CONC_PMON_METHOD.

Concurrent:Report Access Level

Determines access privileges to report output files and log files generated by a concurrent program. This profile option can be set by a System Administrator to User or Responsibility.

If your Concurrent:Report Access Level profile option is set to ”User” you may:

- View the completed report output for your requests online
- View the diagnostic log file for those requests online. (system administrator also has this privilege)
- Reprint your completed reports, if the Concurrent:Save Output profile option is set to ”Yes”.
- If you change responsibilities, then the reports and log files available for online review do not change.

If your Concurrent:Report Access Level profile option is set to ”Responsibility”, access to reports and diagnostic log files is based on the your current responsibility.

- If you change responsibilities, then the reports and log files available for online review change to match your new responsibility. You can always see the output and log files from reports you personally submit, but you also see reports and log files submitted by any user from the current responsibility.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at the site, responsibility, and user levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	No	No
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_REPORT_ACCESS_LEVEL.

Concurrent:Report Copies

You can set the number of output copies that print for each concurrent request. The default is set to 1.

- Changing this value does not affect requests that you have already submitted.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_COPIES.

Concurrent:Request Priority

This displays the default priority number for your concurrent requests. Only a system administrator can change your request priority.

Requests normally run according to start time, on a "first–submitted, first–run" basis. Priority overrides request start time. A higher priority request starts before an earlier request.

Priorities range from 1 (highest) to 99 (lowest). The standard default is 50.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_PRIORITY.

Concurrent:Request Start Time

You can set the date and time that your requests are available to start running.

- If the start date and time is at or before the current date and time, requests are available to run immediately.
- If you want to start a request in the future, for example, at 3:45 pm on June 12, 1998, you enter 12–JUN–1998 15:45:00 as the profile option value.
- You must include both a date and a time.
- Changing this value does not affect requests that you have already submitted.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_REQ_START.

Concurrent:Save Output

You can save your request outputs in a file.

- ”Yes” saves request outputs.
- Some concurrent requests do not generate an output file.
- If your request output is saved, you can reprint a request. This is useful when requests complete with an Error status, for example, the request runs successfully but a printer malfunctions.
- Changing this value does not affect requests you have already submitted.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_SAVE_OUTPUT.

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.

- Concurrent programs are incompatible if simultaneously accessing the same database tables incorrectly affects the values each program retrieves.

- When concurrent programs are defined as incompatible with one another, they cannot run at the same time.

"Yes" prevents your requests from running concurrently. Requests run sequentially in the order they are submitted.

"No" means your requests *can* run concurrently when their concurrent programs are compatible.

Changing this value does not affect requests you have already submitted.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_SINGLE_THREAD.

Concurrent:Wait for Available TM

You can specify the maximum number of seconds that the client will wait for a given transaction manager (TM) to become available before moving on to try a different TM.

Users can see and update this profile option.

This profile option is visible and updatable at the site and application levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	No	No
User	No	No

The internal name for this profile option is CONC_TOKEN_TIMEOUT.

Concurrent:URL Lifetime

The numeric value you enter for this profile option determines the length of time in minutes a URL for a request ouput is maintained. After this time period the URL will be deleted from the system. This

profile option only affects URLs created for requests where the user has entered values in the notify field of the Submit Request or Submit Request Set windows.

Attention: All request ouput URLs are deleted when the Pruge Concurrent Requests and Manager... program is run even if the URL liftime has not expired.

Users can see and update this profile option.

This profile option is visible and updatable at the all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CONC_URL_LIFETIME.

Concurrent:Use ICM

"Yes" allows you to use the Internal Concurrent Manager to resolve request conflicts instead of the Conflict Resolution Manager.

Users can see this profile option, but they cannot update it.

This profile option is visible at all four levels and updatable at the Site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is ICM_RESOLVE.

Currency:Mixed Precision

Use Mixed Currency Precision to specify how many spaces are available to the right of the decimal point when displaying numbers representing different currencies.

- Normally, currency numbers are right–justified.
- Each currency has its own precision value that is the number of digits displayed to the right of a decimal point. For U.S. dollars the precision default is 2, so an example display is 345.*70*.

- Set Mixed Currency Precision to be equal to or greater than the *maximum* precision value of the currencies you are displaying.

  For example, if you are reporting on rows displaying U.S. dollars (precision=2), Japanese yen (precision=0), and Bahraini dinar (precision=3), set Mixed Currency Precision=3.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CURRENCY:MIXED_PRECISION.

Currency:Negative Format

You can use different formats to identify negative currency. The default identifier is a hyphen (–) preceding the currency amount, as in ”–xxx”. You can also select:

Angle brackets < >	< xxx >
Trailing hyphen –	xxx –
Parentheses ( )	(xxx)
Square Brackets [ ]	[xxx]

If you use the negative number formats of “(xxx)” or “[xxx],” in Oracle Applications Release 11, your negative numbers appear as “<xxx>”.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CURRENCY:NEGATIVE_FORMAT.

Note: Currency:Negative Format only affects the display currency. Non–currency negative numbers appear with a preceding hyphen regardless of the option selected here.

Currency:Positive Format

You can use different formats to identify positive currency values. The default condition is no special identifier.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CURRENCY:POSITIVE_FORMAT.

Currency:Thousands Separator

You can separate your currency amounts in thousands by placing a thousands separator. For example, one million appears as 1,000,000.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is CURRENCY:THOUSANDS_SEPARATOR.

Database Instance

Entering a valid two_task connect string allows you to override the default two_task. This profile is specifically designed for use with Oracle Parallel Server, to allow different responsibilities and users to connect to different nodes of the server.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is INSTANCE_PATH.

Default Country

This is the default source for the Country field for all address zones and is used by the Flexible Address Formats feature, the Flexible Bank Structures feature and the Tax Registration Number and Taxpayer ID validation routines.

The profile can be set to any valid country listed in the Maintain Countries and Territories form and can be set to a different value for each user.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is DEFAULT_COUNTRY.

Flexfields:AutoSkip

You can save keystrokes when entering data in your flexfields by automatically skipping to the next segment as soon as you enter a complete valid value into a segment.

- "Yes" means after entering a valid value in a segment, you automatically move to the next segment.
- "No" means after entering a valid value in a segment, you must press [Return] to go to the next segment.

  Note: You may still be required to use tab to leave some segments if the valid value for the segment does not have the

same number of characters as the segment. For example, if a segment in the flexfield holds values up to 5 characters and a valid value for the segment is 4 characters, AutoSkip will not move you to the next segment.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:AUTOSKIP.

Flexfields:BiDi Direction

This profile option controls the appearance of the flexfields window in Applications running in Semitic languages. Possible values are ”Left To Right” and ”Right To Left”. If the profile option is not defined on a particular installation, the default value is ”Right To Left”, where the window appears in a normal, left to right fashion, and the text and layout are reversed to accommodate the right–to–left nature of the Semitic language environment.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:BIDI_DIRECTION.

Flexfields:LOV Warning Limit

Use Flexfields:LOV Warning Limit to improve efficiency when retrieving a list of values.

Sometimes, particularly when no reduction criteria has been specified, an LOV can take a very long time to run if there is a very significant

amount of data in it. Set this profile option to the number of rows to be returned before the user is asked whether to continue retrieving the entire list.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is QUICKPICK_ROWS_BEFORE_WARN.

Flexfields:Open Descr Window

You can control whether a descriptive flexfield window automatically opens when you navigate to a customized descriptive flexfield.

- "Yes" means that the descriptive flexfield window automatically opens when you navigate to a customized descriptive flexfield.
- "No" means that when you navigate to a customized descriptive flexfield, you must choose **Edit Field** from the Edit menu or use the List of Values to open the descriptive flexfield window.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:OPEN_DESCR_WINDOW.

> **Note:** This profile option does not apply to descriptive flexfields in folders.

Flexfields:Shared Table Lock

This profile option is reserved for a future release. You should not alter the value of this profile option.

The internal name for this profile option is FLEXFIELDS:SHARED_TABLE_LOCK.

Flexfields:Shorthand Entry

If shorthand flexfield entry is defined for your flexfield, you can use a shorthand alias to automatically fill in values for some or all of the segments in a flexfield.

Not Enabled Shorthand Entry is not available for any flexfields for this user, regardless of whether shorthand aliases are defined.

New Entries Only Shorthand Entry is available for entering new records in most foreign key forms. It is not available for combinations forms, updating existing records, or entering queries.

Query and New Entry Shorthand Entry is available for entering new records or for entering queries. It is not available for updating existing records.

All Entries Shorthand Entry is available for entering new records or updating old records. It is not available for entering queries.

Always Shorthand Entry is available for inserting, updating, or querying flexfields for which shorthand aliases are defined.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:SHORTHAND_ENTRY.

Flexfields:Show Full Value

If an alias defines valid values for *all* of the segments in a flexfield, and Flexfields: Shorthand Entry is enabled, when you enter the alias the flexfield window does not appear.

”Yes” displays the full flexfield window with the cursor resting on the last segment.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:SHOW_FULL_VALUE.

Flexfields:Validate On Server

This profile option is set to ”Yes” to enable server side, PL/SQL flexfields validation for Key Flexfields. This improves performance when using Key Flexfields over a wide area network by reducing the number of network round trips needed to validate the entered segment combinations.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FLEXFIELDS:VALIDATE_ON_SERVER

Folders:Allow Customization

Your system administrator controls whether you can create or customize a folder definition layout in folder block.

- “Yes” means that you can create or customize a folder definition, that is, the entire Folder menu is enabled in the folder block.
- “No” means that you can only open an existing folder definition in a folder block, that is, only the Open option is enabled in the Folder menu.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at the user level.

Level	Visible	Allow Update
Site	No	No
Application	No	No
Responsibility	No	No
User	Yes	Yes

The internal name for this profile option is FLEXVIEW:CUSTOMIZATION.

Gateway User ID

Oracle login for gateway account. For example, *applsyspub/pub*.

Users can see and but not update this profile option.

This profile option is visible at all levels but may only be updated at the site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is GWYUID.

Help System Base URL

Help System Base URL provides the URL to the Applications Help System. Your System Administrator sets this profile option during the install process.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at the all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is HELP_BASE_URL.

Indicate Attachments

This profile option allows you to turn off indication of attachments when querying records (for performance reasons).

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is ATCHMT_SET_INDICATOR.

MO:Operating Unit

Determines the Operating Unit the responsibility logs onto.

Users can see and update this profile option.

This profile option is visible and updatable at the responsibility level only.

Level	Visible	Allow Update
Site	No	No
Application	No	No
Responsibility	Yes	Yes
User	No	No

The internal name for this profile option is ORG_ID.

Personnel Employee:Installed

When enabled, ”Personnel Employee:Installed” allows you as System Administrator to link an application username and password to an employee name.

- The ”Person” field is usable on the Define Application User form (\ **N**avigate **S**ecurity **U**ser).

Oracle Purchasing uses this capability to associate an employee in your organization with an Oracle Applications user.

The installation process enables this profile option. You cannot change the value of ”Personnel Employee: Installed”.

Users cannot see nor change this profile option.

This profile option is visible at the site level, but cannot be updated.

Level	Visible	Allow Update
Site	Yes	No
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is PER_EMPLOYEE:INSTALLED.

Printer

You can select the printer which prints your reports. If a printer cannot be selected, contact your system administrator. Printers must be registered with Oracle Applications.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is PRINTER.

RRA:Delete Temporary Files

When using a custom editor to view a concurrent output or log file, the Report Review Agent will make a temporary copy of the file on the client. Set this profile to "Yes" to automatically delete these files when the user exits Oracle Applications.

Only the System Administrator can update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FS_DELETE.

RRA:Enabled

Set this user profile to "Yes" to use the Report Review Agent to access files on concurrent processing nodes.

Only the System Administrator can update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FS_ENABLED.

RRA:URL

Specify a URL which points to the CGI script on your WebServer to use the Report Review Agent to access files on concurrent processing nodes.

Only the System Administrator can update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FILE_SERVER_URL.

> **Note:** This profile option is valid only for those environments which support Oracle Web–Deployed Applications report viewer.

RRA:Maximum Transfer Size

Specify, in bytes, the maximum allowable size of files transferred by the Report Review Agent, including those downloaded by a user with the "Copy File..." menu option in the Oracle Applications Report File Viewer and those "temporary" files which are automatically downloaded by custom editors. For example, to set the size to 64K you enter 65536. If this profile is null, there is no size limit.

Only the System Administrator can update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is FS_MAX_TRANS.

Sequential Numbering

Sequential Numbering assigns numbers to documents created by forms in Oracle financial products. For example, when you are in a form that creates invoices, each invoice document can be numbered sequentially.

Sequential numbering provides a method of checking whether documents have been posted or lost. Not all forms within an application may be selected to support sequential numbering.

Sequential Numbering has the following profile option settings:

Always Used You may not enter a document if no sequence exists for it.

Not Used You may always enter a document.

Partially Used You will be warned, but not prevented from entering a document, when no sequence exists.

Users can see this profile option, but they cannot update it.

This profile option is visible and updatable at the site, application, and responsibility levels.

> **Note:** If you need to control Sequential Numbering for each of your set of books, use the 'Responsibility' level. Otherwise, we recommend that you use either the 'Site' or 'Application' level to set this option.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	No	No

The internal name for this profile option is UNIQUE:SEQ_NUMBERS.

Session ID

This runtime profile option contains the session ID number of the last database session that was created.

Users can see this profile option, but they cannot update it.

This profile option is neither visible nor updatable from the System Profile Options form.

Level	Visible	Allow Update
Site	No	No
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is DB_SESSION_ID.

Sign–On:Audit Level

Sign–On:Audit Level allows you to select a level at which to audit users who sign on to Oracle Applications. Four audit levels increase in functionality: None, User, Responsibility, and Form.

None is the default value, and means do not audit any users who sign on to Oracle Applications.

Auditing at the User level tracks:

- who signs on to your system
- the times users log on and off
- the terminals in use

Auditing at the Responsibility level performs the User level audit functions and tracks:

- the responsibilities users choose
- how much time users spend using each responsibility

Auditing at the Form level performs the Responsibility level audit functions and tracks:

- the forms users choose
- how long users spend using each form
- System Administrator visible, updatable at all levels.

Users cannot see nor change this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is SIGNONAUDIT:LEVEL.

Sign–On:Notification

”Yes” displays a message at login that indicates:

- If any concurrent requests failed since your last session,
- How many times someone tried to log on to Oracle Applications with your username but an incorrect password, and
- When the default printer identified in your user profile is unregistered or not specified.

Users can see and update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is SIGNONAUDIT:NOTIFY.

Signon Password Length

Signon Password Length sets the minimum length of an Applications signon password. If no value is entered the minimum length defaults to 5.

Users can see but not update this profile option.

This profile option is visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal name for this profile option is SIGNON_PASSWORD_LENGTH.

Site Name

Site Name identifies an installation of Oracle Applications. The installation process sets this to "No Site Name Specified".

You should set a value for "Site Name" after installation.

Users cannot see nor change this profile option.

This profile option is visible and updatable at the site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is SITENAME.

Stored Procedure Log Directory

Specifying a log directory enables stored procedures used with Oracle 7.3 to generate and store log files. You must also set this log directory in the init.ora file of the database.

For example, if the Stored Procedure Log Directory is /rladev/rla/1.1/log and the Stored Procedure Output Directory is /rladev/rla/1.1/out, then the following entry should be made in the init.ora file of the database containing stored procedures that write to these directories:

```
UTL_FILE_DIR = /rladev/rla/1.1/log,
               /rladev/rla/1.1/out
```

Users cannot see nor change this profile option.

This profile option is visible and updatable at the site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is UTL_FILE_LOG.

Stored Procedure Output Directory

Specifying a output directory enables stored procedures used with Oracle 7.3 to generate and store output files. You must also set this output directory in the init.ora file of the database.

For example, if the Stored Procedure Log Directory is /rladev/rla/1.1/log and the Stored Procedure Output Directory is /rladev/rla/1.1/out, then the following entry should be made in the init.ora file of the database containing stored procedures that write to these directories:

```
UTL_FILE_DIR = /rladev/rla/1.1/log,
               /rladev/rla/1.1/out
```

Users cannot see nor change this profile option.

This profile option is visible and updatable at the site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	No	No
Responsibility	No	No
User	No	No

The internal name for this profile option is UTL_FILE_OUT.

Two Task

TWO_TASK for the database. This profile is used in conjunction with the Gateway User ID profile to construct a connect string for use in creating dynamic URLs for the Web Server. This should be set to the SQL*NET. alias for the database.

> **Note:** The TWO_TASK must be valid on the node upon which the WebServer is running

Users can see and but not update this profile option.

This profile option is visible at all levels but may only be updated at site level.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	No
Responsibility	Yes	No
User	Yes	No

The internal name for this profile option is TWO_TASK.

Utilities:SQL Trace

SQL trace files can now be generated for individual concurrent programs. The trace can be enabled at the user level by setting the profile "Utilities:SQL Trace" to "Yes". This profile can be enabled for a user only by System Administrator so that it is not accidentally turned on and disk usage can be monitored.

For more information on SQL trace, see the ORACLE7 Server SQL Language Reference Manual.

Users cannot see nor change this profile option.

This profile option is visible and updatable at the all levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

Viewer: Text, PostScript, PDF, and HTML

You can use the viewer of your choice when reviewing online the log and output files from your concurrent requests. In almost all cases, you need to specify different viewers for reviewing text files, postscript (bitmap) files, PDF files, and HTML files. Oracle Report programs with VERSION=2.0b as the concurrent program's execution option produce postscript files.

To link the Oracle Applications concurrent processing facility with a text, postscript, PDF, or HTML, with the appropriate application installed on your computer, enter as the profile option value the:

- Name of the application's executable file, when the file can be found in your computer's default path (for example, **write.exe**), or
- Full path to the application's executable file, when the file is not in the default path of your compute (for example, **c:\windows\bin\write.exe**).

You should also enter any required parameters that may be necessary to invoke the application in a particular mode.

Attention: In a client–server environment, PC clients can access log and output files on the servers by using either a Report Review Agent or a PC–NFS mount.

Users can see and update the Viewer profile options.

These profile options are both visible and updatable at all four levels.

Level	Visible	Allow Update
Site	Yes	Yes
Application	Yes	Yes
Responsibility	Yes	Yes
User	Yes	Yes

The internal names for these four profile options are EDITOR_CHAR, EDITOR_PS, VIEWER_PDF, and VIEWER_HTML.

APPENDIX

B Loaders

This appendix lists the loader programs that the system administrator can use to load messages, user profiles and user profile values.

Message Dictionary Generator

The Message Dictionary Generator (FNDMDGEN) is a concurrent program that transfers Oracle Applications Message Dictionary messages between three different kinds of message repositories: the database, readable text files and binary runtime files. The following sections describe the operation of the Message Dictionary Generator.

Message Repositories

Message information is stored in three different repositories, each of which has its own format and serves a specific need. Following is a description for each of the three 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: `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: `NAME, NUMBER, TEXT`

Script

A script file is an ASCII file that stores messages for viewing and editing by humans. Although a script file can contain multiple

languages and applications, it normally contains only one language and one application.

Script files end with the suffix ".msg". The default file name for script files is "text.msg". The script format stores the listed attributes, as well as others that are for use by the Oracle Applications translation team.

Script Attributes: `APPLICATION, LANGUAGE, NAME, NUMBER, TEXT, DESCRIPTION, MAX_LENGTH, TYPE`

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 one of:

DB_TO_RUNTIME From Database to Runtime file (.msb)

DB_TO_SCRIPT From Database to Script file (.msg)

RUNTIME_TO_DB From Runtime file (.msb) to Database

SCRIPT_TO_DB From Script file (.msg) to Database

The argument filename is used only for script files. The default script filename, if none is specified, is text.msg. Otherwise, the script file is found relative to the current directory.

Wildcards

Normally, the Message Dictionary Generator is run to transfer messages for a specific application/language pair from one repository to another. Either <language codename> or [application shortname] can be wildcarded by passing the value "ALL". The following list describes how wildcards are used in the various modes:

To DB Messages get merged into the FND_NEW_MESSAGES table. If a message already exists (same application, language, and message name), it is updated; otherwise, a new message is inserted. Other messages remain untouched.

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.
From RUNTIME	A wildcard for language_codename causes all files in the message directory with valid LANG.msb filenames to be scanned. A wildcard for application_shortname causes all application basepaths to be scanned.
To SCRIPT	Wildcards cause all the messages for all applications and languages to be deposited into one file.
From SCRIPT	All the messages in the file that match the language_codename and application_shortname to be transferred.

Examples of Message Dictionary Generator Usage

Here are some examples of how to run the Message Dictionary Generator:

FNDMDGEN dbuser/dbpassword 0 Y US FND DB_TO_RUNTIME

Creates a US.msb runtime file $FND_TOP/mesg/US.msb. Note that if this command is run against an invalid database, then the runtime message files that the application depends on will be clobbered.

FNDMDGEN dbuser/dbpassword 0 Y ALL ALL SCRIPT_TO_DB patch.msg

Merges a patch file containing various messages into the FND_MESSAGES table. In this example, the file would exist in the directory $FND_TOP/mesg/patch.msg.

Attention: Because the runtime file is not alphabetized, if you use the RUNTIME_TO_SCRIPT mode, the messages in the script will not be ordered by name, as they would be if you used DB_TO_SCRIPT.

Attention: You will not get the same script back if you convert a SCRIPT format to another repository format and then back again, because the SCRIPT format stores more information than the other formats.

Script File Format

A script file looks like this:

```
<HEADER>
<APPLICATION=FND>
<CHARACTER_SET=WE8ISO8859P1>
<LANGUAGE=US>
<RCS_HEADER=$Header$>
<VERSION=>
<LEVEL=>
<TRANSLATOR=>
<TRANSLATED_DATE=>
<REVIEWER=>
<REVIEWED_DATE=>
<TESTER=>
<TESTED_DATE=>
<MESSAGES>
<NAME=AUDIT–START END DATE1>
<NUMBER=718>
<TOKEN=START, END>
<TYPE=>
<DESCRIPTION=>
<STATUS=NEW>
<NOTES=>
<MAX_LENGTH=#432>
<TEXT=>
Please enter a start date less than or equal to the
end date[\n]
[\n]
Cause:[\t]You entered a start date [START] that is
greater than the end date
[END].[\n]
[\n]
Action:[\t]Enter a start date less than or equal to
the end date.[\n]
[\n]
Action:[\t]Change the end date to be greater than or
equal to the start date.
 You must temporarily change the start date to be
less than or equal to the
```

```
end date in order to move from the start date field
to the end date field.[\n]
<NAME=CONC–FDPSEV NO ENV VAR>
<NUMBER=1362>
<TOKEN=>
<TYPE=>
<DESCRIPTION=This is a description of the message for
translators.  It
can span multiple lines if necessary, but shouldn’t
have newlines in it.>
<STATUS=NEW>
<NOTES=>
<MAX_LENGTH=#353>
<TEXT=>
You have specified invalid arguments for concurrent
program FDPSEV[\n]
[\n]
Cause:[\t]You entered an incorrect number of
arguments to the concurrent
program FDPSEV, or the arguments you entered are not
the type expected.[\n]
[\n]
Action:[\t]Enter the correct arguments for FDPSEV as
follows.[\n]
[\n]
Syntax: FDPSEV <<Environment Variable>>=<<Value>>
<<Environment
Variable>>=<<Value>> ...[\n]
....
...
..
```

All of the messages in the <MESSAGES> section belong to the language and application specified in the <HEADER> section. There can be multiple <HEADER> and <MESSAGE> sections in the file, each with a different language / application combination.

This SCRIPT format contains many tags, but some are ignored by the generator when reading the SCRIPT file. The only tags read by the generator are:

APPLICATION The application short name

LANGUAGE The language code set name

NAME The message name

NUMBER The message number

DESCRIPTION A description of the context of the message for translators.

TEXT The translated message text. The message text is the only tag that appears on a separate line from its tag.

String	Meaning
[\t]	Tab (only valid at the beginning of a line)
[\n]	Newline (really means ”end of paragraph”)
[[	Literal ’[’
]]	Literal ’]’
<<	Literal ’<’
>>	Literal ’>’

Table 11 – 1 (Page 1 of 1)

Upon reading the text from a script file:

- Blank lines are ignored
- Spaces are inserted after lines that are not blank and do not end in newline ([\n])
- Character conversions are performed as per the above table.
- [TOKEN] goes to &TOKEN

Upon converting the text into a script file:

- The message text and description are word wrapped
- Character conversions are performed as per the above table
- &TOKEN goes to [TOKEN]

These conversions occur for all of the tag values, not just the TEXT. Also note that any spaces at the end of a message will be stripped on conversion of the message into script format. So there is no way to represent messages that have trailing spaces.

Function Security Loader

The Function Security Loader (FNDSLOAD) is a concurrent program that can move Oracle Applications function security information between database and text file representations. The following sections describe the operation of the Function Security Loader.

Use the Function Security Loader to preserve custom responsibility and menu information across upgrades by downloading files before the upgrade and then uploading them afterward.

Supported Operations

The Function Security Loader allows you to move function security data between the database (where it is used for runtime operation) and a text file representation (where it can be used for distribution). Specifically, you can:

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. In other words, the FUNCTION_NAME, not the FUNCTION_ID, is used to identify records.

Upload (merge) the information in a text file to the database

If a row is already correct in the database, it is not touched. If a row exists, but has different attributes, the row is updated. If a row does not exist, a new row is inserted. No rows are deleted, even if they are absent from the uploaded file.

These download and upload capabilities allow function security 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 function security definitions from a primary site to other sites.

The text file version of function security data is also useful for bulk editing operations, which can be accomplished more efficiently with a text editor than with a form.

Usage

The Function Security Loader takes the following arguments:

```
FNDSLOAD <username/password> 0 Y
  [{ LOCAL <filepath> } |
```

```
  { <appsname> <subdir>/.../<subdir>/<fname>.ext<}]
[UPLOAD | {DOWNLOAD <menuname> ... <menuname>}]
```

where

<username/ password> is the APPLSYS account

<appsname> is an application short name or ”LOCAL”

<fname> is the file that will be read or written

<menuname> is a list of menu names to download

The location of the file is determined by the <appsname> and <fname> arguments. <appsname> indicates whether the file is in the install/import directory for a particular application, or whether it is in the local directory.

Appsname	File	Location
<APP>	<file>	$<APP_TOP>/install/import/<file>.slt
LOCAL	<file>	<file>

Table 11 – 2 (Page 1 of 1)

Examples

FNDSLOAD applsys/fnd 0 Y FND install/import/sysadmin.slt UPLOAD

references file ”$FND_TOP/install/import/sysadmin.slt”

FNDSLOAD applsys/fnd 0 Y LOCAL sysadmin.slt UPLOAD

references file ”sysadmin.slt”

UPLOAD and DOWNLOAD

The loader works in either upload or download mode. In UPLOAD mode the entire file is read and merged into the database. (Any existing entries for a menu are deleted in favor of the entry list in the file, if conflicts occur.)

In DOWNLOAD mode the menu tree rooted at each <menuname> listed is read from the database and written to the file, along with every

function and form referenced by any menu. Note the <menuname> is the actual menu name, not the User Menu Name.

FNDSLOAD applsys/fnd 0 Y FND sysadmin.slt DOWNLOAD FND_NAVIGATE4.0

will download all menus under ”FND_NAVIGATE4.0” to the file sysadmin.slt

FNDSLOAD applsys/fnd 0 Y FND sysadmin.slt UPLOAD

will upload everything in the file sysadmin.slt to the database

Function Security File Format

Function security information is stored in text file format by writing out a series of ”records” for each form, function, and menu. The file is plain text that can be viewed and edited with any standard text editor. You need to understand this file format only if you plan to edit the file.

The file consists of a series of text lines, each of which must be less than 1024 bytes in length (when written, no line will exceed 80 bytes in length).

A line may be a comment line, meaning it has no effect on the data contained in the file. Comment lines begin with a ”#” and end with a newline.

```
# This is a comment line
```

The information in the file consists of a series of tokens delimited by white space. Note that a token may itself contain a space, in which case it is delimited with double–quote marks. Tokens may not contain non–printing characters unless they are preceded by the ”escape” character, which is a backslash. Predefined escapes are:

String	Meaning
\e	Escape
\n	Newline
\r	Carriage return
\b	Backspace
\v	Vertical tab

Table 11 – 3 (Page 1 of 2)

String	Meaning
\f	Form feed
\"	The double quote itself
\ \	The backslash itself

Table 11 – 3 (Page 2 of 2)

Other escapes consist of octal values preceded by the backslash, as in "\007". If a line is too long to fit within 80 characters, the line is broken into as many "continuation" lines as necessary. A line ending with a single backslash is taken to be continued on the next line. In such a case, the newline following the backslash is ignored, rather than being taken as a white space delimiter. The line following such a line is taken to be a continuation of the "broken" line, and such a line can itself end in a backslash, indicating further continuation. Note that in the case of continuation lines, a leading "#" is not taken as a comment indicator, but rather as part of the continued data item.

The file starts with three lines as follows:

```
LANGUAGE = "AMERICAN"
CODESET = "WE8ISO8859P1"
TRANSLATED = "Y"
```

These lines indicate the NLS_LANG language and codeset of the file and the current translation status of any translated column values.

Then there is a series of object definitions of the format:

```
BEGIN <object type> <object identifier>
 <attribute> = <value>
 ...
BEGIN <sub–object type> <object identifier>
 ...
END <sub–object type>
 ...
END <object type>
```

There are three main object types, and one sub–object type. The main objects are FORMs, FUNCTIONs, and MENUs, and the sub–object is ENTRY. Each object type has some identifier, and then a series of attribute values. If an attribute value is not specified, it is assumed to be NULL.

FORM record

Form information is written out in the following format:

```
BEGIN FORM <app_short_name> <form_name>
 USER_FORM_NAME = "<user_form_name: 80 char
 translated>"
 DESCRIPTION = "<description: 240 char translated>"
END FORM
```

Forms are identified by app_short_name and form_name. A FORM record is written out for each form referenced by the menus being downloaded.

Example:

```
BEGIN FORM SQLGL GLXJEENT
 USER_FORM_NAME = "Enter Journals/Encumbrances    "
 DESCRIPTION = "Enter journals/encumbrances  "
END FORM
```

FUNCTION record

Function information is written out in the following format:

```
BEGIN FUNCTION <function_name>
 FORM = <app_short_name> <form_name>
 TYPE = <type:30 char>
 PARAMETERS = "<parameters: 240 char>"
 USER_FUNCTION_NAME = "<user_function_name: 80 char
 translated>"
 DESCRIPTION = "<description: 240 char translated>"
END FUNCTION
```

Functions are identified by the developer function_name. A FUNCTION record is written out for each function referenced by the menus being downloaded.

Example:

```
BEGIN FUNCTION SQLGL_GLXJEENT_A
 FORM = SQLGL GLXJEENT
 TYPE = FORM
 PARAMETERS = "actual_flag=\"A\"
 HELP_TARGET=\"GLXJEENT_A\""
 USER_FUNCTION_NAME = "Enter Journals"
 DESCRIPTION = "Enter journals"
END FUNCTION
```

MENU record

Menu information is really a compound record that consists of an MENU record, which contains (encloses) some number of ENTRY records. Menu information is formatted as follows:

```
MENU <menu_name>
 USER_MENU_NAME = "<user_menu_name: 80 char
 translated>"
 DESCRIPTION = "<description: 240 char translated>"
 BEGIN ENTRY
  PROMPT = "<prompt: 30 char translated>"
  DESCRIPTION = "<description: 240 char translated>"
  SUBMENU = <sub_menu_name>
  FUNCTION = <function_name>
 END ENTRY
 ... more entry records ...
END MENU
```

Example:

```
MENU GL_SUPERVISOR_GUI
 USER_MENU_NAME = GL_SUPERVISOR_GUI
 BEGIN ENTRY
  PROMPT = Journals
  DESCRIPTION = "Enter and post actual and
   encumbrance journals"
  SUBMENU = GL_SU_JOURNAL_GUI
 END ENTRY
 BEGIN ENTRY
  PROMPT = Budgets
  DESCRIPTION = "Define and enter budgets"
  SUBMENU = GL_SU_BUDGET_GUI
 END ENTRY
 BEGIN ENTRY
  PROMPT = "Run Reports"
  DESCRIPTION = "Run a report"
  FUNCTION = SQLGL_FNDRSRUN_GLMODE
  END ENTRY
END MENU
```

Menus are identified by the developer menu name. Menu entries have no identification per se, but are assumed to be sequenced based on the order in which they are defined, and thus the sequence number forms

an implicit identifier. Each listed menu is downloaded along with all referenced submenus (recursively).

Sample Function Security File

```
#
# $Header$
#
LANGUAGE = "AMERICAN"
CODESET = "WE8ISO8859P1"
TRANSLATED = "Y"
BEGIN FORM FND FNDCPQCR
  USER_FORM_NAME = "View Requests"
  DESCRIPTION = "User form to view personal, explicit
concurrent requests"
END FORM
BEGIN FORM FND FNDPOMSV
  USER_FORM_NAME = "Update Personal Profile Values"
  DESCRIPTION = "User form to set a personal profile
option"
END FORM
BEGIN FORM FND FNDRSRUN
  USER_FORM_NAME = "Run Reports"
END FORM
BEGIN FORM FND FNDCPDIA
  USER_FORM_NAME = "Request Help"
  DESCRIPTION = "Request Diagnostics Form (also View
Reports)"
END FORM
BEGIN FORM FND FNDRSSET
  USER_FORM_NAME = "Administer Report Sets"
  DESCRIPTION = "System Administrator form to
administer all report sets"
END FORM
BEGIN FUNCTION FND_FNDCPQCR
  FORM = FND FNDCPQCR
  TYPE = FORM
  USER_FUNCTION_NAME = "Concurrent Requests: View
All"
```

```
  DESCRIPTION = ”Concurrent Requests: View Form”
END FUNCTION
BEGIN FUNCTION FND_FNDPOMSV
  FORM = FND FNDPOMSV
  TYPE = FORM
  USER_FUNCTION_NAME = ”Profile User Values”
  DESCRIPTION = ”Profile User Values Form”
END FUNCTION
BEGIN FUNCTION FND_FNDRSRUN
  FORM = FND FNDRSRUN
  TYPE = FORM
  USER_FUNCTION_NAME = ”Requests: Submit”
  DESCRIPTION = ”Requests: Run Form”
END FUNCTION
BEGIN FUNCTION FND_FNDCPDIA_VIEW
  FORM = FND FNDCPDIA
  TYPE = FORM
  PARAMETERS = ”MODE=\”VIEW\””
  USER_FUNCTION_NAME = ”Concurrent Requests: View
Completed”
  DESCRIPTION = ”Concurrent Request Diagnostics Form:
View Requests Mode”
END FUNCTION
BEGIN FUNCTION FND_FNDRSSET_USER
  FORM = FND FNDRSSET
  TYPE = FORM
  PARAMETERS = ”MODE=\”USER\””
  USER_FUNCTION_NAME = ”Request Sets (User Mode)”
  DESCRIPTION = ”Request Sets: User Mode Form”
END FUNCTION
BEGIN MENU GL_SU_MANAGER_GUI
  USER_MENU_NAME = GL_SU_MANAGER_GUI
  DESCRIPTION = ”Called by GL_SUPERUSER4.0,
GL_USER4.0, GL_SUPERVISOR4.0, GL_B\
UDGETUSER4.0, GL_BUDGETSUPER4.0”
  BEGIN ENTRY
    PROMPT = Requests
    DESCRIPTION = ”Review concurrent manager
requests”
```

```
    FUNCTION = FND_FNDCPQCR
  END ENTRY
  BEGIN ENTRY
    PROMPT = Profile
    DESCRIPTION = ”Review personal profile options”
    FUNCTION = FND_FNDPOMSV
  END ENTRY
  BEGIN ENTRY
    PROMPT = Report
    DESCRIPTION = ”Group and request standard
reports”
    SUBMENU = FND_REPORT4.0
  END ENTRY
END MENU
BEGIN MENU FND_REPORT4.0
  USER_MENU_NAME = ”FND_REPORT 4.0”
  DESCRIPTION = ”Standard Report Submission and view
report for Forms 4.0”
  BEGIN ENTRY
    PROMPT = Run
    DESCRIPTION = ”Submit requests”
    FUNCTION = FND_FNDRSRUN
  END ENTRY
  BEGIN ENTRY
    PROMPT = View
    DESCRIPTION = ”View completed requests”
    FUNCTION = FND_FNDCPDIA_VIEW
  END ENTRY
  BEGIN ENTRY
    PROMPT = Set
    DESCRIPTION = ”Define standard request sets”
    FUNCTION = FND_FNDRSSET_USER
  END ENTRY
END MENU
```

User Profile Loader

The User Profile Loader (FNDPLOAD) is a concurrent program that can move Oracle Applications user profile information between database and text file representations. The following sections describe the operation of the User Profile Loader.

Supported Operations

The User Profile Loader allows you to move profile data between the database (where it is used for runtime operation) and a text file representation (where it can be used for distribution). Specifically, you can:

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. In other words, the PROFILE_OPTION_NAME, not the PROFILE_OPTION_ID, is used to identify records.

Upload (merge) the information in a text file to the database

If a row is already correct in the database, it is not touched. If a row exists, but has different attributes, the row is updated. If a row does not exist, a new row is inserted. No rows are deleted, even if they are absent from the uploaded file.

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.

Usage

The User Profile Loader takes the following arguments:

```
FNDPLOAD <username/password> 0 Y
  <appsname> <filename> UPLOAD | DOWNLOAD
  <profappsname> ...
```

where

<username/ password>	is the APPLSYS account
<appsname>	is an application short name or ”LOCAL”
<filename>	is the file that will be read or written
<profappsname>	is a list of applications profiles to download, or ”ALL”

The location of the file is determined by the <appsname> and <filename> arguments. <appsname> indicates whether the file is in the install/import directory for a particular application, or whether it is in the local directory.

Appsname	File	Location
<APP>	<file>	$<APP_TOP>/install/import/<file>.slt
LOCAL	<file>	<file>

Table 11 – 4 (Page 1 of 1)

Examples

FNDPLOAD applsys/fnd 0 Y FND install/import/FND.plt UPLOAD

references file ”$FND_TOP/install/import/FND.plt”

FNDPLOAD applsys/fnd 0 Y FND.plt UPLOAD

references file ”FND.plt”

UPLOAD and DOWNLOAD

The loader works in either upload or download mode. In UPLOAD mode the entire file is read and merged into the database. In DOWNLOAD mode for one or more application names, the loader will download all profiles belonging to those applications to the file. ”DOWNLOAD ALL” will download all profiles belonging to all applications to the file.

FNDPLOAD applsys/fnd 0 Y FND FND.plt DOWNLOAD FND

will download all profiles belonging to application FND to the file FND.plt

FNDPLOAD applsys/fnd 0 Y FND FND.plt UPLOAD

will upload everything in the file FND.plt to the database

Profile File Format

Profile information is stored in text file format by writing out a series of ”records” for each profile. The file is plain text that can be viewed and edited with any standard text editor. You need to understand this file format only if you plan to edit the file.

The file consists of a series of text lines, each of which must be less than 1024 bytes in length (when written, no line will exceed 80 bytes in length).

A line may be a comment line, meaning it has no effect on the data contained in the file. Comment lines begin with a ”#” and end with a newline.

```
# This is a comment line
```

The information in the file consists of a series of tokens delimited by white space. Note that a token may itself contain a space, in which case it is delimited with double–quote marks. Tokens may not contain non–printing characters unless they are preceded by the ”escape” character, which is a backslash. Predefined escapes are:

String	Meaning
\e	Escape
\n	Newline
\r	Carriage return
\b	Backspace
\v	Vertical tab
\f	Form feed
\”	The double quote itself
\\	The backslash itself

Table 11 – 5 (Page 1 of 1)

Other escapes consist of octal values preceded by the backslash, as in ”\007”. If a line is too long to fit within 80 characters, the line is broken into as many ”continuation” lines as necessary. A line ending with a

single backslash is taken to be continued on the next line. In such a case, the newline following the backslash is ignored, rather than being taken as a white space delimiter. The line following such a line is taken to be a continuation of the "broken" line, and such a line can itself end in a backslash, indicating further continuation. Note that in the case of continuation lines, a leading "#" is not taken as a comment indicator, but rather as part of the continued data item.

The file starts with three lines as follows:

```
LANGUAGE = "AMERICAN"
CODESET = "WE8ISO8859P1"
TRANSLATED = "Y"
```

These lines indicate the NLS_LANG language and codeset of the file and the current translation status of any translated column values.

Then there is a series of object definitions of the format:

```
BEGIN <object type> <object identifier>
 <attribute> = <value>
 ...
END <object type>
```

There is only one type of object in the file: PROFILE. Each object type has some identifier, followed by a series of attribute values. If an attribute value is not specified, it is assumed to be NULL.

PROFILE record

Profiles are identified by PROFILE_OPTION_NAME. Profile information is formatted as follows:

```
BEGIN PROFILE <profile_name>
 APPLICATION_SHORT_NAME =   50-character value
 USER_PROFILE_OPTION_NAME = 240-character translated
                            value
 DESCRIPTION =              240-character translated
                            value
 USER_CHANGEABLE_FLAG =     1-character value
 USER_VISIBLE_FLAG =        1-character value
 READ_ALLOWED_FLAG =        1-character value
 WRITE_ALLOWED_FLAG =       1-character value
 SITE_ENABLED_FLAG =        1-character value
 SITE_UPDATE_ALLOWED_FLAG = 1-character value
 APP_ENABLED_FLAG =         1-character value
```

```
 APP_UPDATE_ALLOWED_FLAG =  1–character value
 RESP_ENABLED_FLAG =        1–character value
 RESP_UPDATE_ALLOWED_FLAG = 1–character value
 USER_ENABLED_FLAG =        1–character value
 USER_UPDATE_ALLOWED_FLAG = 1–character value
 START_DATE_ACTIVE =        DD/MM/YYYY
 END_DATE_ACTIVE =          DD/MM/YYYY or NULL
 SQL_VALIDATION =           2000–character value
END PROFILE
```

Example:

```
BEGIN PROFILE ATCHMT_SET_INDICATOR
 APPLICATION_SHORT_NAME = FND
 USER_PROFILE_OPTION_NAME = ”Indicate Attachments”
 DESCRIPTION = ”Indicate whether attachments exist”
 USER_CHANGEABLE_FLAG = Y
 USER_VISIBLE_FLAG = Y
 READ_ALLOWED_FLAG = Y
 WRITE_ALLOWED_FLAG = Y
 SITE_ENABLED_FLAG = Y
 SITE_UPDATE_ALLOWED_FLAG = Y
 APP_ENABLED_FLAG = Y
 APP_UPDATE_ALLOWED_FLAG = Y
 RESP_ENABLED_FLAG = Y
 RESP_UPDATE_ALLOWED_FLAG = Y
 USER_ENABLED_FLAG = Y
 USER_UPDATE_ALLOWED_FLAG = Y
 START_DATE_ACTIVE = 01/01/1900
 END_DATE_ACTIVE = ””
 SQL_VALIDATION = ”SQL=\”SELECT MEANING \\\”Indicate
Attachments Exist\\\”, LOOKUP_CODE\ninto\
:visible_option_value,\n
:profile_option_value\nfrom\ fnd_lookups\nwhere
lookup_type =\ ’YES_NO’\”\nCOLUMN=\”\\\”Indicate
Attachments Exist\\\”(10)\””
END PROFILE
```

Sample Profile File

```
#
# $Header$
#
LANGUAGE = ”AMERICAN”
CODESET = ”US7ASCII”
TRANSLATED = ”Y”#
BEGIN PROFILE ATCHMT_SET_INDICATOR
  APPLICATION_SHORT_NAME = FND
  USER_PROFILE_OPTION_NAME = ”Indicate Attachments”
  DESCRIPTION = ”Indicate whether attachments exist”
  USER_CHANGEABLE_FLAG = Y
  USER_VISIBLE_FLAG = Y
  READ_ALLOWED_FLAG = Y
  WRITE_ALLOWED_FLAG = Y
  SITE_ENABLED_FLAG = Y
  SITE_UPDATE_ALLOWED_FLAG = Y
  APP_ENABLED_FLAG = Y
  APP_UPDATE_ALLOWED_FLAG = Y
  RESP_ENABLED_FLAG = Y
  RESP_UPDATE_ALLOWED_FLAG = Y
  USER_ENABLED_FLAG = Y
  USER_UPDATE_ALLOWED_FLAG = Y
  START_DATE_ACTIVE = 01/01/1900
  END_DATE_ACTIVE = ””
  SQL_VALIDATION = ”SQL=\”SELECT MEANING \\\”Indicate
Attachments Exist\\\”, LOOKUP_CODE\ninto
:visible_option_value,\n
:profile_option_value\nfrom fnd_\ lookups\nwhere
lookup_type = ’YES_NO’\”\nCOLUMN=\”\\\”Indicate
Attachments Exist\\\”(10)\””
END PROFILE
BEGIN PROFILE AUDITTRAIL:ACTIVATE
  APPLICATION_SHORT_NAME = FND
  USER_PROFILE_OPTION_NAME = ”AuditTrail:Activate”
  DESCRIPTION = ”Activate AuditTrail”
  USER_CHANGEABLE_FLAG = N
  USER_VISIBLE_FLAG = N
  READ_ALLOWED_FLAG = Y
```

```
  WRITE_ALLOWED_FLAG = Y
  SITE_ENABLED_FLAG = Y
  SITE_UPDATE_ALLOWED_FLAG = Y
  APP_ENABLED_FLAG = Y
  APP_UPDATE_ALLOWED_FLAG = Y
  RESP_ENABLED_FLAG = N
  RESP_UPDATE_ALLOWED_FLAG = N
  USER_ENABLED_FLAG = N
  USER_UPDATE_ALLOWED_FLAG = N
  START_DATE_ACTIVE = 01/01/1900
  END_DATE_ACTIVE = ""
  SQL_VALIDATION = "SQL=\"SELECT MEANING
\\\"AuditTrail:Activate\\\", LOOKUP_CODE\n   INTO
:VISIBLE_OPTION_VALUE, :PROFILE_OPTION_VALUE\n   FROM
FND_LOOKUPS\n  WHERE LOOKUP_TYPE =
'YES_NO'\"\nCOLUMN=\"\\\"AuditTrail:Activate\\\"(*)\"
END PROFILE
BEGIN PROFILE AUTOCOMMIT
  APPLICATION_SHORT_NAME = FND
  USER_PROFILE_OPTION_NAME = "AutoCommit"
  DESCRIPTION = "Automatically commit all changes
when leaving a form"
  USER_CHANGEABLE_FLAG = Y
  USER_VISIBLE_FLAG = Y
  READ_ALLOWED_FLAG = Y
  WRITE_ALLOWED_FLAG = Y
  SITE_ENABLED_FLAG = Y
  SITE_UPDATE_ALLOWED_FLAG = Y
  APP_ENABLED_FLAG = Y
  APP_UPDATE_ALLOWED_FLAG = Y
  RESP_ENABLED_FLAG = Y
  RESP_UPDATE_ALLOWED_FLAG = Y
  USER_ENABLED_FLAG = Y
  USER_UPDATE_ALLOWED_FLAG = Y
  START_DATE_ACTIVE = 01/01/1900
  END_DATE_ACTIVE = ""
  SQL_VALIDATION = "SQL=\"select meaning
AUTOCOMMIT,lookup_code\n   into
:visible_option_value,:profile_option_value\n
```

```
from fnd_lookups\n  where\
 lookup_type = 'YES_NO'\"\nCOLUMN=\"AutoCommit(*)\""
END PROFILE
BEGIN PROFILE CALCULATOR:TYPE
  APPLICATION_SHORT_NAME = FND
  USER_PROFILE_OPTION_NAME = "Calculator:Type"
  DESCRIPTION = "Calculator or Adding Machine"
  USER_CHANGEABLE_FLAG = Y
  USER_VISIBLE_FLAG = Y
  READ_ALLOWED_FLAG = Y
  WRITE_ALLOWED_FLAG = Y
  SITE_ENABLED_FLAG = Y
  SITE_UPDATE_ALLOWED_FLAG = Y
  APP_ENABLED_FLAG = Y
  APP_UPDATE_ALLOWED_FLAG = Y
  RESP_ENABLED_FLAG = Y
  RESP_UPDATE_ALLOWED_FLAG = Y
  USER_ENABLED_FLAG = Y
  USER_UPDATE_ALLOWED_FLAG = Y
  START_DATE_ACTIVE = 01/01/1900
  END_DATE_ACTIVE = ""
  SQL_VALIDATION = "SQL=\"select MEANING
\\\"Calculator:type\\\", LOOKUP_CODE \
into\n   :VISIBLE_OPTION_VALUE, :PROFILE_OPTION_VALUE
from FND_LOOKUPS w\ here\n
LOOKUP_TYPE='CALCULATOR_TYPE'\"
\nCOLUMN=\"\\\"Calculator:type\\\"(*)\""
END PROFILE
BEGIN PROFILE CONC_COPIES
  APPLICATION_SHORT_NAME = FND
  USER_PROFILE_OPTION_NAME = "Concurrent:Report
Copies"
  DESCRIPTION = "Number of copies to print for a
concurrent process"
  USER_CHANGEABLE_FLAG = Y
  USER_VISIBLE_FLAG = Y
  READ_ALLOWED_FLAG = Y
  WRITE_ALLOWED_FLAG = Y
  SITE_ENABLED_FLAG = Y
```

```
  SITE_UPDATE_ALLOWED_FLAG = Y
  APP_ENABLED_FLAG = Y
  APP_UPDATE_ALLOWED_FLAG = Y
  RESP_ENABLED_FLAG = Y
  RESP_UPDATE_ALLOWED_FLAG = Y
  USER_ENABLED_FLAG = Y
  USER_UPDATE_ALLOWED_FLAG = Y
  START_DATE_ACTIVE = 01/01/1900
  END_DATE_ACTIVE = ""
  SQL_VALIDATION = ""
END PROFILE
```

User Profile Value Loader

The User Profile Value Loader (FNDVLOAD) is a concurrent program that can move Oracle Applications user profile value information between database and text file representations. The following sections describe the operation of the User Profile Value Loader.

Supported Operations

The User Profile Value Loader allows you to move profile value data between the database (where it is used for runtime operation) and a text file representation (where it can be used for distribution). Specifically, you can:

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. In other words, the PROFILE_OPTION_NAME, not the PROFILE_OPTION_ID, is used to identify records.

Upload (merge) the information in a text file to the database

If a row is already correct in the database, it is not touched. If a row exists, but has different attributes, the row is updated. If a row does not exist, a new row is inserted. No rows are deleted, even if they are absent from the uploaded file.

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.

Usage

The User Profile Value Loader takes the following arguments:

```
FNDVLOAD <username/password> 0 Y
  <appsname> <filename> UPLOAD | DOWNLOAD
```

where

<username/ password> is the APPLSYS account

<appsname> is an application short name or ”LOCAL”

<filename> is the file that will be read or written

The location of the file is determined by the <appsname> and <filename> arguments. <appsname> indicates whether the file is in the install/import directory for a particular application, or whether it is in the local directory.

Appsname	File	Location
<APP>	<file>	$<APP_TOP>/install/import/<file>.slt
LOCAL	<file>	<file>

Table 11 – 6 (Page 1 of 1)

Examples

FNDVLOAD applsys/fnd 0 Y FND install/import/FND.vlt UPLOAD

references file ”$FND_TOP/install/import/FND.vlt”

FNDVLOAD applsys/fnd 0 Y FND.vlt UPLOAD

references file ”FND.vlt”

UPLOAD and DOWNLOAD

The loader works in either upload or download mode. In UPLOAD mode the entire file is read and merged into the database. In DOWNLOAD mode the loader will download all profile values to the file.

FNDVLOAD applsys/fnd 0 Y FND FND.vlt DOWNLOAD

will download all profile values to the file FND.vlt

FNDVLOAD applsys/fnd 0 Y FND FND.vlt UPLOAD

will upload everything in the file FND.vlt to the database

Profile Value File Format

Profile value information is stored in text file format by writing out a series of ”records” for each profile value. The file is plain text that can be viewed and edited with any standard text editor. You need to understand this file format only if you plan to edit the file.

The file consists of a series of text lines, each of which must be less than 1024 bytes in length (when written, no line will exceed 80 bytes in length).

A line may be a comment line, meaning it has no effect on the data contained in the file. Comment lines begin with a ”#” and end with a newline.

```
# This is a comment line
```

The information in the file consists of a series of tokens delimited by white space. Note that a token may itself contain a space, in which case it is delimited with double–quote marks. Tokens may not contain non–printing characters unless they are preceded by the ”escape” character, which is a backslash. Predefined escapes are:

String	Meaning
\e	Escape
\n	Newline
\r	Carriage return
\b	Backspace
\v	Vertical tab
\f	Form feed
\”	The double quote itself
\\	The backslash itself

Table 11 – 7 (Page 1 of 1)

Other escapes consist of octal values preceded by the backslash, as in ”\007”. If a line is too long to fit within 80 characters, the line is broken into as many ”continuation” lines as necessary. A line ending with a single backslash is taken to be continued on the next line. In such a case, the newline following the backslash is ignored, rather than being taken as a white space delimiter. The line following such a line is taken to be a continuation of the ”broken” line, and such a line can itself end in a backslash, indicating further continuation. Note that in the case of

continuation lines, a leading "#" is not taken as a comment indicator, but rather as part of the continued data item.

The file starts with three lines as follows:

```
LANGUAGE = "AMERICAN"
CODESET = "WE8ISO8859P1"
TRANSLATED = "Y"
```

These lines indicate the NLS_LANG language and codeset of the file and the current translation status of any translated column values.

Then there is a series of object definitions of the format:

```
BEGIN <level type> <level identifier>
 PROFILE <profile name> = <profile value>
 ...
END <level type>
```

There are four types of levels in the file: SITE, APPLICATION, RESPONSIBILITY, and USER. Each level type has some identifier, followed by a series of profile values. If a profile value is not specified on a given level, it will not be written to the file on a DOWNLOAD, and the current setting will not be changed on an UPLOAD.

SITE record

Sites are not identified, since there can be only one SITE per database. Site level information is formatted as follows:

```
BEGIN SITE
 PROFILE <profile name: 80 chars> = <profile value:
 240 chars>
 ...
END SITE
```

Example:

```
BEGIN SITE
 PROFILE ACCOUNT_ISSUE_TXN = 1
 PROFILE AR_ADJUST_CREDIT_UNCONFIRMED_INVOICE = NONE
 PROFILE AR_ALLOW_BATCHING = B
 PROFILE AR_ALLOW_TAX_CODE_OVERRIDE = N
 PROFILE AR_ALLOW_TAX_UPDATE = Y
 PROFILE AR_CHANGE_CUST_NAME = Y
 PROFILE AR_CHANGE_CUST_ON_TRX = Y
 PROFILE AR_CMERGE_SET_SIZE = 1
 PROFILE AR_GLPOST_BALANCE_CHECK_FLAG = Y
 PROFILE AR_ITEM_FLEXFIELD_MODE = M
END SITE
```

APPLICATION record

Applications are identified by APPLICATION_SHORT_NAME. Application information is formatted as follows:

```
BEGIN APPLICATION <applname>
 PROFILE <profile name: 80 chars> = <profile value:
 240 chars>
 ...
END APPLICATION
```

Example:

```
BEGIN APPLICATION BOM
 PROFILE CURRENCY:NEGATIVE_FORMAT = 3
 PROFILE SHARED_MESG_APPL_ID = 700
END APPLICATION
```

RESPONSIBILITY record

Responsibilities are identified by APPLICATION_SHORT_NAME and RESPONSIBILITY_NAME. Responsibility level information is formatted as follows:

```
BEGIN RESPONSIBILITY <applname> <respname>
 PROFILE <profile name: 80 chars> = <profile value:
 240 chars>
 ...
END RESPONSIBILITY
```

Example:

```
BEGIN RESPONSIBILITY AX ”AX General Ledger User”
 PROFILE AX_APPLICATION_ID = 101
 PROFILE AX_APPLICATION_NAME = ”Oracle General
 Ledger”
END RESPONSIBILITY
```

USER record

Users are identified by USER_NAME. User level information is formatted as follows:

```
BEGIN USER <username>
 PROFILE <profile name: 80 chars> = <profile value:
 240 chars
 ...
END USER
```

Example:

```
BEGIN USER DATAMERGE
 PROFILE BOM:UPDATE_RESOURCE_UOM = 2
 PROFILE QUICKPICK:AUTOREDUCTION = Y
 PROFILE SAVE_SEARCH_ITEMS = N
END USER
```

Sample Profile Value File

```
#
# $Header$
#
LANGUAGE = ”AMERICAN”
CODESET = ”US7ASCII”
TRANSLATED = ”Y”#
BEGIN SITE
  PROFILE ACCOUNT_ISSUE_TXN = 1
  PROFILE AR_ADJUST_CREDIT_UNCONFIRMED_INVOICE = NONE
  PROFILE AR_ALLOW_BATCHING = B
  PROFILE AR_ALLOW_TAX_CODE_OVERRIDE = N
```

```
PROFILE AR_ALLOW_TAX_UPDATE = Y
PROFILE AR_CHANGE_CUST_NAME = Y
PROFILE AR_CHANGE_CUST_ON_TRX = Y
PROFILE AR_CMERGE_SET_SIZE = 1
PROFILE AR_GLPOST_BALANCE_CHECK_FLAG = Y
PROFILE AR_ITEM_FLEXFIELD_MODE = M
PROFILE AR_OVERRIDE_ADJUSTMENT_ACTIVITY_ACCOUNT = Y
PROFILE AR_PA_CODE = 1
PROFILE AS_MANAGING_EMPLOYEE_HAS_ACCESS_FLAG = Y
PROFILE AS_NOTES_UPDATE_DAY_RANGE = 1
PROFILE AS_TERR_MIN_NUM_PARALLEL_PROC = 100
PROFILE AS_TERR_NUM_CHILD_PROCESSES = 5
PROFILE ATCHMT_SET_INDICATOR = Y
PROFILE AUDITTRAIL:ACTIVATE = N
PROFILE AUTOCOMMIT = N
PROFILE BASE_LANGUAGE = 0
PROFILE BOM:CHECK_DUPL_CONFIG = 2
PROFILE BOM:CONFIG_INHERIT_OP_SEQ = 2
PROFILE BOM:CONFIG_ITEM_DELIMITER = *
PROFILE BOM:CONFIG_ITEM_TYPE = I
PROFILE BOM:DEFAULT_BOM_LEVELS = 1
PROFILE BOM:ITEM_SEQUENCE_INCREMENT = 10
PROFILE BOM:MODEL_ITEM_ACCESS = 1
PROFILE BOM:PLANNING_ITEM_ACCESS = 1
PROFILE BOM:STANDARD_ITEM_ACCESS = 1
PROFILE BOM:UPDATE_RESOURCE_UOM = 2
PROFILE BUDGETARY_CONTROL_OPTION = 0
PROFILE CONC_COPIES = 0
PROFILE CONC_HOLD = N
PROFILE CONC_PMON_METHOD = RDBMS
PROFILE CONC_PRIORITY = 50
PROFILE CONC_REPORT_ACCESS_LEVEL = U
PROFILE CONC_SAVE_OUTPUT = Y
PROFILE CONC_SINGLE_THREAD = N
PROFILE CST_EXCHANGE_RATE_TYPE = 2
PROFILE CST_RU_WAIT_FOR_LOCKS = 2
PROFILE CST_UPDATE_COSTS = 1
PROFILE CST_UPDATE_DEBUG = 1
PROFILE CST_VIEW_COSTS = 1
```

```
  PROFILE CURRENCY:MIXED_PRECISION = 2
  PROFILE CURRENCY:NEGATIVE_FORMAT = 0
  PROFILE CURRENCY:POSITIVE_FORMAT = 0
  PROFILE CURRENCY:THOUSANDS_SEPARATOR = Y
  PROFILE CURRENCY_CONVERSION_TYPE = Corporate
  PROFILE CYCLE_COUNT_APPROVALS_TXN = 1
  PROFILE CYCLE_COUNT_ENTRIES_TXN = 1
  PROFILE DIAGNOSTICS = N
  PROFILE DISPLAY_INVERSE_RATE = N
  PROFILE ENG:ENG_ITEM_ECN_ACCESS = 1
  PROFILE ENG:MANDATORY_ECO_DEPT = 2
  PROFILE ENG:MODEL_ITEM_ECN_ACCESS = 1
  PROFILE ENG:PLANNING_ITEM_ECN_ACCESS = 1
  PROFILE ENG:STANDARD_ITEM_ECN_ACCESS = 1
  PROFILE ENG_SYSTEM_AUTONUMBERING = 2
END SITE
BEGIN APPLICATION BOM
  PROFILE CURRENCY:NEGATIVE_FORMAT = 3
  PROFILE SHARED_MESG_APPL_ID = 700
END APPLICATION
BEGIN APPLICATION CRP
  PROFILE SHARED_MESG_APPL_ID = 700
END APPLICATION
BEGIN APPLICATION ENG
  PROFILE CURRENCY:NEGATIVE_FORMAT = 3
  PROFILE SHARED_MESG_APPL_ID = 700
END APPLICATION
BEGIN APPLICATION INV
  PROFILE INV_STATUS_DEFAULT = Active
  PROFILE INV_UOM_DEFAULT = Each
END APPLICATION
BEGIN APPLICATION MRP
  PROFILE CURRENCY:NEGATIVE_FORMAT = 3
  PROFILE SHARED_MESG_APPL_ID = 700
END APPLICATION
BEGIN APPLICATION PA
  PROFILE PA_NUM_CDL_PER_SET = 2000
  PROFILE PA_NUM_EXPENDITURES_PER_SET = 500
  PROFILE PA_NUM_EXP_ITEMS_PER_SET = 1000
```

```
  PROFILE PA_RULE_BASED_OPTIMIZER = N
  PROFILE PA_STRMLN_SLEEP_INTERVAL = 60
END APPLICATION
BEGIN APPLICATION PAY
  PROFILE DATETRACK:DELETE_MODE = PROMPT
  PROFILE DATETRACK:ENABLED = Y
  PROFILE DATETRACK:OVERRIDE_MODE = PROMPT
  PROFILE DATETRACK:UPDATE_MODE = PROMPT
  PROFILE HR_USER_TYPE = INT
  PROFILE PER_BUSINESS_GROUP_ID = 0
END APPLICATION
BEGIN APPLICATION QA
  PROFILE QA_ACTION_PROCESSING_MODE = 2
  PROFILE QA_BLIND_ENTRY = 2
END APPLICATION
BEGIN RESPONSIBILITY AX "AX General Ledger User"
  PROFILE AX_APPLICATION_ID = 101
  PROFILE AX_APPLICATION_NAME = "Oracle General
  Ledger"
END RESPONSIBILITY
BEGIN RESPONSIBILITY AX "AX Payables Supervisor"
  PROFILE AX_APPLICATION_NAME = "Oracle Payables"
END RESPONSIBILITY
BEGIN RESPONSIBILITY AX "AX Payables User"
  PROFILE AX_APPLICATION_NAME = "Oracle Payables"
END RESPONSIBILITY
BEGIN RESPONSIBILITY AX "AX Receivables Supervisor"
  PROFILE AX_APPLICATION_NAME = "Oracle Receivables"
END RESPONSIBILITY
BEGIN RESPONSIBILITY AX "AX Receivables User"
  PROFILE AX_APPLICATION_NAME = "Oracle Receivables"
END RESPONSIBILITY
BEGIN RESPONSIBILITY CN "Sales Compensation Super
User"
  PROFILE CN_ALLOW_CREATE_PAYRUNS = Y
END RESPONSIBILITY
BEGIN RESPONSIBILITY PA "PB Accounting Supervisor"
  PROFILE PA_SUPER_PROJECT = Y
END RESPONSIBILITY
```

```
BEGIN RESPONSIBILITY PA ”PB Administrator”
  PROFILE PA_SUPER_PROJECT = Y
END RESPONSIBILITY
BEGIN RESPONSIBILITY PA ”PC Accounting Supervisor”
  PROFILE PA_SUPER_PROJECT = Y
END RESPONSIBILITY
BEGIN RESPONSIBILITY PA ”PC Administrator”
  PROFILE PA_SUPER_PROJECT = Y
END RESPONSIBILITY
BEGIN RESPONSIBILITY PAY ”Oracle Human Resources with
Payroll”
  PROFILE HR_USER_TYPE = INT
END RESPONSIBILITY
BEGIN RESPONSIBILITY PER ”Oracle Human Resources”
  PROFILE HR_USER_TYPE = PER
END RESPONSIBILITY
BEGIN RESPONSIBILITY QA Quality
  PROFILE QA_BLIND_ENTRY = 2
END RESPONSIBILITY
BEGIN RESPONSIBILITY SQLGL ”General Ledger Budget
User”
  PROFILE CONC_SINGLE_THREAD = N
END RESPONSIBILITY
BEGIN RESPONSIBILITY SQLGL ”General Ledger
Supervisor”
  PROFILE CONC_SINGLE_THREAD = N
END RESPONSIBILITY
BEGIN USER DATAMERGE
  PROFILE BOM:UPDATE_RESOURCE_UOM = 2
  PROFILE QUICKPICK:AUTOREDUCTION = Y
  PROFILE SAVE_SEARCH_ITEMS = N
END USER
BEGIN USER SYSADMIN
  PROFILE SHARED_MESG_APPL_ID = 700
END USER
```

APPENDIX

C

Using Predefined Alerts

This chapter gives you an overview of Oracle Alert and how to use the predefined alerts that are packaged with your Oracle Applications product. Specifically, this chapter describes:

- The business needs for Oracle Alert
- How to run predefined alerts
- How to modify the predefined alerts to fit your needs

Attention: This chapter focuses on using and customizing the *predefined alerts* that are packaged with Oracle Applications products. If you have a license of the full Oracle Alert product, you should refer to the *Oracle Alert Reference Manual* for information on how to create and customize your own alerts.

Overview of Oracle Alert

Oracle Alert is your complete exception control solution.

Oracle Alert gives you an immediate view of the critical activity in your database. It helps you keep on top of important or unusual business events you need to know about, as they happen. Oracle Alert gives you real–time measurements of staff and organization performance, so you can zero in on potential trouble spots immediately. You can automate routine transactions with Oracle Alert, saving your valuable time for more essential tasks. And, Oracle Alert does all this online, so you do not have to contend with a pile of paperwork.

Oracle Alert gives you the flexibility you need to monitor your business information the way you want.

Basic Business Needs

Oracle Alert meets the following basic business needs:

- Informs you of exception conditions as they occur
- Lets you specify the exception conditions you want to know about, as often as you want to know about them
- Informs you of exception conditions by sending alert messages through a single application—your electronic mail
- Takes actions you specify, based upon your response to an alert message
- Automatically performs routine database tasks, according to a schedule you define
- Integrates fully with your electronic mail system

Oracle Alert Runtime Features

If you do not have a licensed copy of the full Oracle Alert product, you may still derive benefit from major Oracle Alert features by using the predefined alerts that are packaged with your Oracle Applications product.

All Oracle Applications products are packaged with a runtime version of Oracle Alert. Although all the Oracle Alert windows are available in this runtime version, not all the features in those windows are enabled. With the runtime version of Oracle Alert, you can run only the

predefined alerts that are packaged with your product; you cannot create new alerts.

Alert Definitions

Alert

A mechanism that checks your database for a specific exception condition. An alert is characterized by the *SQL SELECT statement* it contains. A SQL SELECT statement tells your application what database exception to identify, as well as what output to produce for that exception.

For example, you can define an alert to flag purchase orders exceeding $10,000, and have that alert output the name of the individual who requested the purchase order, as well as the name of the individual's manager. All predefined alerts are listed in the Alerts window of Oracle Alert.

Event Alert

An event alert monitors the occurrence of a specific exception or change in your database. An exception in your database results if you add or update information using your Oracle Applications windows. The event alert monitors the database for exceptions based on its SQL SELECT statement.

Periodic Alert

A periodic alert periodically reports key information according to a schedule that you define. Rather than notify you of immediate exceptions in the database like an event alert, a periodic alert scans for specific database information specified by its SQL SELECT statement at scheduled intervals.

Alert Action

An alert action is an action you want your alert to perform. An alert action can be dependent on the output from the alert. An alert action can fall under one of three categories:

- Detail action—an action that represents one exception found in the database
- Summary action—an action that represents multiple exceptions found in the database

- No exception action—an action that represents no exceptions found in the database

An action can include sending an electronic mail message to a mail ID, running an Oracle Applications program, running a program or script from your operating system, or running a SQL script to modify information in your database.

You can have more than one action for an alert and an action can incorporate the output of the alert. For example, you may want a particular alert to send a message to a manager, as well as run an Oracle Applications program when an exception occurs.

Action Sets

An action set is a sequence of alert actions that are enabled for a particular alert. Each action that you include in an action set can be assigned a sequence number so that you can specify the order in which the actions are performed. Some predefined alerts may also have more than one action set. You can also assign a sequence number to each action set to specify the order in which each action set is performed.

Predefined Alerts

There are two types of predefined alerts:

- **Event alerts**—for example, the Receiving Notification alert for Oracle Purchasing notifies the requestor with a mail message when an item is received and entered in the Receipts window.
- **Periodic alerts**—for example, the Forecast Over–Consumption alert for Oracle Material Planning checks every day for over–consumption of the forecast and sends you a mail message if the current forecast quantity listed in the Forecast Entries window goes below zero.

Suggestion: See your product's reference guide for a list of the predefined alerts that are packaged with your Oracle Applications product.

Using Predefined Alerts

All predefined alerts are initially disabled. You must enable the alerts you want to use. Select the Oracle Alert Manager responsibility when you start Oracle Applications to view or use a predefined alert. The Alert Manager responsibility gives you access to the Oracle Alert menu.

Navigate to the Alerts window to enable or edit predefined alerts. To display the predefined alert(s) for your Oracle Applications product, execute a query with your Oracle Applications product name in the Application field.

The Name field displays the name of the predefined alert. The Type field indicates if the alert is an event or a periodic alert.

You can enable an alert to run by checking the Enabled check box. You can also enter an End Date to specify the date until you want this alert run.

Choose the Alert Details button to open the Alert Details window. Choose the Alert Installations alternative region to display the available Installations.

Enter the Oracle ID of the application installation you want your alert to run against. You can select only the Oracle IDs that are associated with the application that owns your alert. You can disable an Oracle ID for the alert temporarily by unchecking the Enabled check box.

Choose the Actions button to open the Actions window. Oracle Alert automatically displays the actions that are defined for the alert.

In the Actions window, if the Action Type is Detail, choose the Action Details button to display details for that action.

The alert action sends an alert action message to the mail ID listed in the To field of the Message Detail zone. If the mail ID is in the format ***&NAME***, where ***Name*** is an output defined by your alert, you need not modify this field. If, however, the mail ID in the To field is not in the above format or if there is no value entered in the field, you must enter the mail ID(s) of the person(s) you wish to receive the alert action message. After modifying the contents of this window, save your work.

Navigate to the Oracle Alert Options window. Use this window to specify the electronic mail application you wish to integrate with the predefined alerts.

In the Alerts window, choose the Actions Sets button to navigate to the Action Sets window. Oracle Alert automatically displays the action sets defined for the alert.

Check the Enabled check box for each action set you wish to use. You may also enter an End Date field to specify the date until you want this alert action set to be enabled.

In addition, in the Action Set Members block, check the Enabled check box for each action set member you want to use in that action set.

You may also enter an End Date to specify the date until you want this alert action set member to be enabled. When you finish, save your work.

Your predefined alert is now ready to use.

Customizing Predefined Alerts

You can customize predefined alerts in the following ways to suit your business needs:

Electronic Mail Integration

Oracle Alert is fully integrated with Oracle Office and can use Oracle Office to send electronic mail messages to your users. Since Oracle Office has gateways to other electronic mail systems, Oracle Alert can send messages to users on those systems as well. Oracle Alert can also use UNIX mail, VMS mail, or a custom mail system to send electronic mail messages to your users.

You open the Oracle Alert Options window and use the Mail Systems alternative region to specify the electronic mail application you wish to

use with your predefined alerts. You enter the Name of your electronic mail application, the operating system Command you use to start the mail application, and any Parameters you wish to pass to the mail program.

If you are using Oracle Office, you need not specify an operating system Command. Once you enter the information for your mail application, check the In Use check box, then save your work. You can have only one mail application enabled at any given time.

Standard Alert Message Text

You can customize the message header and footer text that appears in all your alert message actions. Navigate to the Message Elements alternative region of the Oracle Alert Options window, and four message elements appear automatically. Each element represents a specific type of message text that appears in all your alert mail messages.

In the runtime version of Oracle Alert, you need to edit only the Message Action Header and Message Action Footer elements. Simply customize the text that appears to alter the text at the beginning and end of every alert message. You may also leave the text blank if you do not want to display any standard text in your alert messages. Save your work when you are done making changes in this window.

Alert Frequency

You can schedule the frequency you wish to run each predefined periodic alert. You may want to check some alerts every day, some only once a month, still others only when you explicitly request them. You have the flexibility to monitor critical exceptions every day, or even multiple times during a 24–hour period. And, you can set less significant exceptions to a more infrequent schedule; for example, a monthly schedule.

To change the frequency of a predefined alert, navigate to the Alerts window. Perform a query to display the predefined periodic alert you wish to modify, then alter the Frequency of the periodic alert.

Alert History

Oracle Alert can keep a history of exceptions and actions for a particular alert. Use the Alerts window to alter the number of days of history you wish to keep for an alert. Simply change the Keep N Days field to the number of days of history you wish to keep.

Suppressing Duplicates

If you do not want Oracle Alert to send repeated messages for the same alert exception, you can choose to suppress duplicate messages. If Oracle Alert finds a duplicate exception condition for the alert, it simply does not execute the action set members for that alert again.

Use the Suppress Duplicates check box in the Action Sets block of the Alerts window to specify this option. The default for the Suppress Duplicates check box is unchecked. If you check the Suppress Duplicates check box, you must also make sure you keep history for the alert at least one day longer than the number of days between alert checks. Oracle Alert uses the history information to determine if an exception is a duplicate.

Message Actions

If a predefined alert involves a message action, you can customize certain aspects of that message action. Navigate to the Actions block in the Alerts window by choosing the Actions button. In this block, move your cursor to the row representing the message action you want to customize, then choose the Action Details button to open the Action Detail window for that message action. You can modify the following features of the message action:

- Recipient list—you can add or delete mail IDs in the List, To, Cc, Bcc, or Print For User fields. You should not modify any mail IDs listed with the format ***&Name***, as they represent mail ID's defined by the alert output.
- Printer—you can modify the name of the printer to which you want Oracle Alert to direct the message.
- Text—you can modify the boilerplate text that you want your alert message to send. Do not edit any of the alert outputs (in the format ***&Name***) used in the body of the text. For summary messages, edit only the opening and closing text within the summary message. Save your work when you finish making modifications.

Summary Threshold

Predefined alerts use one of three action types: detail action, summary action, and no exception action. A no exception action is straightforward in that Oracle Alert performs the defined action when no exceptions are found for the alert.

But how does Oracle Alert know when to perform a detail or a summary action? Oracle Alert can perform a detail action for every

exception it finds, regardless of the number of exceptions, or Oracle Alert can perform a summary action for a unique set of exceptions. For example, you can receive individual mail messages for each exception found by an alert, or you can receive a single mail message summarizing all the exceptions found by the alert.

In the Members alternative region of the Action Sets block of the Alerts window, you can set a Summary Threshold to specify how many exceptions Oracle Alert can find before it should change the action from a detail action to a summary action.

Oracle Alert Precoded Alerts

Your Oracle Alert installation contains custom alerts that are designed to help you manage your database and the data you generate when you use Oracle Alert. Oracle Alert provides eight alerts that systematically monitor your system for potential tablespace, disk space, and allocation problems, making your Database Administrators more efficient, and increasing database performance.

Occasionally, you will want to purge your database of obsolete concurrent requests, alert checks, and action set checks. Oracle Alert provides two alerts that let you periodically remove old files, freeing up valuable tablespace and increasing database performance. Oracle Alert also provides an alert that clears your Oracle Alert electronic mail folders of older messages, keeping your send mail and response mail accounts to a manageable size.

This section gives you an overview of these eleven alerts, and suggestions on how to use them to enhance your system performance.

Terms

Before reading this discussion of precoded alerts, you may want to familiarize yourself with the following Glossary terms:

- Periodic Alert
- Exception
- Action
- Detail Action
- Summary Action
- No Exception Action
- Input

Oracle Alert DBA Alerts

Oracle Alert DBA alerts help you manage your database by notifying you regularly of:

- Tables and indexes unable to allocate another extent
- Users who are nearing their tablespace quota

- Tablespaces without adequate free space
- Tables and indexes that are too large or are fragmented
- Tables and indexes that are near their maximum extents

Customizable Alert Frequencies

Oracle Alert DBA alerts are periodic alerts, so you determine how often they check your database. Set them to run daily, weekly, or monthly, according to your database needs.

Summary and No Exception Messages

If Oracle Alert finds the database exceptions specified in a DBA alert, it sends you a message summarizing all exceptions found. If Oracle Alert finds no exceptions, it sends you a message reporting that no exceptions were found. Oracle Alert keeps you notified of the status of your database, even if it is unchanging.

Customizable Alert Inputs

Inputs let you customize your DBA alerts. You can specify the ORACLE username, table, or index you want your alerts to target, and you can specify the threshold number of extents, maximum extents, or blocks Oracle Alert should look for. You can also define your input values at the action set level, so you can create multiple action sets that target different usernames, tables, and indexes. You can create as many action sets as you need.

Support for Multiple Database Instances

The Applications DBA application owns the Oracle Alert DBA alerts. This lets Oracle Alert perform the DBA alerts for every database instance you create, even those that reside outside Oracle Alert's database.

Applications DBA Alerts Descriptions

The following descriptions list the customizable frequency and inputs of each DBA alert.

Tables Unable to Allocate Another Extent

This alert looks for tables where the next extent is larger than the largest free extent.

Frequency Every N Calendar Days

Inputs Table Name, ORACLE Username

Indexes Unable to Allocate Another Extent

This alert looks for indexes where the next extent is larger than the largest free extent.

Frequency Every N Calendar Days

Inputs Index Name, ORACLE Username

Users Near Their Tablespace Quota

This alert detects users that are near their tablespace quota.

Frequency Every N Calendar Days

Inputs ORACLE Username

Tablespace Name

Check minimum percent free space remaining

Check maximum percent space use

Minimum total free space remaining (in bytes)

Maximum percent space used

Tablespaces Without Adequate Free Space

This alert looks for tablespaces without a specified minimum amount of free space.

Frequency Every N Calendar Days

Inputs Tablespace Name

Check total free space remaining

Check maximum size of free extents available

Maximum size of free extents available (in bytes)

Minimum total free space remaining (in bytes)

Indexes Too Large or Fragmented

This alert detects indexes that exceed a specified number of blocks or extents.

Frequency Every N Calendar Days

Inputs Index Name

ORACLE Username

Check maximum number of blocks

Check maximum number of extents

Maximum number of blocks

Maximum number of extents

Tables Too Large or Fragmented

This alert detects tables that exceed a specified number of blocks or extents.

Frequency Every N Calendar Days

Inputs Table Name

ORACLE Username

Check maximum number of blocks

Check maximum number of extents

Maximum number of blocks

Maximum number of extents

Tables Near Maximum Extents

This alert searches for tables and indexes that are within a specified number of extents of their maximum extents.

Frequency Every N Calendar Days

Inputs Table Name

ORACLE Username

Minimum number of extents remaining

Indexes Near Maximum Extents

This alert searches for tables and indexes that are within a specified number of extents of their maximum extents.

Frequency Every N Calendar Days

Inputs Index Name

ORACLE Username

Minimum number of extents remaining

Oracle Alert Purging Alerts

Two of the Oracle Alert precoded alerts are designed to help you manage the data you generate when you use Oracle Alert. While using Oracle Alert you should be able to:

- Automatically delete concurrent requests older than a specified number of days
- Automatically clean out alert checks and action set checks that are older than a specified number of days

Customizable Alert Frequencies

You determine the schedule for running your purge alerts. On the schedule you define, Oracle Alert submits the purge alerts to the Concurrent Manager, and deletes all old concurrent requests.

Customizable Alert Inputs

Inputs let you customize your alerts. You specify which application and which concurrent program you want your purge alerts to target, and you decide when your data becomes unnecessary or "old." You define your input values at the action set level, so you can create multiple action sets that target different applications and different concurrent programs. You can create as many action sets as you need, so you can keep your system free from unnecessary files.

Oracle Alert Purging Alerts Descriptions

The following descriptions list the customizable frequency and inputs of each purging alert.

Purge Alert and Action Set Checks

This alert looks for alert and action set checks older than the number of days you specify, and runs a SQL statement script that deletes them.

Alert Type Periodic

Periodicity Every N Calendar Days

Inputs Application Name, Number of days since alert check

> **Note:** Oracle Alert will not delete alert checks and / or action set checks for a response processing alert that has open responses.

Purge Concurrent Requests

This alert looks for concurrent requests and their log and out files that are older than the number of days you specify, and runs a concurrent program that deletes them. If you enter a concurrent program name input, you should use the program name (located in the column USER_CONCURRENT_PROGRAM_NAME in the table FND_CONCURRENT_REQUESTS), and not the optional description that may accompany the concurrent program name in the Requests window.

Alert Type	Periodic
Periodicity	Every N Calendar Days
Inputs	Application Name
	Concurrent Program Name
	Number of days since concurrent request was submitted to the Concurrent Manager
Operating System Program	Deletes log file, out file, and corresponding record of each concurrent request
Arguments	Concurrent request ID

Oracle Alert Purge Mail Alert

One of the Oracle Alert precoded alerts is designed to help you keep your Oracle Office folders to a manageable size. In particular, if you are using response processing, you will want to keep your response account(s) clear of old messages. While using Oracle Alert you should be able to:

- Automatically delete old, obsolete mail messages from your defined Oracle Alert Oracle Office accounts
- Specify which Oracle Office accounts and the Oracle Office folders you want to clear of old messages
- Determine which messages you want to delete

Customizable Alert Frequencies

You determine the schedule for running your alert. On the schedule you define, Oracle Alert submits the purge mail alert to the Concurrent Manager.

Customizable Alert Inputs

Use inputs to tell Oracle Alert which Oracle Office account, which mail folders, and which messages to purge. You define your input values at

the action set level, so you can create multiple action sets that target different mail accounts and different mail folders. You can create as many action sets as you need to keep your mail accounts up–to–date.

Oracle Alert Purge Mail Alert Description

The following description provides the customizable frequency and inputs of the purge mail alert.

Purge Oracle Office Messages

Frequency	Weekly
Inputs	Expiration Days
	Folder
	Oracle Office Account

APPENDIX

D

Menu Appendix

This appendix lists the forms available in the System Administration responsibility along with their navigation paths.

System Administration Menu Paths

This section shows you the first window title and default menu path for each Oracle Applications System Administration form. In addition, we provide a page number reference for the description of each form in this manual, or a reference for the descriptions of forms that are located in other manuals.

See...	Refer to this manual for a complete form description
Flex	*Oracle Applications Flexfields Manual*
User	*Oracle Applications User's Guide*
GL	*Oracle Applications General Ledger User's Guide*

First Window Title	Standard Menu Path
Account Generator Processes: page 7 – 72	Application Flexfield Key Accounts
Administer Concurrent Managers: page 7 – 72	Concurrent Manager Administer
Applications: page 9 – 10	Application Register
Assign Security Rules: ***flex***:	Security Responsibility ValueSet Assign
Audit Groups: page 3 – 35	Security AuditTrail Groups
Audit Installations: page 3 – 34	Security AuditTrail Installations
Audit Tables: page 3 – 38	Security AuditTrail Tables
Combined Specialization Rules: page 7 – 88	Concurrent Manager Rule
Concurrent Conflicts Domains: page 9 – 9	Concurrent Conflicts Domains
Concurrent Managers: page 7 – 80	Concurrent Manager Define
Concurrent Program Executable: page 6 – 48	Concurrent Program Executable
Concurrent Programs: page 6 – 51	Concurrent Program Define
Concurrent Request Types: page 7 – 91	Concurrent Program Types
Cross–Validation Rules: ***flex***	Application Flexfield Key CrossValidation
Currencies: ***GL***	Application Currency
Data Groups: page 6 – 63	Security ORACLE DataGroup
Define Security Rules: ***flex***	Security Responsibility ValueSet Define
Descriptive Flexfield Segments: ***flex***	Application Flexfield Descriptive Segments

Document Categories: page 10 – 12	Application Document Categories
Document Sequences: page 10 – 9	Application Document Define
Find Default Folders: page 9 – 15	Application Administer Folders
Find Requests: ***user***	Concurrent Requests, Requests View
Form Functions: page 2 – 33	Application Function
Key Flexfield Segments: ***flex***	Application Flexfield Key Segments
Languages: page 9 – 18	Install Languages
Menus: page 2 – 36	Application Menu
Monitor Users: page 3 – 9	Security User Monitor
Network Test: page 9 – 13	Application Network Test
Nodes: page 7 – 92	Install Nodes
Notifications Summary: page 7 – 92	Workflow Notifications
ORACLE Users: page 9 – 5	Security ORACLE Register
Personal Profile Values: ***user***	Profile Personal
Print Styles: page 8 – 36	Install Printer Style
Printer Drivers: page 8 – 38	Install Printer Driver
Printer Types: page 8 – 33	Install Printer Types
Printers: page 8 – 35	Install Printer Register
Request Groups: page 6 – 46	Security Responsibility Request
Request Set: ***user***	Concurrent Set, Requests Set
Responsibilities: page 2 – 9	Security Responsibility Define
Rollup Groups: ***flex***	Application Flexfield Key Groups
Segment Values: ***flex***	Application Validation Values, Application Flexfield Key Values, and Application Flexfield Descriptive Values
Sequence Assignments: page 10 – 14	Application Document Assign
Shorthand Aliases: page ***flex***	Application Flexfield Key Aliases
Submit a New Request: ***user***	Requests Run
System Profile Values: page 5 – 6	Profile System
Territories: page 9 – 19	Install Territories
Users: page 2 – 14	Security User Define

Value Sets: ***flex***	Application Validation Set
Work Item: page 7 – 86	Workflow Status
Work Shifts: page 7 – 86	Concurrent Manager WorkShifts

APPENDIX

E

Implementation Appendix

Setting Up Oracle Applications System Administrator

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

Oracle Applications Implementation Wizard

If you are implementing more than one Oracle Applications product, you may want to use the Oracle Applications Implementation Wizard to coordinate your setup activities. The Implementation Wizard guides you through the setup steps for the applications you have installed, suggesting a logical sequence that satisfies cross–product implementation dependencies and reduces redundant setup steps. The Wizard also identifies steps that can be completed independently—by several teams working in parallel—to help you manage your implementation process most efficiently.

You can use the Implementation Wizard as a resource center to see a graphical overview of setup steps, read online help for a setup activity, and open the appropriate setup window. You can also document your implementation, for further reference and review, by using the Wizard to record comments for each step.

Setup Checklist

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

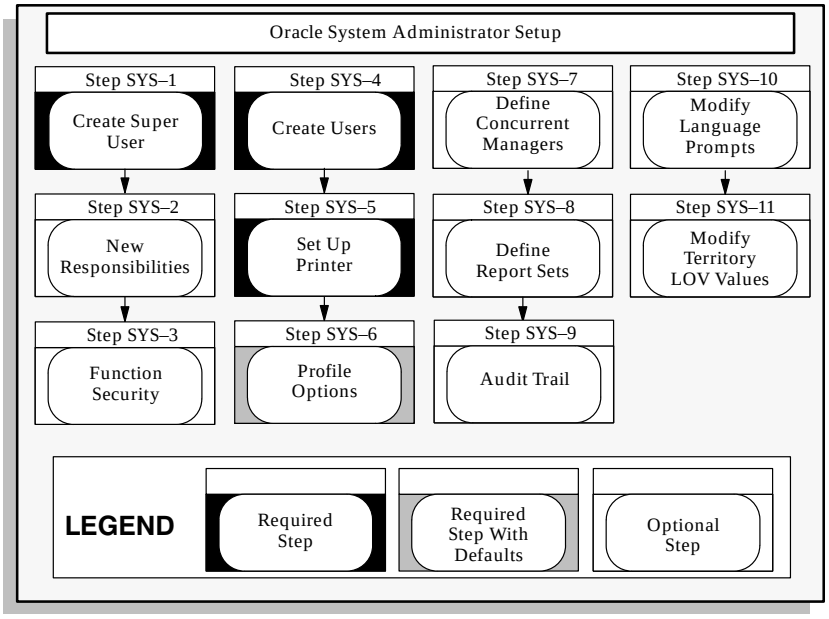
- ❑ Step 1: Create an Oracle Applications User to Complete Setting Up (Required)
- ❑ Step 2: Create New Responsibilities (Optional)
- ❑ Step 3: Implement Function Security (Optional)
- ❑ Step 4: Create Additional Users (Required)
- ❑ Step 5: Set Up Your Printers (Required)
- ❑ Step 6: Specify Your Site–level and Application–level Profile Options (Required with Defaults)
- ❑ Step 7: Define Your Concurrent Managers (Optional)
- ❑ Step 8: Define Report Sets (Optional)
- ❑ Step 9: Set Up AuditTrail (Optional)

- ❑ Step 10: Modify Language Prompts (Optional)
- ❑ Step 11: Modify Territory LOV Values (Optional)

Setup Flowchart

While you can set up Oracle System Administration in many different ways, and defer optional set up steps until you are ready to use the corresponding functionality, we recommend you use the order suggested in the following flowchart:

Figure 11 – 1 Oracle System Administrator Setup



Setup Steps

Step 1 Create an Oracle Applications User to Complete Setting Up

You must create an Oracle Applications user to complete the setup of your applications. You can create one user to set up all of your applications, or you can define one for each product or group of products.

To log on to Oracle Applications, double–click on the Oracle Applications icon.

The sign–on window appears. Enter user name SYSADMIN and password SYSADMIN to access the system administrator responsibility and choose Connect.

Attention: In some cases, a consultant installing your application may have changed the command, username and / or password to something appropriate to your organization. If so, please refer to your consultant for the correct logon instructions.

To define your user, navigate to the Users window by choosing Security > User > Define from the navigation list. Enter a username (for example, INSTALL) in the User Name field and choose your own password. Oracle Applications asks you to change the password the first time you sign on as this new user. You should enter a description to remind yourself that this user is for the setting up of Oracle Applications.

Set the end date to the current date for each of the users you create in this step. This ensures that no one can access your applications with these usernames later.

Assign responsibilities to your new user. You should assign to your new user the full–function responsibility for each of the applications you install.

See Also

Users Window: page 2 – 14

Step 2 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.

You associate each responsibility with a data group, request group, and a menu. The data group defines the pairing of application and ORACLE username. The ORACLE username determines the database tables and table privileges accessible by your responsibility. The request group permits the user with this responsibility to run requests, request sets, or concurrent programs from the Submit Requests form.

Select a predefined menu. A menu provides access to application functions through a hierarchical arrangement of functions and menus of functions

Use the Responsibilities window to define a new responsibility. You can then assign your new responsibility to a user using the Users window.

See Also

Responsibilities Window: page 2 – 9

Users Window: page 2 – 14

Step 3 Implement Function Security (Optional)

Function security is the mechanism by which user access to applications functionality is controlled.

Use the Responsibilities form to limit a responsibility's functionality by excluding menus and functions.

Or

Use the Menus form to create new menus that point to functions you want to make available to a responsibility.

See Also

Responsibilities Window: page 2 – 9

Menus Window: page 2 – 36

Step 4 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 Also

Users Window: page 2 – 14

Step 5 Set Up Your Printers

Read the Setting Up Your Printers 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.

See Also

Defining Printer Types and Registering Printers: page 8 – 11

Print Styles: page 8 – 11

Printer Drivers: page 8 – 12

Overview of Printers and Printing: page 8 – 2

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

Step 6 Specify Your Site–level and Application–level Profile Options

Navigate to the System Profile Values form (Profile > System). In the Find window check Site and Application as your Display levels. Enter System Administration for the Application field and enter Site Name in the Profile field. Oracle Applications displays 'Not Specified' as the site name. Change **Not Specified** to your site name.

To specify the remaining options, return to the find window, clear the Profile field and choose Find.

You should also examine the values set by AutoInstall for the other profile options and determine which ones you want to change. The site–level profile options serve as the defaults for your system until you override them at other levels.

A description for each of the System Administration profile options is available in the Common User Profile Options: page A – 2

See Also

Overview of User Profiles: page 5 – 2

Setting User Profile Options: page 5 – 2

System Profile Values: page 5 – 6

Step 7 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.

See Also

Memory Requirements *(Oracle Applications Installation Manual)*

Preparing to Install or Upgrade *(Oracle Applications Installation Manual)*

Defining Managers and their Work Shifts: page 7 – 23

Overview of Concurrent Processing: page 7 – 2

Step 8 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.

Your users can also define their own report sets.

See Also

Define Request Sets *(Oracle Applications User's Guide)*

Step 9 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.

See Also

Define Audit Groups: page 3 – 35

Define Audit Installations: page 3 – 34

Overview of User and Data Auditing: page 3 – 2

Step 10 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.

See Also

Languages window: page 9 – 18

Step 11 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.

See Also

Territories window: page 9 – 19

APPENDIX

F

Character Mode to GUI Appendix

Oracle Applications System Administrator's Character Mode Forms and Corresponding GUI Windows

This table shows you System Adiministration character mode forms and the windows or processes that have the same functionality in the GUI product.

Most windows are accessible when you use the System Admistrator responsibility. All Navigation paths below assume you are using that responsibility.

Unless otherwise noted, refer to the Oracle Applications System Administrator's Guide Release 11 for more information on GUI windows or processes.

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Administer Concurrent Managers \ Navigate Concurrent Manager Administer	Administer Concurrent Managers window See: Administer Concurrent Managers Navigator: Concurrent > Manager > Administer
Administer Request Sets \ Navigate Concurrent Sets \ Navigate Report Sets	Request Set window See: Request Set *(Oralce Applications User's Guide)* Navigator: Concurrent > Set OR Navigator: Reports > Set
Assign Descriptive Flexfield Security Rules \ Navigate Security Responsibility Flexfield Descriptive Assign	Assign Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Assign In the Find window choose: Descriptive Flexfield
Assign Function Parameters \ Navigate Application Flexfield FlexBuilder Assign	The FlexBuilder feature is replaced by the Account Generator feature using Oracle Workflow See: Account Generator *(Oracle Applications Flexfield Guide)*
Assign Key Flexfield Security Rules \ Navigate Security Responsibility Flexfield Key Assign	Assign Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Assign In the Find window choose: Key Flexfield

Table 11 – 8 (Page 1 of 7)

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Assign Parameter Security Rules \ Navigate Security Responsibility Report Rules Assign	Assign Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Assign In the Find window choose: Concurrent Program
Assign Printer Drivers \ Navigate Install Printer Driver Assign	Printer Drivers window See: Printer Drivers Navigator: Install > Printer > Driver
Assign Security Rules \ Navigate Security Responsibility ValueSet Assign	Assign Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Assign
Assign Terminal Security \ Navigate Security Responsibility Terminal Assign	Obsolete in GUI
Define Application User \ Navigate Security User Define	Users window See: Users Navigator: Security > User > Define
Define Combined Specialization Rules \ Navigate Concurrent Manager Rule	Combined Specialization Rules window See: Combined Specialization Rules Navigator: Concurrent > Manager > Rule
Define Concurrent Manager \ Navigate Concurrent Manager Define	Concurrent Managers window See: Concurrent Managers Navigator: Concurrent > Manager > Define
Define Concurrent Program \ Navigate Concurrent Program Define	Concurrent Programs window See: Concurrent Programs Navigator: Concurrent > Program > Define
Define Concurrent Program Executable \ Navigate Concurrent Program Executable	Concurrent Program Executable window See: Concurrent Program Executable Navigator: Concurrent > Program > Executable
Define Concurrent Request Types \ Navigate Concurrent Program Types	Concurrent Request Types window See: Concurrent Request Types Navigator: Concurrent > Program > Types

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Define Cross–Validation Rule \ Navigate Application Flexfield Key CrossValidation	Cross–Validation window See: Cross–Validation Rules *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Key > CrossValidation
Define Currency \ Navigate Application Currency	Currencies window See: Defining Currencies *(Oracle Applications General Ledger User's Guide)* Navigator: Application > Currency
Define DataGroup \ Navigate Security ORACLE DataGroup	Data Groups window See: Data Groups Navigator: Security > ORACLE > DataGroup
Define Descriptive Flexfield Security Rule \ Navigate Security Responsibility Flexfield Descriptive Define	Define Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Define In the Find window choose Descriptive Flexfield
Define Descriptive Flexfield Segments \ Navigate Application Flexfield Descriptive Segments	Descriptive Flexfield Segments window See: Descriptive Flexfield Segments *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Descriptive > Segments
Define Document Categories \ Navigate Application Document Categories	Document Categories window See: Document Categories Navigator: Application > Document > Categories
Define Document Sequences \ Navigate Application Document Define	Document Sequences window See: Document Sequences Navigator: Application > Document > Define
Define FlexBuilder Parameter \ Navigate Application Flexfield FlexBuilder Define	The FlexBuilder feature is replaced by the Account Generator feature using Oracle Workflow See: Account Generator *(Oracle Applications Flexfield Guide)*
Define Help Text \ Navigate Application Text	Obsolete in GUI

Table 11 – 8 (Page 3 of 7)

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Define Key Flexfield Security Rule \ Navigate Security Responsibility Flexfield Key Define	Define Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Define In the Find window choose Key Flexfield
Define Key Flexfield Segments \ Navigate Application Flexfield Key Segments	Key Flexfield Segments See: Key Flexfield Segments *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Key > Segments
Define Key Segment Values \ Navigate Application Flexfield Key Values	Segment Values window See: Segment Values *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Key > Values
Define Logical Databases \ Navigate Concurrent Databases	Obsolete in Release 11 Logical Databases have been replaced with Concurrent Conflicts Domains See: Concurrent Conflicts Domains Navigator: Concurrent > Conflicts Domains
Define Menu \ Navigate Application Menu	Menus window See: Menus Navigator: Application > Menu
Define Parameter Security Rule \ Navigate Security Responsibility Report Rules Define	Define Security Rules window See: Assign Security Rules *(Oracle Applications Flexfields Guide)* Navigator: Security > Responsibility > ValueSet > Define In the Find window choose Concurrent Program
Define Parameter Values \ Navigate Application Validation Report	Define Value Set Values See: Segment Values *(Oracle Applications Flexfields Guide)* Navigator: Application > Validation > Values In the Find window choose Value Set
Define Print Style \ Navigate Install Printer Style	Print Styles window See: Print Styles Navigator: Install > Printer > Style

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Define Printer Driver \ Navigate Install Printer Driver	Printer Drivers window See: Printer Drivers Navigator: Install > Printer > Driver
Define Printer Types \ Navigate Install Printer Type	Printer Types window See: Printer Types Navigator: Install > Printer > Types
Define Report Group \ Navigate Security Responsibility Report Group	Request Groups window See: Request Groups Navigator: Security > Responsibility > Request
Define Responsibility \ Navigate Security Responsibility Define	Responsibilities window See: Responsibilities Navigator: Security > Responsibility > Define
Define Rollup Groups \ Navigate Application Flexfield Key Groups	Rollup Groups window See: Rollup Groups *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Key > Groups
Define Segment Values \ Navigate Application Validation Values	Segment Values window See: Segment Values *(Oracle Applications Flexfields Guide)* Navigator: Application > Validation > Values or Navigator: Application > Flexfield > Key > Values
Define Shorthand Aliases \ Navigate Application Flexfield Key Aliases	Shorthand Aliases window See: Shorthand Aliases *(Oracle Applications Flexfields Guide)* Navigator: Application > Flexfield > Key > Aliases
Define Terminal Group \ Navigate Security Responsibility Terminal Group	Obsolete in GUI
Define Value Set \ Navigate Application Validation Set	Value Sets window See: Value Sets *(Oracle Applications Flexfields Guide)* Navigator: Application > Validation > Set
Define Work Shifts \ Navigate Concurrent Manager WorkShifts	Work Shifts window See: Work Shifts Navigator: Concurrent > Manager > WorkShifts

Table 11 – 8 (Page 5 of 7)

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Define Zoom \ Navigate Application Zoom	Obsolete in GUI
Flexbuilder Test Screen \ Navigate Application Flexfield FlexBuilder Test	The FlexBuilder feature is replaced by the Account Generator feature using Oracle Workflow See: Account Generator *(Oracle Applications Flexfield Guide)*
Monitor Application Users \ Navigate Security User Monitor	Monitor Users window See: Monitor Users Navigator: Security > User > Monitor
Register Applications \ Navigate Application Register	Applications window See: Applications Navigator: Application > Register
Register Nodes \ Navigate Install Nodes	Nodes window See: Nodes Navigator: Install > Nodes
Register Oracle IDs \ Navigate Security ORACLE Register	ORACLE Users window See: ORACLE Users Navigator: Security > ORACLE > Register
Register Printers \ Navigate Install Printer Register	Printers See: Printers Navigator: Install > Printer > Register
Register Terminals \ Navigate Install Terminals	Obsolete in GUI
Retrieve Audit Data \ Navigate Security AuditTrail	Obsolete in GUI
Run Reports \ Navigate Report Run	Submit Requests window See: Submit Requests *(Oracle Applications User's Guide)* Navigator: Reports > Run
Update Installation Information \ Navigate Install Information	Obsolete in GUI

Table 11 – 8 (Page 6 of 7)

Character Mode Form and Menu Path	GUI Window or Process, and Navigator Path
Update Personal Profile Options \ Navigate Profile Personal	Personal Profile Values window See: Personal Profile Values *(Oracle Applications Flexfields Guide)* Navigator: Profile > Personal
Update System Profile Options \ Navigate Profile System	System Profile Values window See: System Profile Values Navigator: Profile > System
View Concurrent Requests \ Navigate Concurrent Requests	Requests window See: Viewing Requests *(Oracle Applications User's Guide)* Navigator: Concurrent > Requests Choose: Report button – to view Request Output Log button – to view Request Log Use the Menu to choose: Special > Manager Log – to view Manager Log
View Reports \ Navigate Report View	Requests window See: Viewing Requests *(Oracle Applications User's Guide)* Navigator: Concurrent > Requests Choose: Report button – to view Request Output Log button – to view Request Log

Table 11 – 8 (Page 7 of 7)

Glossary

This glossary contains definitions of terms you might encounter while reading this *Oracle Applications System Administrator's User's Guide*. You can consult this glossary to learn more about:

- *Oracle Applications System Administrator's* terminology
- Other manufacturing terminology
- Key Oracle Applications terminology

Glossary

Accounting Flexfield The code you use to identify a general ledger account in an Oracle Financials application. Each Accounting Flexfield value corresponds to a summary or rollup account within your chart of accounts.

Action Set A sequence of alert actions that are enabled for a particular alert. You can assign a sequence number to each action you include in an action set to specify the order in which the actions are performed.

Alert A mechanism that checks your database for a specific exception condition. An alert is characterized by the *SQL SELECT statement* it contains. A SQL SELECT statement tells your application what database exception to identify as well as what output to produce for that exception.

Alert Action An action you want your alert to perform. An alert action can depend on the output from the alert. An action can include sending an electronic mail message to a mail ID, running an Oracle Applications program, running a program or script from your operating system, or running a SQL script to modify information in your database. You can have more than one action for an alert, and an action can incorporate the output of the alert.

AuditTrail AuditTrail tracks which rows in a database table(s) were updated at what time and which user was logged in using the form(s). Several updates can be tracked, establishing a trail of audit data that documents the database table changes.

Concurrent Manager A mechanism that runs concurrent programs. A manager operates during the time and days defined by a work shift. A manager can run any concurrent program, or be specialized to run only certain kinds of programs.

Concurrent Program A program that runs concurrently (at the same time) as other programs. Concurrent programs run as background processes, while you continue to work at your terminal.

Concurrent Request A command to start a concurrent program. An example of a concurrent request is a command to generate and print a report.

Data Group A data group is a group list of Oracle Applications and the Oracle ID each application is assigned to. An Oracle ID grants access privileges to tables in an ORACLE database.

Detail Action An alert action that represents one exception found in the database.

Document Sequence A definition of how to assign a unique number to each document generated when a selected database table is updated.

Event Alert An alert that monitors the occurrence of a specific exception or change in your database. An exception in your database results if you add or update information using Oracle Applications forms. The event alert monitors the database for exceptions based on its SQL SELECT statement.

No Exception Action An alert action that represents no exceptions found in the database.

Periodic Alert An alert that periodically reports key information according to a schedule you define. Rather than notify you of immediate exceptions in the database like an event alert, a periodic alert scans for specific database information specified by its SQL SELECT statement at scheduled intervals.

Summary Action An alert action that represents multiple exceptions found in the database.

Summary Threshold The number of exceptions Oracle Alert can find before it changes an alert action from a detail action to a summary action.

Index

A

C

D

E

F

H

I

L

M

N

O

P

R

S

T

U

W

Z

Reader's Comment Form

Oracle Applications System Administrator's Guide
A58194–01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information we use for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual? What did you like least about it?

If you find any errors or have any other suggestions for improvement, please indicate the topic, chapter, and page number below:

_____

_____

_____

_____

_____

_____

_____

_____

Please send your comments to:

Oracle Applications Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065 USA
Phone: (650) 506–7000 Fax: (650) 506–7200

If you would like a reply, please give your name, address, and telephone number below:

_____

_____

_____

Thank you for helping us improve our documentation.