Oracle® Fusion Middleware

Reference Guide for Oracle Business Intelligence Applications

11*g* Release 1 (11.1.1)

E16816-05

August 2012

Explains various topics related to administering Oracle Business Intelligence Applications, including security, multiple language support and naming conventions in the Oracle Business Analytics Warehouse, applying patches, moving Oracle BI Applications components across environments, configuring the Oracle BI Repository, and customizing ETL mappings.

ORACLE®

Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications 11*g* Release 1 (11.1.1)

E16816-05

Primary Authors: Jill Arehart, Peter Brownbridge, Nick Fry, Stefanie Rhone

Contributors: Oracle Business Intelligence development, product management, and quality assurance teams

Contents

Preface

Oracle Business Intelligence Applications are comprehensive prebuilt solutions that deliver pervasive intelligence across an organization, empowering users at all levels — from front line operational users to senior management — with the key information they need to maximize effectiveness. Intuitive and role-based, these solutions transform and integrate data from a range of enterprise sources and corporate data warehouses into actionable insight that enables more effective actions, decisions, and processes.

Oracle BI Applications is built on Oracle Business Intelligence Enterprise Edition, a comprehensive next-generation business intelligence and analytics platform.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Audience

This document is intended for managers and implementors of Oracle BI Applications.

Conventions

The following text conventions are used in this document:

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

Related Documents

For more information, see the following Oracle Business Intelligence Applications 11*g* Release 1 (11.1.1) documents:

- The Oracle Business Intelligence Applications and Data Warehouse Administration Console chapter in the *Oracle Fusion Middleware Release Notes* for your platform: http://docs.oracle.com/cd/E23943_01/relnotes.htm
- *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*
- *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*

Also see the following documents in the Oracle Business Intelligence Enterprise Edition 11*g* Release 1 (11.1.1) documentation set:

- The Oracle Business Intelligence chapter in *Oracle Fusion Middleware Release Notes* for your platform
- *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence*
- *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Developer's Guide for Oracle Business Intelligence Enterprise Edition*
- *Oracle Fusion Middleware Scheduling Jobs Guide for Oracle Business Intelligence Enterprise Edition*

What's New in This Release

This chapter describes the new features in Oracle Business Intelligence Applications 11*g* Release 1 (11.1.1) that are documented in this guide, *Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications*.

This chapter contains the following topics:

- New Features in Oracle BI Applications Documented in This Guide
- Updates to Revision 3 of This Guide
- Updates to Revision 4 of This Guide
- System Requirements and Certification

For additional new features related to Oracle BI Applications 11*g* Release 1 (11.1.1) that are documented in other guides, see the What's New sections in:

- *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*
- *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*

New Features in Oracle BI Applications Documented in This Guide

New features in Oracle BI Applications 11*g* Release 1 (11.1.1) that are documented in *Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications* include the following:

New Security Architecture

Oracle BI Applications 11*g* is tightly integrated with the Oracle Fusion Middleware Security architecture, and delegates core security functionality to components of that architecture. Chapter 1, "Oracle BI Applications Security," provides detailed information about how security is implemented in Oracle BI Applications 11*g* Release 1 (11.1.1).

New Multi-Language Support

Oracle BI Applications provides multi-language support for metadata level objects exposed in Oracle BI Enterprise Edition dashboards and reports, as well as for data, which enables users to see records translated in their preferred language. Chapter 2, "About Multi-Language Support," explains how multi-language support is implemented in Oracle BI Applications 11*g* Release 1 (11.1.1).

New Oracle BI Applications Patching Support

Oracle BI Applications 11*g* Release 1 (11.1.1) supports patching, which can include bug fixes, metadata, and binary file updates. Chapter 4, "Oracle BI Applications Patching," describes the Oracle BI Applications patching process and provides instructions for applying and rolling back patches as well as diagnosing errors in the patch application process. Chapter 4 supplements the information provided in *Oracle Fusion Middleware Patching Guide* for patching Oracle Fusion Middleware products.

Section 5.4.1.1 'How to Apply Changes to JAZN Settings' also explains how to apply an Oracle BI Applications patch that contains changes to JAZN settings (that is, in the biapps_policystore.xml file).

Information About Configuring Oracle BI Repository for Use With Oracle BI Applications

Chapter 6, "Configuring the Oracle BI Repository," provides instructions for configuring the Oracle BI Repository for use with Oracle BI Applications.

Updates to Revision 3 of This Guide

Revision 3 of *Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications (11.1.1)* contains the following updates:

- Updates to the Oracle BI Applications patching process, as documented in Chapter 4, "Oracle BI Applications Patching."
- A new chapter, Chapter 5, "Moving Oracle BI Applications Components," which provides information about moving Oracle BI Applications from one environment to another.

Updates to Revision 4 of This Guide

Revision 4 of *Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications (11.1.1)* contains the following updates:

- Updates to Informatica PowerCenter references to reflect the supported version.

System Requirements and Certification

Refer to the system requirements and certification documentation for information about hardware and software requirements, platforms, databases, and other information. Both of these documents are available on Oracle Technology Network (OTN).

The system requirements document covers information such as hardware and software requirements, minimum disk space and memory requirements, and required system libraries, packages, or patches:

`http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-requirements-100147.html`

The certification document covers supported installation types, platforms, operating systems, databases, JDKs, and third-party products:

`http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html`

1

Oracle BI Applications Security

This chapter describes how to set up and maintain security for Oracle BI Applications.

This chapter contains the following topics:

Tip: To set up security for new deployment of Oracle BI Applications, follow the steps in Section 1.1.1, "High Level Steps for Setting Up Security in Oracle BI Applications".

1.1 Introduction

Oracle BI Applications 11*g* is tightly integrated with the Oracle Fusion Middleware Security architecture, and delegates core security functionality to components of that architecture.

Oracle BI Applications 11*g* is built on the Oracle BI EE platform, and security is configured using tools available to the Oracle BI EE platform.

For more information about the background and context of security for the Oracle BI EE platform, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

You should be thoroughly familiar with the security features of Oracle BI Enterprise Edition before you begin working with Oracle BI Applications.

This section contains the following topics:

- Section 1.1.5, "What Is a Security Policy and Where Is it Maintained?"
- Section 1.1.6, "Controlling User Access Using Roles"
- Section 1.1.7, "About Object-Level Security"
- Section 1.1.8, "What Privileges Are Required to Run the ETL Process?"
- Section 1.1.9, "Where Is Content Authenticated?"
- Section 1.1.10, "Security Overview of Oracle BI Applications Configuration Manager and Functional Setup Manager"
- Section 1.1.11, "About Permissions in DAC, Configuration Manager and Functional Setup Manager"
- Section 1.1.12, "Additional Sources of Information About Oracle BI Applications Security"

1.1.1 High Level Steps for Setting Up Security in Oracle BI Applications

To set up security in Oracle BI Applications, you must do the following:

1. Read the rest of Section 1.1, "Introduction" to get an overview of security concepts, tools, and terminology.

   For background information about security in the Oracle Business Intelligence Enterprise Edition platform, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

2. Learn about Duty Roles, Job Roles, Data Roles, and Abstract Roles and how they control user privileges. For more information, see Section 1.1.6, "Controlling User Access Using Roles".

   Note: Job Roles, Data Roles, and Abstract Roles are sometimes collectively referred to as Enterprise Roles.

3. The Oracle Fusion Applications provisioning process configures Users and default Fusion Applications Enterprise Roles in your LDAP directory.

   For more information, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".

   If you need to manually map Users to Enterprise Roles, follow the steps in Section 1.3.1, "Creating Users and Assigning Them to Job Roles".

4. Fusion Applications Enterprise Roles in the LDAP directory are mapped by default to Duty Roles for Oracle BI Applications, which provides users with access to dashboards and data.

   For background information about Oracle APM to manage Duty Roles, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges".

   If you want to change the default role mappings, follow the steps in Section 1.3.2, "Assigning Enterprise Roles (or Job Roles) to Duty Roles".

5. Set up appropriate users for Oracle BI Applications components (for example, Oracle BI Applications Configuration Manager, and DAC).

   This task is performed as step 'Grant User Access to Oracle BI Applications Components', in the post-installation steps for Oracle BI Applications. For more information, see the topic 'Grant User Access to Oracle BI Applications Components' in Chapter 4 of *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

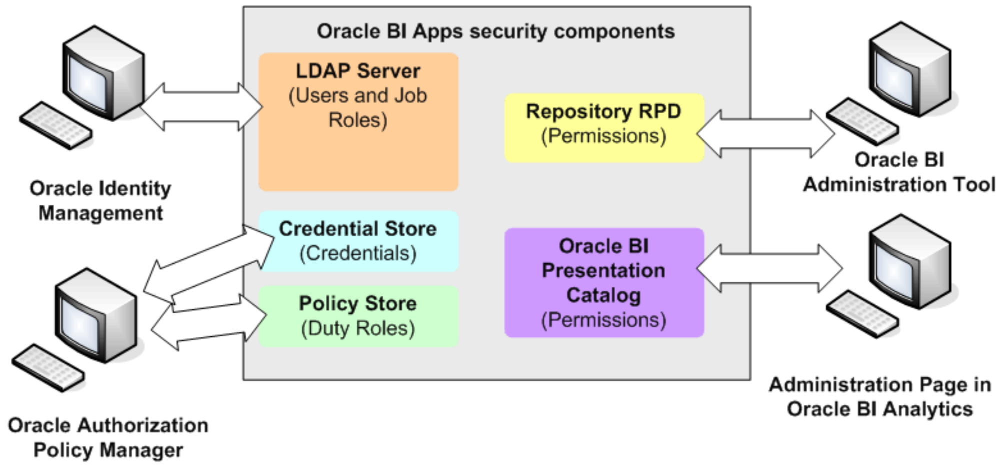
When determining your user requirements, read Section 1.1.10, "Security Overview of Oracle BI Applications Configuration Manager and Functional Setup Manager".

6. The Oracle BI Repository is installed with default access permissions on installation. If the default access permissions do not meet your business requirements, then fine tune the access permissions for Roles by following the steps in Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository".
7. The Oracle BI Presentation Catalog is installed with default privileges on installation. If the default privileges do not meet your business requirements, then fine tune the privileges granted for Roles by following the steps in Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server".
8. If required, configure Single Sign-On (SSO). For more information, see "Enabling SSO Authentication" in *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.
9. If required, configure Secure Sockets Layer (SSL). For more information, see Section 1.4, "Configuring SSL for Oracle BI Applications".
10. For examples and typical setup scenarios, see Section 1.3, "Setting Up Security in Oracle BI Applications".

1.1.2 What Tools Configure Security in Oracle Business Intelligence Applications?

The figure below summarizes the tools used to configure security in a default installation of Oracle BI Applications using OID as the LDAP Server.

Figure 1–1 Summary of Tools for Configuring Security in a Default Installation


Oracle BI Apps security components
LDAP Server
(Users and Job Roles)
Repository RPD
(Permissions)
Credential Store
(Credentials)
Policy Store
(Duty Roles)
Oracle BI Presentation Catalog
(Permissions)
Oracle Identity Management
Oracle Authorization Policy Manager
Oracle BI Administration Tool
Administration Page in Oracle BI Analytics

Use these Security configuration tools as follows:

- Oracle Identity Management

  Use Oracle Identity Management to manage Users and Job Roles (referred to as Groups in OID) in the LDAP directory. For more information about using OID tools, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".

For detailed information about deploying OID, see *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management (Oracle Fusion Applications Edition)*.

- Oracle Authorization Policy Manager

  Use Oracle Authorization Policy Manager (Oracle APM) to manage Duty Roles (also known as Application Roles), and permissions for determining functional access. For more information about using Oracle APM, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges".

  For detailed information about Oracle APM, see *Oracle Fusion Middleware Oracle Authorization Policy Manager Administrator's Guide (Oracle Fusion Applications Edition)*.

- Oracle BI Administration Tool

  Use the Oracle BI Administration Tool to perform tasks such as setting permissions for business models, tables, columns, and subject areas; specifying filters to limit data accessibility; and setting authentication options. For more information about using Oracle BI Administration Tool, see Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository".

  For detailed information, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

- Oracle BI Presentation Services Administration

  Use Oracle BI Presentation Services Administration to perform tasks such as setting permissions for Presentation Catalog objects, including dashboards and dashboard pages. For more information about using Oracle BI EE Administration Page, see Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server".

  For detailed information, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

For detailed information about using these tools, see Section 1.2, "About Security Configuration Tools in Oracle BI Applications".

1.1.3 What Security Levels Does Oracle BI Applications Use?

Security in Oracle BI Applications can be classified broadly into the following three levels:

- **User-level security (authentication of users).** User-level security concerns the authentication and confirmation of the identity of a user based on the credentials provided, such as username and password. By default, user-level security is set up in the LDAP server and policy store. For more information, see *Oracle Fusion Applications Security Guide*.

- **Object-level security.** Object-level security controls the visibility to business logical objects based on a user's role. You can set up object-level security for Oracle BI Repository objects, such as business models and subject areas, and for Web objects, such as dashboards and dashboard pages, which are defined in the Presentation Catalog. For more information, see Section 1.1.7, "About Object-Level Security."

- **Data-level security.** Data-level security controls the visibility of data (content rendered in subject areas, dashboards, Oracle BI Answers, and so on) based on the user's association to data in the transactional system. For more information, see Section 1.7, "Advanced Security Topics - About Data-Level Security."

1.1.4 What Security Providers Does Oracle BI Applications Use?

Oracle BI Applications uses the following security providers:

- **Credential store provider** - the credential store enables access to the credentials required for authentication of users.
- **Identity store provider** - the identity store contains information about user access to Oracle BI Applications, and is responsible for authenticating users.

  For more information, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".
- **Policy store provider** - the policy store authorizes access of Oracle BI Applications Duty Roles, and determines what users can and cannot see and do in Oracle BI Applications. This forms a core part of the security policy, described in Section 1.1.5.

For more information about roles, see Section 1.1.6, "Controlling User Access Using Roles".

1.1.5 What Is a Security Policy and Where Is it Maintained?

A security policy comprises a number of access privileges that are associated with user roles. In Oracle BI Applications release 11*g*, the security policy definition is split across the following components:

- **Presentation Services Catalog** – Defines access to catalog objects and Oracle BI Presentation Services functionality using specific roles associated with users.

  For more information, see Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server".
- **BI Repository** – Defines access permissions in presentation area folders using specific roles associated with users.

  For more information, see Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository"
- **Policy Store** – Defines access to Oracle BI Server, BI Publisher, and Real Time Decisions functionality using specific roles associated with users.

  For more information, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".

For more information about configuring these components, see Section 1.2, "About Security Configuration Tools in Oracle BI Applications".

1.1.6 Controlling User Access Using Roles

This topic describes how Oracle BI Applications controls user access using roles, and contains the following sections:

- Section 1.1.6.1, "Authorizing User Access Using Roles"
- Section 1.1.6.2, "About Roles in Oracle BI Applications"

1.1.6.1 Authorizing User Access Using Roles

After a user has been authenticated, the next critical aspect of security is to ensure that the user can do and see what they are authorized to do and see. Authorization for Oracle BI Applications release 11g is controlled by security policies (Oracle BI Applications privileges) defined for users using a role-based model (for more information, see Section 1.1.5, "What Is a Security Policy and Where Is it Maintained?").

Every Oracle Applications user is hired by their company to do a job in the organization. For example, Payroll Manager, AP Manager. Each job involves one or more duties. For example, a Software Development V.P. might hire people, carry out appraisals, make salary changes and stock grants to people in the group, manage project plans, approve expense reports, fill out vacation time, expense reports and update other personal details. Similarly an AP Manager might involve payable invoice approval, payable invoice processing, payables period closing, and some personal duties like filling out vacation time and submitting expenses. Thus job and duties form a hierarchy where a job has multiple duties and a person is hired to do a job. A job is represented by a Job Role. A duty is represented by Duty Role. A Job Role has access to one or more Duty Roles.

An Oracle Applications user is granted a Job Role and thus inherits one or more Duty Roles that were granted to the Job Role.

It is possible that a Job Role is built using other Job Roles, that is, Job Roles can form their own hierarchy. Similarly, Duty Roles can form a hierarchy.

It is possible to grant multiple Job Roles to a user; however Oracle recommends that Job Roles are defined in such a way that need to grant multiple Job Roles to user should be minimized.

1.1.6.2 About Roles in Oracle BI Applications

Roles in Oracle BI Applications fall into two classes, application-wide roles (Duty Roles), and enterprise-wide roles (Job Roles, Data Roles, and Abstract Roles).

Application-wide roles are defined in terms of the tasks that are needed to perform a job, and are held in the policy store with associated Oracle BI Applications privileges.

Enterprise-wide roles are defined in terms of an occupation within an enterprise, and are held in the identity store with associated users.

Four common role types are used in Oracle BI Applications:

- **Job Roles** (enterprise-wide roles, also referred to as Fusion Applications Enterprise Roles)

  A Job Role represents the job definition of a person in your organization, for example, AP manager, HR specialist. Users are assigned to Job Roles. A Job Role inherits one or more Duty Roles, and therefore, users inherit Duty Roles through Job Roles.

  For example, the Job Role AP_ACCOUNTS_PAYABLE_MANAGER_JOB might be associated with the Duty Role OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY.

  For more information about the Oracle BI Applications Job Role hierarchy, see Section 1.3.5, "Understanding Oracle BI Applications Job Role Hierarchies".

  For more information about managing Job Roles, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles").

- **Duty Roles** (application-wide roles, also referred to as Application Roles)

A Duty Role represents a specific task needed to do a job, and comprises the privileges required to perform that job. For example, the Duty Role BIA_ADMINISTRATOR_DUTY enables a user to administer and manage Oracle BI Applications, providing access to Oracle BI Applications Configuration Manager, and the Oracle BI Data Warehouse Administration Console.

The privileges assigned to a Duty Role are defined in application policies within the policy store. For more information, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges".

- **Data Roles** (enterprise-wide roles)

  A Data Role is implemented as Job Roles for a defined set of data. This role describes the job a user does within that defined set of data. The Data Role will inherit a Job Role and will be granted applicable data security privileges.

  – Data Roles grant specific data to users. For example, HR Admin UK has access to all UK employees, or Sales Rep West Coast can access West Coast customer accounts. Data Roles are built at the customer site as they are data dependant.

  – Data Roles are built using Job Role and permission grants on custom data. For example:

    An Accounts Payables Specialist might be assigned the Data Role 'Accounts Payables Specialist - US Business Unit'. This Data Role inherits the Job Role 'Accounts Payable Specialist' and grants access to transactions in the US Business Unit.

    A Benefits Administrator is assigned the Data Role 'Benefits Administrator - Surname A-E'. This Data Role inherits the Job Role 'Benefits Administrator' and grants access to Employees with the Surname starting from A to E. Typically, a person is hired into a job, which might trigger an event to automatically assign a Job Role without being assigned a defined set of data. A person in a functional department might at a later point in time assign his staff a Data Role that describes the job of that person within that defined set of data.

    Data roles can be provisioned to a user on request. Data Roles can be formed by declaring child Duty, Abstract, and Job Roles.

    For example, the Data Role OBIA_COSTING_ORGANIZATION_DATA_SECURITY might be associated with the Job Role OBIA_COST_ACCOUNTING_ANALYSIS_JOB.

- **Abstract Roles** (enterprise-wide roles)

  These roles are associated with a user irrespective of their job or job function, and are not associated with a job or duty. For example, Employee (Human Resources), Manager (Human Resources), Customer, Supplier etc. Abstract Roles are normally assigned by the system (based on user attributes) but can be provisioned to a user on request.

  An example of an Abstract Role is ASM_APPLICATION_IMPLEMENTATION_ADMIN_ABSTRACT.

Figure 1–2 illustrates the relationship between Users, Job Roles, Duty Roles, and Oracle BI Applications privileges.

Figure 1–2 Example Users, Job Roles, Duty Roles, and Oracle BI Applications Privileges

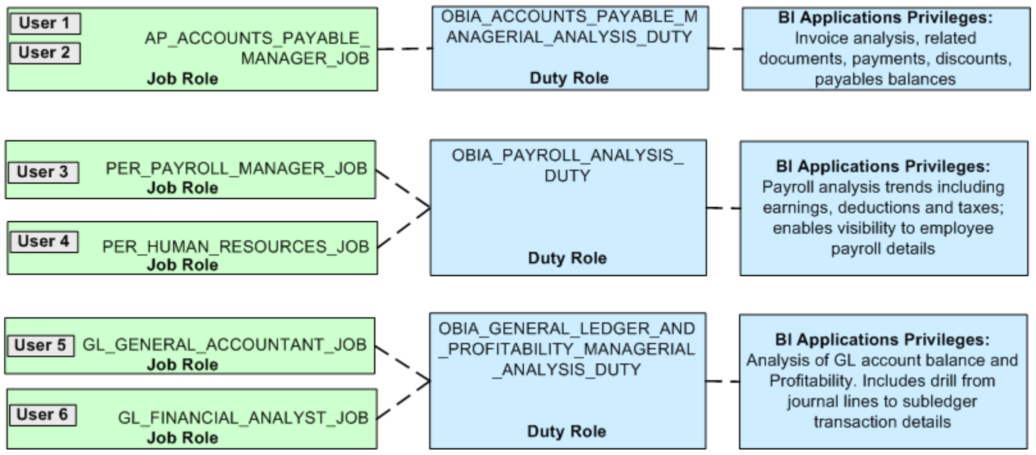

User 1
User 2
AP_ACCOUNTS_PAYABLE_MANAGER_JOB
Job Role
OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY
Duty Role
BI Applications Privileges: Invoice analysis, related documents, payments, discounts, payables balances
User 3
PER_PAYROLL_MANAGER_JOB
Job Role
User 4
PER_HUMAN_RESOURCES_JOB
Job Role
OBIA_PAYROLL_ANALYSIS_DUTY
Duty Role
BI Applications Privileges: Payroll analysis trends including earnings, deductions and taxes; enables visibility to employee payroll details
User 5
GL_GENERAL_ACCOUNTANT_JOB
Job Role
User 6
GL_FINANCIAL_ANALYST_JOB
Job Role
OBIA_GENERAL_LEDGER_AND_PROFITABILITY_MANAGERIAL_ANALYSIS_DUTY
Duty Role
BI Applications Privileges: Analysis of GL account balance and Profitability. Includes drill from journal lines to subledger transaction details

Figure 1–2 illustrates the following:

- Users 1 and 2 have the Oracle BI Applications Job Role 'AP_ACCOUNTS_PAYABLE_MANAGER_JOB'. This Job Role is associated with the Duty Role 'OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY' which enables associated users to analyze invoices and related documents along with payments, discounts, and payables balances.
- User 3 has the Oracle BI Applications Job Role 'PER_PAYROLL_MANAGER_JOB', and User 4 has the Job Role 'PER_HUMAN_RESOURCES_JOB'. Both of these Job Roles are associated with the Duty Role 'OBIA_PAYROLL_ANALYSIS_DUTY' which enables these users to analyze payroll trends including earnings, deductions and taxes, and giving visibility to employee payroll details.
- User 5 has the Oracle BI Applications Job Role 'GL_GENERAL_ACCOUNT_JOB', and User 6 has the Job Role 'GL_FINANCIAL_ANALYST_JOB'. Both of these Job Roles are associated with the Duty Role 'OBIA_GENERAL_LEDGER_AND_PROFITABILITY_MANAGERIAL_ANALYSIS_DUTY' which enables users to analyze GL account balance and profitability. It also enables drill from journal lines to subledger transaction details.

1.1.7 About Object-Level Security

Object-level security is the specification of Duty Role permissions on Oracle BI Repository objects such as subject areas, tables, and columns, and Oracle BI Presentation Services objects, such as dashboard pages.

You secure these objects using Duty Roles and their associated policies and privileges, defined for Oracle BI Applications.

Duty Roles are stored in the Fusion Policy Store, which is accessed by the Oracle BI Repository and the Oracle BI Presentation Catalog.

For more information about default role mapping, see Section 1.5, "Mapping Roles to Secure Objects in the Oracle BI Repository and Oracle BI Presentation Catalog".

You implement object level security using:

- Oracle Identity Management

  To configure Users and Job Roles.
- Oracle BI Administration Tool

To configure Oracle BI Repository Duty Role object permissions (for example, read, write against a subject area, table or column).

- Administration page in the Oracle BI Presentation Catalog

  To configure Oracle BI Presentation Services Duty Role object privileges (for example, access, view, or edit, a dashboard page).

For more information, see Section 1.2, "About Security Configuration Tools in Oracle BI Applications"

1.1.8 What Privileges Are Required to Run the ETL Process?

The Extract Transform and Load (ETL) process must be run by a user with appropriate data security privileges granted on the Fusion Application tables from which data is extracted into Oracle Business Analytics Warehouse. For this purpose, the Group named FUSION_APPS_OBIA_BIEE_APPID is provisioned during install with the appropriate ETL security privileges (by default, this Group is mapped to the Duty Role named OBIA_EXTRACT_TRANSFORM_LOAD_DUTY).

To create a user for ETL, use Oracle Identity Management (or an appropriate security tool) to create a new user and password. For example, you might create a new user named OBIA_ETL_USER. Then, make the user a member of the Group FUSION_APPS_OBIA_BIEE_APPID. For example, you might assign the user OBIA_ETL_USER to the Group FUSION_APPS_OBIA_BIEE_APPID. Finally, make a note of the user's credentials. This process is described in the post-installation setup steps in the topic 'Create a User for ETL' in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*".

Note: The ETL user password is stored with the DAC connection details, and is not automatically synchronized with the LDAP version. Therefore, if you change the LDAP password of the ETL user is changed later, then you must also make the same changes to the DAC connection information. If the ETL user password expires, then you must reset it in LDAP, in the DAC connection details, and in Informatica > Relational Connections. To avoid this issue you can choose not to set an expiry date for the ETL user password (if your security best practices allow).

1.1.9 Where Is Content Authenticated?

Oracle BI Applications content can be authenticated as follows:

- Fusion Reporting Pane

  User credentials are authenticated in the Reporting Pane login.

- Oracle BI Applications dashboards accessed directly

  The BI Presentation Server passes an authentication request to the BI Server, which uses the Identity Store.

- Fusion Applications UI ("embedded Analytics")

  In this scenario the user is already in Fusion Applications and therefore has already been authenticated.

1.1.10 Security Overview of Oracle BI Applications Configuration Manager and Functional Setup Manager

To access Oracle BI Applications Configuration Manager or Functional Setup Manager (for Oracle BI Applications), a user must be granted one of the following Duty Roles:

- BI Applications Administrator Duty (BIA_ADMINISTRATOR_DUTY)

  Users with the BI Applications Administrator Duty Role have access to all Configuration Manager User Interfaces and all Functional Setup Manager User Interfaces. For Configuration Manager, only users with this Duty Role are able to perform system setup tasks.

- BI Applications Implementation Manager Duty (BIA_IMPLEMENTATION_MANAGER_DUTY)

  Users with the BI Applications Implementation Manager Duty Role have access to Configuration Manager Overview page and the Export and Import of Setup Data. In Functional Setup Manager, these users have access to Configure Offerings and Manage Implementation Projects User Interfaces but cannot execute a setup task.

- BI Applications Functional Developer Duty (BIA_FUNCTIONAL_DEVELOPER_DUTY)

  Users with the BI Applications Functional Developer Duty Role have access to Configuration Manager User Interfaces, except for the System Setup screens. In Functional Setup Manager, these users have access to the list of functional setup tasks assigned to them and have the ability to execute the setup tasks.

Assigning these Duty Roles to appropriate Users and Job Roles is performed as Step 8: 'Grant User Access to Oracle BI Applications Components', in the post-installation steps for Oracle BI Applications. For more information, see section 4.3.8 'Grant User Access to Oracle BI Applications Components' in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

1.1.11 About Permissions in DAC, Configuration Manager and Functional Setup Manager

This section describes permissions in DAC, Configuration Manager and Functional Setup Manager, and contains the following sections.

- Section 1.1.11.1, "About Permissions in DAC"
- Section 1.1.11.2, "About Permissions in Configuration Manager"
- Section 1.1.11.3, "About Permissions in Functional Setup Manager"

1.1.11.1 About Permissions in DAC

For Oracle BI Applications, DAC permissions are granted through the Oracle BI Applications Duty Roles as follows:

- BI Applications Administrator

  - Read DAC Repository

  - Administer DAC Repository

- BI Applications Implementation Manager

  - Read DAC Repository

  - Manage DAC ETL

- BI Applications Developer

  - Read DAC Repository

  - Design DAC Metadata

You login to DAC using the 'BI' connection type, because the DAC Server will run in Web mode. User authentication is handled by the LDAP directory where the DAC Server is deployed.

DAC permissions are as follows:

- Read DAC Repository

  resourceType=bi.dac.permission,resourceName=oracle.bi.dac.readDACRepository
- Manage DAC ETL

  resourceType=bi.dac.permission,resourceName=oracle.bi.dac.manageDACETL
- Design DAC Metadata

  resourceType=bi.dac.permission,resourceName=oracle.bi.dac.designDACMetadat a
- Administer DAC Repository

  resourceType=bi.dac.permission,resourceName=oracle.bi.dac.administerDACRep ository

1.1.11.2 About Permissions in Configuration Manager

Table 1–1 shows the list of Configuration Manager screens visible to each of the Oracle BI Applications Duty Roles.

Table 1–1 List of Configuration Manager Screens Visible to Each Oracle BI Applications Duty Role

Oracle BI Applications Duty Role	Oracle BI Applications Configuration Manager screen	Associated Privilege
BI Applications Administrator	Overview	BIA_OVERVIEW_PRIV
BI Applications Administrator	System Setups - Define Oracle BI Applications Instance	BIA_DEFINE_INSTANCE_ PRIV
BI Applications Administrator	System Setups - Manage Oracle BI Applications	BIA_MANAGE_INSTANCE_ PRIV
BI Applications Administrator	System Setups - Manage Preferred Currencies	BIA_MANAGE_INSTANCE_ PRIV
BI Applications Administrator	Functional Configurations - "Perform Functional Configurations" link to launch Functional Setup Manager	BIA_FUNCTIONAL_ SETUPS_PRIV
BI Applications Administrator	Setup Data Maintenance and Administration - Manage Domains and Mappings	BIA_CONFIGURE_ DOMAINS_PRIV
BI Applications Administrator	Setup Data Maintenance and Administration - Manage Data Load Parameters	BIA_CONFIGURE_ DATALOAD_PARAMS_PRIV
BI Applications Administrator	Setup Data Maintenance and Administration - Manage Reporting Parameters	BIA_CONFIGURE_RPD_ PARAMS_PRIV
BI Applications Administrator	Setup Data Export and Import - Export Setup Data	BIA_EXPORT_SETUPS_PRIV

Table 1–1 (Cont.) List of Configuration Manager Screens Visible to Each Oracle BI Applications Duty Role

Oracle BI Applications Duty Role	Oracle BI Applications Configuration Manager screen	Associated Privilege
BI Applications Administrator	Setup Data Export and Import - Import Setup Data	BIA_IMPORT_SETUPS_PRIV
BI Applications Functional Developer	Overview	BIA_OVERVIEW_PRIV
BI Applications Functional Developer	Functional Configurations - "Perform Functional Configurations" link to launch Functional Setup Manager	BIA_FUNCTIONAL_ SETUPS_PRIV
BI Applications Functional Developer	Setup Data Maintenance and Administration - Manage Domains and Mappings	BIA_CONFIGURE_ DOMAINS_PRIV
BI Applications Functional Developer	Setup Data Maintenance and Administration - Manage Data Load Parameters	BIA_CONFIGURE_ DATALOAD_PARAMS_PRIV
BI Applications Functional Developer	Setup Data Maintenance and Administration - Manage Reporting Parameters	BIA_CONFIGURE_RPD_ PARAMS_PRIV
BI Applications Functional Developer	Setup Data Export and Import - Export Setup Data	BIA_EXPORT_SETUPS_PRIV
BI Applications Functional Developer	Setup Data Export and Import - Import Setup Data	BIA_IMPORT_SETUPS_PRIV
BI Applications Implementation Manager	Overview	BIA_OVERVIEW_PRIV
BI Applications Implementation Manager	Setup Data Export and Import - Export Setup Data	BIA_EXPORT_SETUPS_PRIV
BI Applications Implementation Manager	Setup Data Export and Import - Import Setup Data	BIA_IMPORT_SETUPS_PRIV

1.1.11.3 About Permissions in Functional Setup Manager

Functional Setup Manager Duty Roles are associated with Oracle BI Applications Duty Roles as follows:

- The BI Applications Administrator Duty role (BIA_ADMINISTRATOR_DUTY) is associated to the following Functional Setup Manager Duty Roles:
  - ASM_FUNCTIONAL_SETUPS_DUTY
  - ASM_IMPLEMENTATION_MANAGER_DUTY
  - ASM_APPLICATION_DEPLOYER_DUTY
  - ASM_APPLICATION_REGISTRATION_DUTY
  - ASM_LOGICAL_ ENTITY_MODELING_DUTY
  - ASM_SETUP_OBJECTS_PROVIDER_DUTY
- The BI Applications Implementation Manager Duty role (BIA_ IMPLEMENTATION_MANAGER_DUTY) is associated to the following Functional Setup Manager duty:

  - ASM_IMPLEMENTATION_MANAGER_DUTY
- The BI Applications Functional Developer Duty role (BIA_FUNCTIONAL_DEVELOPER_DUTY) is associated to the following Functional Setup Manager duty:
  - ASM_FUNCTIONAL_SETUPS_DUTY

1.1.12 Additional Sources of Information About Oracle BI Applications Security

When configuring security in Oracle BI Applications, in some circumstances you might need to refer to security in other areas, as follows:

- Oracle Fusion Applications security

  For more information, see:
  - *Oracle Fusion Applications Security Guide*
  - *Oracle Fusion Applications Common Security Reference Manual*.
- Oracle Business Intelligence Enterprise Edition security implementation

  For more information, see:
  - *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*
  - *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*

    For information about Oracle BI EE platform integration with VOs, see the section 'Working with ADF Business Component Data Sources'.

1.2 About Security Configuration Tools in Oracle BI Applications

To configure security in Oracle BI Applications, you typically use the following tools:

- Oracle Identity Management - for more information, see Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".
- Oracle Authorization Policy Manager (APM) - for more information, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges".
- Oracle BI EE Administration Tool - for more information, see Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository".
- Oracle BI EE Administration Page - for more information, see Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server".

1.2.1 Using Oracle Identity Management to Manage Users and Enterprise Roles

Oracle Identity Management enables you to manage authenticated users, Job Roles, and Data Roles in the LDAP directory.

Note: If you use an alternative authentication provider (for example, Active Directory) you must use the console provided with the alternative authentication provider to create Users, Job Roles, and Data Roles. Oracle WebLogic Server Administration Console will also display this information.

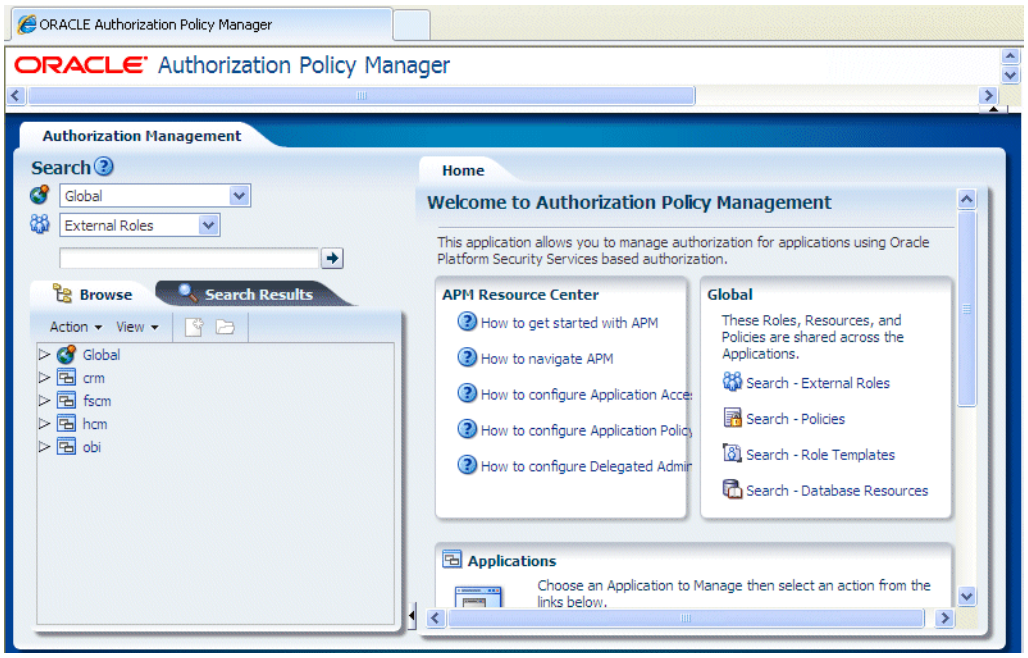
For more information about managing Users and Groups (Job Roles, Abstract Roles, Data Roles), see:

- Section 1.3.1, "Creating Users and Assigning Them to Job Roles"
- *Oracle Fusion Applications Security Guide*
- *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management (Oracle Fusion Applications Edition)*

1.2.2 Using Oracle Authorization Policy Manager to Configure Roles and Privileges

Oracle Authorization Policy Manager (APM) enables you to create and manage Duty Roles that control access privileges to Oracle Business Intelligence resources (for Oracle BI Applications Configuration Manager, FSM, DAC, and dashboards and data in Oracle BI EE platform).

Figure 1–3 Oracle Authorization Policy Manager Main Screen


ORACLE Authorization Policy Manager
ORACLE Authorization Policy Manager
Authorization Management
Search
Global
External Roles
Browse
Search Results
Action
View
Global
crm
fscm
hcm
obi
Home
APM Resource Center
How to get started with APM
How to navigate APM
How to configure Application Acce
How to configure Application Policy
How to configure Delegated Admir
Global
Search - External Roles
Search - Policies
Search - Role Templates
Search - Database Resources
Applications

Welcome to Authorization Policy Management

This application allows you to manage authorization for applications using Oracle Platform Security Services based authorization.

These Roles, Resources, and Policies are shared across the Applications.

Choose an Application to Manage then select an action from the links below.

Job Roles (or Fusion Applications Enterprise Roles) in the LDAP directory are mapped by default to the pre-configured Duty Roles that are installed with Oracle BI Applications. For detailed steps on mapping Enterprise Roles to Duty Roles, see Section 1.3.2, "Assigning Enterprise Roles (or Job Roles) to Duty Roles".

Note: Fusion Middleware Control also enables you to manage Duty Roles for the BI Platform, but Oracle recommends using Oracle APM to create and manage Duty Roles.

For a detailed list of the default mapping between roles, see Section 1.5, "Mapping Roles to Secure Objects in the Oracle BI Repository and Oracle BI Presentation Catalog".

In Oracle APM, you can search on roles related to a specific functional area. To do so, navigate to the Search Role tab of the Role Catalog page, and select the appropriate functional area from the Category list.

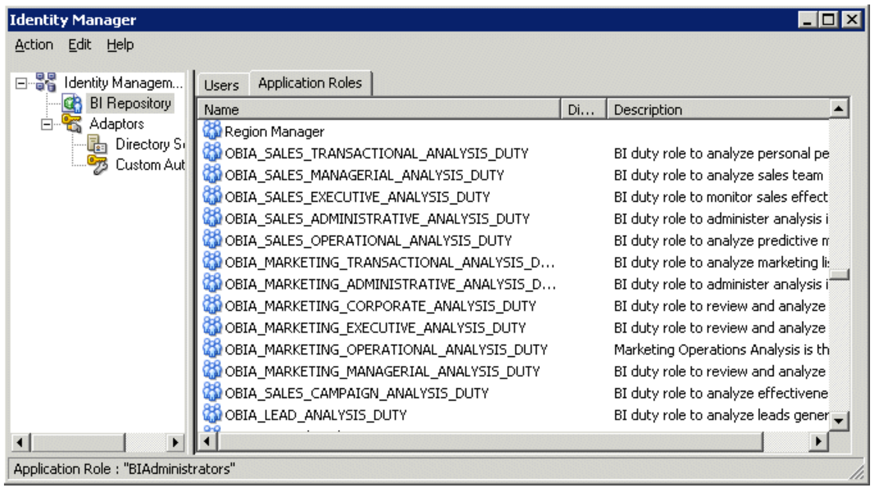
For more information about configuring roles and privileges, see:

- For background information about Duty Roles, see Section 4 'Role-Based Access Control' in *Oracle Fusion Applications Security Guide*
- For additional information about using Oracle APM to create and manage Duty Roles (referred to as Application Roles in Oracle APM) see Chapter 5 'Managing Security Artifacts' in *Oracle Fusion Middleware Oracle Authorization Policy Manager Administrator's Guide (Oracle Fusion Applications Edition)*

1.2.3 Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository

Oracle BI Administration Tool enables you to configure Duty Role permissions (for example, Read or Write), for business models, tables, columns, and subject areas in the Oracle BI Repository.

Figure 1–4 Oracle BI Administration Tool - Identity Manager



Name	Di...	Description
Region Manager		
OBIA_SALES_TRANSACTIONAL_ANALYSIS_DUTY		BI duty role to analyze personal pe
OBIA_SALES_MANAGERIAL_ANALYSIS_DUTY		BI duty role to analyze sales team
OBIA_SALES_EXECUTIVE_ANALYSIS_DUTY		BI duty role to monitor sales effect
OBIA_SALES_ADMINISTRATIVE_ANALYSIS_DUTY		BI duty role to administer analysis i
OBIA_SALES_OPERATIONAL_ANALYSIS_DUTY		BI duty role to analyze predictive m
OBIA_MARKETING_TRANSACTIONAL_ANALYSIS_D...		BI duty role to analyze marketing li
OBIA_MARKETING_ADMINISTRATIVE_ANALYSIS_D...		BI duty role to administer analysis i
OBIA_MARKETING_CORPORATE_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_MARKETING_EXECUTIVE_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_MARKETING_OPERATIONAL_ANALYSIS_DUTY		Marketing Operations Analysis is th
OBIA_MARKETING_MANAGERIAL_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_SALES_CAMPAIGN_ANALYSIS_DUTY		BI duty role to analyze effectivene
OBIA_LEAD_ANALYSIS_DUTY		BI duty role to analyze leads gener

Application Role : "BIAdministrators"

For more information, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

Note: You use Fusion Middleware Control to manage the Oracle BI Repository (RPD) password (for more information, see Changing the Repository Password in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*).

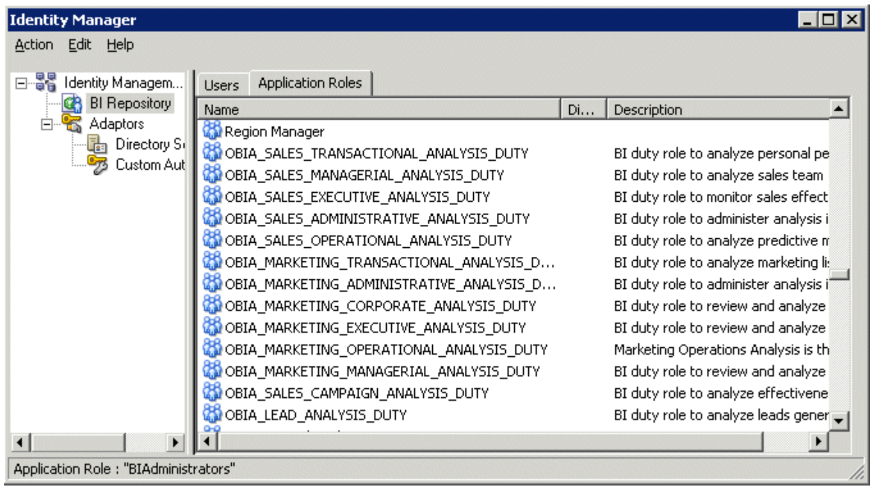
To configure Duty Role permissions on subject areas and tables in the Oracle BI Repository using the Oracle BI Administration Tool:

1. From the Windows Start menu, select All Programs, and Oracle Business Intelligence.
2. Log in to the Administration Tool.

   Note: If you log in to the Administration Tool in online mode, then you can view all users from the LDAP directory. If you log in to Administration Tool in offline mode, then you can only view users that are stored in the repository.
3. To configure Duty Role permissions on subject areas and tables:

   a. Choose Manage, then Identity to display the Identity Manager dialog.

      The Application Roles tab displays, Application Roles, and Duty Roles.



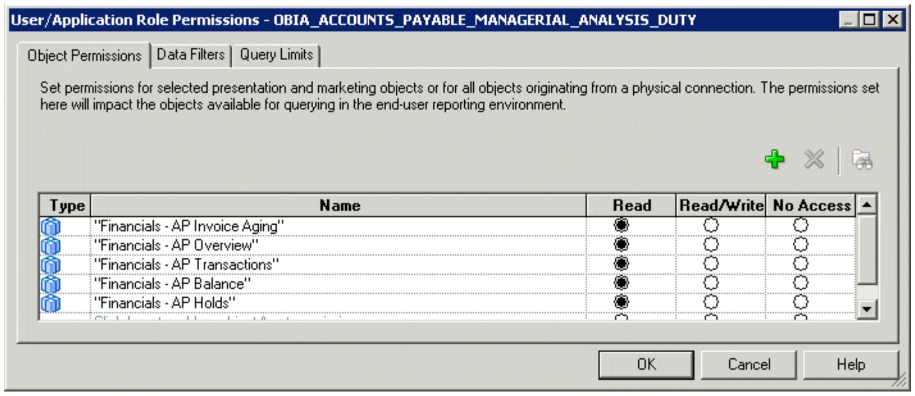
Name	Di...	Description
Region Manager		
OBIA_SALES_TRANSACTIONAL_ANALYSIS_DUTY		BI duty role to analyze personal pe
OBIA_SALES_MANAGERIAL_ANALYSIS_DUTY		BI duty role to analyze sales team
OBIA_SALES_EXECUTIVE_ANALYSIS_DUTY		BI duty role to monitor sales effect
OBIA_SALES_ADMINISTRATIVE_ANALYSIS_DUTY		BI duty role to administer analysis i
OBIA_SALES_OPERATIONAL_ANALYSIS_DUTY		BI duty role to analyze predictive m
OBIA_MARKETING_TRANSACTIONAL_ANALYSIS_D...		BI duty role to analyze marketing li
OBIA_MARKETING_ADMINISTRATIVE_ANALYSIS_D...		BI duty role to administer analysis i
OBIA_MARKETING_CORPORATE_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_MARKETING_EXECUTIVE_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_MARKETING_OPERATIONAL_ANALYSIS_DUTY		Marketing Operations Analysis is th
OBIA_MARKETING_MANAGERIAL_ANALYSIS_DUTY		BI duty role to review and analyze
OBIA_SALES_CAMPAIGN_ANALYSIS_DUTY		BI duty role to analyze effectivene
OBIA_LEAD_ANALYSIS_DUTY		BI duty role to analyze leads gener

Application Role : "BIAdministrators"

b. Double-click a Duty Role to display the Application Role <*Name*> dialog

c. Click **Permissions** to display the User/Application Role Permissions dialog.

d. Display the Object Permissions tab.


User/Application Role Permissions - OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY
Object Permissions
Data Filters
Query Limits
OK
Cancel
Help

Set permissions for selected presentation and marketing objects or for all objects originating from a physical connection. The permissions set here will impact the objects available for querying in the end-user reporting environment.

Type	Name	Read	Read/Write	No Access
	"Financials - AP Invoice Aging"	●	○	○
	"Financials - AP Overview"	●	○	○
	"Financials - AP Transactions"	●	○	○
	"Financials - AP Balance"	●	○	○
	"Financials - AP Holds"	●	○	○

Use the radio buttons (Read, Read Write, No Access) to set permissions for a Duty Role, on Oracle BI Repository objects.

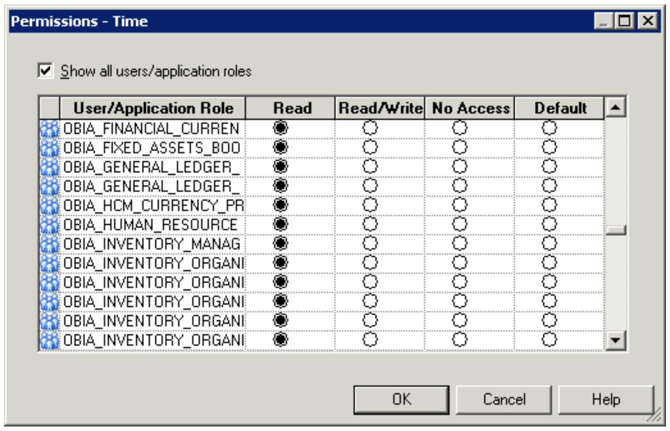
e. Click **OK** to save your changes.

f. Close the Identity Manager dialogs.

4. To configure permissions for Duty Roles on a subject area or a table.

   a. In the Presentation pane, double-click either a subject area icon (a cube), or expand a subject area, and double-click a presentation table to display the Permissions <*Subject Area name*>/<*Table name*> dialog for the chosen object.

   b. Click **Permissions** to display the Permissions <*Subject Area name*>/<*Table name*> dialog.



☑ Show all users/application roles

User/Application Role	Read	Read/Write	No Access	Default
OBIA_FINANCIAL_CURREN	●	○	○	○
OBIA_FIXED_ASSETS_BOO	●	○	○	○
OBIA_GENERAL_LEDGER_	●	○	○	○
OBIA_GENERAL_LEDGER_	●	○	○	○
OBIA_HCM_CURRENCY_PR	●	○	○	○
OBIA_HUMAN_RESOURCE	●	○	○	○
OBIA_INVENTORY_MANAG	●	○	○	○
OBIA_INVENTORY_ORGANI	●	○	○	○
OBIA_INVENTORY_ORGANI	●	○	○	○
OBIA_INVENTORY_ORGANI	●	○	○	○
OBIA_INVENTORY_ORGANI	●	○	○	○
OBIA_INVENTORY_ORGANI	●	○	○	○

c. Use the radio buttons (Read, Read/Write, No Access, and Default), to set permissions for Duty Roles on the selected Oracle BI Repository object.

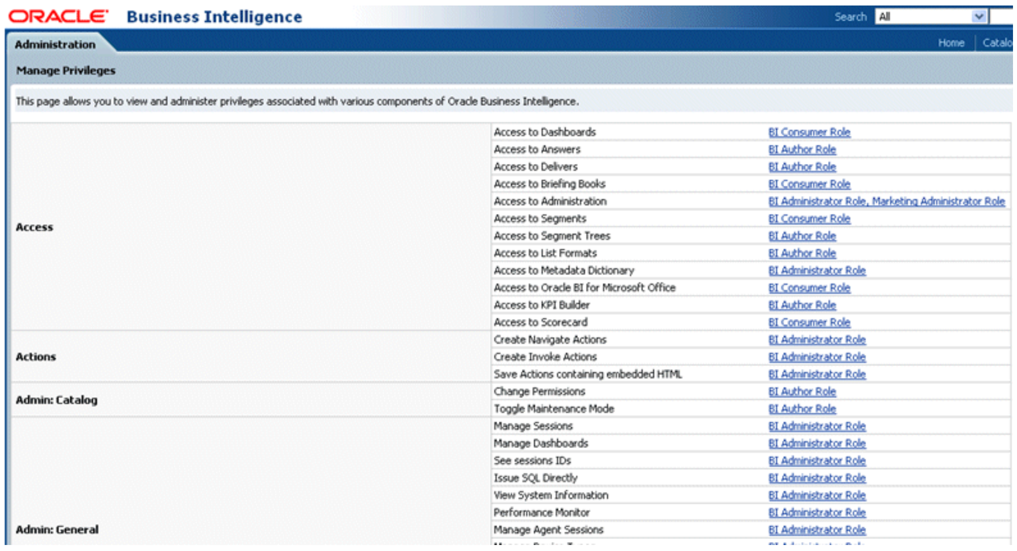
d. Click **OK** to save your changes.

e. Close the Permissions dialogs.

1.2.4 Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server

Oracle BI EE Administration Page enables you to configure Oracle BI Presentation Catalog Duty Role privileges. For example, for dashboard and other content. For more information, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

Figure 1–5 Oracle BI EE Administration Page



Manage Privileges

This page allows you to view and administer privileges associated with various components of Oracle Business Intelligence.

Access	Access to Dashboards	BI Consumer Role
	Access to Answers	BI Author Role
	Access to Delivers	BI Author Role
	Access to Briefing Books	BI Consumer Role
	Access to Administration	BI Administrator Role, Marketing Administrator Role
	Access to Segments	BI Consumer Role
	Access to Segment Trees	BI Author Role
	Access to List Formats	BI Author Role
	Access to Metadata Dictionary	BI Administrator Role
	Access to Oracle BI for Microsoft Office	BI Consumer Role
	Access to KPI Builder	BI Author Role
	Access to Scorecard	BI Consumer Role
Actions	Create Navigate Actions	BI Administrator Role
	Create Invoke Actions	BI Administrator Role
	Save Actions containing embedded HTML	BI Administrator Role
Admin: Catalog	Change Permissions	BI Author Role
	Toggle Maintenance Mode	BI Author Role
Admin: General	Manage Sessions	BI Administrator Role
	Manage Dashboards	BI Administrator Role
	See sessions IDs	BI Administrator Role
	Issue SQL Directly	BI Administrator Role
	View System Information	BI Administrator Role
	Performance Monitor	BI Administrator Role
	Manage Agent Sessions	BI Administrator Role

You can use this page to grant specific privileges, for example to:

- Manage Device Types
- Access a subject area within BI Answers (for example, Subject Area: "General Ledger - Balances Real Time")
- View a Dashboard Prompt

To configure Oracle BI Presentation Catalog Duty Role privileges:

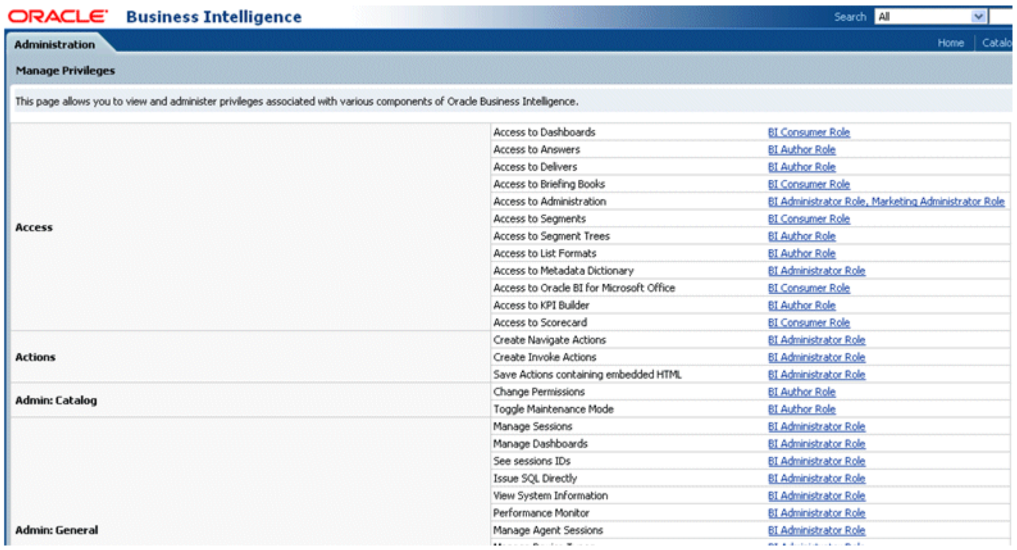
1. Log in to Oracle Business Intelligence with Administrator privileges.

   http://<*hostname*>:7001/analytics

2. Select the **Administration** link to display the Administration page.

3. Select the **Manage Privileges** link.

   The following screenshot shows components, privileges and associated roles.

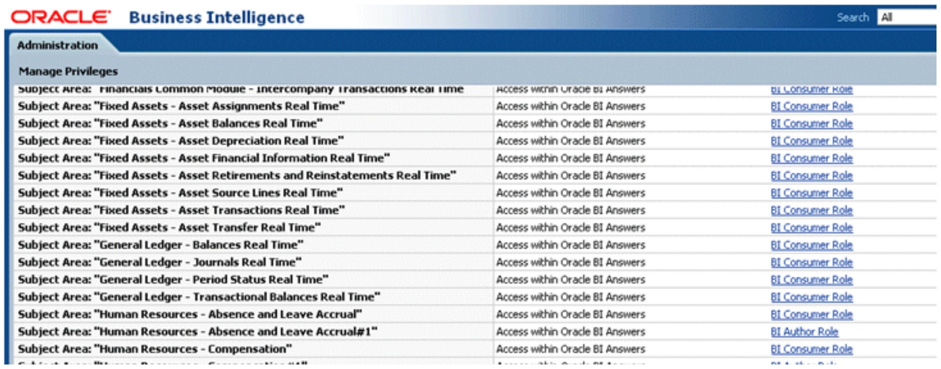


Manage Privileges

This page allows you to view and administer privileges associated with various components of Oracle Business Intelligence.

Access	Access to Dashboards	BI Consumer Role
	Access to Answers	BI Author Role
	Access to Delivers	BI Author Role
	Access to Briefing Books	BI Consumer Role
	Access to Administration	BI Administrator Role, Marketing Administrator Role
	Access to Segments	BI Consumer Role
	Access to Segment Trees	BI Author Role
	Access to List Formats	BI Author Role
	Access to Metadata Dictionary	BI Administrator Role
	Access to Oracle BI for Microsoft Office	BI Consumer Role
	Access to KPI Builder	BI Author Role
	Access to Scorecard	BI Consumer Role
Actions	Create Navigate Actions	BI Administrator Role
	Create Invoke Actions	BI Administrator Role
	Save Actions containing embedded HTML	BI Administrator Role
Admin: Catalog	Change Permissions	BI Author Role
	Toggle Maintenance Mode	BI Author Role
Admin: General	Manage Sessions	BI Administrator Role
	Manage Dashboards	BI Administrator Role
	See sessions IDs	BI Administrator Role
	Issue SQL Directly	BI Administrator Role
	View System Information	BI Administrator Role
	Performance Monitor	BI Administrator Role
	Manage Agent Sessions	BI Administrator Role

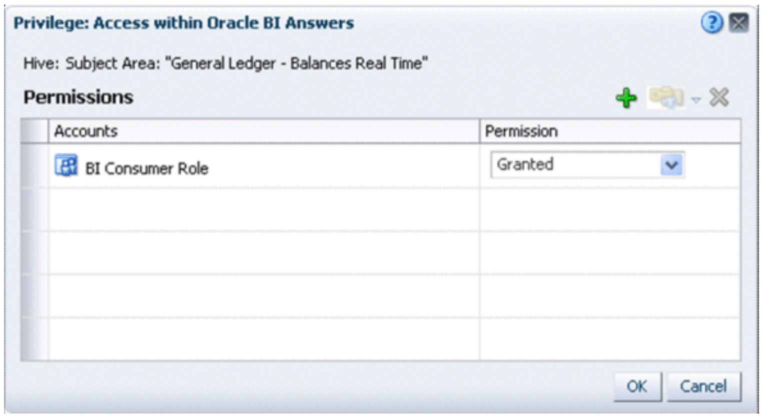
   The following screenshot shows another view of the Manage Privileges page with subject area folders, privileges and associated roles.



Manage Privileges

Subject Area: "Financials Common Module - Intercompany Transactions Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Assignments Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Balances Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Depreciation Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Financial Information Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Retirements and Reinstatements Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Source Lines Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Transactions Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Fixed Assets - Asset Transfer Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "General Ledger - Balances Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "General Ledger - Journals Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "General Ledger - Period Status Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "General Ledger - Transactional Balances Real Time"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Human Resources - Absence and Leave Accrual"	Access within Oracle BI Answers	BI Consumer Role
Subject Area: "Human Resources - Absence and Leave Accrual#1"	Access within Oracle BI Answers	BI Author Role
Subject Area: "Human Resources - Compensation"	Access within Oracle BI Answers	BI Consumer Role

4. Select a link to display the Privilege dialog, where you can grant or deny the privilege to the currently selected role.


Privilege: Access within Oracle BI Answers

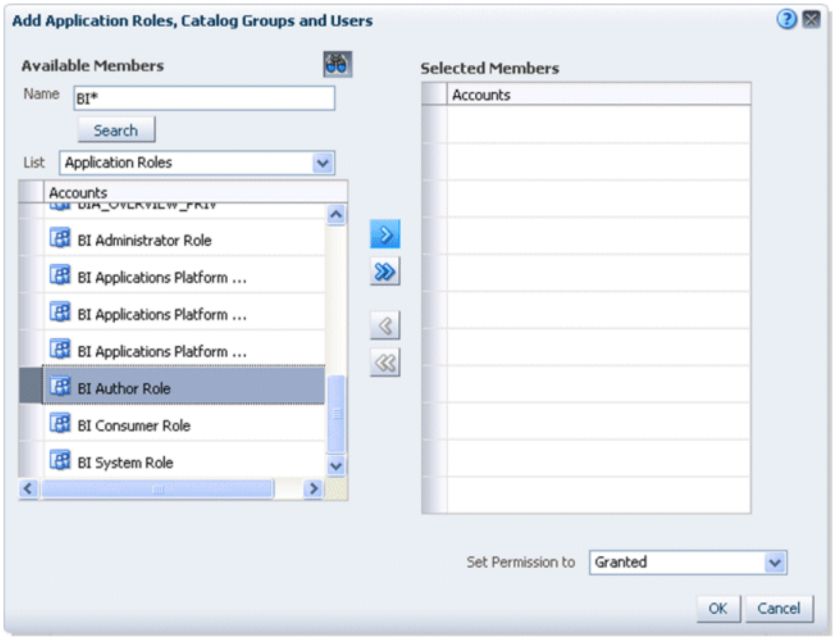
Hive: Subject Area: "General Ledger - Balances Real Time"

Permissions

Accounts	Permission
BI Consumer Role	Granted

5. Click the Add users/roles icon (+) to display the Add Application Roles, Catalog Groups, and Users dialog.

   The following screenshot shows the available Duty Roles, which can be assigned to this privilege.


Add Application Roles, Catalog Groups and Users

Available Members

Name: BI*

List: Application Roles

Accounts
BIA_OVERVIEW_PRIV
BI Administrator Role
BI Applications Platform ...
BI Applications Platform ...
BI Applications Platform ...
BI Author Role
BI Consumer Role
BI System Role

Selected Members

Accounts

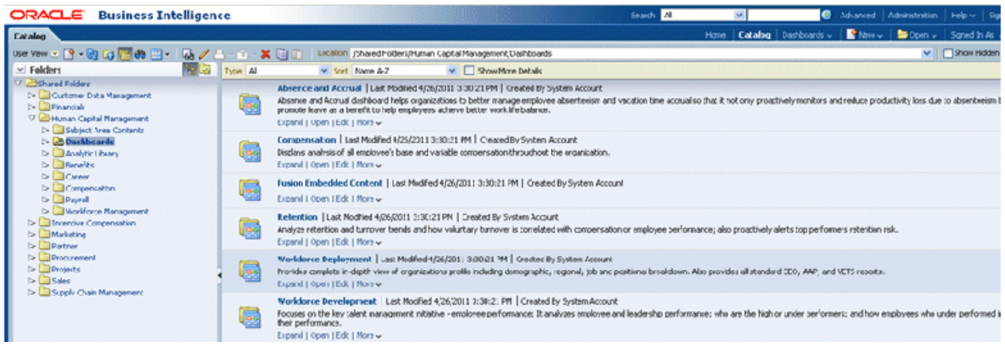
Set Permission to: Granted

6. To configure permissions for Duty Roles on functional dashboards and reports:

   a. In the top, click **Catalog** and expand Shared Folders in the left pane.

   b. Click the required pillar or functional area folder to open it.

      For example, select the Workforce Deployment dashboard under the Human Capital Management shared folder.


ORACLE Business Intelligence
Search All
Advanced | Administration | Help | Sign Out
Home | Catalog | Dashboards | New | Open | Signed In As
Catalog
User View
Location /Shared Folders/Human Capital Management/Dashboards
Show Hidden
Folders
Type All
Sort Name A-Z
Show More Details

- Shared Folders
  - Customer Data Management
  - Financials
  - Human Capital Management
    - Subject Area Contents
    - Dashboards
    - Analytic Library
    - Benefits
    - Career
    - Compensation
    - Payroll
    - Workforce Management
  - Incentive Compensation
  - Marketing
  - Partner
  - Procurement
  - Projects
  - Sales
  - Supply Chain Management

Absence and Accrual | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Absence and Accrual dashboard helps organizations to better manage employee absenteeism and vacation time accrual so that it not only proactively monitors and reduce productivity loss due to absenteeism b
promote leave as a benefit to help employees achieve better work life balance.
Expand | Open | Edit | More

Compensation | Last Modified 4/25/2011 3:30:21 PM | Created By System Account
Displays analysis of all employee's base and variable compensation throughout the organization.
Expand | Open | Edit | More

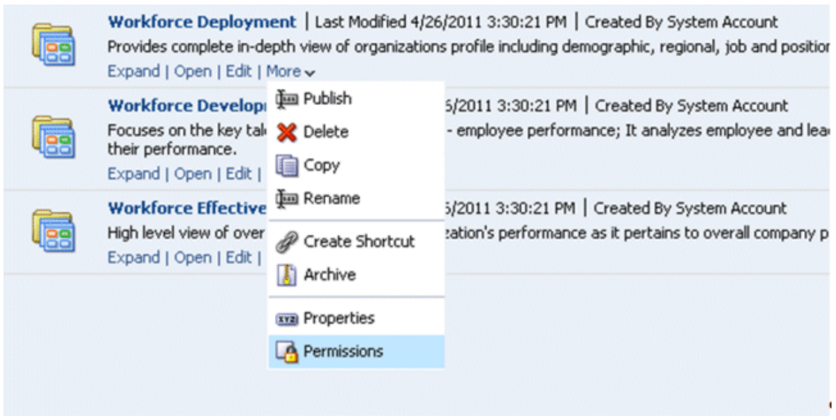
Fusion Embedded Content | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Expand | Open | Edit | More

Retention | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Analyze retention and turnover trends and how voluntary turnover is correlated with compensation or employee performance; also proactively alerts top performers retention risk.
Expand | Open | Edit | More

Workforce Deployment | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Provides complete in-depth view of organizations profile including demographic, regional, job and positional breakdown. Also provides all standard EEO, AAP, and VETS reports.
Expand | Open | Edit | More

Workforce Development | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Focuses on the key talent management initiative - employee performance; It analyzes employee and leadership performance; who are the high or under performers; and how employees who under performed i
their performance.
Expand | Open | Edit | More

c. Click the **More** link for the Workforce Deployment dashboard to open the Permissions dialog box.

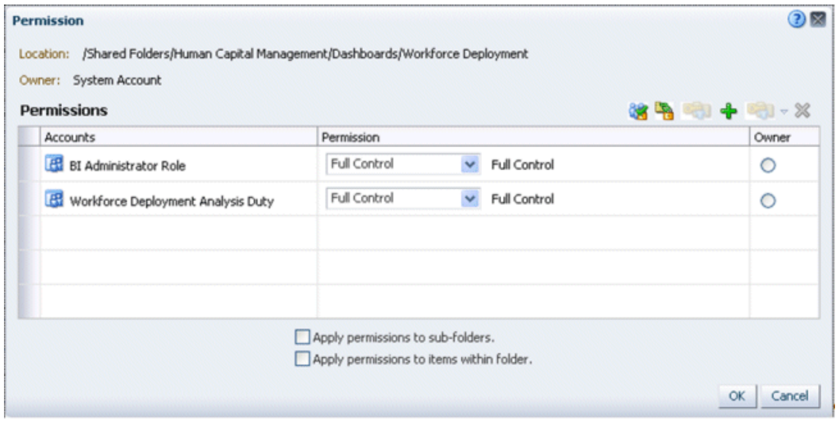


Workforce Deployment | Last Modified 4/26/2011 3:30:21 PM | Created By System Account
Provides complete in-depth view of organizations profile including demographic, regional, job and positior
Expand | Open | Edit | More

Workforce Developı ...5/2011 3:30:21 PM | Created By System Account
Focuses on the key tal... - employee performance; It analyzes employee and lea
their performance.
Expand | Open | Edit |

Workforce Effective ...5/2011 3:30:21 PM | Created By System Account
High level view of over... zation's performance as it pertains to overall company p
Expand | Open | Edit |

d. Use the Permission dialog box to add or modify role permissions for the Workforce Deployment dashboard.


Permission

Location: /Shared Folders/Human Capital Management/Dashboards/Workforce Deployment

Owner: System Account

Permissions

Accounts	Permission		Owner
BI Administrator Role	Full Control	Full Control	○
Workforce Deployment Analysis Duty	Full Control	Full Control	○

☐ Apply permissions to sub-folders.
☐ Apply permissions to items within folder.

Note: Similar steps can be taken to configure security for other objects in the Oracle BI Presentation Catalog.

1.3 Setting Up Security in Oracle BI Applications

This section describes how to set up security in Oracle BI Applications using the pre-configured Job Roles and Duty Roles that are installed. The setup process involves the following key tasks:

- Users and default Enterprise Roles are set up by the Fusion Applications provisioning process.

  If you want to change the default Users and Enterprise Roles in the LDAP directory, then follow the steps in Section 1.3.1, "Creating Users and Assigning Them to Job Roles".

- Default Duty Roles are installed with Oracle BI Applications, and are mapped by default to appropriate Enterprise Roles.

  If you want to change the default mappings for Duty Roles and Enterprise Roles in the Role Catalog, then follow the steps in Section 1.3.2, "Assigning Enterprise Roles (or Job Roles) to Duty Roles".

After you have installed Oracle BI Applications, you typically evaluate the product using the pre-configured users and roles. This section also describes how to create and develop your own modifications iteratively to meet your business requirements.

This topic contains the following sections:

- Section 1.3.1, "Creating Users and Assigning Them to Job Roles"
- Section 1.3.2, "Assigning Enterprise Roles (or Job Roles) to Duty Roles"
- Section 1.3.3, "Creating Duty Roles"
- Section 1.3.4, "Granting Users Access to Oracle BI Applications Dashboards"
- Section 1.3.5, "Understanding Oracle BI Applications Job Role Hierarchies"
- Section 1.3.6, "Assigning Duty Role Permissions to Oracle BI Repository Subject Areas and Tables"
- Section 1.3.7, "Assigning a Data Security Filter to a Duty Role"
- Section 1.3.8, "Assigning Duty Role Privileges to Oracle BI Presentation Catalog Objects"

For information about configuring security for Oracle Business Intelligence, see the *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

1.3.1 Creating Users and Assigning Them to Job Roles

When you create a new user you must assign them to one or more Job Roles or Data Roles using Oracle Identity Management. The new user inherits appropriate privileges from the Duty Roles that are associated with the Job Role or Data Role. New users are automatically added to the Fusion Applications Identity Store.

For more information, see Section 1.3.5, "Understanding Oracle BI Applications Job Role Hierarchies".

For Fusion Business Intelligence, Oracle Business Intelligence authenticates against Fusion Applications Identity Store at runtime.

To create a new user and assign the user to a Job Role or Data Role:

1. Use Oracle Identity Management as described in Section 1.2.1, "Using Oracle Identity Management to Manage Users and Enterprise Roles".
2. Create a new user using Oracle Identity Management (OIM).

   For more information about creating users, and assigning to Job Roles and Data Roles), see "Role-Based Access Control" in *Oracle Fusion Applications Security Guide*.
3. Assign the new user to a Job Role or Data Role.

If the default Job Roles or Data Roles do not meet your business requirements, then create your own Job Roles or Data Roles.

For more information, see *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Identity Management (Oracle Fusion Applications Edition)*'

Note: When you create a new User (or Job Role), at a minimum you must assign membership to two roles:

- BIAuthor (an Oracle BI EE 'Application Role')

  - for Oracle BI EE access privileges

- OBIA_BUSINESS_INTELLIGENCE_APPLICATIONS_WORKER (an Oracle BI Applications 'Abstract Role')

  - for Oracle BI Applications VO access privileges, if the user or role is Oracle BI Applications only

---

1.3.2 Assigning Enterprise Roles (or Job Roles) to Duty Roles

Oracle BI Applications is installed with a set of pre-configured Duty Roles that are mapped by default to appropriate Fusion Applications Enterprise Roles. For example, for Oracle HR Analytics, the Duty Role 'Workforce Deployment Analysis Duty' (OBIA_WORKFORCE_DEPLOYMENT_ANALYSIS_DUTY) is mapped to the Enterprise Role PER_LINE_MANAGER_ABSTRACT.

If the default security settings meet your business needs, then you do not need to alter the mappings. If the default security settings do not meet your needs, then you might:

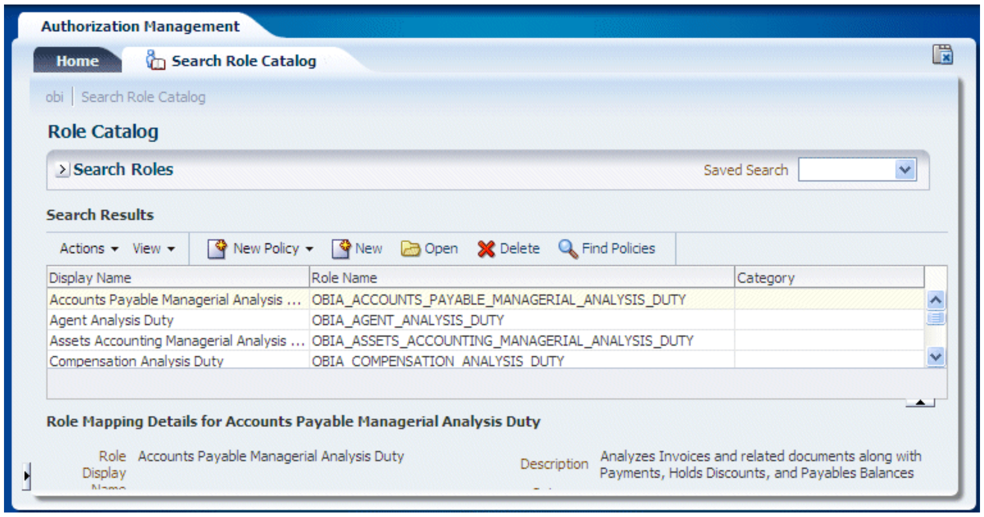
- edit the default Duty Roles and map them to different Enterprise Roles.
- create new Duty Roles and map them to Enterprise Roles.

Follow the steps below to map an Enterprise Role to a Duty Role.

Prerequisite: Before you start, the Enterprise Role must already exist in the LDAP directory, and the Duty Role must already exist in the Role Catalog. You can either use one of the pre-configured Enterprise Roles that are installed with Oracle BI Applications, or you can use OID to create your own Enterprise Roles.

To map an Enterprise Role to a Duty Role:

1. In Oracle APM, navigate to the 'obi' Application and use the Search options to locate the Duty Role that you want to map.



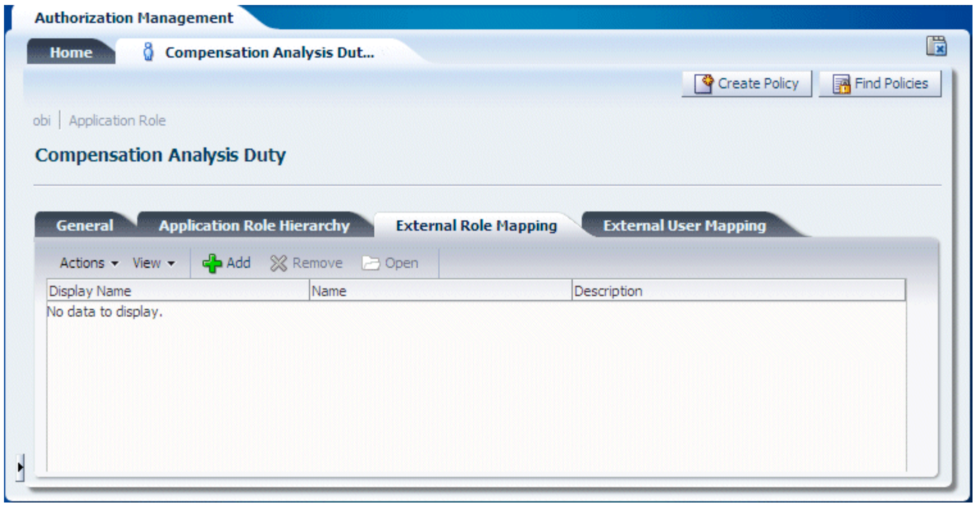
Display Name	Role Name	Category
Accounts Payable Managerial Analysis ...	OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY	
Agent Analysis Duty	OBIA_AGENT_ANALYSIS_DUTY	
Assets Accounting Managerial Analysis ...	OBIA_ASSETS_ACCOUNTING_MANAGERIAL_ANALYSIS_DUTY	
Compensation Analysis Duty	OBIA_COMPENSATION_ANALYSIS_DUTY	

Role Mapping Details for Accounts Payable Managerial Analysis Duty

Role Display Name: Accounts Payable Managerial Analysis Duty

Description: Analyzes Invoices and related documents along with Payments, Holds Discounts, and Payables Balances

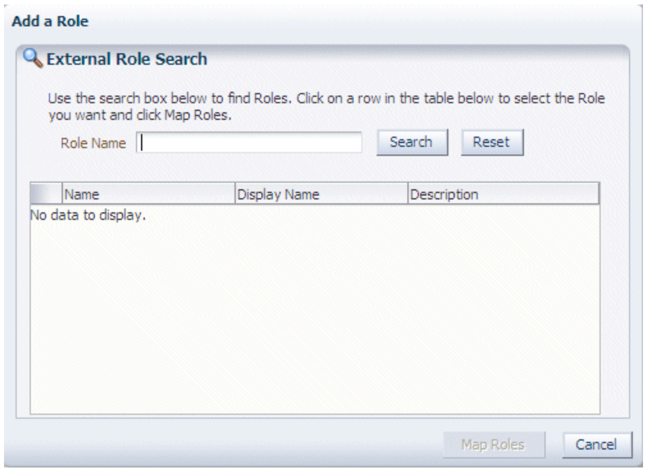
2. Select the Duty Role, then click Open to display the <*Application*> | Application Role dialog.
3. Display the External Role Mapping tab.



Display Name	Name	Description
No data to display.		

4. Use the External Role Mapping tab to search for and select the Enterprise Roles that you want to map to that Duty Role.

   Click Add to display the External Role Search dialog.



Use the search box below to find Roles. Click on a row in the table below to select the Role you want and click Map Roles.

	Name	Display Name	Description
	No data to display.		

For example, the OBIA_WORKFORCE_DEPLOYMENT_ANALYSIS_DUTY Duty Role must be mapped to the PER_LINE_MANAGER_ABSTRACT Enterprise Role and the PER_HUMAN_RESOURCES_VP_JOB Enterprise Role.

5. Click Map Roles to map the Enterprise Roles selected to the Duty Role, and return to the External Role Mapping tab.

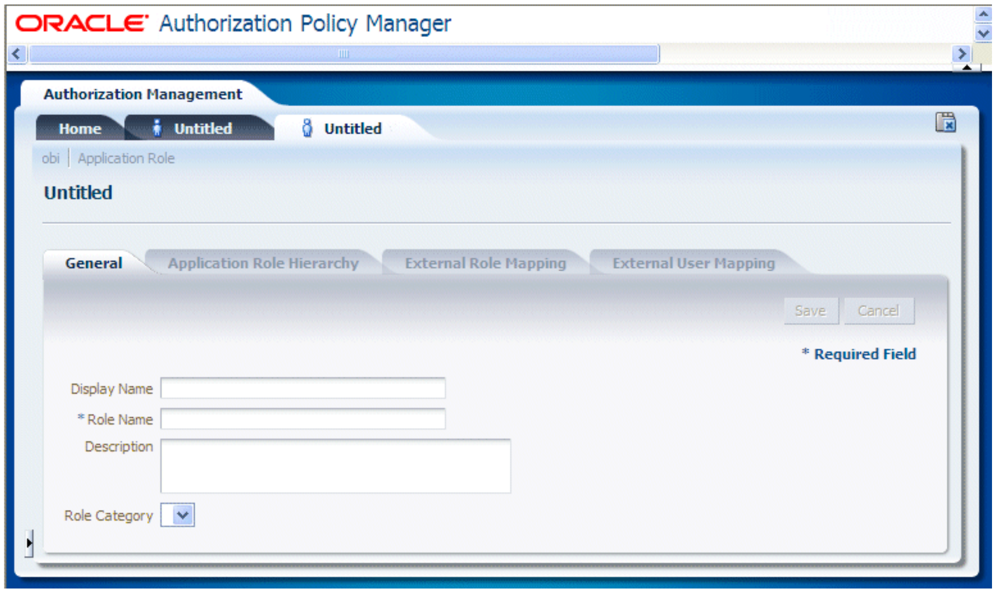
1.3.3 Creating Duty Roles

Oracle BI Applications is installed with a set of pre-configured Duty Roles. For example, for Oracle HR Analytics, the Duty Role 'Workforce Deployment Analysis Duty' (OBIA_WORKFORCE_DEPLOYMENT_ANALYSIS_DUTY) provides users with the required BI security privileges for that functional area.

If the default Duty Roles meet your business needs, then you do not need to change them. If the default Duty Roles do not meet your needs, then you might create new Duty Roles and map them to Job Roles.

Follow the steps below to create a Duty Role.

1. Log into Oracle APM, and select the 'obi' Application area.
2. In the Applications pane\Create area select the **New Application Role** link to display the <*Application*> | Application Role dialog.



3. In the General tab, use the **Display Name** field to specify the Duty Role name.
4. In the General tab, use the **Role Name** field to specify the Duty Role name in upper-case, with words separated with underscores, and prefixed with OBIA.

   For example, for the Duty Role named 'Workforce Deployment Analysis Duty', specify 'OBIA_WORKFORCE_DEPLOYMENT_ANALYSIS_DUTY'.

   Leave the **Role Category** field blank.
5. Click Save.

   The other tabs only become active after you click Save.

   At this point, you might want to map Job Roles to the new Duty Role now by following the steps in Section 1.3.2, "Assigning Enterprise Roles (or Job Roles) to Duty Roles", or you might want to map Job Roles to the new Duty Role at a later date.

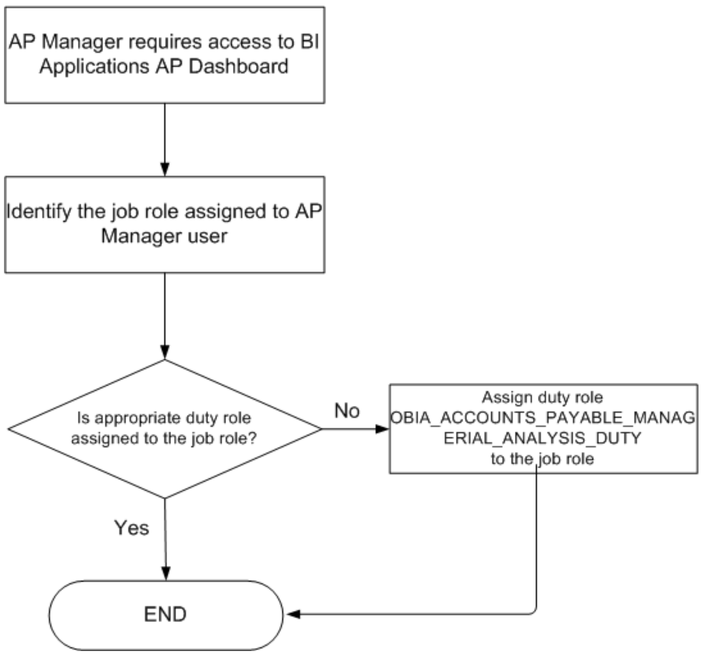
1.3.4 Granting Users Access to Oracle BI Applications Dashboards

Oracle BI Applications is installed with fully configured Enterprise Roles and Duty Roles that enable users to access BI dashboards. A user assigned to an Enterprise Role (also known as a Job Role) can access all BI dashboards and data that are appropriate to that role. The following examples illustrate how to provide access to BI dashboards using appropriate Enterprise Roles and Duty Roles.

The following examples describe granting a user access to a BI Applications dashboard:

- Example 1: Granting a user access to an Oracle BI Applications AP dashboard without changing the default security settings.

  In this scenario you would identify the appropriate Job Role assigned to the user, and check whether the appropriate Duty Role is assigned to that Job Role. If it is not, you assign the appropriate Duty Role to the Job Role to grant access to the Oracle BI Applications AP dashboard (for more information, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges").


AP Manager requires access to BI Applications AP Dashboard
Identify the job role assigned to AP Manager user
Is appropriate duty role assigned to the job role?
No
Assign duty role OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY to the job role
Yes
END

- Example 2: Granting a user access to an Accounts Payable (AP) dashboard by making changes to the installed security settings

  In this scenario you would identify the appropriate Job Role assigned to the user, and check whether the appropriate Duty Role is assigned to that Job Role, if it is not, you would assign the appropriate Duty Role to the AP Manager Job Role to grant access to the Oracle BI Applications AP dashboard. For more information, see Section 1.2.2, "Using Oracle Authorization Policy Manager to Configure Roles and Privileges".

  If suitable Oracle BI Repository permissions are not granted to a Duty Role, you can modify them for example, to enable or disable read/write permissions on a subject area or sub-folder. For more information, see:

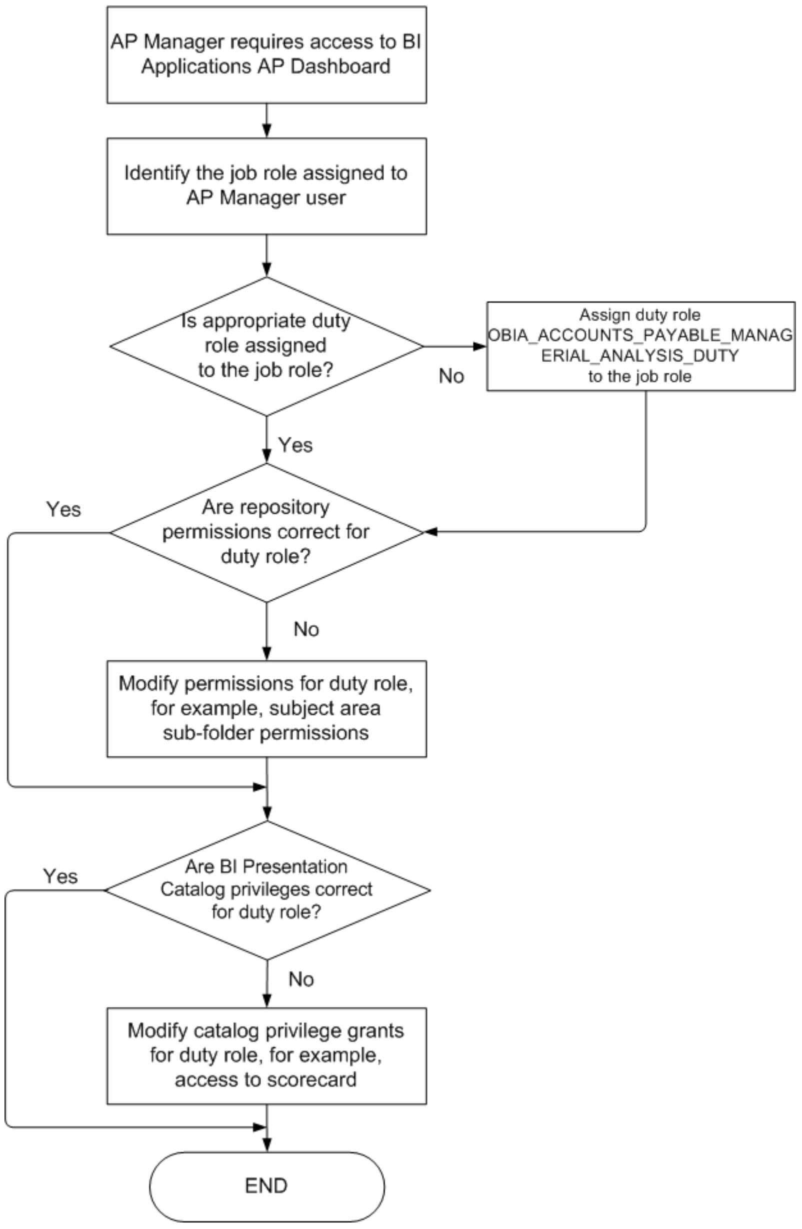
  – Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository"

  – Section 1.3.6, "Assigning Duty Role Permissions to Oracle BI Repository Subject Areas and Tables"

  If suitable Oracle BI Presentation Catalog privileges are not granted to a Duty Role, you can modify them for example, to enable or disable access to Dashboards, or view catalog objects. For more information, see:

  – Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server"

  – Section 1.3.8, "Assigning Duty Role Privileges to Oracle BI Presentation Catalog Objects"


AP Manager requires access to BI Applications AP Dashboard
Identify the job role assigned to AP Manager user
Is appropriate duty role assigned to the job role?
No
Assign duty role OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY to the job role
Yes
Yes
Are repository permissions correct for duty role?
No
Modify permissions for duty role, for example, subject area sub-folder permissions
Yes
Are BI Presentation Catalog privileges correct for duty role?
No
Modify catalog privilege grants for duty role, for example, access to scorecard
END

1.3.5 Understanding Oracle BI Applications Job Role Hierarchies

Oracle BI Applications provides pre-built Job Role hierarchies for security to enable access to Oracle BI EE.

When you create a new role you must comply with the pre-built role hierarchies and naming convention that applies to Oracle BI Applications and OLTP roles:

- Section 1.3.5.1, "Assigning Job Roles to Other Roles"
- Section 1.3.5.2, "Oracle BI Applications Pre-Built Role Hierarchy"

1.3.5.1 Assigning Job Roles to Other Roles

Every Job Role must be assigned to two types of roles:

- **Oracle BI Applications BI Duty Roles** - these provide:

    - access to one or more subject areas in the Oracle BI Repository and folders in Oracle BI Presentation Catalog
    - access to various Oracle BI EE platform permissions through inheritance of the BIAuthor role.

      For example, to run reports and ad-hoc queries.

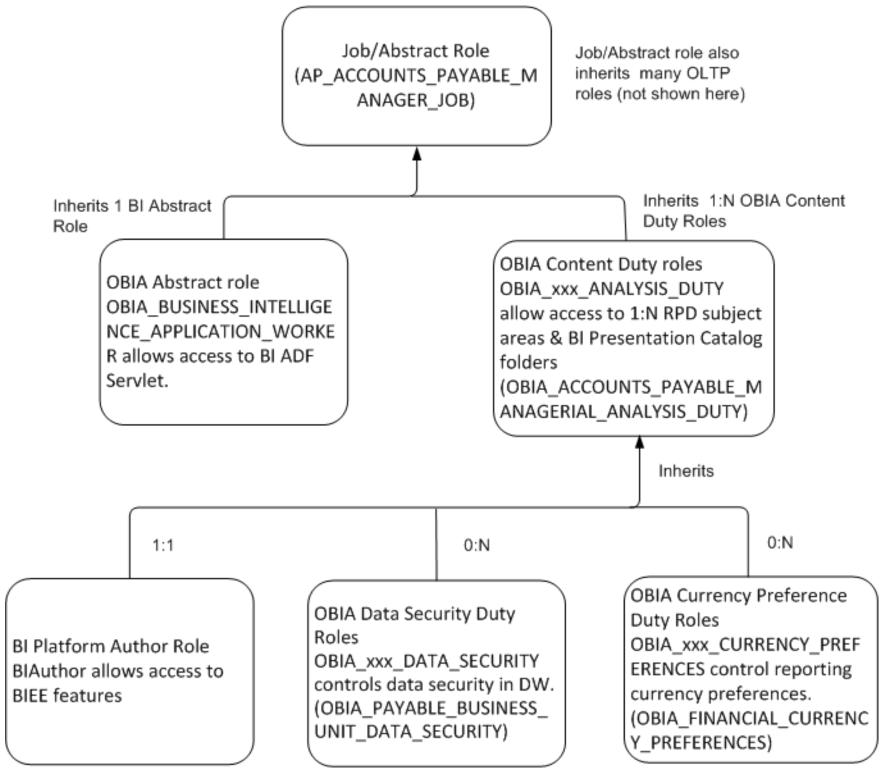
- **Oracle BI Applications BI Abstract Roles** - these provide access to Oracle Business Intelligence View Objects (VOs)

  For example, an Oracle BI Applications Job Role must be assigned to OBIA_BUSINESS_INTELLIGENCE_APPLICATIONS_WORKER Abstract Role for access to Oracle Business Intelligence View Objects.

1.3.5.2 Oracle BI Applications Pre-Built Role Hierarchy

Figure 1–6 illustrates the Oracle BI Applications pre-built role hierarchy and naming convention that applies to Job Roles and Abstract Roles.

Figure 1–6 Oracle BI Applications Pre-Built Role Hierarchy Sample


Job/Abstract Role (AP_ACCOUNTS_PAYABLE_MANAGER_JOB)
Job/Abstract role also inherits many OLTP roles (not shown here)
Inherits 1 BI Abstract Role
Inherits 1:N OBIA Content Duty Roles
OBIA Abstract role OBIA_BUSINESS_INTELLIGENCE_APPLICATION_WORKER allows access to BI ADF Servlet.
OBIA Content Duty roles OBIA_xxx_ANALYSIS_DUTY allow access to 1:N RPD subject areas & BI Presentation Catalog folders (OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY)
Inherits
1:1
0:N
0:N
BI Platform Author Role BIAuthor allows access to BIEE features
OBIA Data Security Duty Roles OBIA_xxx_DATA_SECURITY controls data security in DW. (OBIA_PAYABLE_BUSINESS_UNIT_DATA_SECURITY)
OBIA Currency Preference Duty Roles OBIA_xxx_CURRENCY_PREFERENCES control reporting currency preferences. (OBIA_FINANCIAL_CURRENCY_PREFERENCES)

Key to the Figure 1–6, "Oracle BI Applications Pre-Built Role Hierarchy Sample":

- **Job/Abstract Roles** (for example, AP_ACCOUNTS_PAYABLE_MANAGER_JOB) inherit the following roles:
    - **OLTP roles** (not shown)

      There will be many inherited OLTP roles.
    - **Oracle BI Applications Content Duty Roles** (OBIA_XXX_ANALYSIS_DUTY) - these allow access to one or more of the following:

    - Oracle BI Repository subject areas and tables
    - Oracle BI Presentation Catalog folders

    For example, OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY.

  - **Oracle BI Applications Abstract Role** - this enables access to the BI ADF Servlet

    Note: There is only one Oracle BI Applications Abstract Role - OBIA_BUSINESS_INTELLIGENCE_APPLICATION_WORKER.

- **Oracle BI Applications Content Duty Roles** (for example, OBIA_ACCOUNTS_PAYABLE_MANAGERIAL_ANALYSIS_DUTY) inherit the following roles:
  - **BIAuthor** Role - this allows access to Oracle BI EE features
  - **Oracle BI Applications Data Security Duty Roles** (OBIA_XXX_DATA_SECURITY) - these control data security in the Data Warehouse (DW)

    For example, OBIA_PAYABLE_BUSINESS_UNIT_DATA_SECURITY.
  - **Oracle BI Applications Currency Preference Duty Roles** (OBIA_XXX_CURRENCY_PREFERENCES) - control reporting financial currency preferences

    For example, OBIA_FINANCIAL_CURRENCY_PREFERENCES.

1.3.6 Assigning Duty Role Permissions to Oracle BI Repository Subject Areas and Tables

You grant Duty Role permissions to a Oracle BI Repository subject area or table to enable users associated with different Duty Roles to be granted different permissions in repository subject areas, and tables.

To assign Duty Role permissions to Oracle BI Repository subject areas and tables:

Edit the Oracle BI Repository, and set up permissions for a Duty Role as described in Section 1.2.3, "Using Administration Tool to Configure Duty Role Permissions in the Oracle BI Repository".

1.3.7 Assigning a Data Security Filter to a Duty Role

You assign a data security filter to a Duty Role to enable users associated with that Duty Role to view different data from a user associated with another Duty Role.

For more information, see Section 1.7.2, "Implementing Data-Level Security in the Oracle BI Repository"

1.3.8 Assigning Duty Role Privileges to Oracle BI Presentation Catalog Objects

You assign Duty Role privileges to Oracle BI Presentation Catalog objects to enable users associated with that Duty Role to be granted specific query permissions (for example, dashboard and other content).

To assign Duty Role Privileges to Oracle BI Presentation Catalog objects:

For more information, see Section 1.2.4, "Using the Oracle BI EE Administration Page to Configure Duty Role Privileges in the Oracle BI EE Presentation Server".

1.4 Configuring SSL for Oracle BI Applications

Secure Sockets Layer (SSL) for Oracle BI Applications enables secure communication between its components. Oracle BI Applications SSL is an extension of the Oracle BI EE platform SSL (SSL Everywhere), and must be manually configured after an Oracle BI Applications installation.

This topic references topics in other books that explain how to configure SSL for the Oracle BI EE platform and Oracle BI Applications, and contains the following sections:

- Section 1.4.1, "Configuring SSL for the Oracle BI EE Platform"
- Section 1.4.2, "Configuring SSL for Oracle BI Applications"

1.4.1 Configuring SSL for the Oracle BI EE Platform

You must configure SSL for the Oracle BI EE platform components before you can configure SSL for Oracle BI Applications components.

For information about configuring SSL for the Oracle BI EE platform components, see *Oracle Fusion Middleware Security Guide for Oracle Business Intelligence Enterprise Edition*.

1.4.2 Configuring SSL for Oracle BI Applications

If Oracle Content Server and the WebCenter application in which you intend to create a repository connection are not on the same system or the same trusted private network, then identity propagation is not secure. To ensure secure identity propagation you must also configure SSL on Oracle Content Server.

For more information, see Securing the WebCenter Spaces Connection to Oracle Content Server with SSL in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.*

You must also secure the connection between Oracle BI Applications Configuration Manager and Functional Setup Manager, as WebCenter portlet provider and consumer.

For more information, see Securing the WebCenter Spaces Connection to Portlet Producers with SSL in *Oracle Fusion Middleware Administrator's Guide for Oracle WebCenter.*

1.5 Mapping Roles to Secure Objects in the Oracle BI Repository and Oracle BI Presentation Catalog

For information about default mappings for Duty Roles, Job Roles, and Data Roles, to secure objects in the Oracle BI Repository and Oracle BI Presentation Catalog, see Document 1333454.1 *Oracle Business Intelligence Applications Duty Role Assignments for Fusion Applications* on My Oracle Support:

`https://support.oracle.com`.

1.6 Implementing GL Segment Security in Oracle BI Applications for Fusion Adaptor

This topic describes how to implement GL segment security in Oracle BI Applications with a Fusion Applications source system, and contains the following sections:

- Section 1.6.1, "Introduction"

- Section 1.6.2, "Configuring GL Segment Security"

1.6.1 Introduction

Oracle Financial Analytics supports a combination of the following security mechanisms for GL subject areas:

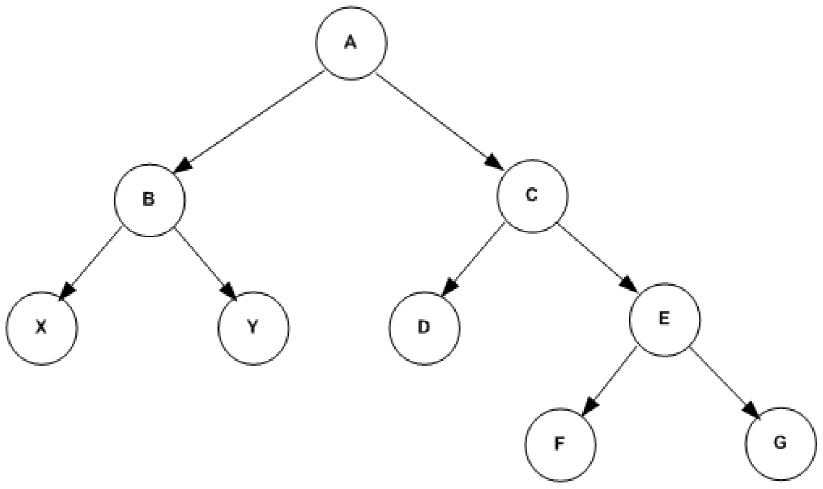
- Security using GL Data Access Sets
- Security using GL Accounting Segments

Data Access Set security is configured during installation and does not require additional configuration. This section gives an overview of the segment security, and describes how to configure security using GL Accounting Segments.

One or more value sets define the accounting segments in your OLTP. You can set up these value sets as a tree value set or non-tree value set. Users can have different types of access for each value set:

- NOACCESS - User has access to none of the values in that value set.
- FULLACCESS - User has access to all the values in that value set.
- FILTEREDACCESS - User has access to specific values in that value set, defined as follows:
  - Tree valueset: If the valueset has a tree, then access to the user can be granted using "is-descendant of" hierarchical operator. This means that the user has access to that node and all the descendants of that node within that value set.

    For example in the following illustration, if the user is granted "is-descendant of" node C, then the user has access to nodes C, D, E, F and G.



  - Non-tree valueset: If the valueset does not have a tree, then the user can be granted access to specific node/s or a range of nodes

1.6.2 Configuring GL Segment Security

Prior to configuring the segment security in the Oracle BI Repository, you should have completed configuring the segment dimensions in the Oracle BI Repository by mapping the segment VOs to the appropriate logical dimensions using BI Extender. Then perform the following tasks for each of the segment that you are securing. Based on the value sets used for those segments, the segment can be a tree enabled segment or a non-tree segment. The security implementation is different for these cases.

- Section 1.6.2.1, "Tree Segment Security Implementation"
- Section 1.6.2.2, "Non-Tree Segment Security Implementation"

1.6.2.1 Tree Segment Security Implementation

Perform the following steps when the segment on which security to be applied is a tree-based segment.

Task 1 Define Initialization Blocks and Session Variables

1. For tree-based value sets, the data security VO "FscmTopModelAM.DataSecurityAM.KFFHierFilter1" will give the different access types for the user as mentioned in the previous section. You will need to create a row wise session initialization block which reads from this VO. A sample SQL for this initialization block is as follows.

   ```
   SET variable DISABLE_SQL_BYPASS=1, ApplicationIdBind='101',
   KeyFlexfieldCodeBind='GL#', SegmentLabelCodeBind='FA_COST_CTR': SELECT DISTINCT
   'COST_CENTER_'||AccessType, CASE WHEN AccessType = 'FULLACCESS' THEN
   ValueSetCode ELSE ValueSetCode||'~'||TreeCode||'~'||TreeNodePk1Value END FROM
   "oracle.apps.fscm.model.analytics.applicationModule.FscmTopModelAM_
   FscmTopModelAMLocal"..."FscmTopModelAM.DataSecurityAM.KFFHierFilter1"
   ```

   Turn ON the "Allow deferred execution" option for this initialization block.

   Use the appropriate segment label code for the particular segment and any suitable prefix for the variable name, which are highlighted in bold text. In the above example, the segment label code used is "FA_COST_CTR" and the variable prefix used is "COST_CENTER_". This SQL will give (a) the value set codes the user has been granted full access to and/or (b) specific parent nodes within a tree the user has been granted access to using "is-descendant of" operator.

2. Create two session variables for the initialization block with names <*prefix*>_ FULLACCESS and <*prefix*>_FILTEREDACCESS, where <*prefix*> is the variable prefix used in the initialization block SQL. For example, in the above case you will define two session variables with the name COST_CENTER_FULLACCESS and COST_CENTER_FILTEREDACCESS. Default them with a value '-1' (Varchar).

3. When the user has filtered access, we need to determine the hierarchy level in the hierarchy/tree where the node falls. For this you will need to create another row wise session initialization block. A sample SQL for this would be as follows. You will need to use the FILTEREDACCESS variable created in the previous step.

   ```
   SELECT DISTINCT 'COST_CENTER_LEVELS', FIXED_HIER_LEVEL FROM "Oracle Data
   Warehouse"."Catalog"."dbo"."W_COST_CENTER_DH" WHERE LEVEL0_SECURITY_ID IN
   (VALUELISTOF(NQ_SESSION.COST_CENTER_FILTEREDACCESS)) AND CURRENT_FLG='Y'
   ```

   Turn ON the "Allow deferred execution" option for this initialization block.

   Please use "<*prefix*>_LEVELS" for the variable name in the select clause, where <*prefix*> is the same variable prefix used in steps 2 and 3. Please note the variable name used (in the where clause), should be the same as defined in the previous initialization block.

4. Create a session variable for the initialization block with the same name as used in the initialization block (COST_CENTER_LEVELS in this example) and default it with a value 0 (number). Set the execution precedence to make the initialization block mentioned in the previous step to run first.

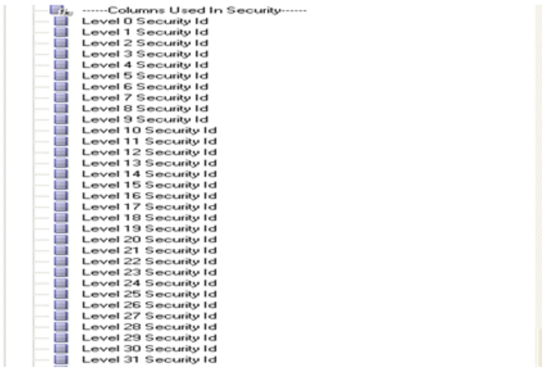
5. You can refer to the initialization blocks "Cost Center Security" and "Cost Center Security Top Node Levels" in the repository installed by default, as a reference to create the above two initialization blocks.

6. Repeat the previous steps for each of the segment to be secured, giving a different name for the two initialization blocks and the three session variables for each segment.

Task 2 Security id Expression in the logical dimensions

Each segment dimension in the Oracle BI Repository (Dim - Cost Center, Dim - Balancing Segment, Dim - Natural Account Segment and Dim - GL Segment 1-10) can be either a tree or non-tree segment based on your requirements. In case you have configured them to be tree segments, perform the steps below after creating the initialization blocks and variables mentioned in Task 1.

1. Each dimension has 32 security columns, Level 0 Security Id through Level 31 Security Id, as shown below. The expression for each of these logical columns needs to be modified using the hierarchy level variable created above.



```
------Columns Used In Security------
  Level 0 Security Id
  Level 1 Security Id
  Level 2 Security Id
  Level 3 Security Id
  Level 4 Security Id
  Level 5 Security Id
  Level 6 Security Id
  Level 7 Security Id
  Level 8 Security Id
  Level 9 Security Id
  Level 10 Security Id
  Level 11 Security Id
  Level 12 Security Id
  Level 13 Security Id
  Level 14 Security Id
  Level 15 Security Id
  Level 16 Security Id
  Level 17 Security Id
  Level 18 Security Id
  Level 19 Security Id
  Level 20 Security Id
  Level 21 Security Id
  Level 22 Security Id
  Level 23 Security Id
  Level 24 Security Id
  Level 25 Security Id
  Level 26 Security Id
  Level 27 Security Id
  Level 28 Security Id
  Level 29 Security Id
  Level 30 Security Id
  Level 31 Security Id
```

2. Open the logical table source of the dimension that maps to the warehouse dimension table and set the expression for each of these columns using the example from "Dim - Cost Center" dimension. For example, if you are securing by "Dim - GL Segment3" and the hierarchy level variable for this segment is "SEGMENT3_LEVELS", you would set the expression for each of the "Level <*n*> Security Id" column with the following:

```
INDEXCOL( IFNULL( VALUEOF(<n>, NQ_SESSION."SEGMENT3_LEVELS"),  VALUEOF(0, NQ_
SESSION."SEGMENT3_LEVELS")),
"Oracle Data Warehouse"."Catalog"."dbo"."Dim_W_GL_SEGMENT_DH_Security_
Segment3"."LEVEL31_SECURITY_ID",
"Oracle Data Warehouse"."Catalog"."dbo"."Dim_W_GL_SEGMENT_DH_Security_
Segment3"."LEVEL30_SECURITY_ID",
"Oracle Data Warehouse"."Catalog"."dbo"."Dim_W_GL_SEGMENT_DH_Security_
Segment3"."LEVEL29_SECURITY_ID",
…and so on for each security id column…
"Oracle Data Warehouse"."Catalog"."dbo"."Dim_W_GL_SEGMENT_DH_Security_
Segment3"."LEVEL1_SECURITY_ID",
"Oracle Data Warehouse"."Catalog"."dbo"."Dim_W_GL_SEGMENT_DH_Security_
Segment3"."LEVEL0_SECURITY_ID")
```

3. Repeat the above steps for each of the segment dimension to be secured.

Task 3 Security Filters in the Data Security Duty Roles

After completing Task 2, filters need to be added to the appropriate Data Role for data security predicates to be applied to queries. For more information, see Section 1.3.5, "Understanding Oracle BI Applications Job Role Hierarchies".

1. Navigate to "Manage -> Identity" from the menu.
2. Open the "OBIA_GENERAL_LEDGER_DATA_SECURITY" Duty Role.
3. Navigate to "Permissions -> Data Filters".

   For each of the logical facts secured under this role, you will see some existing filters, which are handling data access security. You will need to append the segment security filters to this with an 'AND' condition. A snippet of the segment security filters to be appended for a given segment dimension is given below, assuming the security is on "Dim - GL Segment3" and the session variable prefix used in the previous steps was "SEGMENT3".

```
(
"Core"."Dim - GL Segment3"."Segment Value Set Code" IS NULL OR
((
"Core"."Dim - GL Segment3"."Segment Value Set Code" = VALUEOF(NQ_
SESSION."SEGMENT3_FULLACCESS") OR
"Core"."Dim - GL Segment3"."Level 0 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 1 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 2 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
...and so on for each security id column...
"Core"."Dim - GL Segment3"."Level 30 Security Id"   = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 31 Security Id"   = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS")
)
AND
"Core"."Dim - GL Segment3"."Current Flag Security" = 'Y')
)
```

4. Repeat the above for each tree based segment dimension that is secured using appropriate variable names for each segment and appending each block of filters with an AND. For example, if you are securing by cost center and segment3 dimensions, the filter including the data access set security, will be as follows:

```
/* data access security filters */
 (
"Core"."Dim - GL Data Access Set Security"."Ledger List" = VALUEOF(NQ_
SESSION."LEDGER_LIST")
OR
"Core"."Dim - GL Data Access Set Security"."Ledger BSV List" = VALUEOF(NQ_
SESSION."LEDGER_BSV_LIST")
OR
"Core"."Dim - GL Data Access Set Security"."Ledger MSV List" = VALUEOF(NQ_
SESSION."LEDGER_MSV_LIST")
)
/* cost center segment security filters */
AND
 (
"Core"."Dim - Cost Center"."Cost Center Value Set Code" IS NULL OR
((
"Core"."Dim - Cost Center"."Cost Center Value Set Code" = VALUEOF(NQ_
SESSION."COST_CENTER_FULLACCESS") OR
"Core"."Dim - Cost Center"."Cost Center Level 0 Security Id"    = VALUEOF(NQ_
SESSION." COST_CENTER _FILTEREDACCESS") OR
"Core"."Dim - Cost Center"."Cost Center Level 1 Security Id"    = VALUEOF(NQ_
SESSION." COST_CENTER _FILTEREDACCESS") OR
"Core"."Dim - Cost Center"."Cost Center Level 2 Security Id"    = VALUEOF(NQ_
SESSION." COST_CENTER _FILTEREDACCESS") OR
...and so on for each security id column...
"Core"."Dim - Cost Center"."Cost Center Level 30 Security Id"   = VALUEOF(NQ_
SESSION." COST_CENTER _FILTEREDACCESS") OR
"Core"."Dim - Cost Center"."Cost Center Level 31 Security Id"   = VALUEOF(NQ_
SESSION." COST_CENTER _FILTEREDACCESS")
)
AND
"Core"."Dim - Cost Center"."Current Flag Security" = 'Y')
)
/* segment3 security filters */
AND
 (
"Core"."Dim - GL Segment3"."Segment Value Set Code" IS NULL OR
((
"Core"."Dim - GL Segment3"."Segment Value Set Code" = VALUEOF(NQ_
SESSION."SEGMENT3_FULLACCESS") OR
"Core"."Dim - GL Segment3"."Level 0 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 1 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 2 Security Id"    = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
...and so on for each security id column...
"Core"."Dim - GL Segment3"."Level 30 Security Id"   = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS") OR
"Core"."Dim - GL Segment3"."Level 31 Security Id"   = VALUEOF(NQ_
SESSION."SEGMENT3_FILTEREDACCESS")
)
AND
"Core"."Dim - GL Segment3"."Current Flag Security" = 'Y')
)
```

Note: When a tree has more than one version, the security filters are always applied on the current version for that tree (CURRENT_FLG='Y'). However, you can navigate through any other version of the tree in the reports but security will always be applied on the current version.

1.6.2.2 Non-Tree Segment Security Implementation

Perform the following steps when the segment on which security to be applied is not a tree based segment.

Task 1 Define Initialization Blocks and Session Variables

1. Determine the name of the VO that was generated for the segment. It will follow a naming pattern such as FLEX_VS_<*label*>_VI, where <*label*> is the segment label defined in the OLTP.
2. Create a session row wise initialization block reading from this VO.

A sample SQL statement might be:

```
SELECT 'GL_MANAGEMENT_FILTEREDACCESS', ValueSetCode||'~'||Value FROM
"oracle.apps.fscm.model.analytics.applicationModule.FscmTopModelAM_
FscmTopModelAMLocal"..."FscmTopModelAM.AccountBIAM.FLEX_VS_GL_MANAGEMENT2_VI"
```

Use an appropriate prefix for the variable name, highlighted above. This initialization block gives a concatenation of value set code and values the user has access to.

3. Create appropriate session variable with the same name as used above and default it with a value '-1' (Varchar). In the above example, the variable name is "GL_MANAGEMENT_FILTEREDACCESS".
4. Repeat the above steps for each non-tree segment that needs to be secured.

Task 2 Security Filters in the "Data Security" Data Roles

After you have completed Task 1, filters need to be added to the appropriate Data Role for data security predicates to be applied to queries.

1. Navigate to "Manage -> Identity" from the menu, and open the "OBIA_GENERAL_LEDGER_DATA_SECURITY" Data Role.
2. Navigate to "Permissions -> Data Filters", and for each of the logical facts secured under this role, append the following filter to any existing filters with an 'AND' condition. The sample filter will look like:

```
(
"Core"."Dim - GL Segment2"."Segment Value Set Code" IS NULL OR
"Core"."Dim - GL Segment2"."Segment Code Id"  = VALUEOF(NQ_SESSION."GL_
MANAGEMENT_FILTEREDACCESS")
)
```

3. Repeat the previous steps for each non-tree segment dimension that is secured using appropriate variable names for each segment and appending each block (one block per segment) with an "AND" condition. If you have a combination of non-tree and tree segments, then apply the data filters accordingly (as explained for each case) appending each filter with an 'AND' condition.

1.7 Advanced Security Topics - About Data-Level Security

Data-level security defines what a user with a particular Duty Role in an OLTP application can access inside a report. The same report, when run by two different users (associated with different Duty Roles), can return different data. This is similar to how the My Opportunities view in an operational application displays different data for different users. However, the structure of the report is the same for all users, unless a user does not have access to a column in a report, in which case the column is not displayed for that user.

Data-level security works by granting a privilege conditionally to a user using initialization blocks that determine what data is available on log in (for example, the hierarchy level in the organization hierarchy, or responsibilities of a role).

BI session variables store information specifying the rows that a logged in user is permitted to see, and are used inside initialization blocks. Session variable initialization blocks are initialized using ADF BC View Objects (VOs).

This topic contains the following sections:

- Section 1.7.1, "Securing Dimensions"

- Section 1.7.2, "Implementing Data-Level Security in the Oracle BI Repository"
- Section 1.7.3, "About Pre-Configured Initialization Blocks Used for Data-Level Security in Oracle BI Applications"
- Section 1.7.4, "About Data-Level Security Design in Oracle BI Applications"

For more information about:

- Oracle BI EE platform integration with View Objects (VOs)

  See 'Working with ADF Business Component Data Sources' in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*

1.7.1 Securing Dimensions

Dimensions in the Oracle BI Repository, can be secured using the following methods:

- In List

  This method obtains a list of rows that the logged in user is allowed to see. For example, for organization-based security, this method might obtain a list of organization IDs. This list is obtained when the user logs into Oracle Business Intelligence, and is stored in an Oracle Business Intelligence session variable.

- Top Node

  This method is used for a hierarchical dimension, and obtains the top node to which the user has access, and stores it in the session variable. When querying the warehouse facts, the cached top node can be used along with flattened hierarchy tables within the warehouse to apply security on Oracle BI Applications tables.

- Resource or Resource Hierarchy

  This method requires ETL to the data warehouse of all data in Sales Resource dimension, Sales Resource Hierarchy dimension, and (Fact - Sale Resource/Sales Resource Hierarchy) helper tables. When querying a fact table, the login user's Party ID (session variable USER_PARTY_ID, initialized by querying UserPVO) determines what rows in the Resource dimension, and/or Resource Hierarchy dimension, and/or the helper table that the user can access, which then determines what data the user can access in the fact table.

Note: Installed dimension security configuration for "In List" and "Top Node" is obtained from Fusion OLTP at runtime. If the dimension security gets into the data warehouse through ETL, then it is obtained from OLTP during ETL time, not runtime.

1.7.2 Implementing Data-Level Security in the Oracle BI Repository

Data-level security in Oracle BI Applications is implemented in three major steps, as described below. For detailed information about this security feature, see 'Applying Data Access Security to Repository Objects' *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

To implement data-level security in the Oracle BI Repository:

1. Set up initialization blocks that obtain specific security-related information when a user logs in, for example related to the user's responsibilities.
2. Set up the joins to the appropriate security tables in the metadata physical and logical layers.

For detailed information about this security feature, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

3. Set up the filters for each Duty Role on each logical table that needs to be secured.

   For detailed information about this security feature, see 'Setting Up Row-Level Security' in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

1.7.3 About Pre-Configured Initialization Blocks Used for Data-Level Security in Oracle BI Applications

Some initialization blocks for obtaining a given user's primary position, primary organization, and the owner ID, are preconfigured in the Oracle BI Repository. For more information, see the Oracle Business Intelligence Administration Tool.

For more information about setting up and managing initialization blocks, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

1.7.4 About Data-Level Security Design in Oracle BI Applications

As discussed in the preceding sections, Oracle BI Applications maintains data-level security Duty Roles that are assigned dynamically to every user at the session level. Each Duty Role has a set of filters associated with it that determines the data each user is allowed to see. A user is assigned a Duty Role through the Authorization initialization block, as discussed in Section 1.7.3, "About Pre-Configured Initialization Blocks Used for Data-Level Security in Oracle BI Applications."

The data security design has the following features:

- **Drill down.** The user can drill down on a particular position in the position hierarchy to slice the data by the next position level in the hierarchy. For example, if the initial report is defined as:

  ```
  select Top Level Position, Revenue from RevenueStar
  ```

  then by drilling down on a value of MyPosition in the TopLevelPosition hierarchy, the report will become:

  ```
  Select Level8 Position, Revenue, where TopLevelPosition = 'MyPosition'
  ```

- **Personalized reports.** Users at different levels of the Position hierarchy can use the same Position-based reports but with each user seeing the data corresponding to his or her level. In such reports, Position is a dynamic column.

  For example, if a report is defined as:

  ```
  select Position, Revenue from RevenueStar
  ```

  the logical query for the user at the top level of the hierarchy will be:

  ```
  select Top Level Position, Revenue from RevenueStar
  ```

  The logical query for the user at the next level of the hierarchy will be:

  ```
  select Level8 Position, Revenue from RevenueStar
  ```

- **CURRENT Position hierarchy columns.** Position hierarchy columns with the prefix CURRENT contain the Current Position hierarchy at any point of time. This feature allows users to see the same data associated with the employee holding the Current Employee position at the time the report runs. This type of Analysis is called As Is.

- **Additional Position hierarchy columns.** The columns EMP_LOGIN and EMPLOYEE_FULL_NAME are used at every level of the Position hierarchy to store additional information about an employee holding a particular position. In the Logical layer, the Employee path and Position path are two drill down paths under the Position hierarchy that allow the user to drill down on a position to see all positions under it. It also allows an employee to see all the employees reporting to him or her.

2 About Multi-Language Support

This chapter provides information about multi-language support in Oracle Business Intelligence Applications.

This chapter contains the following topics:

2.1 Introduction to Multi-Language Support

Oracle BI Applications provides multi-language support for metadata level objects exposed in Oracle BI Enterprise Edition dashboards and reports, as well as for data, which enables users to see records translated in their preferred language.

Configuring Base and Installed Data Warehouse Languages

After installing Oracle BI Applications, you use the Oracle BI Applications Configuration Manager (Configuration Manager) to configure which languages you want to support in the Oracle Business Analytics Warehouse. You must configure one "Base" language, and you can also configure any number of "Installed" languages. Typically, the Base language specified for the data warehouse should match the Base language of the source system. The Installed languages that you specify for the data warehouse do not have to match the languages that are installed in the source system. The data warehouse can have more, fewer, or completely different Installed languages compared to the source system. Note that for languages that match between the transactional system and the data warehouse, the corresponding record is extracted from the transactional system; languages that do not match will have a pseudo-translated record generated.

Note: You should only install the languages that you expect to use, because each installed language can significantly increase the number of records stored in the data warehouse and can affect overall database performance.

For information about how to configure data warehouse languages, see *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

Translation Tables

There are two types of translation tables: the Domains translation table and Dimension translation tables. There is a single Domain translation table which holds a translated value in each supported language for a domain. Dimension translation tables are extension tables associated with a given dimension. Depending on certain characteristics of a translatable attribute, it will be found in either the domain or a dimension translation table.

The user's session language is captured in an Oracle BI Enterprise Edition session variable named USER_LANGUAGE_CODE. This is set when users log in from Answers, where they select their preferred language. If users decide to change their preferred language in the middle of a session by using the Administration option to change the current language, this session variable will detect this change. Records returned from a translation table are filtered to those records with a LANGUAGE_CODE value that matches this session variable.

2.2 About Pseudo-Translations

The ETL process extracts translation records from the source system that correspond to the languages installed in the data warehouse. If a record cannot be found in the source system that corresponds to a language that has been installed in the data warehouse, a pseudo-translated record will be generated. Without a pseudo-translated record, a user that logs in with the missing language as their preferred language will not see any records.

A pseudo-translated record is generated by copying the translation record that corresponds to the data warehouse Base language and flagging it with the missing record's language by populating the LANGUAGE_CODE column with the language value. SRC_LANGUAGE_CODE stores the language from which the pseudo-translated record was generated; this will always match the data warehouse Base language.

In the future, if a translation record is created in the source system, it will be extracted and the pseudo-translated record will be overwritten to reflect the actual translated value. Table 2–1 provides an example in which "US" is the data warehouse Base language, and "IT" and "SP" are the Installed languages. The source system only had translated records for "US" and "IT" but did not have a translated record for "SP". The "US" and "IT" records are extracted and loaded into the data warehouse. Because there is no translation record in the source system for the "SP" language, a pseudo-translated record is generated by copying the "US" record and flagging LANGUAGE_CODE as if it were an "SP" record. The pseudo-translated record can be identified because SRC_LANGUAGE_CODE is different from LANGUAGE_CODE, matching the Base Language.

Table 2–1 Example of Pseudo-Translated Record

INTEGRATION_ID	NAME	LANGUAGE_CODE	SRC_LANGUAGE_CODE
ABC	Executive	US	US
ABC	Executive	IT	IT
ABC	Executive	SP	US

2.3 About Oracle BI Applications Domains

A domain refers to the possible, unique values of a table column in a relational database. In transactional systems, domains are often referred to as list of values (LOVs), which present attribute selections in the user's session language. The storage of the transaction is independent of the user's language; and, therefore, the field is stored using a language independent identifier. This identifier is typically a character code but can also be a numeric ID. The LOV or domain is then based on an ID-value pair, referred to as a member, and the LOV presents the values in the user's session language. At run time, the IDs are resolved to the value for the user's session language.

In the Oracle Business Analytics Warehouse, the number of unique values in any particular domain is relatively small and can have a low cardinality relative to the dimension it is associated with. For example, the Person dimension may have the domain 'Gender' associated with. The dimension may have millions of records, but the domain will generally have two or three members (M, F and possibly U). In the Oracle Business Analytics Warehouse, the Gender Code is stored in the Person dimension which acts as a foreign key to the Domains Translation table which stores the translated values. When a query is run, the user-friendly text associated with the code value is returned in the user's session language.

Depending on certain properties associated with a domain, domains can be configured in the Configuration Manager. In addition to serving as a mechanism for supporting translations, domains can be used to conform disparate source data into a common set of data.

Data Model

Oracle BI Applications domains are associated with dimensions as fields in the dimension table that follow the %_CODE naming convention. For example, the Person dimension W_PARTY_PER_D would store the Gender domain in the GENDER_CODE column.

Oracle BI Applications domains are stored in the domain translation table W_DOMAIN_MEMBER_LKP_TL. This table stores the translated values for each domain member code. Translated values are usually either a Name or a Description value which are stored in the NAME and DESCR columns of this table. The DOMAIN_MEMBER_CODE column acts as a key column when joining with the %_CODE column in the dimension table. As domains come from various systems, a DATASOURCE_NUM_ID column is used to identify which system the translated value comes from and is used as part of the join key with the dimension table. A LANGUAGE_CODE column is used to identify the language the translated values are associated with. Note that the LANGUAGE_CODE column follows the %_CODE naming convention. Language is considered a domain with a given set of unique values.

ETL Process

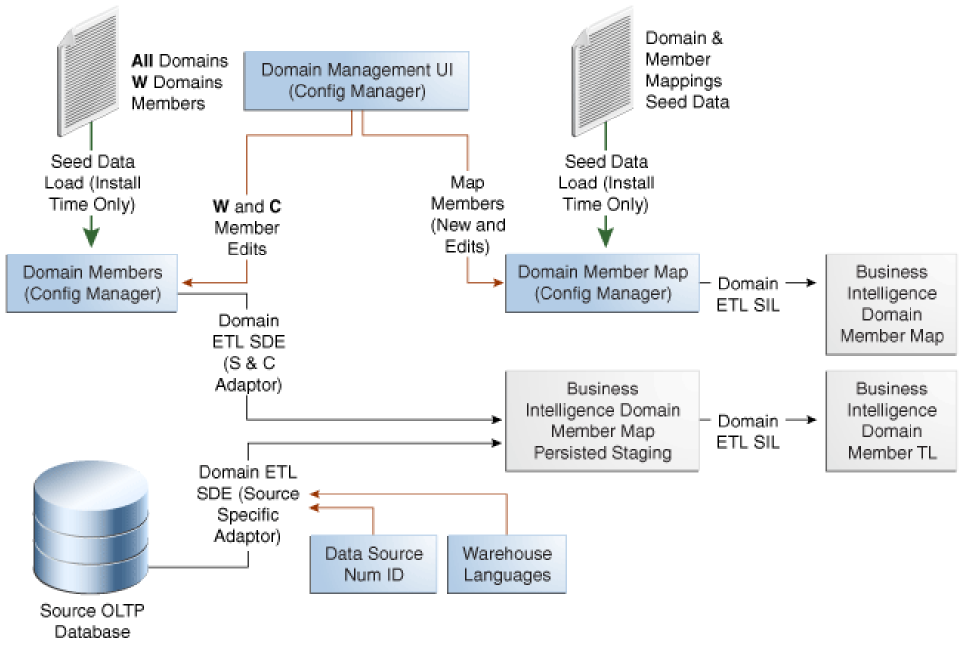
The W_DOMAIN_MEMBER_LKP_TL table stores both domains that are extracted from the source system as well as internally defined domains that are seeded in the Configuration Manager. For each of the %_CODE columns that have translated values available in the source system, an ETL process extracts the domain members from the transactional system and loads them into W_DOMAIN_MEMBER_LKP_TL. Internally defined domains—usually domains specific to the Oracle Business Analytics Warehouse and known as conformed domains but can also include source domains—are stored in the Configuration Manager schema and are similarly extracted and loaded into the W_DOMAIN_MEMBER_LKP_TL table through ETL processes.

Only those translation records that match one of the languages that have been installed in the data warehouse are extracted from the transactional system. If translated records are not found in the transactional system matching an installed language, the ETL will generate a 'pseudo-translated' record for that language.

Some source applications store translations that can be extracted and loaded into the translation table. Some source applications do not maintain translations for an entity that corresponds to a dimension table. In these cases, whatever record is available is extracted and used as the Base language record to generate pseudo-translations for all other installed languages.

Figure 2–1 shows an overview of the Oracle BI Applications domain ETL process.

Figure 2–1 Overview of BI Applications Domain ETL Process


All Domains
W Domains
Members
Domain Management UI (Config Manager)
Domain & Member Mappings Seed Data
Seed Data Load (Install Time Only)
W and **C** Member Edits
Map Members (New and Edits)
Seed Data Load (Install Time Only)
Domain Members (Config Manager)
Domain Member Map (Config Manager)
Domain ETL SIL
Business Intelligence Domain Member Map
Domain ETL SDE (S & C Adaptor)
Business Intelligence Domain Member Map Persisted Staging
Domain ETL SIL
Business Intelligence Domain Member TL
Domain ETL SDE (Source Specific Adaptor)
Data Source Num ID
Warehouse Languages
Source OLTP Database

About Oracle BI Applications Domains and Oracle BI Enterprise Edition

The exact mechanism used to retrieve the translated value in Oracle BI Enterprise Edition is the LOOKUP() function. When the LOOKUP() function is used, Oracle BI Enterprise Edition performs all aggregations before joining to the lookup table. The aggregated result set is then joined to the lookup table. Low-cardinality attributes tend to be involved in several aggregations, so it is useful to be joined after results are aggregated rather than before.

In a logical dimension, a Name or Description attribute will use the LOOKUP() function, passing the value in the %_CODE column associated with that Name or Description to the Domain Lookup Table. The LOOKUP() function includes the Domain Name to be used when looking up values. The results from the Domain Lookup table are filtered to match the user's session language and returned as part of the query results.

Domains can be either source or conformed (internally defined warehouse domains). Source domains can come from a variety of transactional systems and so must include a Datasource_Num_Id value to resolve. Conformed domains are defined as part of the Oracle BI Applications and do not require a Datasource_Num_ID to resolve. As a result, there are two lookup tables implemented in the Oracle BI Repository that are aliases of W_DOMAIN_MEMBER_LKP_TL. When resolving a source domain, the source domain lookup requires Datasource_Num_Id to be passed as part of the LOOKUP() function while the conformed domain lookup does not.

2.4 About Dimension Translation Tables

As mentioned in Section 2.3, "About Oracle BI Applications Domains,", domains are dimensional attributes that have a relatively small number of distinct members, have a low cardinality relative to the number of records in the dimension, and are often used in aggregations. Dimensions have other attributes that require translation that may not fit one or more of these criteria. Dimensions may have translatable attributes that have a high cardinality relative to the dimension or may have a large number of members, and, thus, are not likely candidates for aggregation. If the domains ETL process was implemented in such cases, performance would be very poor. As a result, these particular attributes are implemented using dimension translation tables.

Data Model

If a dimension has such high-cardinality attributes that cannot be treated as domains, the dimension will have an extension table that follows the _TL naming convention. If the _TL table has a one-to-one relationship with the dimension table (after filtering for languages), the _TL table name will match the dimension table name. For example, W_JOB_D_TL is the translation table associated with the W_JOB_D dimension table. If the _TL table does not have a one-to-one relationship with any dimension table, its name will reflect content.

The dimension and dimension translation table are joined on the translation table's INTEGRATION_ID + DATASOURCE_NUM_ID. If the translation and dimension tables have a one-to-one relationship (after filtering for language), the join to the dimension table is on its INTEGRATION_ID + DATASOURCE_NUM_ID. Otherwise, there will be a %_ID column in the dimension table that is used to join to the translation table.

ETL Process

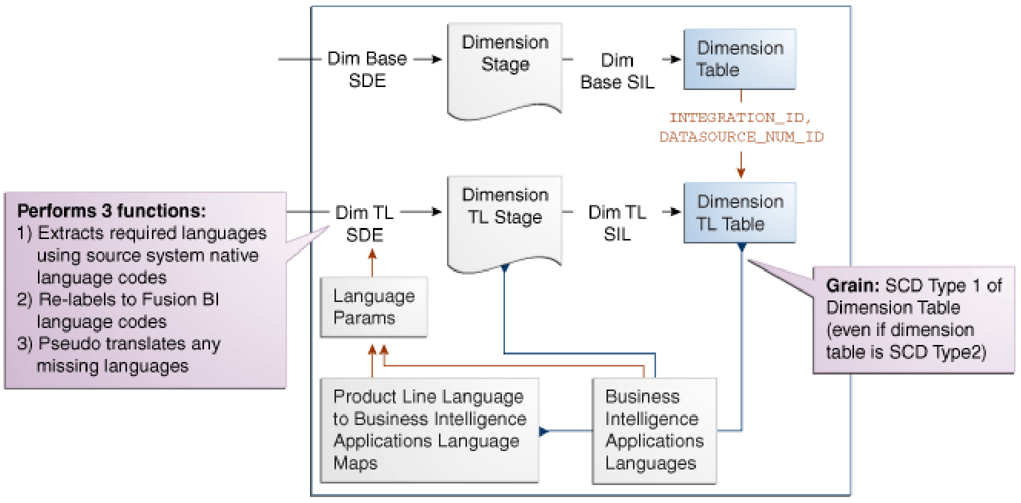
Similar to the Oracle BI Applications domain ETL process, when using dimension translation tables, ETL tasks extract the translated values from the transactional system. Rather than the domain staging table being loaded, the dimension's translation staging table is loaded. The ETL process then moves these records into the dimension translation table.

Only those translation records that match one of the languages that have been installed in the data warehouse are extracted from the transactional system. If translated records are not found in the transactional system matching a data warehouse Installed language, the ETL will generate a 'pseudo-translated' record for that language by copying the record that corresponds to the data warehouse Base language.

Some source applications store translations that can be extracted and loaded into the translation table. Some source applications do not maintain translations for an entity that corresponds to a dimension table. In these cases, whatever record is available is extracted and used as the Base language record, which is then used to generate pseudo-translations for all other Installed languages.

Oracle BI Applications does not support Type 2 SCD tracking of dimension translation attributes when the dimension and translation tables have a one-to-one relationship with each other. These tables are joined on INTEGRATION_ID + DATASOURCE_NUM_ID, and, therefore, can be joined to a single record in the translation table. Attributes in the dimension table can be Type 2-enabled, but the current and prior records will always have the same translated value. Figure 2–2 describes the ETL domain process.

Figure 2–2 Domain ETL Process


Dim Base SDE
Dimension Stage
Dim Base SIL
Dimension Table
INTEGRATION_ID, DATASOURCE_NUM_ID
Performs 3 functions:
1) Extracts required languages using source system native language codes
2) Re-labels to Fusion BI language codes
3) Pseudo translates any missing languages
Dim TL SDE
Dimension TL Stage
Dim TL SIL
Dimension TL Table
Language Params
Grain: SCD Type 1 of Dimension Table (even if dimension table is SCD Type2)
Product Line Language to Business Intelligence Applications Language Maps
Business Intelligence Applications Languages

Oracle BI Enterprise Edition

In Oracle BI Enterprise Edition, joins are created between the dimension and translation tables as normal. The translation table is brought in as another supporting table in the logical table source. If a user selects an attribute from the translation table, it will be included as a joined table in the SQL that Oracle BI Enterprise Edition generates. If the user does not select a translation attribute, the translation table will not be included in the generated SQL.

To ensure this behavior, the physical join between the dimension and translation tables is configured as one-to-many with the dimension table on the many side.

An important consideration is filtering on a user's language. If the language filter is included in the logical table source as a content filter, the translation table will always be joined whether a user selects a translation attribute or not. To avoid this behavior, opaque views are created in the physical layer that include a WHERE clause on the user's session language. Filtering on the user's language is still possible, but as the filter criteria is not implemented as a logical table source content filter, it is ensured that the translation table is only joined when necessary.

3 Oracle Business Analytics Warehouse Naming Conventions

This chapter includes information on the types of tables and columns in the Oracle Business Analytics Warehouse, including the naming conventions used.

Note: This chapter contains naming conventions used for database tables and columns in the Oracle Business Analytics Warehouse. This information does not apply to objects in the Oracle Business Intelligence repository.

This chapter contains the following topics:

- Section 3.1, "Naming Conventions for Oracle Business Analytics Warehouse Tables"
- Section 3.2, "Table Types for Oracle Business Analytics Warehouse"
- Section 3.3, "Standard Column Prefixes in Oracle Business Analytics Warehouse"
- Section 3.4, "Standard Column Suffixes in Oracle Business Analytics Warehouse"
- Section 3.5, "System Columns in Oracle Business Analytics Warehouse Tables"
- Section 3.6, "Multi-Currency Support for System Columns"
- Section 3.7, "Oracle Business Analytics Warehouse Primary Data Values"
- Section 3.8, "About Multi-Language Support in the Oracle Business Analytics Warehouse"
- Section 3.9, "Oracle Business Analytics Warehouse Currency Preferences"

3.1 Naming Conventions for Oracle Business Analytics Warehouse Tables

Oracle Business Analytics Warehouse tables use a three-part naming convention: PREFIX_NAME_SUFFIX, as shown in Table 3–1.

Table 3–1 Naming Conventions for Oracle Business Analytics Data Warehouse Tables

Part	Meaning	Table Type
PREFIX	Shows Oracle Business Analytics-specific data warehouse application tables.	W_ = Warehouse

Table 3–1 (Cont.) Naming Conventions for Oracle Business Analytics Data Warehouse

Part	Meaning	Table Type
NAME	Unique table name.	All tables.

SUFFIX	Indicates the table type.	_A = Aggregate
		_D = Dimension
		_DEL = Delete
		_DH = Dimension Hierarchy
		_DHL = Dimension Helper
		_DHLS = Staging for Dimension Helper
		_DHS = Staging for Dimension Hierarchy
		_DS = Staging for Dimension
		_F = Fact
		_FS = Staging for Fact
		_G, _GS = Internal
		_H = Helper
		_HS = Staging for Helper
		_MD = Mini Dimension
		_PE = Primary Extract
		_PS = Persisted Staging
		_RH = Row Flattened Hierarchy
		_TL = Translation Staging (supports multi-language support)
		_TMP = Pre-staging or post-staging temporary table
		_UD = Unbounded Dimension
		_WS = Staging for Usage Accelerator

3.2 Table Types for Oracle Business Analytics Warehouse

Table 3–2 lists the types of tables used in the Oracle Business Analytics Warehouse.

Table 3–2 Table Types Used in the Oracle Business Analytics Warehouse

Table Type	Description
Aggregate tables (_A)	Contain summed (aggregated) data.
Dimension tables (_D)	Star analysis dimensions.
Delete tables (_DEL)	Tables that store IDs of the entities that were physically deleted from the source system and should be flagged as deleted from the data warehouse. Note that there are two types of delete tables: _DEL and _PE. For more information about the _PE table type, see the row for Primary extract tables (_PE) in this table.
Dimension Hierarchy tables (_DH)	Tables that store the dimension's hierarchical structure.
Dimension Helper tables (_DHL)	Tables that store many-to-many relationships between two joining dimension tables.
Staging tables for Dimension Helper (_DHLS)	Staging tables for storing many-to-many relationships between two joining dimension tables.
Dimension Hierarchy Staging table (_DHS)	Staging tables for storing the hierarchy structures of dimensions that have not been through the final extract-transform-load (ETL) transformations.
Dimension Staging tables (_DS)	Tables used to hold information about dimensions that have not been through the final ETL transformations.
Fact tables (_F)	Contain the metrics being analyzed by dimensions.

Table 3–2 (Cont.) Table Types Used in the Oracle Business Analytics Warehouse

Table Type	**Description**
Fact Staging tables (_FS)	Staging tables used to hold the metrics being analyzed by dimensions that have not been through the final ETL transformations.
Internal tables (_G, _GS)	General tables used to support ETL processing.
Helper tables (_H)	Inserted between the fact and dimension tables to support a many-to-many relationship between fact and dimension records.
Helper Staging tables (_HS)	Tables used to hold information about helper tables that have not been through the final ETL transformations.
Mini dimension tables (_MD)	Include combinations of the most queried attributes of their parent dimensions. The database joins these small tables to the fact tables.
Primary extract tables (_PE)	Tables used to support the soft delete feature. The table includes all the primary key columns (integration ID column) from the source system. When a delete event happens, the full extract from the source compares the data previously extracted in the primary extract table to determine if a physical deletion was done in the Siebel application. The soft delete feature is disabled by default. Therefore, the primary extract tables are not populated until you enable the soft delete feature. Note that there are two types of delete tables: _DEL and _PE. For more information about the _DEL table type, see the row for Delete table (_DEL) in this table.
Persisted Staging table (_PS)	Tables that source multiple data extracts from the same source table. These tables perform some common transformations required by multiple target objects. They also simplify the source object to a form that is consumable by the warehouse needed for multiple target objects. These tables are never truncated during the life of the data warehouse. These are truncated only during full load, and therefore, persist the data throughout.
Row Flattened Hierarchy Table (_RH)	Tables that record a node in the hierarchy by a set of ancestor-child relationships (parent-child for all parent levels).
Translation Staging tables (_TL)	Tables store names and descriptions in the languages supported by Oracle BI Applications.
Pre-staging or post-staging Temporary table (_TMP)	Source-specific tables used as part of the ETL processes to conform the data to fit the universal staging tables (table types_DS and _FS). These tables contain intermediate results that are created as part of the conforming process.
Unbounded dimension (_UD)	Tables containing information that is not bounded in transactional database data but should be treated as bounded data in the Oracle Business Analytics Warehouse.
Staging tables for Usage Accelerator (_WS)	Tables containing the necessary columns for the ETL transformations.

3.2.1 Aggregate Tables in Oracle Business Analytics Warehouse

One of the main uses of a data warehouse is to sum up fact data with respect to a given dimension, for example, by date or by sales region. Performing this summation on-demand is resource-intensive, and slows down response time. The Oracle Business Analytics Warehouse precalculates some of these sums and stores the information in *aggregate tables*. In the Oracle Business Analytics Warehouse, the aggregate tables have been suffixed with _A.

3.2.2 Dimension Class Tables in Oracle Business Analytics Warehouse

A class table is a single physical table that can store multiple logical entities that have similar business attributes. Various logical dimensions are separated by a separator column, such as, type or category. W_XACT_TYPE_D is an example of a dimension class table. Different transaction types, such as, sales order types, sales invoice types, purchase order types, and so on, can be housed in the same physical table.

You can add additional transaction types to an existing physical table and so reduce the effort of designing and maintaining new physical tables. However, while doing so, you should consider that attributes specific to a particular logical dimension cannot be defined in this physical table. Also, if a particular logical dimension has a large number of records, it might be a good design practice to define a separate physical table for that particular logical entity.

3.2.3 Dimension Tables in Oracle Business Analytics Warehouse

The unique numeric key (ROW_WID) for each dimension table is generated during the load process. This key is used to join each dimension table with its corresponding fact table or tables. It is also used to join the dimension with any associated hierarchy table or extension table. The ROW_WID columns in the Oracle Business Analytics Warehouse tables are numeric. In every dimension table, the ROW_WID value of zero is reserved for Unspecified. If one or more dimensions for a given record in a fact table is unspecified, the corresponding key fields in that record are set to zero.

3.2.4 Dimension Tables With Business Role-Based Flags

This design approach is used when the entity is logically the same but participates as different roles in the business process. As an example, an employee could participate in a Human Resources business process as an employee, in the sales process as a sales representative, in the receivables process as a collector, and in the purchase process as a buyer. However, the employee is still the same. For such logical entities, flags have been provided in the corresponding physical table (for example, W_EMPLOYEE_D) to describe the record's participation in business as different roles.

While configuring the presentation layer, the same physical table can be used as a specific logical entity by flag-based filters. For example, if a particular star schema requires Buyer as a dimension, the Employee table can be used with a filter where the Buyer flag is set to Y.

3.2.5 Fact Tables in Oracle Business Analytics Warehouse

Each fact table contains one or more numeric foreign key columns to link it to various dimension tables.

3.2.6 Helper Tables in Oracle Business Analytics Warehouse

Helper tables are used by the Oracle Business Analytics Warehouse to solve complex problems that cannot be resolved by simple dimensional schemas.

In a typical dimensional schema, fact records join to dimension records with a many-to-one relationship. To support a many-to-many relationship between fact and dimension records, a helper table is inserted between the fact and dimension tables.

The helper table can have multiple records for each fact and dimension key combination. This allows queries to retrieve facts for any given dimension value. It should be noted that any aggregation of fact records over a set of dimension values might contain overlaps (due to a many-to-many relationship) and can result in double counting.

At times there is a requirement to query facts related to the children of a given parent in the dimension by only specifying the parent value (example: manager's sales fact that includes sales facts of the manager's subordinates). In this situation, one helper table containing multiple records for each parent-child dimension key combination is inserted between the fact and the dimension. This allows queries to be run for all subordinates by specifying only the parent in the dimension.

3.2.7 Hierarchy Tables in Oracle Business Analytics Warehouse

Some dimension tables have hierarchies into which each record rolls. This hierarchy information is stored in a separate table, with one record for each record in the corresponding dimension table. This information allows users to drill up and down through the hierarchy in reports.

There are two types of hierarchies in the Oracle Business Analytics Warehouse: a structured hierarchy in which there are fixed levels, and a hierarchy with parent-child relationships. Structured hierarchies are simple to model, since each child has a fixed number of parents and a child cannot be a parent. The second hierarchy, with unstructured parent-child relationships is difficult to model because each child record can potentially be a parent and the number of levels of parent-child relationships is not fixed. Hierarchy tables have a suffix of _DH.

3.2.8 Mini-Dimension Tables in Oracle Business Analytics Warehouse

Mini-dimension tables include combinations of the most queried attributes of their parent dimensions. They improve query performance because the database does not need to join the fact tables to the big parent dimensions but can join these small tables to the fact tables instead.

Table 3–3 lists the mini-dimension tables in the Oracle Business Analytics Warehouse.

Table 3–3 Mini-Dimension Tables in Oracle Business Analytics Warehouse

Table Name	Parent Dimension
W_RESPONSE_MD	Parent W_RESPONSE_D
W_AGREE_MD	Parent W_AGREE_D
W_ASSET_MD	Parent W_ASSET_D
W_OPTY_MD	Parent W_OPTY_D
W_ORDER_MD	Parent W_ORDER_D
W_QUOTE_MD	Parent W_QUOTE_D
W_SRVREQ_MD	Parent W_SRVREQ_D

3.2.9 Staging Tables in Oracle Business Analytics Warehouse

Staging tables are used primarily to stage incremental data from the transactional database. When the ETL process runs, staging tables are truncated before they are populated with change capture data. During the initial full ETL load, these staging tables hold the entire source data set for a defined period of history, but they hold only a much smaller volume during subsequent refresh ETL runs.

This staging data (list of values translations, computations, currency conversions) is transformed and loaded to the dimension and fact staging tables. These tables are typically tagged as <TableName>_DS or <TableName>_FS. The staging tables for the Usage Accelerator are tagged as WS_<TableName>.

The staging table structure is independent of source data structures and resembles the structure of data warehouse tables. This resemblance allows staging tables to also be used as interface tables between the transactional database sources and data warehouse target tables.

3.2.10 Translation Tables in Oracle Business Analytics Warehouse

Translation tables provide multi-language support by storing names and descriptions in each language that Oracle Business Analytics Warehouse supports. There are two types of translation tables:

- Domain tables that provide multi-language support associated with the values stored in the %_CODE columns.
- Tables that provide multi-language support for dimensions.

Domains and their associated translated values are stored in a single table named W_DOMAIN_MEMBER_LKP_TL. Each dimension requiring multi-language support that cannot be achieved with domains has an associated _TL table. These tables have a one-to-many relationship with the dimension table. For each record in the dimension table, you will see multiple records in the associated translation table (one record for each supported language).

3.3 Standard Column Prefixes in Oracle Business Analytics Warehouse

The Oracle Business Analytics Warehouse uses a standard prefix to indicate fields that must contain specific values, as shown in Table 3–4.

Table 3–4 Standard Column Prefix

Prefix	Description	In Table Types
W_	Used to store Oracle BI Applications standard or standardized values. For example, W_%_CODE (Warehouse Conformed Domain) and W_TYPE, W_INSERT_DT (Date records inserted into Warehouse).	_A _D _F

3.4 Standard Column Suffixes in Oracle Business Analytics Warehouse

The Oracle Business Analytics Warehouse uses suffixes to indicate fields that must contain specific values, as shown in Table 3–5.

Table 3–5 Standard Column Suffixes

Suffix	Description	In Table Types
_CODE	Code field. (Especially used for domain codes.)	_D, _DS, _FS, _G, _GS
_DT	Date field.	_D, _DS, _FS, _G, _DHL, _DHLS
_ID	Correspond to the _WID columns of the corresponding _F table.	_FS, _DS
_FLG	Indicator or Flag.	_D, _DHL, _DS, _FS, _F, _G, _DHLS
_WID	Identifier generated by Oracle Business Intelligence linking dimension and fact tables, except for ROW_WID.	_F, _A, _DHL
_NAME	A multi-language support column that holds the name associated with an attribute in all languages supported by the data warehouse.	_TL
_DESCR	A multi-language support column that holds the description associated with an attribute in all languages supported by the data warehouse	_TL

3.5 System Columns in Oracle Business Analytics Warehouse Tables

Oracle Business Analytics Warehouse tables contain system fields. These system fields are populated automatically and should not be modified by the user. Table 3–6 lists the system columns used in data warehouse dimension tables.

Table 3–6 System Columns Used in Data Warehouse Tables

System Column	Description
ROW_WID	Surrogate key to identify a record uniquely.
CREATED_BY_WID	Foreign key to the W_USER_D dimension that specifies the user who created the record in the source system.
CHANGED_BY_WID	Foreign key to the W_USER_D dimension that specifies the user who last modified the record in the source system.
CREATED_ON_DT	The date and time when the record was initially created in the source system.
CHANGED_ON_DT	The date and time when the record was last modified in the source system.
AUX1_CHANGED_ON_DT	System field. This column identifies the last modified date and time of the auxiliary table's record that acts as a source for the current table.
AUX2_CHANGED_ON_DT	System field. This column identifies the last modified date and time of the auxiliary table's record that acts as a source for the current table.
AUX3_CHANGED_ON_DT	System field. This column identifies the last modified date and time of the auxiliary table's record that acts as a source for the current table.
AUX4_CHANGED_ON_DT	System field. This column identifies the last modified date and time of the auxiliary table's record that acts as a source for the current table.

Table 3–6 (Cont.) System Columns Used in Data Warehouse Tables

System Column	Description
DELETE_FLG	This flag indicates the deletion status of the record in the source system. A value of Y indicates the record is deleted from the source system and logically deleted from the data warehouse. A value of N indicates that the record is active.
W_INSERT_DT	Stores the date on which the record was inserted in the data warehouse table.
W_UPDATE_DT	Stores the date on which the record was last updated in the data warehouse table.
DATASOURCE_NUM_ID	Unique identifier of the source system from which data was extracted. In order to be able to trace the data back to its source, it is recommended that you define separate unique source IDs for each of your different source instances.
ETL_PROC_WID	System field. This column is the unique identifier for the specific ETL process used to create or update this data.
INTEGRATION_ID	Unique identifier of a dimension or fact entity in its source system. In case of composite keys, the value in this column can consist of concatenated parts.
TENANT_ID	Unique identifier for a tenant in a multi-tenant environment. This column is typically be used in an Application Service Provider (ASP)/Software as a Service (SaaS) model.
X_CUSTOM	Column used as a generic field for customer extensions.
CURRENT_FLG	This is a flag for marking dimension records as "Y" in order to represent the current state of a dimension entity. This flag is typically critical for Type II slowly changing dimensions, as records in a Type II situation tend to be numerous.
EFFECTIVE_FROM_DT	This column stores the date from which the dimension record is effective. A value is either assigned by Oracle BI Applications or extracted from the source.
EFFECTIVE_TO_DT	This column stores the date up to which the dimension record is effective. A value is either assigned by Oracle BI Applications or extracted from the source.
SRC_EFF_FROM_DT	This column stores the date from which the source record (in the Source system) is effective. The value is extracted from the source (whenever available).
STC_EFF_TO_DT	This column stores the date up to which the source record (in the Source system) is effective. The value is extracted from the source (whenever available).

3.6 Multi-Currency Support for System Columns

Table 3–7 lists the currency codes and rates for related system columns.

Table 3–7 Currency Codes and Rates for Related System Columns

System Column	Description
DOC_CURR_CODE	Code for the currency in which the document was created in the source system.
LOC_CURR_CODE	Usually the reporting currency code for the financial company in which the document was created.

Table 3–7 (Cont.) Currency Codes and Rates for Related System Columns

System Column	Description
GRP_CURR_CODE	The primary group reporting currency code for the group of companies or organizations in which the document was created.
LOC_EXCHANGE_RATE	Currency conversion rate from the document currency code to the local currency code.
GLOBAL1_EXCHANGE_RATE	Currency conversion rate from the document currency code to the Global1 currency code.
GLOBAL2_EXCHANGE_RATE	Currency conversion rate from the document currency code to the GLOBAL2 currency code.
GLOBAL3_EXCHANGE_RATE	Currency conversion rate from document currency code to the GLOBAL3 currency code.
PROJ_CURR_CODE	Code used in Project Analytics that corresponds to the project currency in the OLTP system.

3.7 Oracle Business Analytics Warehouse Primary Data Values

It is possible for various dimensions to have one-to-many and many-to-many relationships with each other. These kinds of relationships can introduce problems in analyses. For example, an Opportunity can be associated with many Sales Representatives and a Sales Representative can be associated with many Opportunities. If your analysis includes both Opportunities and Sales Representatives, a count of Opportunities would not be accurate because the same Opportunity would be counted for each Sales Representative with which it is associated.

To avoid these kinds of problems, the Oracle Business Analytics Warehouse reflects the primary member in the "many" part of the relationship. In the example where an Opportunity can be associated with many Sales Representatives, only the Primary Sales Representative is associated with that Opportunity. In an analysis that includes both Opportunity and Sales Representative, only a single Opportunity will display and a count of Opportunities returns the correct result.

There are a few important exceptions to this rule. The Person star schema supports a many-to-many relationship between Contacts and Accounts. Therefore, when querying the Person star schema on both Accounts and Contacts, every combination of Account and Contact is returned. The Opportunity-Competitor star schema supports a many-to-many relationship between Opportunities and Competitor Accounts, and the Campaign-Opportunity star schema supports a many-to-many relationship between Campaigns and Opportunities. In other star schemas, however, querying returns only the primary account for a given contact.

3.8 About Multi-Language Support in the Oracle Business Analytics Warehouse

Oracle BI Applications provides multi-language support for metadata level objects exposed in Oracle BI Enterprise Edition dashboards and reports, as well as data, which enables users to see records translated in their preferred language. For more information about multi-language support, see Chapter 2, "About Multi-Language Support."

3.9 Oracle Business Analytics Warehouse Currency Preferences

For information about setting up currencies, refer to the following task in Functional Setup Manager: **Common Areas and Dimensions Configurations\ Configure Global Currencies**.

The Oracle Business Analytics Warehouse supports the following currency preferences.

- **Contract currency.** The currency used to define the contract amount. This currency is used only in Project Analytics.
- **CRM currency.** The CRM corporate currency as defined in the Fusion CRM application. This currency is used only in CRM Analytics applications.
- **Document currency.** The currency in which the transaction was done and the related document created.
- **Global currency.** The Oracle Business Analytics Warehouse stores up to three group currencies. These need to be pre-configured so as to allow global reporting by the different currencies. The exchange rates are stored in the table W_EXCH_RATE_G.
- **Local currency.** The accounting currency of the legal entity in which the transaction occurred.
- **Project currency.** The currency in which the project is managed. This may be different from the functional currency. This applies only to Project Analytics.

4

Oracle BI Applications Patching

This chapter describes how to apply Oracle BI Applications patches.

This chapter contains the following topics:

4.1 Introduction to Oracle BI Applications Patching

For Oracle Fusion Applications source systems, this chapter supplements the information provided in *Oracle Fusion Applications Patching Guide*.

This chapter also supplements information provided in *Oracle Fusion Middleware Patching Guide* for patching Oracle Fusion Middleware products.

Patching involves using OPatch to copy a small collection of files over an existing installation. A patch is normally associated with a particular version of an Oracle product. A patch set contains a collection of patches that are designed to be applied at the same time.

Oracle Fusion Middleware Patching Guide provides a full description of the types of patches available and the instructions to apply them. Before applying a patch to an Oracle Business Intelligence Applications system, make sure to review *Oracle Fusion Middleware Patching Guide*, as well as the readme.txt file provided with the patch. The readme file will describe the content of the patch and may contain additional information and instructions.

4.2 What Is Included in Oracle BI Applications Patches?

An Oracle BI Applications patch can include bug fixes, metadata, and binary file updates. The exact updates made will depend on the patch contents, and can include any of the following:

- **Metadata patch** - updates pre-built content, as follows:
  – Oracle BI Presentation Catalog - dashboard and report updates.
  – Oracle Business Intelligence Repository (RPD) - Presentation layer, Business Model and Mapping layer, and Physical layer updates.
  – Oracle BI Applications Configuration Manager and Functional Setup Manager.
  – Oracle Business Intelligence Data Warehouse Administration Console (DAC) - table and column updates.
  – Informatica Repository.
  – Changes to JAZN security settings (that is, in the `system-jazn-data.xml` file).
- **Binary patch** - updates binary files (files with extensions such as DLL, JAR, and EXE) as follows:
  – Oracle BI Applications Configuration Manager
  – DAC Client and DAC Server

Note: An Oracle BI Applications patch does not include the following:

- Oracle Business Intelligence platform

  For information about patching the Oracle Business Intelligence platform, see 'Patching Oracle Business Intelligence Systems' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.
- Informatica PowerCenter tools

  Patches to Informatica PowerCenter binary files are available as hot fixes from Informatica Corporation.

4.3 Where Updates Are Made by Oracle BI Applications Patches

The exact updates made by a patch depends on what is in that particular patch, but it can consist of a combination of the content types described in Section 4.2.

If the content of the patch contains just binary updates, then these updates are made directly to the Oracle home, and only to the Oracle home.

If the content of the patch contains metadata updates it is important to note that it is not only the Oracle home that will be updated. For each metadata type that can be patched (for more information, see Section 4.2), one copy is kept in the Oracle home and another is deployed for use at runtime. A patch containing metadata updates will update both the Oracle home content as well as the runtime, or deployed content.

For example, patch updates to the Oracle BI Presentation Catalog will be copied to the Oracle home location that keeps track of the version of the Oracle BI Presentation Catalog last delivered by Oracle, and will also update the Oracle BI Presentation Catalog that has been deployed for use at runtime.

4.4 Applying Oracle BI Applications Patches

You can apply a patch to Oracle BI Applications or Oracle BI Applications Configuration Manager using the following procedure:

- Section 4.4.1, "Applying an Oracle BI Applications Patch"
- Section 4.4.2, "Applying an Oracle BI Applications Configuration Manager Patch"

4.4.1 Applying an Oracle BI Applications Patch

To apply an Oracle BI Applications patch:

1. Copy or download the OPatch archive.

   For example, you might copy a downloaded archive called 1234567.zip, to the folder /scratch/patchBase.

   For information about what is included in the patch:

   - See the README.txt file in the OPatch archive.
   - Run the 'opatch query' command (for example, `opatch query 123456789.zip`) to list bugs fixed, and the contents of the patch.

2. (Required for metadata patches only.) Prepare the Patch Properties file (apply-patch.properties).

   To apply changes to the metadata types being patched, you specify values in the apply-patch.properties file for the different metadata types. For example, the location of the runtime Oracle BI Presentation Catalog.

   A sample properties file described in Section 4.8, "Sample Patch Properties File", lists each metadata type. You only need to supply values that correspond to the metadata types being patched.

   Once the property values are completed, save this to: /scratch/patchBase/apply-patch.properties

   Note: If you do not specify this file in the OPatch command, properties that are required will be requested when you are applying the patch.

3. Back up all metadata.

   Metadata must be backed up using your normal backup mechanisms.

4. Stop the Oracle Business Intelligence components.

   Since the patch will make updates to the runtime metadata, you must do this to avoid locking issues; the patch might fail if you do not do this. In addition, not all metadata updates will be visible until Oracle Business Intelligence is restarted.

   Use Fusion Middleware Control to stop Oracle Business Intelligence components. For more information, see 'Starting and Stopping Oracle Business Intelligence' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

   Note: If the patch contains RPD updates only, you need stop only the BI Server component. If the patch contains Oracle BI Presentation Catalog updates only, you need stop only the BI Presentation Services component.

5. Apply the Oracle BI Applications metadata patch.

   Apply the patch using the copy of OPatch in the Oracle home, as follows:

   For UNIX use the following syntax:

   ```
   ./opatch apply -pre <path to patch properties file> -opatch_
   pre_end <path to unzipped patch dir>
   ```

   For example, if you copied your patch archive to /scratch/patchBase/1234567.zip, and your properties file is in /scratch/patchBase/apply-patch.properties, enter the following command:

   ```
   ./opatch apply -pre /scratch/patchBase/apply-patch.properties
   -opatch_pre_end /scratch/patchBase/1234567.zip
   ```

6. If the patch contains changes to JAZN settings, then follow the steps in Section 4.4.1.1, "How to Apply Changes to JAZN Settings".

   The task described in Section 4.4.1.1, "How to Apply Changes to JAZN Settings" must be performed using a copy of system-jazn-data.xml that is made before the patch is applied in Step 5 above.

   Tip: To determine whether a patch includes changes to JAZN settings, look for a `biapps-policystore.xml` file in the patch file, or look in the readme file that accompanies the patch for changes to JAZN settings.

7. If the patch includes Oracle BI Repository (RPD) updates and the environment is clustered, you must scale out the RPD so that the repository can be shared by all Oracle BI Servers participating in a cluster.

   For more information, see 'Uploading and Sharing the Oracle BI Repository' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

8. Re-start the Oracle Business Intelligence components.

   For more information, see 'Starting and Stopping Oracle Business Intelligence' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

   Note: If the patch contained only RPD updates, and you stopped only the BI Server in step 4, then you need only restart the BI Server. If the patch contained only Oracle BI Presentation Catalog updates, and you stopped the BI Presentation Services in step 4, then you need only restart the BI Presentation Services.

4.4.1.1 How to Apply Changes to JAZN Settings

Note: This task forms part of Step 6 in Section 4.4.1, "Applying an Oracle BI Applications Patch". After completing this task, continue from Step 7 in Section 4.4.1, "Applying an Oracle BI Applications Patch".

If you are applying an Oracle BI Applications patch that contains changes to JAZN settings (that is, in the `biapps-policystore.xml` file), then you must follow the steps below.

Notes:

- You must make a backup of the production policy store before applying the changes included in a JAZN patch. For more information about backing up a policy store, see Section 8.6.2 'Migrating with the Script migrateSecurityStore' in *Oracle Fusion Middleware Application Security Guide*.
- During patching, only the system-jazn-data.xml file in the Oracle Home is patched. The changes to BI JAZN data need to be applied to the production JAZN policy store using either:

  - the WLST script patchPolicyStore() - see Steps 5, 6, and 7 in the task below.

  - Oracle Authorization Policy Manager (Oracle APM). If you use Oracle APM to apply JAZN changes to the production JAZN policy store, then you can omit Steps 5, 6, and 7 in the task below.

How to Apply Changes to JAZN Settings:

1. Make a copy of the production jps-config.xml file.

   For example, you might use the following command:

   ```
   cp ${ORACLE_HOME}/BIDomain/config/fmwconfig/jps-config.xml
   ```

```
${ORACLE_HOME}/BIDomain/config/fmwconfig/copy-jps-config.xml
```

2. Edit the copy of the jps-config.xml, and commenting out all service instances in the default JPS context so that only the policystore service instance remains.

   For example:

   ```
   <jpsContext name="default">
          <!--serviceInstanceRef ref="credstore.ldap"/-->
           <!--serviceInstanceRef ref="keystore.ldap"/-->
           <serviceInstanceRef ref="policystore.ldap"/>
           <!--serviceInstanceRef ref="audit"/-->
           <!--serviceInstanceRef ref="idstore.ldap"/-->
           <!--serviceInstanceRef ref="trust"/-->
           <!--serviceInstanceRef ref="pdp.service"/-->
   </jpsContext>
   ```

3. Start the WLST interactively using $ORACLE_HOME/common/bin/wlst.sh.
4. If SSO is used to connect to the policy store then set the following properties:

   ```
   java.lang.System.setProperty("javax.net.ssl.trustStore",'<truststore>');
   java.lang.System.setProperty("javax.net.ssl.trustStorePassword", '<password>');
   ```

   Where <truststore> is the location of the truststore. For example: `/scratch/aime1/APPTOP/fusionapps/wlserver_10.3/server/lib/fusion_trust.jks`. And where <password> is the truststore password.

5. Call the patchPolicyStore script as follows:

   ```
   patchPolicyStore(phase="analyze", baselineFile=<previous_jazn>,
   patchFile=<system_jazn>,
   productionJpsConfig=<copy_of_jps_config>, patchDeltaFolder=<delta_dir>,
   baselineAppStripe="obi")
   ```

   Where:

   - <previous_jazn> = a copy of the biapps-policystore.xml from before the patch is applied (see Step 1 above). For example, $[ORACLE_HOME]/bifoundation/admin/provisioning/biapps-policystore.xml

     Opatch stores the copy of the biapps-policystore.xml file at:

     $[BI ORACLE_HOME]/.patch_storage/<patch_number>_<date_time>/original_patch/files/bifoundation/admin/provisioning/biapps-policystore.xml
   - <system_jazn> = the new biapps_policystore.xm in the Oracle Home. For example, $[ORACLE_HOME]/bifoundation/admin/provisioning/biapps-policystore.xml
   - <copy_of_jps_config> = the jps-config edited as mentioned above. For example, $[ORACLE_HOME]/BIDomain/config/fmwconfig/copy-jps-config.xml
   - <delta_dir> = a directory to store the results of the analysis. For example, $tmp/jazn_patch_delta

6. Resolve any conflicts.

   For more information, refer to OPSS Patching Tool documentation.

7. Call patchPolicyStore() with 'apply', that is:

```
patchPolicyStore(phase="apply", productionJpsConfig=<jps_config>,
patchDeltaFolder=<delta_dir>,
productionAppStripe="obi")
```

4.4.2 Applying an Oracle BI Applications Configuration Manager Patch

To apply an Oracle BI Applications Configuration Manager patch:

1. Copy or download the OPatch archive.

   For example, you might copy a downloaded archive called 1234567.zip, to the folder /scratch/patchBase.

2. Backup all metadata.

   Metadata must be backed up using your normal backup mechanisms.

3. Stop the Oracle Business Intelligence components.

   Since the patch will make updates to the runtime metadata you must do this to avoid locking issues; the patch might fail if you do not do this. In addition, not all metadata updates will be visible until Oracle Business Intelligence is restarted.

   Use Fusion Middleware Control to stop Oracle Business Intelligence components. For more information, see 'Using Fusion Middleware Control to Start and Stop Oracle Business Intelligence System Components and Java Components' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

4. If the patch contains BI Applications Configuration Manager binary updates, you must also stop the Admin server.

   Use Oracle WebLogic Server Administration Console to stop the Admin server. For more information, see 'Using Oracle WebLogic Server Administration Console to Start and Stop Java Components ' in Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition.

5. Apply a BI Applications Configuration Manager patch.

   You apply a BI Applications Configuration Manager patch using the copy of OPatch in the Oracle home.

   Note: A data warehouse schema must have already been created using the Repository Creation Utility (RCU).

   For UNIX use the following syntax:

   ```
   ./opatch apply -post <hostname:port:servicename/SID> <data_
   warehouse_schema_username> <data_warehouse_schema_password>
   -opatch_post_end <path to unzipped patch dir>
   ```

   For example, if you copied your patch archive to /scratch/patchBase/456789.zip, enter the following command:

   ```
   ./opatch apply -post mycomputer.mycompany.com:1521:mydatabase
   mydbusername mydbpassword -opatch_post_end
   /scratch/patchBase/456789.zip
   ```

6. Re-start the Oracle Business Intelligence components

   For more information, see 'Using Fusion Middleware Control to Start and Stop Oracle Business Intelligence System Components and Java Components' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

7. Re-start the Admin server (if stopped in step 4).

   For more information, see 'Using Oracle WebLogic Server Administration Console to Start and Stop Java Components' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

4.5 Diagnosing Whether Oracle BI Applications Patches Are Applied Correctly

During the application of a patch, OPatch will output status results to the command window. If the patch was applied correctly, the last statement will be "OPatch succeeded". If there are any errors, these will be reported and captured in the log file located in <*ORACLE_HOME*>/cfgtoollogs/opatch/opatch<*timestamp*>.log.

4.6 Rolling Back Failed Oracle BI Applications Patches

If you roll back a patch with metadata updates, different steps are required compared to if you roll back a patch with just binary file updates.

To rollback a failed Oracle BI Applications patch:

1. Rollback the OPatch archive.

   This step removes the OPatch inventory metadata, allowing the patch to be applied again in the future. However, it does not update the changes made to the metadata (see next step).

   ```
   ./opatch rollback -ID 1234567
   ```

   Where 1234567 is the patch id.

2. Replace the updated metadata from your backup area.

   Only complete this step if the patch contains metadata updates to the Oracle Business Intelligence Repository (RPD), Oracle BI Presentation Catalog, Informatica Repository, or DAC.

   In order to get the metadata that was updated by the patch to be as it was before the patch, you need to manually copy the metadata from your backup area (see step 3 of the 'apply' flow above). You must copy this to both the Oracle home and the runtime locations.

   Warning: If you roll back, you will lose any customizations made since the patch was applied.

4.7 What Happens if Conflicts Are Detected When Applying an Oracle BI Applications Patch?

For DAC and Informatica PowerCenter metadata patches, the updates included in the patch will replace metadata in customer repositories. If customers have made any changes to the metadata since the initial install, these changes are lost, and will need to be reapplied after the patch. For this reason there are no conflicts during the patch apply cycle itself for these metadata types.

When you patch the Oracle BI Repository (RPD) and Oracle BI Presentation Catalog, updates are merged into the existing customer runtime repositories. However, there will be cases where the metadata included in the patch cannot be merged without user intervention, that is, a choice needs to be made.

Where such a conflict occurs the patch will fail to apply, and details of the conflicts will be given in the log file. These conflicts need to be resolved first and then the patch re-applied.

In order to resolve the conflicts customers can use the standard Oracle Business Intelligence platform tools appropriate for the metadata type in questions. For example, for conflicts in the Oracle Business Intelligence Repository (RPD) use the Oracle BI Administration Tool, and for conflicts in the Oracle BI Presentation Catalog use the Catalog Manager. Using these tools customers need to resolve the conflicts, re-apply the patch, and then apply their customizations on top of the patched metadata.

- Section 4.7.1, "Resolving Conflicts for Oracle BI Presentation Catalog Updates"
- Section 4.7.2, "Resolving Conflicts for Oracle BI Repository (RPD) Updates"

4.7.1 Resolving Conflicts for Oracle BI Presentation Catalog Updates

Follow the instructions outlined here to resolve conflicts for the Oracle BI Presentation Catalog updates.

For more information, see 'Configuring and Managing the Oracle BI Presentation Catalog' in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

To resolve conflicts for Oracle BI Presentation Catalog updates:

1. Start the Catalog Manager (CatMan1), and open your runtime Oracle BI Presentation Catalog in offline mode.
2. Open a second instance of the Catalog Manager (CatMan2), and open the Oracle home Presentation Catalog in offline mode.
3. Review the list of objects with conflicts in the patch log.
4. Archive the runtime Oracle BI Presentation Catalog:

   For instructions, see "Archiving and Unarchiving Using Catalog Manager," in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.
5. For each object with a conflict, do the following:
   a. In the Oracle home Presentation Catalog (CatMan2) copy the object.
   b. Paste the object into the runtime Oracle BI Presentation Catalog (CatMan1) using the "Force" paste option.

      For more information about copying and pasting objects, see "Copying and Pasting Objects" in Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition.

   The runtime Oracle BI Presentation Catalog now has its conflicting objects reset to the Oracle home equivalent.
6. Re-apply the patch.

4.7.2 Resolving Conflicts for Oracle BI Repository (RPD) Updates

Follow the instructions outlined here to resolve conflicts for Oracle BI Repository (RPD) updates.

For more information, see 'Applying a Repository Patch' in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

To resolve conflicts for Oracle BI Repository (RPD) updates:

1. Retrieve the Oracle BI Repository 'diff' file from the location in the log.
2. Open the Oracle BI Administration Tool.
3. Open the modified/customer repository.
4. Select File, and Merge from the menu.
5. Select merge type as Patch Repository Merge.
6. Select the patch 'diff' file.

   The location for this file is specified in the patch apply logs.
7. Choose the original repository to be the "original" Oracle Business Intelligence Repository (RPD) in the Oracle home.
8. Enter the password for the original RPD.
9. Display the next page of the Wizard to see a list of conflicts.
10. For each conflict mentioned in the patch log, select "current", (accepting the change from the patch).
11. Re-apply the patch.

4.8 Sample Patch Properties File

You can modify the example apply-patch.properties file described in this section, to reference in the OPatch apply command.

The OPatch command must include the full file path, for example if the file is in: /scratch/biapps/mwhome/Oracle_BI1/apply-patch.properties, then you would enter the following opatch command:

```
./OPatch/opatch apply -pre /scratch/biapps/mwhome/Oracle_
BI1/apply-patch.properties -opatch_pre_end 1234567.zip
```

Note: The latest version of this file is available in the patch archive at custom/ patch.properties.

The contents of the apply-patch.properties file is as follows:

```
#
#BI Applications Apply Patch Properties
#

#RPD
#The following properties are required if the patch contains updates to the RPD
(aka Oracle Business Intelligence Repository)

#Full path to target RPD File (that is, the RPD file used in the runtime system)
rpd.targetRPDFile =

#password for the target RPD
rpd.targetPassword =

#password for the original RPD (that is, the RPD file shipped by Oracle, in Oracle
home)
```

```
rpd.originalPassword =



#Web Catalog
#The following properties are required if the patch contains updates to the Web
Catalog (aka Oracle BI Presentation Catalog)

#Root Directory of target Web Catalog (that is, the Web Catalog used in the
runtime system)
webcat.target.root =

#Root Directory of target Oracle Instance (under instances)
oracle.instance.root =



#Informatica Content
#The following properties are required if the patch contains updates to the
shipped Informatica content

#Informatica Home Directory (under which server/bin/pmrep lives)
infa.home =

#Informatica Repository Name
infa.repo.name =

#Informatica Repository Domain
infa.repo.domain =

#Informatica Repository Username
infa.repo.username =

#Informatica Repository Password
infa.repo.password =

#Informatica Repository Hostname
infa.repo.hostname =

#Informatica Repository Server Port (default is 6001)
infa.repo.port =



#DAC Content
#The following properties are required if the patch contains updates to the
shipped DAC content

#DAC repository hostname
dac.repo.hostname =

#DAC repository server port (default is 1521)
dac.repo.port =

#DAC repository server servicename/SID
dac.repo.serviceName =

#DAC repository username
dac.repo.username =
#DAC repository password
dac.repo.password =
```

5

Moving Oracle BI Applications Components

This chapter provides information about moving Oracle BI Applications from one environment to another.

This chapter contains the following sections:

5.1 About Moving Oracle BI Applications

You can move Oracle BI Applications components from one environment to another using different movement scenarios. This chapter focuses on the following scenarios:

- **Full-movement scenario**

  In this scenario, the *target* Oracle BI Applications environment, such as a production environment, does not exist. First, the *source* environment (development or test) is created, configured, customized, and tested. Then, the target environment is created by moving all the components along with their configurations from the source environment.

  For example, suppose you deployed Oracle BI Applications along with Oracle Fusion Applications in a development or test environment. You could move the components and configurations of the deployment into production using the full-movement scenario.

  For instructions on moving Oracle BI Applications using the full-movement scenario, see Section 5.2, "Moving Oracle BI Applications Using the Full-Movement Scenario".

- **Incremental-movement scenario**

  In this scenario, a target (production) environment exists but Oracle BI Applications has not been deployed.

  For example, suppose you deployed Oracle Fusion Applications, including Oracle Transactional Business Intelligence (which requires an Oracle BI metadata repository and presentation catalog) in a production environment, but you did not deploy Oracle BI Applications. You can deploy Oracle BI Applications in an Oracle Fusion Applications production environment at a later time by using the incremental-movement scenario.

For instructions on deploying Oracle BI Applications into an existing environment, see Section 5.3, "Moving Oracle BI Applications Using the Incremental-Movement Scenario".

5.2 Moving Oracle BI Applications Using the Full-Movement Scenario

When performing a full movement of Oracle BI Applications, you create a target (production) environment by moving all the components and configurations of the source (development or test) environment using movement scripts.

If you are replicating an Oracle BI Applications environment that is part of an Oracle Fusion Applications deployment, you will use the Oracle Fusion Applications movement scripts to move the components and configurations of both Oracle Fusion Applications and Oracle BI Applications at the same time.

For instructions on moving an Oracle BI Applications environment along with an Oracle Fusion Applications environment, see the chapter "Moving Components for Oracle Fusion Applications Across Environments" in *Oracle Fusion Applications Administrator's Guide*.

After completing the Oracle Fusion Applications movement process, additional tasks are required to complete the Oracle BI Applications move. For instructions on completing the required tasks, see the following sections:

- Section 5.2.1, "Preconditions for Moving an Oracle BI Applications Environment"
- Section 5.2.2, "Moving the Informatica Repository"
- Section 5.2.3, "Moving Oracle BI Applications Configuration Manager Components"

5.2.1 Preconditions for Moving an Oracle BI Applications Environment

The following preconditions must be met before moving an Oracle BI Applications environment.

- Oracle BI Enterprise Edition is available in the target environment.
- The RCU for Oracle BI Enterprise Edition was run, as described in the section "Loading the Oracle Business Intelligence Schemas in the Oracle RAC Database" in *Oracle Fusion Middleware Enterprise Deployment Guide for Oracle Business Intelligence*.
- The RCU for Oracle BI Applications was run, as described in the section "Creating Oracle BI Applications Schemas Using Oracle BI Applications RCU" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.
- A database has been created in the target environment for the Informatica Repository, and this database platform is of the same type as that in the source environment.
- Informatica PowerCenter Client and Services have been installed and set up in the target environment.

  Note: Make sure you have installed and set up Informatica PowerCenter Client and Services in the target environment according to the instructions in the section "Manual Installation and Set Up for Informatica PowerCenter Services" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*. The high-level set up requirements include the following:

- Install and configure database connectivity software on the computers that host the Informatica Integration Services and Repository Service.
- Create the Informatica Repository Service and point it to the Informatica Repository database for the production environment.
- Create the Informatica Integration Services.
- On the computer that hosts the Integration Services, configure the database client connectivity to the data warehouse and the ODBC DSN to the BI Server.
- Make sure the code page of the Repository Service in the target environment is the same as the code page of the Repository Service in the source environment
- Copy the source files from the Oracle BI Applications installation directory to the Informatica directory on the Informatica PowerCenter Services computer.
- Configure Integration Services for relaxed code page validation.
- Set the Integration Services overrideMpltVarWithMapVar custom property.
- Create the Informatica Repository user in the native security domain.
- Configure Informatica relational connections.
- Register the Integration Services in DAC.

5.2.2 Moving the Informatica Repository

This section contains instructions for moving the Informatica Repository from one environment to another.

If you are moving Oracle BI Applications in the context of an Oracle Fusion Applications movement, you must perform this additional procedure for moving the Informatica Repository after you perform the steps for moving Oracle Fusion Applications, as described in the chapter "Moving Components for Oracle Fusion Applications Across Environments" in *Oracle Fusion Applications Administrator's Guide*.

Use the Informatica Backup and Restore functionality to move the Informatica Repository if the Informatica domain in the source and target environments is different from one another. See Section 5.2.2.1 for instructions.

Use the Informatica Copy Repository functionality to move the Informatica Repository if the Informatica domain in the source and target environments is the same. See Section 5.2.2.2 for instructions.

5.2.2.1 Backing Up and Restoring the Informatica Repository

When you back up a repository, the Repository Service saves the repository in a binary file (.rep), including the repository objects, connection information, and code page information. You can restore metadata from a repository binary backup file. When you restore a repository, you must have a database available for the repository. You can restore the repository in a database that has a compatible code page with the original database.

Before you back up a repository and restore a repository into a different domain, verify that users and groups with privileges for the source Repository Service exist in the target domain. The Service Manager periodically synchronizes the list of users and groups in the repository with the users and groups in the domain configuration database. During synchronization, users and groups that do not exist in the target domain are deleted from the repository. You can use infacmd to export users and groups from the source domain and import them into the target domain.

To back up and restore the Informatica Repository:

1. Back up the source repository:
   a. In the source environment, log into the Informatica Administrator.
   b. In the navigator, select the Repository Service for the Informatica Repository.
   c. In the **Actions** list, select **Back Up Contents**.
   d. Enter the user name and password. (The Security Domain field appears when the PowerCenter domain contains an LDAP security domain. If necessary, enter the Security Domain.)
   e. Enter a file name and description for the repository backup file.
   f. Choose to skip or back up workflow and session logs, deployment group history, and MX data. You might want to skip these operations to increase performance when you restore the repository.
   g. Click **OK**. The results of the backup operation appear in the activity log
2. Restore the repository in the target environment:
   a. In the target environment, log into the Informatica Administrator.
   b. In the navigator, select the Repository Service.
   c. In the Actions list, select **Restore Contents**. The Restore Contents options appear.
   d. Select the appropriate backup file to restore.
   e. Select whether or not to restore the repository as new. When you restore a repository as new, the Repository Service restores the repository with a new repository ID and deletes the log event files.
   f. Optionally, choose to skip workflow and session logs, deployment group history, and MX data. Skipping the data improves performance.
   g. Click **OK**. The activity log indicates whether the restore operation succeeded or failed

5.2.2.2 Copying the Informatica Repository

You can use the Informatica Copy functionality to copy content into a repository when no content exists for the repository and you want to use the content from a different repository. Copying repository content provides a quick way to copy the metadata that you want to use as a basis for a new repository. To copy repository content, you must create the Repository Service for the target repository. When you create the Repository Service, set the creation mode to create the Repository Service without content. If a repository exists in the target database, the copy operation fails. You must back up the existing repository and delete it from the target database before copying the repository content.

To copy the Informatica Repository:

1. Log into the Informatica Administrator. (Since the Informatica domain is the same for both source and target environments, there is just one Informatica Administrator.)
2. In the navigator, select the Repository Service to which you want to add copied content.

   You cannot copy content to a repository that already contains content.

3. In the **Actions** list, select **Copy From**.

   The page displays the options for the **Copy From** operation.

4. Select the name of the Repository Service.

   The source Repository Service and the Repository Service to which you want to add copied content (target) must be in the same domain and must be of the same service version.

5. Enter the user name and password. (The Security Domain field appears when the PowerCenter domain contains an LDAP security domain. If necessary, enter the Security Domain.)

6. To skip workflow and session logs, deployment group history, and MX data, select the check boxes in the **advanced options**. Skipping this data can increase performance

7. Click **OK**.

   The activity log displays the results of the copy operation.

5.2.3 Moving Oracle BI Applications Configuration Manager Components

This section contains instructions for moving Oracle BI Applications Configuration Manager components from one environment to another.

If you are moving Oracle BI Applications in the context of an Oracle Fusion Applications movement, you must perform this additional procedure for moving Configuration Manager components after you perform the steps for moving Oracle Fusion Applications, as described in the chapter "Moving Components for Oracle Fusion Applications Across Environments" in *Oracle Fusion Applications Administrator's Guide*.

To move Oracle BI Applications Configuration Manager components:

1. Register the source system with the same data source number using Oracle BI Applications Configuration Manager, as described in the section "Registering Source Systems" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

2. Run the Domains ETL, as described in the section "Run Domains ETL" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

3. Migrate the setup data using Oracle BI Applications Configuration Manager export and import functionality, as described in the section "About Migrating Setup Data" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

4. Perform setup steps in Functional Setup Manager:

   a. Configure the offerings and modules to deploy, and select Feature Choices, as described in the section "How to Enable Offerings and Select Feature Choices" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

   b. Create an implementation project and select an offering and one or more modules, as described in the section "How to Create an Implementation Project and Select Offerings" in *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

5.3 Moving Oracle BI Applications Using the Incremental-Movement Scenario

The Oracle BI Applications incremental-movement scenario is meant for situations in which you have already deployed a source system, such as Oracle Fusion Applications, in a development, test, or production environment, and you want to later deploy Oracle BI Applications.

This section provides information about first deploying Oracle BI Applications in a pre-production environment, such as a development or test environment, and then moving this deployment to a production environment.

This section contains the following topics:

- Section 5.3.1, "Deploying Oracle BI Applications Into an Existing Pre-Production Environment"
- Section 5.3.2, "Moving Oracle BI Applications Into an Existing Production Environment"

5.3.1 Deploying Oracle BI Applications Into an Existing Pre-Production Environment

This section provides information about deploying Oracle BI Applications into an existing pre-production source system environment, such as a development or test environment, for example, if you have an existing Oracle Fusion Applications development environment and you want to deploy Oracle BI Applications.

After you create, configure, customize, and test the Oracle BI Applications deployment in a pre-production environment, you can then move it into a production environment by following the instructions in Section 5.3.2, "Moving Oracle BI Applications Into an Existing Production Environment".

For Oracle Fusion Applications deployments, the Oracle Fusion Applications installation and provisioning process installs the Oracle BI Applications software components in the Business Intelligence Oracle Home but does no further setup. You can deploy Oracle BI Applications at any time. To do so, you must follow the instructions in the chapter "Setting Up Oracle Business Intelligence Applications" in the *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*.

After you complete setting up Oracle BI Applications, you then need to perform the functional configuration, using Oracle BI Applications Configuration Manager and Functional Setup Manager. For instructions, see "Functional Configuration for Oracle Business Intelligence Applications" in Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications.

After functional configuration is complete, your environment is then ready for loading data into the Oracle Business Analytics Warehouse. The Oracle BI Applications ETL process is accomplished using Oracle Business Intelligence Data Warehouse Administration Console (DAC) and Informatica PowerCenter. For more information about configuring, running, and monitoring ETL processes, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

In addition, after Oracle BI Applications reports and dashboards have been tested and the data validated, the Oracle BI Applications system administrator will need to inform the Fusion Applications administrator to enable the appropriate embedded reports and dashboard (ADR) regions in Fusion Applications. The ADR regions are enabled using Functional Setup Manager for Fusion Applications.

5.3.2 Moving Oracle BI Applications Into an Existing Production Environment

Moving Oracle BI Applications from a pre-production environment into an existing production source system environment, such as an Oracle Fusion Applications environment, involves two major processes:

- **Setting up the Oracle BI Applications binary components in the production environment.**

  For an Oracle Fusion Applications deployment, the move plan process that would have been carried out when Oracle Fusion Applications was moved from a pre-production environment into the production environment also would have moved the Oracle BI Applications binary component files. In this process, you set up and configure these binary components.

  For instructions, see Section 5.3.2.1, "Setting Up the Oracle BI Applications Binary Components in Production".

- **Moving the metadata components of the pre-production Oracle BI Applications environment into production.**

  If you made changes to any of the metadata components, you need to migrate the customized metadata into the production environment.

  In this process, you migrate the customized metadata for the following components:

  – Data warehouse schema

  – DAC Repository

  – Informatica Repository

  – Configuration Manger and Functional Setup Manager

  – Oracle BI Repository

  – Presentation Catalog

  For instructions on migrating the metadata components, see Section 5.3.2.2, "Moving Oracle BI Applications Metadata Components Into Production".

5.3.2.1 Setting Up the Oracle BI Applications Binary Components in Production

Follow this procedure to set up the Oracle BI Applications binary components in a production environment when the binary components have been installed but not deployed.

Note: The process for setting up the Oracle BI Applications binary components differs based on the operating system platform. For Windows, the setup process is manual. For UNIX and Linux, the setup process is partially automated by the Warehouse Configuration Deployment Assistant (WCA).

Note: You must complete the steps in the order they are listed.

To set up the binary components in a production environment:

1. For all operating systems, complete the following procedures in Section 3.4, "Tasks for Setting Up Oracle BI Applications," in the *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*:

   a. "Setup Step: Create Required Databases."

   b. "Setup Step: Create Schemas and Data Warehouse Objects for Oracle BI Applications."

Note: When you run the RCU, deselect the **Create/Upgrade Schema** option. In this step you will run the Oracle BI Applications RCU in order to create Configuration Manger and Function Setup Manager tables. You will create the data warehouse schema using DAC in a later step.

   c. "Setup Step: Install and Set Up Informatica PowerCenter."
   d. "Setup Step: Create an ODBC DSN to the Oracle BI Server."
   e. "Setup Step: Create a User for ETL."
   f. "Setup Step: Grant User Access to Oracle BI Applications Components."

2. For UNIX and Linux platforms, perform the procedure in Section 3.4, "Setup Step: Configure Oracle BI Applications Components Using Warehouse Component Deployment Assistant (WCA)."

   Note: The WCA is not supported on Windows.

3. For the Windows platform, complete the following procedures in Section 3.5, "Platform-Specific and Topology-Specific Setup Steps," in the Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications:
   a. "Setup Step: Creating the Informatica Repository Service."
   b. "Setup Step: Creating the Informatica Integration Service."
   c. "Setup Step: Copying Source Files to the Informatica PowerCenter Services Machine."
   d. "Setup Step: Setting PowerCenter Integration Services Relaxed Code Page Validation."
   e. "Setup Step: Setting PowerCenter Integration Services Custom Properties."
   f. "Setup Step: Creating the Repository Administrator User in the Native Security Domain."
   g. "Setup Step: Extending the BI Domain."
   h. "Setup Step: Configuring the Oracle BI Connection Pools."
   i. "Setup Step: Configuring Physical Data Source Connections in DAC."
   j. "Setup Step: Configuring Relational Connections in Informatica."
   k. "Setup Step: Setting the SiebelUnicodeDB Property in Informatica Integration Services."
   l. "Setup Step: Enabling User Currency Preference Settings."
   m. "Setup Step: Registering Source Systems."
   n. "Setup Step: Configure DAC Integration Settings."

4. For all operating systems, complete the following procedures in Section 3.4, "Tasks for Setting Up Oracle BI Applications," in the *Oracle Fusion Middleware Installation and Configuration Guide for Oracle Business Intelligence Applications*:
   a. "Setup Step: Start DAC Server."
   b. "Setup Step: Validate the Oracle BI Applications Component Configuration."
   c. "Setup Step: Accessing the Informatica Domain and Repository from Informatica Client Tools."
   d. "Setup Step: Install Oracle BI Administration Tool."

e. "Setup Step: Configure Oracle HTTP Server."

f. "Setup Step: Configure SSO and Portlet Provider for Oracle BI Applications Configuration Manager and Functional Setup Manager."

g. "Setup Step: Enable Offerings for Deployment."

h. "Setup Step: Set Languages for Data Load."

i. "Setup Step: Editing Preferred Currency Display Names."

j. "Setup Step: Enable Document Currency."

k. "Setup Step: Install DAC Client."

   Note: This step is optional. You do not have to install the DAC Client in the production environment.

5.3.2.2 Moving Oracle BI Applications Metadata Components Into Production

This section provides instructions for moving metadata components into a production environment. Follow these procedures for any metadata component for which you made metadata changes.

If you did not change the metadata for a particular component, you do not need to perform the move procedure.

This section contains the following topics:

- Section 5.3.2.2.1, "Moving DAC Metadata"
- Section 5.3.2.2.2, "Creating the Data Warehouse Schema in a Production Environment"
- Section 5.3.2.2.3, "Moving Informatica Repository Metadata"
- Section 5.3.2.2.4, "Moving Configuration Manager and Functional Setup Manager Metadata"
- Section 5.3.2.2.5, "Moving Oracle BI Repository Metadata"
- Section 5.3.2.2.6, "Moving Oracle BI Presentation Catalog Metadata"

5.3.2.2.1 Moving DAC Metadata The DAC metadata is stored in a repository database and is used to configure the schema definition, index definition, ETL process names (task names), subject areas, and execution plans.

When you move the DAC metadata from one environment to another, the data in the source environment (for example, a development or test environment) is exported as serialized XML files and then imported into the target environment (for example, a production environment).

To export DAC metadata from the source environment:

- Follow the instructions in the section "Exporting DAC Metadata" in the *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

  Note: The first time you migrate the DAC metadata into a production environment export only the Logical and System data categories (as indicated by check boxes in the Export dialog). For subsequent data migration moves, you should only export the Logical data category. If you export the System data category for subsequent migrations, you will overwrite the configurations.

To import DAC metadata into the target environment:

- Follow the instructions in the section "Importing DAC Metadata" in the *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

Note: You can also use a command line to export and import DAC metadata. For instructions, see "DAC Repository Command Line Parameters" in *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

Note: If you need to move a subset of DAC metadata from one environment to another, you can use the DAC patching feature, which enables you to export a subset of data as an XML file from one environment and then import the data into another environment.

For more information about the DAC patching feature, see "Working With DAC Metadata Patches" in *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console.*

5.3.2.2.2 Creating the Data Warehouse Schema in a Production Environment Data warehouse schema information for the following objects is stored in the DAC repository:

- Tables
- Table columns
- Foreign key tables (part of the table columns definition)
- Related tables (table to table relationship)
- Indexes
- Index columns

In a pre-production environment, typically, changes are made to the data warehouse schema in the database using database client utilities, such as SQL Developer or SQL*Plus. Schema changes made in the database should then be imported into the DAC repository. You can then use DAC functionality to create the schema in the production environment.

To use DAC to create the data warehouse schema in a production environment:

- If you are using an Oracle database, follow the instructions in the section "Creating, Upgrading or Dropping an Entire Schema for Oracle Databases" in *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*. In the "Generate DW Table Scripts for Oracle" dialog, select the option **Create New**.

- If you are using a non-Oracle database follow the instructions in the section "Managing Data Warehouse Schemas for Non-Oracle Databases" in *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*. In the "Data Warehouse Configuration Wizard" dialog, select the option **Create\Upgrade Data Warehouse Tables**.

5.3.2.2.3 Moving Informatica Repository Metadata For instructions on moving an Informatica Repository from one environment to another, see Section 5.2.2, "Moving the Informatica Repository".

5.3.2.2.4 Moving Configuration Manager and Functional Setup Manager Metadata For instructions on moving Configuration Manager and Functional Setup Manager metadata, see Section 5.2.3, "Moving Oracle BI Applications Configuration Manager Components".

5.3.2.2.5 Moving Oracle BI Repository Metadata This section provides instructions for applying Oracle BI repository customizations in a pre-production environment to an existing Oracle BI repository in a production environment as well as re-applying objects from the default repository that may have been trimmed.

This procedure uses the Oracle BI Administration Tool to perform a three-way merge of the following repositories:

- The original, unmodified Oracle BI repository that you received with the current version of Oracle BI Applications. In the Administration Tool user interface, this repository is referred to as the "Original Master Repository."
- The customized Oracle BI repository in your production environment. In the Administration Tool user interface, this repository is referred to as the "Current Repository."
- The customized Oracle BI repository in your pre-production environment. In the Administration Tool user interface, this repository is referred to as the "Modified Repository."

To move the Oracle BI repository into a production environment:

1. Back up your current, production repository by copying the file from your runtime location to a safe location of your choosing.
2. Open the Oracle BI Administration Tool.
3. Equalize the three repositories to be merged.

   For instructions, see the section "Equalizing Objects," in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition*.
4. Open the production Oracle BI repository in offline mode.
5. From the File menu, select **Merge**.
6. In the Merge Repository Wizard, select **Full Repository Merge** as the Merge Type.
7. In the Original Master Repository field, browse for and select the original, unmodified Oracle BI repository that you received with the current version of Oracle BI Applications. Provide the password for this repository.
8. In the Modified Repository field, browse for and select the customized, pre-production Oracle BI repository. Provide the password for this repository.
9. Click **Next**.

   The Define Merge Strategy dialog displays a list of conflicts.
10. For each conflict displayed, select **Current**. This will apply your customization in the customized, pre-production Oracle BI repository to the production Oracle BI repository.

    For detailed information about merging repositories using the Oracle BI Administration Tool, see "Merging Repositories," in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition*.

5.3.2.2.6 Moving Oracle BI Presentation Catalog Metadata The process for moving the Oracle BI Presentation Catalog involves copying the customized objects in the source catalog and pasting them into the target catalog. For instructions, see the section titled "Copying and Pasting Objects" in *Oracle Fusion Middleware System Administrator's Guide for Oracle Business Intelligence Enterprise Edition*.

6 Configuring the Oracle BI Repository

This chapter describes how to configure the Oracle BI Repository for use with Oracle BI Applications.

This chapter contains the following topics:

6.1 How to Set Up Date-Specific Metrics

The time dimension in the Oracle BI Repository for Oracle Business Analytics Warehouse is a standard or canonical time dimension that links to the most important time role in each star schema. The Physical table alias used as a canonical time dimension is W_DAY_D_Common.

If a fact table contains a distinct set of metrics that needs to be reported by different dates, the metadata is organized so that each metric is reported by its causal date.

For example, the Invoice fact table has three metrics called Invoice Amount, Fulfill Amount, and Paid Amount, and each of these metrics need to be reported by the corresponding date—Invoice Date, Fulfill Date, and Payment Date. Additional dates in a fact table that a metric could be queried by are known as Secondary dates. These are available to the end users inside a detailed presentation folder. The detailed presentation folder is typically called the Details folder.

In Table 6–1 each of the metrics reflect the activity related to that event for the entire period, for example, Invoice Amount by Invoice Date, Fulfill Amount by Fulfill date, and Payment Amount by Payment Date.

Table 6–1 Invoice Fact Table Example

Date	Invoice Amount	Fulfill Amount	Payment Amount
January	4000	5000	4500

To implement date-specific metrics

1. Using Oracle BI Administration Tool, open OracleBIAnalyticsApps.rpd.
2. In the Physical layer, right-click on Oracle Data Warehouse, and create a new physical alias for the fact table.
3. Create Joins for the physical alias which are similar to the base fact table.

   The Join to the date dimension is changed to use the date role in question.
4. Create a new logical table source in the logical fact table that maps the metrics for the physical fact alias.

   The grain of the fact table is the same as the base fact table.

> **Note:** You need to map each metric to one logical table source at the Detail Level.

6.2 How to Set Up Additional Time Series Metrics for Oracle Business Analytics Warehouse

The Oracle BI Repository provides a framework to add Period Ago metrics. The Oracle Business Analytics Warehouse is pre-configured with pre-mapped period ago metrics; however, you can map other metrics by using the following procedure.

To set up additional time series metrics

1. Using Oracle BI Administration Tool, open OracleBIAnalyticsApps.rpd.
2. In the Physical layer, right-click on Oracle Data Warehouse, and create a new Period Ago physical alias table.
3. In the Physical layer, create additional tables for each Period Ago alias required.

   For example, Quarter Ago, Year Ago, and so on.

   These aliases need to have the same joins as the base fact table, except for the date join, which you can change in the next step. Setting up this alias is easier to accomplish by copying the base table.
4. Change the join to the date dimension (W_DAY_D) to use the appropriate Period Ago Key.
5. Map the Period Ago metrics in the logical table using the new fact alias by creating a new logical table source under the fact table.
6. Set the content pane levels for the period ago logical table source, to specify the level of the source data.

   These settings are the same as the base fact table.
7. Save and close the OracleBIAnalyticsApps.rpd file.

6.3 How to Set Up Additional Dimension Tables for Oracle Business Analytics Warehouse

Oracle Business Analytics Warehouse is pre-configured to map dimension tables required for analysis. The Physical layer in the Oracle BI Repository provides several other dimension table keys that can be used for certain specific analysis. If you need to set up any of the additional dimensions tables to the Physical layer, perform the following procedure.

To set up additional dimension tables

1. Validate that the dimension table key is resolved appropriately for the data source that you are using.
2. Using Oracle BI Administration Tool, open OracleBIAnalyticsApps.rpd.
3. Add a dimension table alias in the Physical layer.
4. Join the dimension table alias to the fact table alias using the appropriate keys.
5. Save and close the OracleBIAnalyticsApps.rpd file.

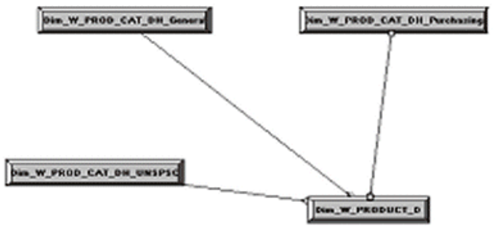
6.4 How to Set Up Product Category Hierarchies

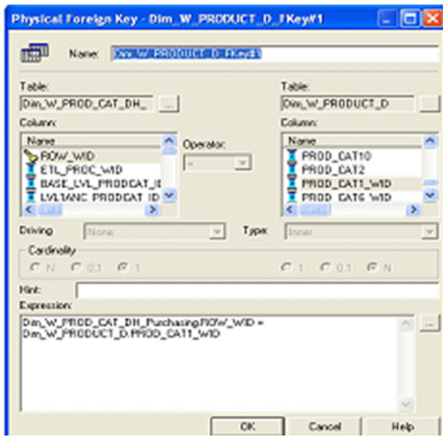
In Oracle Business Intelligence Applications, the following three hierarchies are supplied out-of-the-box:

- General Hierarchy
- Purchasing Hierarchy
- UNSPSC Hierarchy

To customize Product Category Hierarchies:

1. Using Oracle BI Administration Tool, open OracleBIAnalyticsApps.rpd.
2. Create a link with the W_PRODUCT_D table's PROD_CATn_WID column.





Name: Dim_W_PRODUCT_D_FKey#1

Table: Dim_W_PROD_CAT_DH_
Column:
Name
ROW_WID
ETL_PROC_WID
BASE_LVL_PRODCAT_I
LVL1ANC PRODCAT ID

Operator: =

Table: Dim_W_PRODUCT_D
Column:
Name
PROD_CAT10
PROD_CAT2
PROD_CAT1_WID
PROD_CAT6_WID

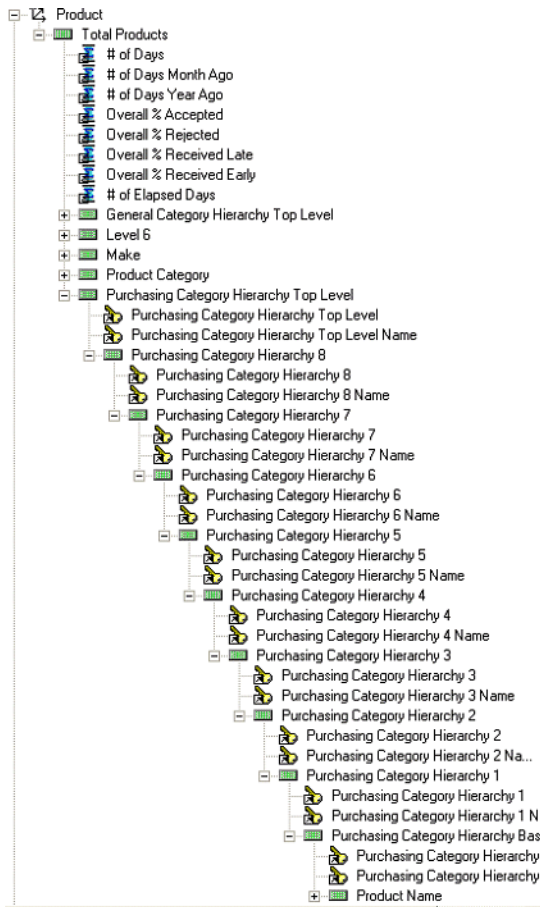
Driving: None
Type: Inner

Cardinality: N 0,1 [✓ 1] — [✓ N] 1 0,1

Hint:

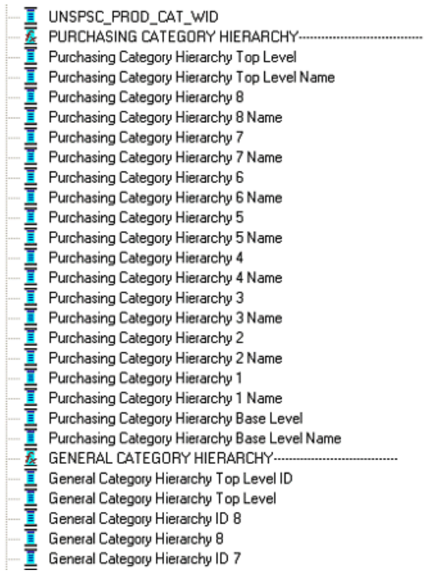
Expression:
Dim_W_PROD_CAT_DH_Purchasing.ROW_WID = Dim_W_PRODUCT_D.PROD_CAT1_WID

3. Create the new Logical Layer Dimension and Logical Layer Dimension Hierarchy.



```
Product
  Total Products
    # of Days
    # of Days Month Ago
    # of Days Year Ago
    Overall % Accepted
    Overall % Rejected
    Overall % Received Late
    Overall % Received Early
    # of Elapsed Days
    General Category Hierarchy Top Level
    Level 6
    Make
    Product Category
    Purchasing Category Hierarchy Top Level
      Purchasing Category Hierarchy Top Level
      Purchasing Category Hierarchy Top Level Name
      Purchasing Category Hierarchy 8
        Purchasing Category Hierarchy 8
        Purchasing Category Hierarchy 8 Name
        Purchasing Category Hierarchy 7
          Purchasing Category Hierarchy 7
          Purchasing Category Hierarchy 7 Name
          Purchasing Category Hierarchy 6
            Purchasing Category Hierarchy 6
            Purchasing Category Hierarchy 6 Name
            Purchasing Category Hierarchy 5
              Purchasing Category Hierarchy 5
              Purchasing Category Hierarchy 5 Name
              Purchasing Category Hierarchy 4
                Purchasing Category Hierarchy 4
                Purchasing Category Hierarchy 4 Name
                Purchasing Category Hierarchy 3
                  Purchasing Category Hierarchy 3
                  Purchasing Category Hierarchy 3 Name
                  Purchasing Category Hierarchy 2
                    Purchasing Category Hierarchy 2
                    Purchasing Category Hierarchy 2 Na...
                    Purchasing Category Hierarchy 1
                      Purchasing Category Hierarchy 1
                      Purchasing Category Hierarchy 1 N
                      Purchasing Category Hierarchy Bas
                        Purchasing Category Hierarchy
                        Purchasing Category Hierarchy
                        Product Name
```

4. Add the new columns to the Presentation Layer.



```
UNSPSC_PROD_CAT_WID
PURCHASING CATEGORY HIERARCHY
Purchasing Category Hierarchy Top Level
Purchasing Category Hierarchy Top Level Name
Purchasing Category Hierarchy 8
Purchasing Category Hierarchy 8 Name
Purchasing Category Hierarchy 7
Purchasing Category Hierarchy 7 Name
Purchasing Category Hierarchy 6
Purchasing Category Hierarchy 6 Name
Purchasing Category Hierarchy 5
Purchasing Category Hierarchy 5 Name
Purchasing Category Hierarchy 4
Purchasing Category Hierarchy 4 Name
Purchasing Category Hierarchy 3
Purchasing Category Hierarchy 3 Name
Purchasing Category Hierarchy 2
Purchasing Category Hierarchy 2 Name
Purchasing Category Hierarchy 1
Purchasing Category Hierarchy 1 Name
Purchasing Category Hierarchy Base Level
Purchasing Category Hierarchy Base Level Name
GENERAL CATEGORY HIERARCHY
General Category Hierarchy Top Level ID
General Category Hierarchy Top Level
General Category Hierarchy ID 8
General Category Hierarchy 8
General Category Hierarchy ID 7
```

6.5 About the Period Ago Keys for Oracle Business Analytics Warehouse

The Period Ago Key fields are used to set up the time series metrics like Year Ago, Quarter Ago, and so on. The Period Ago Key fields represent metrics for a prior period, for example, Quarter Ago Revenue, Year Ago Revenue, and so on. Oracle

Business Analytics Warehouse is pre-configured with a set of fields in the W_DAY_D table. These fields are:

- MONTH_AGO_WID
- QUARTER_AGO_WID
- TRIMESTER_AGO_WID
- WEEK_AGO_WID
- YEAR_AGO_WID

These fields are used in joins to Oracle Business Analytics Warehouse fact tables to achieve the period ago metrics. The joins in Oracle Business Analytics Warehouse uses the Period Ago fields in the W_DAY_D table.

6.6 About Oracle BI Time Repository Variables

The Oracle BI Repository is pre-configured with variables that are used for both reporting and internal usage.

Table 6–2 lists some example Oracle BI repository date variables and their descriptions. For a full list of variables, in Oracle BI Administration Tool, choose Manage, then Variables, to display the Variable Manager, and refer to the Description fields for a brief description.

Note: Repository variables with _OTBI and _OBIA should not be directly used. They are only used to switch between Oracle Business Intelligence Applications and Oracle Transactional Business Intelligence sourcing.

Table 6–2 Oracle BI Repository Date Variables

Variable Name	Description
CAL_MONTH_YEAR_AGO	Returns the value of Previous Year Month in the YYYY/MM format.
CURRENT_BALANCE_DK_AP	Returns the value of the last date key for the available Accounts Payable balance. It is used in Accounts Payable Account Balance Computation.
CURRENT_BALANCE_DK_AR	Returns the value of the last date key for the available Accounts Receivables balance. It is used in Accounts Receivable Account Balance Computation.
CURRENT_BALANCE_DK_GL	Returns the value of the last date key for the available General Ledger balance. It is used in General Ledger Account Balance Computation.
CURRENT_DAY	Returns the value of Current Date in the MM/DD/YYYY format.
CURRENT_FSCL_MONTH	Returns the value of Current Fiscal Month in the YYYY/MM format.
CURRENT_FSCL_QUARTER	Returns the value of Current Quarter in the YYYY Q n format.
CURRENT_FSCL_WEEK	Returns the value of Current Fiscal Week in the YYYY Week nn format.
CURRENT_FSCL_YEAR	Returns the value of Current Fiscal Year in the FYYYYY format.
CURRENT_JULIAN_DAY_NUM	Returns the value of Current Julian Date Number.
CURRENT_MONTH	Returns the value of Current Month in the YYYY/MM format.

Table 6–2 (Cont.) Oracle BI Repository Date Variables

Variable Name	Description
CURRENT_QTR	Returns the value of Current Quarter in YYYY Q n format.
CURRENT_WEEK	Returns the value of Current Week in the YYYY Week nn format.
CURRENT_YEAR	Returns the value of Current Year in the YYYY format.
FSCL_MONTH_YEAR_AGO	Returns the value of Previous Year Fiscal Month in YYYY/MM format.
FSCL_QTR_YEAR_AGO	Returns the value of Previous Year Quarter in YYYY Q n format.
NEXT_FSCL_MONTH	Returns the value of Next Fiscal Month in the YYYY / MM format.
NEXT_FSCL_QUARTER	Returns the value of Next Quarter in the YYYY Q n.
NEXT_FSCL_WEEK	Returns the value of Next Fiscal Week in the YYYY Weeknn format.
NEXT_FSCL_YEAR	Returns the value of Next Fiscal Year in the FYYYYY format.
NEXT_MONTH	Returns the value of Next Month in the YYYY / MM format.
NEXT_QUARTER	Returns the value of Next Quarter in the YYYY Q n.
NEXT_WEEK	Returns the value of Next Week in the YYYY Weeknn format.
NEXT_YEAR	Returns the value of Next Year in the YYYY format.
PREVIOUS_FSCL_MONTH	Returns the value of Previous Fiscal Month in the YYYY/MM format.
PREVIOUS_FSCL_QUARTER	Returns the value of Previous Quarter in the YYYY Q n format.
PREVIOUS_FSCL_WEEK	Returns the value of Previous Fiscal Week in the YYYY Weeknn format.
PREVIOUS_FSCL_YEAR	Returns the value of Previous Fiscal Year in the FYYYYY format.
PREVIOUS_MONTH	Returns the value of Previous Month in the YYYY/MM format.
PREVIOUS_QUARTER	Returns the value of Previous Quarter in the YYYY Q n.
PREVIOUS_WEEK	Returns the value of Previous Week in the YYYY Weeknn format.
PREVIOUS_YEAR	Returns the value of Previous Year in the YYYY format.
REF_JULIAN_DATE	Stores the start date of the Julian calendar and should not be changed.
REF_JULIAN_DATE_NUM	Stores the Julian number for the start of the Julian calendar and should not be changed.
TIME_OFFSET	Returns the difference between the current date and a given number of days value. It is primarily used for testing to simulate an earlier or later date. You could set the variable to the number of days you want the preceding date variables to be moved back.
YEAR_AGO_DAY	Returns the value of year ago date in the mm/dd/yyyy format.

6.7 About Configuring Usage Tracking for Oracle Business Analytics Warehouse

Oracle Business Analytics Warehouse supports the accumulation of usage tracking statistics. For more information on the Usage Tracking application, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)* .

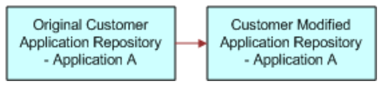
6.8 About the Incremental Deployment of the Oracle BI Applications Repository

Oracle BI Applications consists of various application families, for example, Supplier Performance Analytics, Contact Center Telephony Analytics, General Ledger and Profitability Analytics, and so on. You can purchase these applications at different times. You can customize functionality and incrementally add new applications.

This section describes the procedure for deploying multiple applications. You can repeat the procedure to add applications incrementally.

The figure below shows a single Oracle BI Applications environment. During installation, you will be asked to specify the application module(s) you have licensed, and the installer will extract the metadata corresponding to this module into one repository file. You can then modify the Oracle BI Repository to suit your business needs.

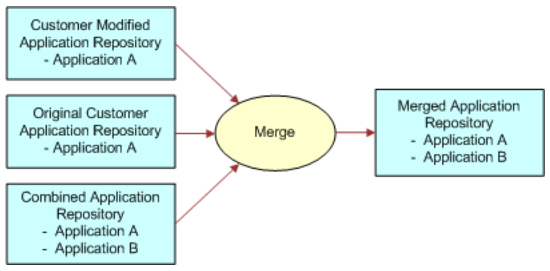
Figure 6–1 Oracle Business Analytics Warehouse Environment


Original Customer Application Repository - Application A
Customer Modified Application Repository - Application A

When you purchase another Oracle BI Applications application, you need to extract new metadata for all the modules that you have licensed. Use the merge utility in Oracle BI Administration Tool to perform a three-way merge of the original repository, the modified repository, and the combined repository. For more information on merging repositories, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

The merged repository preserves your modifications from the original Oracle BI Repository and appends the information with the new Oracle BI Repository, as shown in the figure below.

Figure 6–2 Merging with an Oracle BI Repository


Customer Modified Application Repository - Application A
Original Customer Application Repository - Application A
Combined Application Repository - Application A - Application B
Merge
Merged Application Repository - Application A - Application B

You can repeat this merging procedure to add more Oracle BI applications to the Oracle BI Repository.

7 Customizing ETL Mappings for Oracle BI Applications

This chapter provides a high-level overview of some of the common concepts and tasks required to customize Extract, Transform, and Load (ETL) mappings for Oracle Business Intelligence Applications.

Customizing Oracle BI Applications is a multifaceted process. Before implementing customizations, you should have a thorough understanding of the following:

- Oracle BI Applications
- Oracle Business Analytics Warehouse
- Your source systems
- Informatica PowerCenter
- Oracle Business Intelligence Data Warehouse Administration Console (DAC)
- Oracle BI Foundation (BI Server metadata model, Administration Tool, Answers, and dashboards)

If you are using Oracle Fusion Applications as a source system, before customizing Oracle BI Applications, you should review the section titled "Customizing Reports and Analytics," in *Oracle Fusion Applications Extensibility Guide*. This section describes how to use Oracle Business Intelligence Suite (Oracle BI Suite) to customize and extend reports and analytics for Oracle Fusion Applications, including customizing Oracle BI Publisher layouts and data models, customizing Oracle BI Enterprise Edition analyses and dashboards, and extending the Oracle BI repository.

This chapter contains the following main topics:

- Section 7.10, "About Domain Values"

7.1 Overview of ETL Customization for Oracle BI Applications

In Oracle BI Applications, ETL customization is defined as changing the preconfigured ETL logic to enable you to analyze information in new ways in your business intelligence dashboards.

This chapter explains how to customize ETL functionality after you have performed a business analysis and technical analysis. It does not cover the following tasks that you also need to perform:

- Business analysis - before you start customization, you typically analyze your current BI dashboards to determine the changes required to support your business or organization.
- Technical analysis - after you have decided upon your business requirements, you need to determine the technical changes required, which entails identifying source tables, staging tables, target tables, and Informatica transformations that you need to modify.
- Oracle BI repository modification - after making customizations to the ETL functionality, you need to modify your Oracle BI repository to expose the new data in your dashboards. For more information about Oracle BI repository modification, see *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition 11g Release 1 (11.1.1)*.
- Oracle BI dashboard modification - you need to modify your Oracle BI dashboards to display the customized data.

The most common Oracle BI Applications customizations are referred to as Category 1 and Category 2 customizations, as described below:

- **Category 1.** In a Category 1 customization, you add additional columns from source systems and load the data into existing data warehouse tables. Category 1 customizations require you to modify existing SDE and SIL mappings. For instructions, see Section 7.3, "Category 1 Customizations: Adding Columns to Existing Fact or Dimension Tables."
- **Category 2.** In a Category 2 customization, you add new fact or dimension tables to the data warehouse. Category 2 customizations normally require you to build new SDE and SIL mappings. For instructions, see Section 7.4, "Category 2 Customizations: Adding Additional Tables."

7.2 Before You Begin ETL Customization for Oracle BI Applications

This section explains important concepts that you need to know before beginning to customize ETL for Oracle BI Applications. This section contains the following topics:

- Section 7.2.1, "Understanding View Objects for Oracle Fusion Applications Sources"
- Section 7.2.2, "Upgrade Considerations"
- Section 7.2.3, "Using Custom Folders in the Oracle Business Analytics Warehouse"
- Section 7.2.4, "Creating Custom Informatica Workflows"
- Section 7.2.5, "Additional Points to Consider When Customizing Oracle BI Applications"

7.2.1 Understanding View Objects for Oracle Fusion Applications Sources

Oracle Business Analytics Warehouse is populated from Oracle Fusion Applications source databases using an ETL (extract, transform, and load) process. The extract from the source system is done using the Oracle Application Development Framework (Oracle ADF) view objects.

For detailed information about how view objects are extracted from an Oracle Fusion Applications source and loaded into Oracle Business Analytics Warehouse, see the chapter titled, "Designing and Securing View Objects for Oracle BI Applications," in *Oracle Fusion Applications Developer's Guide*. This chapter provides guidelines and best practices for designing and securing Oracle Application Development Framework (Oracle ADF) view objects and other supporting business component objects for use by Oracle BI Applications.

In addition, you can configure and enable the BI Extender functionality to propagate flexfield changes to the data warehouse. See the following sections in the *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)* for more information about these topics:

- Importing Metadata From ADF Business Component Data Sources
- Using the BI Extender to Propagate Flex Object Changes

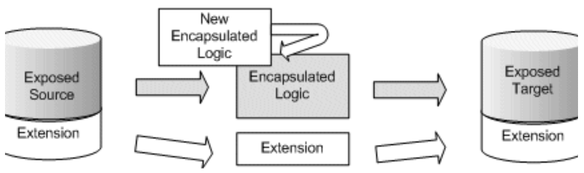
7.2.2 Upgrade Considerations

One of the most difficult aspects about working with customizations is handling the customizations at the time of an upgrade. Informatica does not provide a 'diff-merge' capability that would automatically detect changes introduced by customers and add them into upgraded mappings. Therefore, customizations must be reapplied manually to upgraded mappings. Oracle BI Applications attempts to minimize the amount of effort required to reapply customizations after an upgrade. Following the customization standards described in this chapter should help to reduce the effort required at the time of upgrade.

7.2.2.1 How Upgrade Impacts Mappings

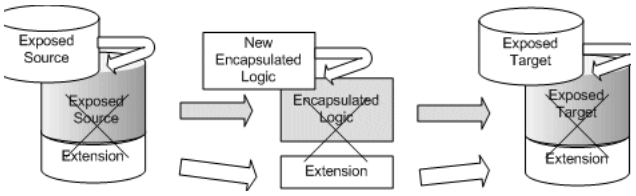
When upgrading, you will deploy customized mappings on an individual basis. By encapsulating the logic as much as possible, any changes made to the preconfigured logic can be switched as either part of a patch release or upgrade without impacting any extension logic, as shown in the following figure.

Figure 7–1 Encapsulating Logic



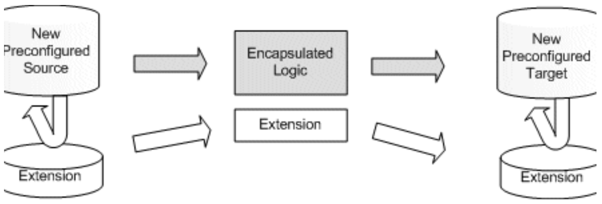
If there is a change to an exposed object, the new logic will always take precedence over the extension logic. However, rather than losing all of the extensions, much of the extension logic is retained and only has to be reapplied to the exposed objects. For example, if you add an additional column from the source and load it into the target, during an upgrade, the upgraded mapping brings the additional column from the source and loads them into the target.

Figure 7–2 Encapsulated Logic and Extension Logic



The source and target are completely replaced so any extensions to these are lost in Informatica (note that the columns will still exist in the database). However, the extension logic itself still exists after the upgrade. The source and target must be re-extended and then reconnected to the extension logic.

Figure 7–3 Re-extending and Reconnecting to Extension Logic



If you extend a mapping and the mapping does not change during an upgrade, all extensions are retained. If changes to the encapsulated logic occur, all extensions are also retained. However, if exposed objects change during the upgrade, extensions to these objects are lost, but the underlying extension logic is retained. Extensions to exposed objects must be manually reapplied.

7.2.2.1.1 Points to Remember Note the following points:

- Encapsulated objects must never be customized unless directed by Oracle. Encapsulated objects are usually mapplets and reusable transformations.
- Exposed objects can be extended but must never be otherwise modified. Exposed objects may be completely replaced at upgrade.
- Custom objects are never changed during an upgrade.
- To minimize the work required for upgrading, try to minimize the number of changes to exposed objects by using custom objects. For example, rather than adding a table to the Source Qualifier to bring in a column from a related table, add a lookup to that table in the mapping.
- In customizing objects, you must evaluate the options and determine the best approach for your environment. If you find the custom object approach allows the ETL to run in an acceptable amount of time, then this is the preferred approach. If the custom object causes the ETL process to take too long, you may want to consider incorporating the extension into an exposed object.

7.2.3 Using Custom Folders in the Oracle Business Analytics Warehouse

If you want to make changes to the Oracle Business Analytics Warehouse, you must create a custom folder and make the changes in it. Do not change objects in any of the shipped folders unless explicitly directed by Oracle. The shipped folders and the objects within them may be overwritten in future upgrades.

The deployed Informatica Repository does not include any custom folders; you must create your own. You should create a custom folder for each SDE folder you have deployed that will have customizations. These folders hold the extract mappings to various sources. You should also create a separate custom folder for customizations you want to make to the SILOS folder. Do not store customized extract and load mappings in the same folder.

The easiest way to modify an object is to copy an existing object from the shipped folder into the corresponding custom folder and to reuse existing business components, source and target definitions, transformations, mapplets, and mappings.

Note: If source tables are extended, then the source tables require manual editing in Informatica PowerCenter Designer. Do not import the tables from the database into the repository, because it changes the source table definitions for the entire Oracle Business Analytics Warehouse.

When importing new tables from the database into the custom folder, use the Oracle Business Analytics Warehouse and transactional database ODBC database connections (using the ODBC drivers supplied by your database vendor) to connect to the source and target databases.

After importing new table definitions, change the database type to Oracle in the Informatica Repository, regardless of the database platform you are using. This has no effect on the choice of a relational database. This step is very important because in Informatica if the database type of the source tables are not identical, all mappings and workflows that refer to the source tables will be invalid.

7.2.4 Creating Custom Informatica Workflows

You must create custom workflows for all customized mappings. The general requirements for creating custom workflows are as follows:

- Create the workflow such that each loads only one table. This helps to integrate the workflow with DAC.
- The workflow name should match a session name that is used inside the workflow. This helps DAC to collect certain statistics.
- The flag 'Fail parent if this task fails' should be checked for all the sessions within a workflow.
- The flag 'Fail parent if this task does not run' should be checked for all the sessions within a workflow.
- The Stop on Errors parameter should be set to 1 for all sessions within a workflow. This parameter is located in the Error Handling area of the Config Object tab in Informatica PowerCenter Designer.
- Set the appropriate source and target connection values in Informatica PowerCenter Designer.
- If the workflow is going to be used for full load command, you can choose to load in bulk mode (applicable only to Oracle and DB2-UDB databases). If the workflow is going to be used for the full load command in DAC, then in the Properties tab of Informatica PowerCenter Designer, set the Target Load type to Bulk. This requires that the target table does not have any indices during the load. DAC will drop the indices automatically; no action is required on your part.
- For all entities, such as dimension and fact tables, create two workflows, one to be used for a full load and the other to be used for an incremental load. Both

workflows are based on the same mapping. The same mapping gets executed during both full and incremental loads. This provides an opportunity to tune each of these load scenarios.

- If the workflow is designed for loading a dimension in a full mode, make sure the workflow also includes a session for the unspecified row creation.
- When defining a task in DAC, you must choose the appropriate truncate option. This helps DAC to decide whether to drop and create indices on the table.
- You should not truncate target tables using the Informatica 'truncate target' option. It is especially important that DAC handle truncating tables when extracting from more than a single source system and loading into the same data warehouse. DAC will dynamically determine when tables need to be truncated. By setting the truncate option in the Informatica workflow, the table will always be truncated, limiting your ability to extract and load data from multiple sources. For example, you cannot extract data from two systems and load the data into the same staging table in parallel as the Informatica session will truncate the staging table when another session is running.
- If some sessions have to be run in serial order and if all the sessions need to be rerun upon failure of the workflow, then you should consider designing a single workflow that has sessions run in serial. If upon failure, not all of the sessions need to be run again, then consider designing separate workflows and define dependencies in DAC.
- The custom workflows can be plugged into an ETL process by registering them in DAC. All new tasks need to be registered in DAC, with the appropriate properties set. Also, you must register in DAC, source and target tables, task definitions, and dependencies. For instructions on registering objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

7.2.5 Additional Points to Consider When Customizing Oracle BI Applications

You should consider the following points before customizing Oracle BI Applications:

- **Table definitions in Informatica.** Ensure that the SQL style is set to Oracle while importing the table definitions from external data sources. Even if the actual data source is of another database type, such as DB2 or MSSQL, it does not affect the logic of how data gets loaded.
- **Update strategy.** For loading new fact and dimension tables, design a custom process on the source side to detect the new and modified records. The SDE process should be designed to load only the changed data (new and modified). If the data is loaded without the incremental process, the data that was previously loaded will be updated again, which is a costly process. For example, the logic in the preconfigured SIL mappings looks up the destination tables based on the INTEGRATION_ID and DATASOURCE_NUM_ID and returns the ROW_WID if the combination exists, in which case it updates the record. If the lookup returns null, it inserts the record instead. In some cases, last update date(s) stored in target tables are also compared in addition to the columns specified above to determine insert or update. Look at the similar mappings in the preconfigured folder for more details.
- **ETL process.** When using multiple sources for the data warehouse, you can decide to load from all of them at the same time or at different time frequencies using different Execution Plans.

- **Truncating target tables.** Truncating should be done through DAC. A single task has place holders for a full load, and one for an incremental load.
  - For the SDE workflows, the commands for full load and incremental load are the same. They should have the Truncate Always flag checked in DAC. For these kinds of tasks, the command for full load and incremental load are based on the same mapping.
  - For SIL workflows, the command can be different for full and incremental loads. They should have the Truncate For Full Load option checked in DAC. When a table gets truncated, the indices are automatically dropped and created after the data is loaded. The workflow associated with the full load command can have the Bulk Load option turned on for an optimized version of the mapping that quickly inserts data. Note that if there are indexes on the table, the bulk load may fail, so it is very important that the indexes are registered in DAC and that you drop all of the indexes on this table during a full load if you use the bulk load option.
  - If a source requires an auxiliary task, it must be run only during incremental mode. So, for these tasks, the full load command is empty. No truncate options should be set.
- **DATASOURCE_NUM_ID.** Use a parameter to define this value in the mapping. DAC will automatically create a parameter file with the correct DATASOURCE_NUM_ID, which will be picked up by the parameter in the mapping. This enables you to make multiple copies of the same mapping when you have multiple instances of the same transactional database type. You do not have to do any additional hard-coding other than to register the sources in DAC.
- **Creating indices and naming conventions.** Staging tables typically do not require any indices. Use care to determine if indices are required on staging tables. Create indices on all the columns that the ETL will use for dimensions and facts (for example, ROW_WIDs of Dimensions and Facts, INTEGRATION_ID and DATASOURCE_NUM_ID, and flags). Carefully consider which columns or combination of columns filter conditions should exist, and define indices to improve query performance. Inspect the OTB objects for guidance. Name all the newly created tables as WC_. This helps visually isolate the new tables from the OTB tables. Keep good documentation of the customizations done; this helps when upgrading your data warehouse. Once the indices are decided upon, they should be registered in DAC, either manually or by right-clicking on the certain table and invoking the Import Indices command.
- **Currency.** For currency-related data, populate the table with the base currency and exchange date field (in order to convert the data appropriately). The data for currency conversion should be maintained in the main data source. (The currency data is maintained by converting all the currency information into a single Base Currency code specified in DAC.)
- **Day Dimension.** For the data that is related to W_DAY_D, use the reusable transformation EXP_DAY_DIMENSION_FK_RESOLUTION. Note that this transformation will take a date as input and return the foreign key to the Day dimension as output in the appropriate format (YYYYMMDD) avoiding the need of costly join or lookup to the W_DAY_D dimension table each time for resolution. Copy the reusable transformation and use it.
- **List Of Values.** This applies in particular to Category 1 and II. The preconfigured columns that depend on a list of values have a language-dependent column and a language-independent column. Use the mapplet MPLT_LOV_TRANSLATION to populate the language-dependent and independent columns in the dimension

tables. For fact tables, use MPLT_LOV_D_ROW_WID to create a new foreign key to the LOV dimension. You could also deal with translations directly in SQL overrides to improve performance.

7.3 Category 1 Customizations: Adding Columns to Existing Fact or Dimension Tables

This section includes the following topics:

- Section 7.3.1, "About Extending Oracle Business Analytics Warehouse"
- Section 7.3.2, "High-Level Steps to Extend Oracle Business Analytics Warehouse"
- Section 7.3.3, "Example of Extending Oracle Business Analytics Warehouse"

7.3.1 About Extending Oracle Business Analytics Warehouse

Category 1 customizations involve extracting additional columns from the source system and loading the data into existing data warehouse tables. In order to see additional columns in the data warehouse, the columns must first be passed through the ETL process, which requires extending the existing mappings and target tables. Oracle BI Applications provides a methodology to extend preconfigured mappings to include additional columns and then load the data into existing tables. This methodology includes sample placeholders to demonstrate how to pass and store additional data.

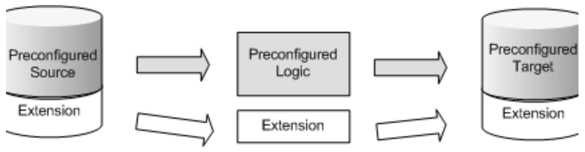
Oracle BI Applications recognizes two types of customization: extension, which is the supported customization method; and modification, which is not supported. The supported extension logic enables you to add to existing objects. For example, you can extract additional columns from a source, pass them through existing mappings, and populate new columns added to an existing table.

In contrast, Oracle BI Applications does not support modification of existing logic or columns. You should not change existing calculations to use different columns, and you should not remap existing columns to be loaded from different sources. For example, if you want to calculate revenue differently from the existing logic, you should create a new transformation and connect that calculation to a new column, for example, X_REVENUE. You can then remap the Oracle Business Intelligence repository to point to the new X_REVENUE column.

Most mappings have a single placeholder column, named X_CUSTOM, that marks a *safe path* through the mapping. All extension logic should follow the same route through the mapping as X_CUSTOM. You can add additional transformations to the mapping, but they should follow the same route through the mapping as X_CUSTOM.

In the following figure, the preconfigured logic is shaded in gray. You should not modify anything contained within these objects. You should add customizations to the existing mapping, which allows them to run parallel to the existing logic.

Figure 7–4 Preconfigured Logic and Customizations



Because some objects need to be modified in order to allow for extensions, Oracle BI Applications breaks down extensions into the following categories:

- **Exposed objects.** These objects allow changes, but the changes must be in the form of extensions (additive), and must never modify existing preconfigured logic. These objects are included in the mapping when shipped and are usually sources, targets, and nonreusable transformations.

  Note: Most SDE adapter folders use the concept of Business Component mapplets. These are extract mapplets that contain relational, application, or flat file sources. Generally, the Business Component mapplet can be treated as an exposed object and is the only mapplet object that should be modified. Keep in mind that you can modify exposed objects, but there is a risk that these changes may be lost at the time of upgrade.

- **Encapsulated objects.** These objects cannot be extended. They attempt to hide as much of the shipped transformation logic as possible to prevent breaking the preconfigured logic. These objects are included in the mapping when shipped and are usually mapplets and reusable transformations.
- **Custom objects.** You add custom objects to a mapping. (They are not shipped by Oracle.) Custom objects can be sources, transformations (reusable and nonreusable), or mapplets. Reusable transformations and mapplets that are shipped are considered encapsulated objects, but when you add such objects to an existing mapping, they are considered custom objects to that particular mapping. For example, if you want to add another amount to a fact table and that amount must be converted from its original currency to the data warehouse currency, you would normally add the existing Currency Exchange mapplet to the mapping to convert this new amount. In this case, the mapplet is considered a custom object to this particular mapping; however, it is also encapsulated, so the internal logic must not be changed.

Note: Targets should not be added to a mapping.

7.3.2 High-Level Steps to Extend Oracle Business Analytics Warehouse

The most common scenario for extending the data warehouse is to extract additional columns from a source and pass them through to an existing data warehouse table (either fact or dimension). This type of change requires extending the source dependent extract (SDE) and source independent load (SIL) mappings.

This section provides typical, high-level instructions for extending the SDE and SIL mappings to add an additional column to a data warehouse table.

To extend the SDE and SIL mappings:

1. In Informatica PowerCenter Repository Manager, create a custom folder for the custom SDE mappings and a custom folder for the custom SIL mappings.

   Common naming conventions for custom folders are the following:

   - `CUSTOM_SDE_<source system>_Adapter` for custom SDE mappings.
   - `CUSTOM_SILOS` for custom SIL mappings.

2. Copy the SDE mapping you want to customize:

   a. Locate the SDE mapping you want to extend in the standard extract folder, for example, SDE_<source system>_Adapter.

   b. In the Repository Manager toolbar, select **Edit**, then select **Copy**.

   c. Open the custom SDE folder, and select **Edit**, then **Paste**.

The mapping, any mapplets, sources, and targets are copied to the custom folder.

3. Copy the SIL mapping you want to customize from the standard SIL folder and paste it into the custom SIL folder.
4. Use a database client tool to add the new columns to the source and target tables in the database.

   Note: The last column in data warehouse tables is X_CUSTOM. As a best practice, Oracle recommends that you add new columns after X_CUSTOM and use X_ as a prefix for all custom columns to ensure there are no name conflicts with any columns Oracle may add later to that table.
5. In Informatica PowerCenter Designer, use the Target Designer to import the source and target definitions into the custom folders.
6. Extend the SDE mapping by bringing in the new columns.
   a. Open the mapping, and then open the business component mapplet. The naming convention for the business component mapplet is mplt_bc_<name>.
   b. Drag and drop the new columns from the source definition to the Source Qualifier.
   c. Modify the SQL override in the Source Qualifier to include the new columns.

      For tips on modifying the SQL override, see Section 7.3.3.3, "Tips for Modifying the SQL Override in a Source Qualifier."
   d. Drag and drop the new columns from the Source Qualifier to the 'Exp_Custom' expression.
   e. Drag and drop the new columns from the 'Exp_Custom' expression to the Output transformation, and then close the mapplet editor.
   f. Add a new custom Expression transformation to the mapping.
   g. Drag and drop the new columns from the mapplet to the new custom Expression transformation and then to the target definition.
7. Extend the SIL mapping by bringing in the new columns.
   a. Open the mapping, and drag and drop the new columns from the source definition to the Source Qualifier.
   b. Edit the SQL override in the Source Qualifier to include the new columns.
   c. Drag and drop the new columns from the Source Qualifier to the 'Exp_Custom' expression.
   d. Drag and drop the new columns from the 'Exp_Custom' expression to the Update Strategy transformation.
   e. Drag and drop the new columns from the update strategy expression to the target definition.
8. Copy the SDE workflow into the custom SDE folder and the SIL workflow into the custom SIL folder. In Informatica PowerCenter Workflow Manager, refresh and validate the workflows.
9. Register the customized mappings in DAC. You must include the table definition, with any additional columns or indexes, and the required changes so the tasks execute the modified sessions in the new custom folders.

For instructions on registering data warehouse objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

7.3.3 Example of Extending Oracle Business Analytics Warehouse

In this example data is passed from an existing source to an existing data warehouse table. The company in this example is using Oracle Fusion Applications version 11.1.3 and has identified additional fields in a base table that need to be added to the data warehouse table W_PARTY_D. The company used an extension field to capture information related to organizations referred to as 'ACCOUNT_LOG.' In addition, the company wants to include the name of the person who last updated the record as an attribute of the organization.

This section includes the following topics:

- Section 7.3.3.1, "Example of Extracting Data from an Oracle Fusion Applications Source"
- Section 7.3.3.2, "Example of Loading Data into an Existing Target Table"
- Section 7.3.3.3, "Tips for Modifying the SQL Override in a Source Qualifier"

7.3.3.1 Example of Extracting Data from an Oracle Fusion Applications Source

In this example, the company has used the HZ_CUST_ACCOUNTS.ATTRIBUTE1 field to capture data related to 'ACCOUNT_LOG.' The name of the last person to update the record is already stored in the HZ_CUST_ACCOUNTS.LAST_UPDATE_LOGIN field. There is no need to join to any additional tables.

Note: When modifying the SQL override in the Source Qualifier, use Oracle BI Server Logical SQL, which includes standard SQL, plus special functions (SQL extensions). For a detailed description of Oracle BI Server Logical SQL, see the section titled "Logical SQL Reference," in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.

If you add another table as a source, in addition to defining the join, you must also include the table's 'LastUpdateDate' in the `WHERE` clause using the following syntax:

```
OR <VIEW_OBJECT_NAME>.LastUpdateDate > TO_DATE('$$LAST_EXTRACT_DATE', 'MM/DD/YYYY
HH24:MI:SS')
)
AND
…
```

This ensures that changes to a record in that table will trigger an extract. If this were the only table to have an update and the other tables were not updated, then this change would not be detected.

Note: Most SDE adapter folders use the concept of Business Component mapplets. These are extract mapplets that contain relational, application, or flat file sources. Generally, the Business Component mapplet can be treated as an exposed object and is the only mapplet object that should be modified. Keep in mind that you can modify exposed objects, but there is a risk that these changes may be lost at upgrade time.

To extract data from an Oracle Fusion Applications source:

1. In Informatica PowerCenter Repository Manager, create a custom folder for the custom SDE mappings, and name the folder CUSTOM_SDE_FUSION_V1_ Adapter.
2. Copy the SDE_ORA_OrganizationDimension_Customer mapping and workflow to this folder.
3. Edit the target definition W_PARTY_DS to include the following columns:

Column Name	Data Type
X_ACCOUNT_LOG	VARCHAR2(10)
X_LAST_LOGIN	VARCHAR2(10)

> **Note:** If the source table had been customized, it would be necessary to re-import the source table into the custom folder, replacing the existing version. For this example, the source table has not changed.

4. Open the SDE_ORA_OrganizationDimension_Customer mapping.
5. Edit the Business Component 'mplt_BC_ORA_OrganizationDimension_Customer' by right-clicking the mapplet and selecting 'Open Mapplet.'

   Remember, the Business Component mapplets are the only mapplets you should edit. You should not edit any other mapplets unless directed by Oracle.
6. Drag the columns LAST_UPDATE_LOGIN and ATTRIBUTE1 to the Source Qualifier, and then drag these columns to the mapplet Output transformation.
7. Edit the Source Qualifier to include the new columns, as shown below:

```
SELECT
…

HZ_PARTIES.SIC_CODE
- Added by J.Smith on 1/10/2007
, HZ_CUST_ACCOUNTS.LAST_UPDATE_LOGIN
, HZ_CUST_ACCOUNTS.ATTRIBUTE1

FROM
HZ_CUST_ACCOUNTS, HZ_PARTIES
WHERE
…
```

8. Return to the mapping.
9. Add a new expression and rename it to 'X_CUSTOM.'
10. Connect the new columns from the Business Component mapplet to this expression.
11. As a best practice, you should rename these ports to indicate both the table and column they came from. If the mapping is changed and the related exposed objects are replaced, this will make it easier to reconnect, because the custom expression will not be replaced
12. Connect these columns to the appropriate columns in the target definition.
13. Save your changes.

14. Refresh and validate the session in Informatica PowerCenter Workflow Manager.

    This is necessary because it is possible that changes made to the mapping may invalidate the session.

15. Register the customized mappings in DAC. You must include the table definition, with any additional columns or indexes, and the required changes so the tasks execute the modified sessions in the new custom folders.

    For instructions on registering data warehouse objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

7.3.3.2 Example of Loading Data into an Existing Target Table

Once the required data has been extracted and staged, it must be loaded into an existing target table in the data warehouse.

To load data into an existing target table in the data warehouse:

1. In Informatica PowerCenter Repository Manager, create a custom folder for the custom SIL mappings, and name the folder CUSTOM_SILOS.
2. Copy the SIL_OrganizationDimension mapping and workflow to this folder.
3. Edit the source definition W_PARTY_DS to include the following columns:

Column Name	Data Type
X_ACCOUNT_LOG	VARCHAR2(10)
X_LAST_LOGIN	VARCHAR2(10)

4. Edit the target definition W_PARTY_D to include the following columns:

Column Name	Data Type
X_ACCOUNT_LOG	VARCHAR2(10)
X_LAST_LOGIN	VARCHAR2(10)

5. Open the SIL_OrganizationDimension mapping.
6. Drag the columns X_ACCOUNT_LOG and X_LAST_LOGIN to the Source Qualifier.
7. Drag the columns X_ACCOUNT_LOG and X_LAST_LOGIN from the Source Qualifier to the filter.

   Normally, existing transformations should not be modified. Filters are active transformations, and it is not possible to route data around an active transformation and bring it back to the same data flow. In this case, the filter is considered an exposed object and may be modified, but any changes are at risk of being lost at upgrade time.

8. Drag the columns X_ACCOUNT_LOG and X_LAST_LOGIN from the Filter to the expression EXP_Custom. If you need to apply any transformations, you should do so in this expression.
9. Drag the columns X_ACCOUNT_LOG and X_LAST_LOGIN from the expression to the Update Strategy.

The Update Strategy is another active transformation and is, therefore, considered an to be an exposed object, just like the Filter.

10. Connect these columns to the appropriate columns in the target definition.
11. Save your changes.
12. Refresh and validate the session in Informatica PowerCenter Workflow Manager.

    This is necessary because it is possible that changes made to the mapping may invalidate the session.
13. Register the customized mappings in DAC. You must include the table definition, with any additional columns or indexes, and the required changes so the tasks execute the modified sessions in the new custom folders.

    For instructions on registering data warehouse objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

7.3.3.3 Tips for Modifying the SQL Override in a Source Qualifier

- You update the SQL override in the Source Qualifier in Informatica PowerCenter Designer as follows: display the Mapping Designer tool, select the Source Qualifier object, right click and choose Edit, display the Properties tab, and edit the SQL Query value.
- When modifying the SQL override in the Source Qualifier, use Oracle BI Server Logical SQL, which includes standard SQL, plus special functions (SQL extensions). For a detailed description of Oracle BI Server Logical SQL, see the section titled "Logical SQL Reference," in *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition (Oracle Fusion Applications Edition)*.
- It is very important that the connected columns in the Source Qualifier appear in the same order in the SQL override. A common mistake is to have the ports appear in the Source Qualifier in a different order than in the SQL override.
- The column in the SELECT clause must reference the aliased name of the table if an alias is used.
- You must include a comma before new columns in a SELECT clause or before a new table in a FROM clause.
- A new table should always be defined using LEFT OUTER join syntax. Do not use INNER join or RIGHT OUTER join syntax, because you could lose records as a result.
- Make sure you define joins to match on a unique set of values. If you do not define a join that ensures a unique relationship, you may get a cartesian product, which changes the granularity and will result in duplicate errors downstream. If you cannot define a unique join, then you should bring the data in with a Lookup transformation, which guarantees that at most one record will be returned.
- As a best practice, you should comment custom code you have introduced. Comments should include at least the developer's name and the date the code was added.

7.4 Category 2 Customizations: Adding Additional Tables

This section includes the following topics:

7.4.1 About Creating New Dimension or Fact Tables

This section describes the process for building entirely new tables that will be loaded with data from a source table from which data is not already extracted. For example, you may want to create a new Project dimension table. In this case, you create new dimension and staging tables as well as new extract and load ETL mappings.

When creating a new custom table, use the prefix WC_ to help distinguish custom tables from tables provided by Oracle as well as to avoid naming conflicts in case Oracle later releases a table with a similar name. For example, for your Project dimension you may create a WC_PROJECT_DS and a WC_PROJECT_D table.

When you create a new dimension or fact table, use the required system columns that are part of each of the data warehouse tables to maintain consistency and the ability to reference existing table structures. When you create a new table, you need to register the tables and indices in DAC. You will also have to register in DAC the new tasks for new Informatica workflows and then reassemble the appropriate subject area and rebuild the appropriate execution plan. For information about assembling subject areas and building execution plans, see the *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.

7.4.1.1 Required Columns

For custom staging tables, the following columns are required:

- **INTEGRATION_ID.** Stores the primary key or the unique identifier of a record as in the source table.
- **DATASOURCE_NUM_ID.** Stores the data source from which the data is extracted.

For dimension and fact tables, the required columns are the INTEGRATION_ID and DATASOURCE_NUM_ID columns as well as the following:

- **ROW_WID.** A sequence number generated during the ETL process, which is used as a unique identifier for the data warehouse.
- **ETL_PROC_WID.** Stores the ID of the ETL process information. The details of the ETL process are stored in the W_ETL_RUN_S table on the data warehouse side. This is also the Process ID on Current Run/Run History screen in DAC.

7.4.1.2 About the Oracle Business Analytics Warehouse DATASOURCE_NUM_ID Column

All the tables in Oracle Business Analytics Warehouse schema have DATASOURCE_NUM_ID as part of their unique user key. While the transactional application normally ensures that a primary key is unique, it is possible that a primary key is duplicated between transactional systems. To avoid problems when loading this data into the data warehouse, uniqueness is ensured by including the DATASOURCE_NUM_ID as part of the user key. This means that the rows can be loaded in the same data warehouse tables from different sources if this column is given a different value for each data source.

Note: The DATASOURCE_NUM_ID is maintained by DAC. Ensure that each source system has a unique value assigned. It is possible to have multiple instances of the same source system (for example, a U.S.-based and a European-based Siebel transactional database both loading into the same data warehouse). The two different transactional database systems should be assigned different DATASOURCE_NUM_ID values in DAC. DAC is predefined with one entry for Siebel and the DATASOURCE_NUM_ID is assigned the value of 1. If you are going to extract from additional Siebel transactional database systems and load the data into the same data warehouse, a different DATASOURCE_NUM_ID must be assigned to each Siebel transactional database system.

7.4.2 Adding a New Dimension in the Oracle Business Analytics Warehouse

Follow this procedure to add a new dimension in the Oracle Business Analytics Warehouse.

To add a new dimension and use it with an existing fact table:

1. Create a DDL for the new dimension based on the standard structure (with appropriate system columns). Create a staging table for this dimension, and include the necessary columns. See Section 7.4.1.1, "Required Columns."
2. Register the new source table and its staging table (if it does not already exist) in the DAC repository and associate it with the appropriate database connection.

   For instructions on registering data warehouse objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.
3. Create a new custom map SDE_XYZ to populate the dimension stage. Instead of the actual source table (for example S_ABC), use the view that will be generated by the change capture process (for example V_ABC) in the SQL so that it extracts only the incremental data. Use existing reference maps as examples of how to populate the system columns. Make sure you truncate the stage table in corresponding tasks.
4. Create a new custom map SIL_XYZ to populate the new dimension from the stage table. Use the above referenced map as example for how to populate the system columns.
5. Register the new dimension table in DAC and associate it with the appropriate database connection.

   If you are planning to build a new dimension incrementally, assign an image suffix to the source table.
6. Register the workflows as tasks in DAC.
7. For SDE mapping of the dimension make sure you set the Build Image flag to True, and the Truncate Always option to True. And in the list of source tables, mark the primary/auxiliary source(s) of this dimension.
8. For SIL workflows of the dimension make sure you set only Truncate for Full Load option to True.

9. Make sure the target table of the SDE_XYZ is defined as source table for SIL_XYZ.

7.4.3 Adding a New Fact Table in Oracle Business Analytics Warehouse

Follow this procedure to add a new fact table in Oracle Business Analytics Warehouse.

To add a new fact table:

1. Create a DDL for the new fact based on the standard structure (with appropriate system columns). Create a staging table for this fact. See Section 7.4.1.1, "Required Columns."
2. Register the new source table (if it does not already exist) in the DAC repository and associate it with a database connection.

   For instructions on registering data warehouse objects in DAC, see *Oracle Fusion Middleware User's Guide for Oracle Business Intelligence Data Warehouse Administration Console*.
3. Create SDE mappings to populate the custom stage table. Use the view created by change capture as the main table in the SQL so that it extracts only the incremental data. Use the reference maps (above) as examples of how to populate the system columns. Be sure to truncate the stage table in corresponding workflows.
4. Create SIL mapping to populate the custom fact table. Use reference maps as examples of how to populate the system columns.
5. Use lookups or SQL override joins to dimension tables for populating dimension foreign keys (ROW_WIDs) pointing to the existing dimension.
6. Register the target tables in DAC.
7. Create new tasks for the workflows.
8. For the SDE task, make sure you have the Build Image flag set to True, and list all the source tables that it queries from. Choose one or more tables as primary or auxiliary. For the target tables choose the staging table. Set the Truncate Always flag to True.
9. For the SIL task, list all the dimensions that will be required under source tables.

7.4.4 Adding a New Dimension Table for a New Fact Table in the Oracle Business Analytics Warehouse

The steps for creating a new dimension table are similar to the steps for incremental change capture.

To add a new dimension table for a new fact table:

1. In the new custom fact loading mapping (SIL), use lookups for getting foreign keys to the new dimension.
2. Use existing maps as examples.

7.5 Other Types of Customizations Requiring Special Handling

This section includes the following topics:

- Section 7.5.1, "Adding Dimensions to Existing Facts"
- Section 7.5.2, "Adding Date Dimensions to Existing Facts"
- Section 7.5.3, "Adding Currencies to an Existing Table"

7.5.1 Adding Dimensions to Existing Facts

This section covers adding a dimension (preexisting or custom) to an existing fact. It assumes you have already built the required process to populate this dimension.

This process involves extending both the fact staging table and the fact data warehouse table to include the new column. In Informatica, remember to define the tables using the Oracle database type. The staging table should be defined as a varchar2(80) field and named with in _ID suffix. The data warehouse table column should be defined as an integer and named with a _WID suffix.

The SDE fact mapping must be modified to pass through the unique identifier of the dimension key. This assumes that there is some relationship between the base table and this unique identifier. It may already be stored in the base table or stored by joining to a related table. Depending on the source system, this identifier may be based on a single column or derived from multiple columns.

If you are adding an existing dimension, the SIL mapping should be extended to include the preexisting reusable Lookup transformation to that dimension. Pass the dimension's INTEGRATION_ID through the mapping along the path identified by the X_CUSTOM column and connect it to the Lookup after the Filter transformation. Also, connect the DATASOURCE_NUM_ID to the Lookup. If the dimension is a slowly changing dimension, the fact table's standard or 'canonical' date should be passed to the lookup as well, even if the dimension has not been enabled to capture Category 2 changes.

Remember to connect the ROW_WID of the Lookup to the X_CUSTOM transformation and include logic to default this value to 0 if no record is returned from the Lookup. Pass this column on to the Update strategy, and then on to the target.

Update DAC to include the foreign key to this dimension in the fact table's definition. You should reassemble the subject Area and rebuild the Execution Plan to ensure that DAC populates this dimension table before this fact table starts to load.

7.5.2 Adding Date Dimensions to Existing Facts

If adding a date dimension to a fact table, you merely have to pass the date itself through the SDE mapping to the stage table. In the SIL mapping, pass the date along the same path as X_CUSTOM. Add the reusable expression EXP_DAY_DIMENSION_FK_RESOLUTION after the Filter. Connect the date to any input and connect the appropriate output to the EXP_Custom transformation, then on to the Update Strategy and finally to the target.

7.5.3 Adding Currencies to an Existing Table

Amounts must be converted from the original currency to the data warehouse currency. Along with the amount, you must pass the currency code if it is not already connected in the mapping. Depending on the source system, there may be more than one currency code

Other sources may have several currency types. Be sure to read the section on configuring currency codes to get a better understanding of how these work.

If the SIL mapping does not already include it, add the mapplet MPLT_CURCY_CONVERSION_RATES after the Filter and connect all required input ports.

Connect the appropriate exchange rate(s) to the EXP_Custom expression. Use the appropriate exchange rate to convert the amount to the data warehouse currency. Pass the converted currency to the Update strategy then onto the target.

7.6 Configuring Extracts

Oracle BI Applications includes prepackaged logic to extract particular data from a particular source. You can configure the extract mappings and mapplets to accommodate additional source data. For example, if your business divides customer information into separate tables based on region, then you would have to set up the extract mapping to include data from these tables.

This section contains the following topics:

- Section 7.6.1, "Extracting New Data Using an Existing Source Table"
- Section 7.6.2, "Extracting Data from a New Source Table"
- Section 7.6.3, "Configuring a Source Adapter Mapplet"

7.6.1 Extracting New Data Using an Existing Source Table

Extract mappings generally consist of a source table or Business Component, an Expression transformation, and a staging table. If you want to extract new data using the existing mapping, you have to modify the extract mapping to include the new data by performing the following task.

To modify an existing mapping to include new data:

1. Modify the existing Business Component to extract information from the source, and add it to an appropriate extension column.

   Tip: You can perform calculation transformations in the Business Component mapplet or the source adapter mapplet in the extract mapping. However, do not use calculations in the extract that could impact performance on your source transaction system. For these types of calculations, it is recommended that you perform them in the source adapter mapplet.

2. Modify the Expression transformation to perform any necessary transformations.
3. Connect all input and output ports within the extract mapping so that the data moves from the source or Business Component to the Expression transformation and through the source adapter mapplet, and finally to the staging table's appropriate extension column.

You have to determine which type of extension column to map the data to in the staging table. After you modify the extract mapping, you would also have to modify the corresponding load mapping to make sure the extension columns that you added are connected all the way from the staging table to the warehouse table.

7.6.2 Extracting Data from a New Source Table

Business Components are packaged as mapplets, which reside in source-specific folders within the repository. Business Components are used to extract data from the source system. You can configure these mapplets to perform the following:

- **Extract data from a new source table**
- **Set incremental extraction logic**

The following procedure contains instructions for adding a new table to the Business Component. The procedure includes adding a new source definition, connecting the ports to the Source Qualifier, editing the Source Qualifier, connecting the ports to the Output transformation, and editing the Output transformation.

To add a new source table to an existing Business Component mapplet:

1. In Informatica PowerCenter Designer, open the appropriate source system configuration folder.
2. Open the Mapplet Designer tool.
3. Drag the Business Component mapplet into Mapplet Designer to view the transformations that comprise the Business Component.
4. Expand the Sources folder, and copy a source table into the mapplet by dragging and dropping the table into the Mapplet Designer.
5. Connect the applicable ports from the new Source Definition to the Source Qualifier by clicking on the port in the new source table and dragging it to the connecting port in the Source Qualifier.
6. Double-click the Source Qualifier to open the Edit Transformations box.

   In the Ports tab, make any changes to the new ports for data type, precision, scale, or all these values, as necessary.
7. Connect the applicable ports from the Source Qualifier to the Mapplet Output transformation (MAPO).

   Note: In some cases, the Business Component contains an Expression transformation between the Source Qualifier and the MAPO.

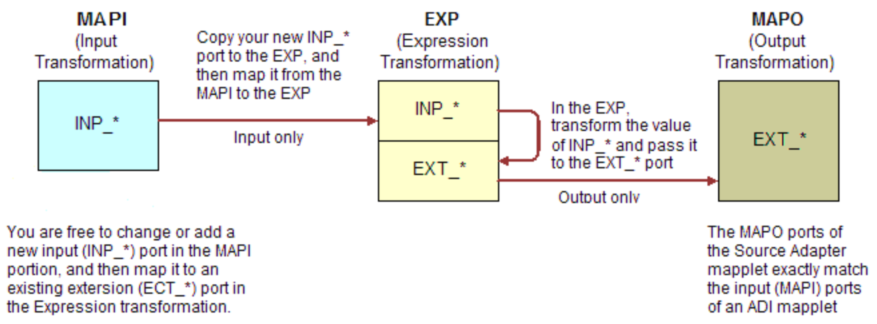
8. In the Properties tab, make changes to the SQL statement as necessary.
9. Validate and save your changes to the repository.

7.6.3 Configuring a Source Adapter Mapplet

The majority of all source-specific transformations occur in the source adapter mapplet; source-independent transformations generally take place in the Analytic Data Interface (load mapping). The source adapter mapplet converts source-specific data elements into standard formats and then stores them in a staging table. The source independent loading mapping then picks up these records, which are already transformed into standard format.

Figure 7–5 illustrates the three components of the source adapter mapplet that allow transformations of data to occur. The three components are Mapplet Input (MAPI), Expression transformation (EXP), and Mapplet Output (MAPO).

Figure 7–5 Components of the Source Adapter Mapplet


MAPI
(Input Transformation)
INP_*
Copy your new INP_* port to the EXP, and then map it from the MAPI to the EXP
Input only
EXP
(Expression Transformation)
INP_*
EXT_*
In the EXP, transform the value of INP_* and pass it to the EXT_* port
Output only
MAPO
(Output Transformation)
EXT_*
You are free to change or add a new input (INP_*) port in the MAPI portion, and then map it to an existing extersion (ECT_*) port in the Expression transformation.
The MAPO ports of the Source Adapter mapplet exactly match the input (MAPI) ports of an ADI mapplet

In Figure 7–5, if the input data is transformed, the data is passed to the Expression transformation (EXP) as input only. After the data is transformed, it is output through a new port, which is prefixed with EXT_. If the data is not transformed, it comes in as input-only and leaves through an output-only port.

If you want to add a new transformation, you must add a new port to contain the expression that is used to transform the data.

To add a new port to the source adapter mapplet:

1. In Informatica PowerCenter Designer, open the applicable source system configuration folder.
2. Open the applicable source adapter mapplet.
3. Double-click the MAPI component of the mapplet, and add a new input port following the `INP_*` naming convention.
4. Copy the new input port from the MAPI to the Expression transformation.
5. Connect the new port from the MAPI to the Expression transformation.
6. In the Expression transformation, uncheck the Output indicator for the new input port; you use the value from this port in a Transformation expression.
7. Perform any necessary transformations within the Expression transformation.

   The transformed data is passed out of an EXT_* output-only port.
8. Connect the port from the Expression transformation to the MAPO.
9. Validate and save your repository.

7.7 Marking Records as "Deleted" in Oracle Business Analytics Warehouse

In a typical implementation, records that are deleted from your source system are not removed from the Oracle Business Analytics Warehouse. If you want to mark these records as deleted in the data warehouse, you must enable the primary extract and delete mappings.

Primary extract mappings flag records that are deleted from the data warehouse. Delete mappings set the DELETE_FLG column to 'Y' for these records in the warehouse tables. When enabled, primary extract and delete mappings by default look for any records removed from the source system's database. If these mappings find that the records no longer exist in that database, the mappings mark them as deleted in the data warehouse.

Note: When you mark a record as deleted in Oracle Business Analytics Warehouse, the delete flag is set to 'Y', but the record is not physically deleted from the table. This type of delete is often referred to as a "soft delete."

Caution: Delete and primary extract mappings must always be disabled together; you may not disable only one type.

7.7.1 About Primary Extract and Delete Mappings

Before you decide to enable primary extract and delete sessions, it is important to understand their function within the Oracle Business Analytics Warehouse. Primary extract and delete mappings allow your analytics system to determine which records are removed from the source system by comparing primary extract staging tables with the most current Oracle Business Analytics Warehouse table.

The primary extract mappings perform a full extract of the primary keys from the source system. Although many rows are generated from this extract, the data only extracts the key ID and source ID information from the source table. The primary extract mappings load these two columns into staging tables that are marked with a *_PE suffix.

Figure 7–6 provides an example of the beginning of the extract process. It shows the sequence of events over a two-day period during which the information in the source table has changed. On day one, the data is extracted from a source table and loaded into the Oracle Business Analytics Warehouse table. On day two, Sales Order number three is deleted and a new sales order is received, creating a disparity between the Sales Order information in the two tables.

Figure 7–6 Extract and Load Mappings

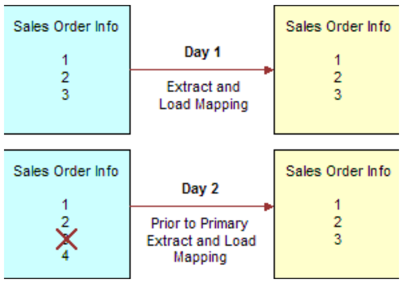
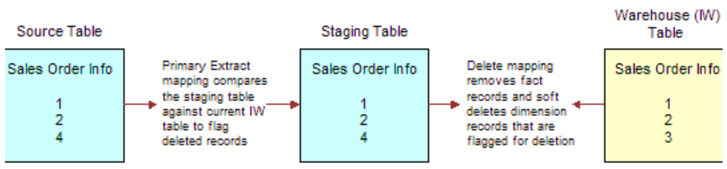


Figure 7–7 shows the primary extract and delete process that occurs when day two's information is extracted and loaded into the Oracle Business Analytics Warehouse from the source. The initial extract brings record four into the Oracle Business Analytics Warehouse. Then, using a primary extract mapping, the system extracts the key IDs and the source IDs from the source table and loads them into a primary extract staging table.

The extract mapping compares the keys in the primary extract staging table with the keys in the most current Oracle Business Analytics Warehouse table. It looks for records that exist in the Oracle Business Analytics Warehouse but do not exist in the staging table (in the preceding example, record three), and sets the delete flag to Y in the source adapter mapplet, causing the corresponding record to be marked as deleted.

The extract mapping also looks for any new records that have been added to the source, and which do not already exist in the Oracle Business Analytics Warehouse; in this case, record four. Based on the information in the staging table, Sales Order number three is marked as deleted in Oracle Business Analytics Warehouse, as shown in Figure 7–7. When the extract and load mappings run, the new sales order is added to the warehouse.

Figure 7–7 Primary Extract and Delete Mappings


Source Table
Sales Order Info
1
2
4
Primary Extract mapping compares the staging table against current IW table to flag deleted records
Staging Table
Sales Order Info
1
2
4
Delete mapping removes fact records and soft deletes dimension records that are flagged for deletion
Warehouse (IW) Table
Sales Order Info
1
2
3

7.7.2 Working with Primary Extract and Delete Mappings

The primary extract (*_Primary) and delete mappings (*_IdentifyDelete and *_ Softdelete) serve a critical role in identifying which records have been physically deleted from the source system.

Because delete mappings use source IDs and key IDs to identify purged data, if you are using multiple source systems, you must modify the SQL query statement to verify that the proper source ID is used in the delete mapping. In addition to the primary extract and delete mappings, the configuration of the delete flag in the load mapping also determines how record deletion is handled.

You can manage the extraction and deletion of data in the following ways:

- Deleting the configuration for source-archived records
- Deleting records from a particular source
- Enabling delete and primary-extract sessions
- Configuring the Record Deletion flag
- Configuring the Record Reject flag

7.7.2.1 Retaining Source-Archived Records in Oracle Business Analytics Warehouse

There are some instances when you may want to disable or remove the primary extract and delete mappings, such as when you want to retain records in the data warehouse that were removed from the source system's database and archived in a separate database.

Some sources archive records in separate databases and retain only the current information in the main database. If you have enabled the delete mappings, you must reconfigure the mappings in the Oracle Business Analytics Warehouse to retain the archived data.

To retain source-archived records in the Oracle Business Analytics Warehouse, make sure the $$LAST_ARCHIVE_DATE parameter value is set properly to reflect your archive date.

7.7.2.2 Enabling Delete and Primary Extract Sessions

If you want to mark your source-deleted records as deleted in the Oracle Business Analytics Warehouse, you need to enable the delete and primary extract tasks for your application.

As shown in the following procedures, there are two different ways to enable the sessions, depending on the modules you implemented.

To enable primary extract and delete sessions on tasks:

1. In DAC, go to the Design view, and select the appropriate custom container from the drop-down list.

2. Display the Tasks tab.
3. Query for all tasks containing the string 'Delete' or 'Primary'.
4. Deselect the Inactive check box for those tasks.
5. Reassemble your subject areas and rebuild your execution plans.

To enable primary extract and delete sessions on Configuration Tags:

1. In DAC, go to the Design view, and select the appropriate custom container from the drop-down list.
2. Display the Configuration Tags tab.
3. Query for all configuration tags containing the string 'Identify and Soft Delete'.
4. Select the tag for your module, and then click the Subject Areas subtab.
5. Deselect the Inactive check box for the subject areas.
6. Reassemble your subject areas and rebuild your execution plans.

7.8 Configuring Slowly Changing Dimensions

This section contains the following topics:

- Section 7.8.1, "About Identifying Historically Significant Attributes"
- Section 7.8.2, "About Type 1 and Type 2 Slowly Changing Dimensions"
- Section 7.8.3, "Modifying Type 2 Slowly Changing Dimensions"

7.8.1 About Identifying Historically Significant Attributes

You may want to retain a history of all the updates to a particular dimension so that you can use them in reports. These dimensions are known as *historically significant* attributes. For example, if a customer moves to a different region and you assign that customer a new regional salesperson and territory ID, you may want to keep records of that customer's account history with the original salesperson and territory ID. In this case, the salesperson and territory IDs are *historically significant* attributes. In contrast, you may have a load that populates the telephone number field. If your business does not perform data analysis on phone number history, then this information may be considered a *historically insignificant* attribute.

Identifying attributes as significant or insignificant enables you to determine the category of SCD you require. However, before you can select the appropriate type of SCD, you must understand their differences.

7.8.1.1 About the Extract View

The extract view of any given table in the staging area consists of four types of records:

- New records
- Changed records with data that is historically insignificant
- Changed records having historical significance
- Changed records whose changes have no significance of any kind and are ignored altogether

Of the four kinds of records, only the first three are of interest for the data mart. Of those three, brand new records and records whose changes are tracked as SCDs are both treated as new and become inserts into the data warehouse. Records with changes that are important but not historically tracked are overwritten in the data warehouse, based on the primary key.

7.8.2 About Type 1 and Type 2 Slowly Changing Dimensions

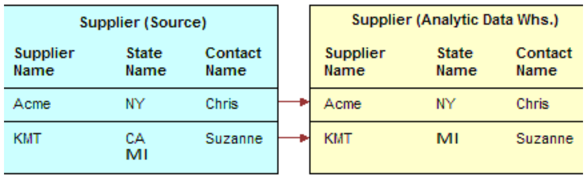
After you have correctly identified your significant and insignificant attributes, you can configure the Oracle Business Analytics Warehouse based on the type of slowly changing dimension (SCD) that best fits your needs—Type 1 or Type 2.

7.8.2.1 Type 1 Slowly Changing Dimension

A Type 1 SCD overwrites the column's value and is the default SCD for the Oracle Business Analytics Warehouse. Although a Type 1 does not maintain history, it is the simplest and fastest way to load dimension data. Type 1 is used when the old value of the changed dimension is not deemed important for tracking or is an historically insignificant attribute. For example, you may want to use Type 1 when changing incorrect values in a column.

In Figure 7–8, the State Name column for the supplier KMT is changed in the source table Suppliers, because it was incorrectly entered as California. When the data is loaded into the data warehouse table, no historical data is retained and the value is overwritten. If you look up supplier values for California, records for KMT do not appear; they only appear for Michigan, as they have from the beginning.

Figure 7–8 An Example Type 1 Slowly Changing Dimension



Supplier (Source)

Supplier Name	State Name	Contact Name
Acme	NY	Chris
KMT	CA MI	Suzanne

Supplier (Analytic Data Whs.)

Supplier Name	State Name	Contact Name
Acme	NY	Chris
KMT	MI	Suzanne

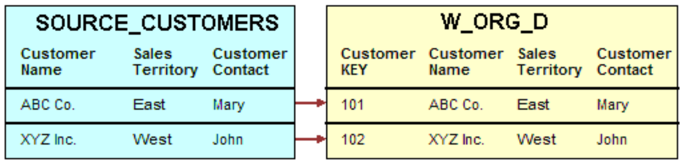
7.8.2.2 Type 2 Slowly Changing Dimension

A Type 2 SCD creates another record and leaves the old record intact. Type 2 is the most common SCD because it enables you to track historically significant attributes. The old records point to all history prior to the latest change, and the new record maintains the most current information.

Slowly changing dimensions work in different parts of a star schema (the fact table and the dimension table). Figure 7–9 shows how an extract table (SOURCE_CUSTOMERS) becomes a data warehouse dimension table (W_PARTY_D). Although there are other attributes that are tracked, such as Customer Contact, in this example there is only one *historically tracked attribute,* Sales Territory. This attribute is of historical importance because businesses frequently compare territory statistics to determine performance and compensation. Then, if a customer changes region, the sales activity is recorded with the region that earned it.

This example deals specifically with a single day's extract, which brings in a new record for each customer. The extracted data from SOURCE_CUSTOMERS is loaded into the target table W_PARTY_D, and each record is assigned a unique primary key (ROW_WID).

Figure 7–9 An Example Type 2 Slowly Changing Dimension



SOURCE_CUSTOMERS

Customer Name	Sales Territory	Customer Contact
ABC Co.	East	Mary
XYZ Inc.	West	John

W_ORG_D

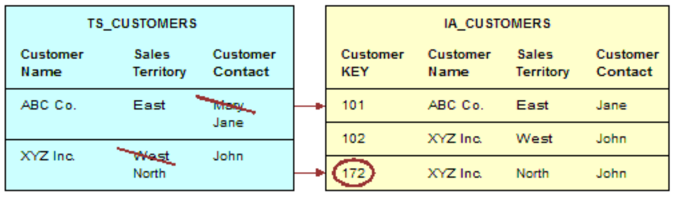
Customer KEY	Customer Name	Sales Territory	Customer Contact
101	ABC Co.	East	Mary
102	XYZ Inc.	West	John

However, this data is not static; the next time a data extract shows a change for your customers in W_PARTY_D, the records must change. This situation occurs when slowly changing dimensions are invoked. Figure 7–10 shows that records for the two customers, ABC Co., and XYZ inc. have changed. Notice that ABC's Customer Contact has changed from Mary to Jane, and XYZ's Sales Territory has changed from West to North.

As discussed earlier in this example, the Customer Contact column is historically insignificant; therefore a Type 1 SCD is applied and Mary is overwritten with Jane. Because the change in ABC's record was a Type 1 SCD, there was no reason to create a new customer record. In contrast, the change in XYZ's record shows a change of sales territory, an attribute that is historically significant. In this example, the Type 2 slowly changing dimension is required.

As shown in Figure 7–10, instead of overwriting the Sales Territory column in the XYZ's record, a new record is added, assigning a new ROW_WID, 172, to XYZ in W_PARTY_D. XYZ's original record, 102, remains and is linked to all the sales that occurred when XYZ was located in the West sales territory. However, new sales records coming in are now attributed to ROW_WID 172 in the North sales territory.

Figure 7–10 An Example Type 2 Slowly Changing Dimension



TS_CUSTOMERS

Customer Name	Sales Territory	Customer Contact
ABC Co.	East	~~Mary~~ Jane
XYZ Inc.	~~West~~ North	John

IA_CUSTOMERS

Customer KEY	Customer Name	Sales Territory	Customer Contact
101	ABC Co.	East	Jane
102	XYZ Inc.	West	John
172	XYZ Inc.	North	John

7.8.2.3 Effective Dates

Effective dates specify when a record was effective. For example, if you load a new customer's address on January 10, 2003 and that customer moves locations on January 20, 2003, the address is only effective between these dates. Effective Dates are handled in the following manner:

- If the source supplies both effective dates, these dates are used in the warehouse table.
- If the source does not supply both the effective to and effective from dates, then the Type 2 logic creates effective dates.
- If the source supplies one of the two effective dates, then the Oracle Business Analytics Warehouse automatically populates the missing effective dates using a wrapper mapping. This situation is discussed in this section.

For example, in the W_PARTY_D table previously discussed, XYZ moved to a new sales territory.

If your source system supplied historical data on the location changes, your table may contain a record for XYZ in the West sales territory with an effective from date of

January 1, 2001 and an effective to date of January 1, 3714. If the next year your source indicates XYZ has moved to the North sales territory, then a second record is inserted with an effective from date of January 1, 2002, and an effective to date of January 1, 3714, as shown in Table 7–1.

Table 7–1 Records Before a Wrapper Session in W_CUSTOMER

Customer Name	Sales Territory	Customer Contact	Effective From	Effective To	Current
ABC	East	Jane	1/1/2001	1/1/3714	Y
XYZ	West	John	1/1/2001	1/1/3714	Y
XYZ	North	John	1/1/2002	1/1/3714	Y

Note your first record for XYZ still shows as effective from January 1, 2001 to January 1, 3714, while a second record has been added for XYZ in the North territory with the new effective from date of January 1, 2002. In this second record the effective to date remains the same, January 1, 3714.

When the wrapper session executes, the effective dates for the first XYZ are corrected (January 1, 2001-January 1, 2002), and the Current Flag is adjusted in the Analytic Data Interface (load mapping) so that only the second record (January 1, 2002-January 1, 3714) is set to Y. After the wrapper session completes its work, you have Type 2 information for XYZ in your data warehouse rather than two disparate records, as shown in Table 7–2.

Table 7–2 Records After a Wrapper Session in W_CUSTOMER

Customer Name	Sales Territory	Customer Contact	Effective From	Effective To	Current
ABC	East	Jane	1/1/2001	1/1/3714	Y
XYZ	West	John	1/1/2001	1/1/2002	N
XYZ	North	John	1/1/2002	1/1/3714	Y

In the previous paragraph, the wrapper session corrected the effective to dates and current flag. However, if the record's dates had been correct, the wrapper mapping would simply have set the current flag as needed, because its logic is set to check dates and flags and only adjust columns that contain discrepancies.

7.8.3 Modifying Type 2 Slowly Changing Dimensions

This section contains the following topics:

- Section 7.8.3.1, "Enabling a Type 2 SCD Dimension"
- Section 7.8.3.2, "Extending the Logic That Triggers a Type 2 Change"

7.8.3.1 Enabling a Type 2 SCD Dimension

If you want to capture historical changes with dimensions that are configured as Type 1 dimensions, you need to modify them so that they can capture Type 2 changes. A common form of customization is to change the criteria that triggers a Type 2 change in a dimension. Most changes in a dimension are treated as Type 1 changes in that the existing column is simply overwritten with the new value.

Oracle Business Analytics Warehouse provides Type 2 slowly changing dimension (SCD) functionality, which enables you to track the history of updates to dimension records. When a record in Oracle Business Analytics Warehouse has an update, the updated information is posted into a new row and the old information is kept for historical reporting purposes.

Oracle Business Analytics Warehouse identifies and applies the slowly changing dimension logic chosen by the user after data has been extracted and transformed to be source-independent. Users may configure Oracle BI Applications to support both Type 1 SCDs, in which data is overwritten with updates, and Type 2 SCDs, in which the original records are maintained while a new record stores the updated data. Choosing Type 1 or Type 2 SCDs depends on identifying your historically significant attributes.

Most dimensions use Type 1 updates by default. If you need to change a dimension to Type 2 SCD update, use the following procedure.

To enable a Type 2 SCD dimension:

1. In DAC, go to the Design view, and select the appropriate custom container from the drop-down list.
2. Display the Tasks tab.
3. Query for the SIL task that populating the dimension.
4. Display the Parameters subtab, and set the value for $$TYPE2_FLG to Y.

Note: If you want to turn off the TYPE2_FLG for any dimension set by default to be Type 2 SCDs, you must customize the source-dependent extract mapping. To do so, make the following changes:

1. If the SQL is not yet overridden, you must override it to filter out the historical as well as the future effective records. In other words, the ultimate SQL should always fetch the latest records, as effective at the time of the session run. If the SQL is already overridden, change it to achieve the same result mentioned above. There should not be any duplicate INTEGRATION_ID values in the staging table as a result of the SQL. Although the unique index, if any, typically is on INTEGRATION_ID, DATASOURCE_NUM_ID, and SRC_EFF_FROM_DT, you should form your SQL in such a way that the uniqueness is not violated, even if SRC_EFF_FROM_DT is kept out of the equation.

   The SQL change will vary depending on the OLTP data model and the entity in question. In some cases, you can get the record based on the MAX(EFFDT) from the source, whereas in other cases, where future data is supported, you can select only records effective at the time of the session run. The Informatica parameter $$$SessStartTime gives the system date and time of the computer where Informatica Server is running. Note that this might be different than the database hosting your OLTP data.
2. Disconnect the ports that bring the source effective from and to dates.
3. In a downstream transformation, hardcode the value of SRC_EFF_FROM_DT as 01/01/1899 (or use the Oracle BI Applications standard $$LOW_DT parameter value). The SRC_EFF_TO_DT can be left as NULL.
4. Set the value of $$TYPE2_FLG as 'N' for the task loading the table where you intend this behavior.
5. Optionally, you can disable the corresponding SCDUpdate mapping, if you want to save on overall ETL time. If you choose to disable it, you might have to rebuild your execution plan.

   Note that keeping the SCDUpdate mapping enabled does not harm the functionality.

7.8.3.2 Extending the Logic That Triggers a Type 2 Change

Once enabled, there are only a small number of columns that will trigger a Type 2 change. You can extend the logic that triggers a Type 2 change by adding additional columns to the logic that tracks Type 2 changes. In addition, you can remove columns from this logic in case you do not want these types of changes to trigger a Type 2 change. Modifying the Type 2 tracking logic is one of the only exceptions to the rule that you should not make changes to shipped logic. The logic that tracks Type 2 changes is contained in exposed objects in each SIL dimension mapping that supports Type 2 changes.

There is a lookup between the Source Qualifier and the filter. This lookup is used to determine if the record already exists in the target and, therefore, must be updated in addition to other system columns. Columns that track Type 2 changes are returned in this lookup and passed to the next expression. The columns returned by the lookup are compared with the columns passed from the staging table. If any of these columns are different, the record is flagged for a Type 2 change.

This expression contains a variable port named 'TYPE2_COLS_DIFF'. If this port is flagged as 'Y' then a Category 2 change will be triggered. If it is flagged as 'N' then a Type 1 change will be triggered.

To change the columns used to determine a Type 2 change, modify the lookup to pass any additional columns you want to be evaluated for Type 2 changes. Then, modify the variable port 'TYPE2_COLS_DIFF' to include this column when being evaluated.

For example, the SIL_BusinessLocationDimension mapping compares the following columns:

```
BUSN_LOC_NAME
CITY_NAME
POSTAL_CODE
```

If you wanted to include COUNTY as part of Type 2 logic, you would change the expression for 'TYPE2_COLS_DIFF' from the following:

```
IIF(VAR_BUSN_LOC_NAME != VAR_LKP_BUSN_LOC_NAME, 'Y',
IIF(VAR_CITY_NAME != VAR_LKP_CITY_NAME, 'Y',
IIF(VAR_POSTAL_CODE != VAR_LKP_POSTAL_CODE, 'Y', 'N')))
```

To this:

```
IIF(VAR_BUSN_LOC_NAME != VAR_LKP_BUSN_LOC_NAME, 'Y',
IIF(VAR_CITY_NAME != VAR_LKP_CITY_NAME, 'Y',
IIF(VAR_POSTAL_CODE != VAR_LKP_POSTAL_CODE, 'Y',
IIF(VAR_COUNTY != VAR_LKP_COUNTY, 'Y', 'N'))))
```

As mentioned previously, you would have to modify the Lookup transformation and pass the COUNTY column and store in LKP_COUNTY, and then null-evaluate it and store it in a variable VAR_LKP_COUNTY. Also, the COUNTY column that came in from the stage table should also be null-evaluated and stored in another variable column called VAR_COUNTY.

Now that COUNTY is part of Type 2 logic, you need to remove this column from any other mappings that still assume this is a Type 1 column. The only other mapping that makes this assumption is SIL_BusinessLocationDimension_SCDUpdate. This mapping consists of the source table, followed by a Source Qualifier transformation, an Expression transformation, a Filter transformation, and finally the target table.

To ensure that the newly added column COUNTY is treated as a Type 2 column in SIL_BusinessLocationDimension_SCDUpdate, make the following required changes:

1. Source Qualifier transformation changes:
   - Delete the COUNTY port.
   - From the SQL override, remove the item "TARGET_TABLE.COUNTY" from the SELECT clause.
2. Expression transformation changes:
   - Delete the input-only port COUNTY.
   - Delete the corresponding variable port COUNTY_VAR.
   - Delete the corresponding output-only port COUNTY_OUT.
3. Filter transformation changes:
   - Delete the port COUNTY_OUT.

Note: The previous example illustrates a situation where you want to "add" a new column to the Type 2 set. In situations where you want to "remove" a column from the Type 2 set, you would need to do the opposite steps. For example, assume that you want to remove CITY_NAME from being a Type 2 column:

1. In the mapping SIL_BusinessLocationDimension, the TYPE2_COLS_DIFF expression would reduce to:

```
IIF(VAR_BUSN_LOC_NAME != VAR_LKP_BUSN_LOC_NAME, 'Y',
IIF(VAR_POSTAL_CODE != VAR_LKP_POSTAL_CODE, 'Y', 'N'))
```

2. In the mapping SIL_BusinessLocationDimension_SCDUpdate, the following changes are required:

   - Source qualifier: Instead of removing a column, you will now add CITY_NAME to the select clause.

   - Expression transformation: Follow other examples of Category 1 columns to add three new ports corresponding to CITY_NAME.

   - Filter transformation: Add the new port CITY_NAME.

   - Make the appropriate connections.

7.9 About Resolving Dimension Keys

By default, dimension key resolution is performed by the Oracle Business Analytics Warehouse in the load mapping. The load mapping uses prepackaged, reusable Lookup transformations to provide prepackaged dimension key resolution. This section describes how dimension keys are looked up and resolved.

There are two commonly used methods for resolving dimension keys. The first method, which is the primary method used, is to perform a lookup for the dimension key. The second method is to supply the dimension key directly into the fact load mapping.

7.9.1 Resolving the Dimension Key Using Lookup

If the dimension key is not provided to the load mapping through database joins, the load mapping performs the lookup in the dimension table. The load mapping does this using prepackaged Lookup transformations.

The load mapping uses the Integration ID, the DATASOURCE_NUM_ID and Lookup date in looking up the dimension key. All these columns are necessary for the load mapping to return the dimension key. The ports are described in Table 7–3.

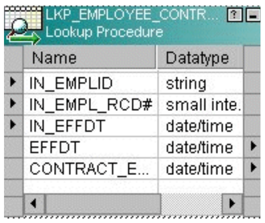
Table 7–3 Columns Used in the load mapping Dimension Key Lookup

Port	Description
INTEGRATION ID	Uniquely identifies the dimension entity within its source system. Formed from the transaction in the source adapter of the fact table.
DATASOURCE_NUM_ID	Unique identifier of the source system instance.
Lookup Date	The primary date of the transaction; for example, receipt date, sales date, and so on.

If Type 2 slowly changing dimensions are enabled, the load mapping uses the unique effective dates for each update of the dimension records. When a dimension key is looked up, it uses the fact's primary date to resolve the appropriate dimension key.

The effective date range gives the effective period for the dimension record. The same entity can have multiple records in the dimension table with different effective periods due to Type 2 slowly changing dimensions. This effective date range is used to exactly identify a record in its dimension, representing the information in a historically accurate manner. In the lookup for Employee Contract Data shown in Figure 7–11, you can see the effective dates used to provide the effective period of employee contracts.

Figure 7–11 Lookup for Employee Contract Data



Name	Datatype
IN_EMPLID	string
IN_EMPL_RCD#	small inte.
IN_EFFDT	date/time
EFFDT	date/time
CONTRACT_E...	date/time

7.10 About Domain Values

The Oracle Business Analytics Warehouse foundation comprises a data model that accommodates data from disparate source systems. Data is sourced from operational systems and systematically molded into a source-independent format. After the data is made source independent, it can then be used to create key metrics for analytic reporting, so that metric calculations are not source dependent. This clear separation enables you to swap source systems or integrate additional source systems without having to reconfigure the metric calculations to accommodate each source system's requirements.

One method for transforming source data into a source-independent format is to convert the source-supplied domain values to conformed domain values. Conformed domain values are a set of distinct values used to calculate prepackaged metrics. These values are provided by the Oracle Business Analytics Warehouse to allow you to create metric calculations independent of source domain values.

Using Oracle BI Applications Configuration Manager, you can extend conformed domains and the domain mappings. For instructions, see the section titled, "About Working With Domains and Domain Mappings," in *Oracle Fusion Middleware Configuration Guide for Oracle Business Intelligence Applications*.

For additional general information about Oracle BI Applications domains, see "About Oracle BI Applications Domains," in *Oracle Fusion Middleware Reference Guide for Oracle Business Intelligence Applications*.

7.10.1 About the Domain Value Conversion Process

To best understand the domain value conversion process, consider an example of two source systems—Source System A and Source System B. Each source system stores two types of employee events—hire and rehire. Source system A uses H to denote a hire event and R to denote a rehire event, whereas source system B uses 1 to denote a hire event and 2 to denote a rehire event. When the Oracle Business Analytics Warehouse extracts data from both systems, it ports those source values through the extract mapping until the data reaches the W_EVENT_GRP_CODE column in the W_EVENT_TYPE_DS staging table.

The load mapping then ports the extracted source values (H and R from source system A, and 1 and 2 from source system B) into the source adapter mapplet. Within the source adapter, source values are translated into domain values (HIR and REH) based on a set of rules that are particular to your business practices.

7.10.1.1 Preparing to Define the Rules

You must define the rules so that the source adapter knows how to map your specific source values to the given set of conformed domain values. Before you set up the rules you must:

1. Analyze all of your source values and how they map to the prepackaged conformed domain values. You may find that you need to create additional domain values for particular columns. The result of this preparation work is a list of each source value and how it is mapped to a conformed domain value.
2. Implement this logic in the applicable source adapter mapplet. To set up the logic, modify the Expression transformation in the source adapter mapplet for each affected column.

Figure 7–12 illustrates how the source values are converted to the domain values—HIR and REH.

Figure 7–12 Source Values Translated to Domain Values

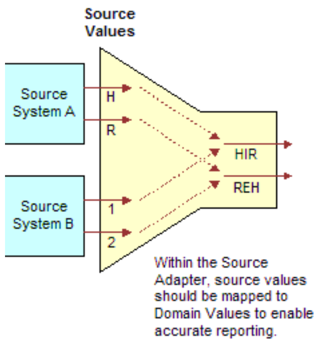
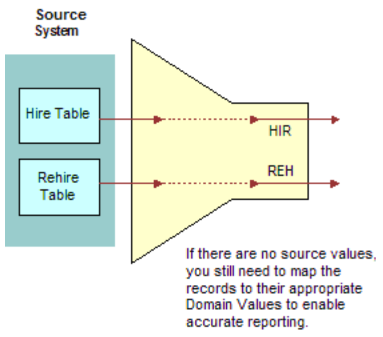

Source Values
Source System A
Source System B
H
R
1
2
HIR
REH
Within the Source Adapter, source values should be mapped to Domain Values to enable accurate reporting.

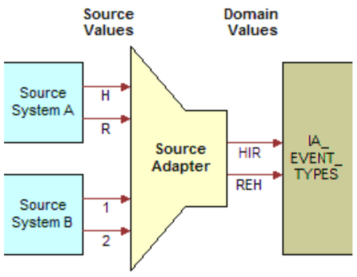
Figure 7–13 illustrates a different situation where the records may not contain a source value that flags the record as Hire or Rehire. In this case, the source system stores hires in one table and rehires in another table. To make this work, one possible solution is to modify the extract mappings to populate the W_EVENT_GRP_CODE column with HIR or REH. If the field is populated in the extract mapping, you can then carry those same values through the source adapter mapplet.

Figure 7–13 Source Values in Different Tables Translated to Domain Values


Source System
Hire Table
Rehire Table
HIR
REH
If there are no source values, you still need to map the records to their appropriate Domain Values to enable accurate reporting.

After the source adapter mapplet converts the source-specific values to conformed domain values, the conformed domain values are inserted into an Oracle Business Analytics Warehouse table. In this example, the HIR and REH values populate the W_EVENT_TYPES table, as illustrated in Figure 7–14.

Figure 7–14 HIR and REH Values Populating the W_EVENT_TYPES Table



7.10.2 About the Importance of Domain Values

Values in the W_EVENT_TYPES table are used to create metrics in the front end. Some metrics are defined using conformed domain values. For example, seven metrics use the HIR and REH event group code in their calculation. The following are the seven metrics, along with their descriptions and calculations:

7.10.2.1 Hire Count

This metric counts all hires for a specified period. The calculation is:

```
SUM(CASE WHEN (CMMNEVTP.W_EVENT_GRP_CODE IN ('HIR','REH')) THEN EVNT.EVENT_CNT
ELSE 0 END)
```

7.10.2.2 Re-hires Ratio

This metric determines the ratio of rehires to all employees hired during a specified period. The calculation is:

```
CASE WHEN SUM(CASE WHEN CMMNEVTP.W_EVENT_GRP_CODE IN ('REH','HIR') THEN
EVNT.EVENT_CNT ELSE 0 END) = 0 THEN 0 ELSE SUM(CASE WHEN CMMNEVTP.W_EVENT_GRP_CODE
IN ('REH') THEN EVNT.EVENT_CNT ELSE 0 END)/SUM(CASE WHEN CMMNEVTP.W_EVENT_GRP_CODE
IN ('REH','HIR') THEN EVNT.EVENT_CNT ELSE 0 END) END
```

7.10.2.3 New Hire Count

This metric counts the head count hired for regular full-time positions. The calculation is:

```
SUM(CASE WHEN CMMNEMPT.FULL_TIME_FLAG = 'Y' AND CMMNEMPT.EMP_CAT_CODE = 'R' AND
(CMMNEVTP.W_EVENT_GRP_CODE = 'HIR' OR CMMNEVTP.W_EVENT_GRP_CODE = 'REH') AND
EVNT.EVENT_DK >= (CMMNDATE.DATE_KEY - 365) AND EVNT.EVENT_DK <= CMMNDATE.DATE_KEY
THEN EVNT.EVENT_CNT ELSE 0 END)
```

7.10.2.4 Newly Separated Veterans - New Hires

This metric counts the regular full-time and part-time employees who belong to this category of veterans and were hired during the previous 12 months. The calculation is:

```
SUM(CASE WHEN CMMNEMPD.VETERAN_STAT_CODE = '4' AND CMMNEMPT.EMP_CAT_CODE = 'R' AND
(CMMNEVTP.W_EVENT_GRP_CODE = 'HIR' OR CMMNEVTP.W_EVENT_GRP_CODE = 'REH') AND
EVNT.EVENT_DK >= (CMMNDATE.DATE_KEY - 365) AND EVNT.EVENT_DK <= CMMNDATE.DATE_KEY
THEN EVNT.EVENT_CNT ELSE 0 END)
```

7.10.2.5 Other Protected Veterans - New Hires

This metric counts regular full-time and part-time employees who belong to this category of veterans. The calculation is:

```
SUM(CASE WHEN CMMNEMPD.VETERAN_STAT_CODE = '3' AND CMMNEMPT.EMP_CAT_CODE = 'R' AND
(CMMNEVTP.W_EVENT_GRP_CODE = 'HIR' OR CMMNEVTP.W_EVENT_GRP_CODE = 'REH') AND
EVNT.EVENT_DK >= (CMMNDATE.DATE_KEY - 365) AND EVNT.EVENT_DK <= CMMNDATE.DATE_KEY
THEN EVNT.EVENT_CNT ELSE 0 END)
```

7.10.2.6 Special Disabled Veteran Head count - New Hires

This metric counts regular full-time and part-time employees who belong to this category of veterans and were hired during the previous 12 months. The calculation is:

```
SUM(CASE WHEN CMMNEMPD.VETERAN_STAT_CODE = '1' AND CMMNEMPT.EMP_CAT_CODE = 'R' AND
(CMMNEVTP.W_EVENT_GRP_CODE = 'HIR' OR CMMNEVTP.W_EVENT_GRP_CODE = 'REH') AND
EVNT.EVENT_DK >= (CMMNDATE.DATE_KEY - 365) AND EVNT.EVENT_DK <= CMMNDATE.DATE_KEY
THEN EVNT.EVENT_CNT ELSE 0 END)
```

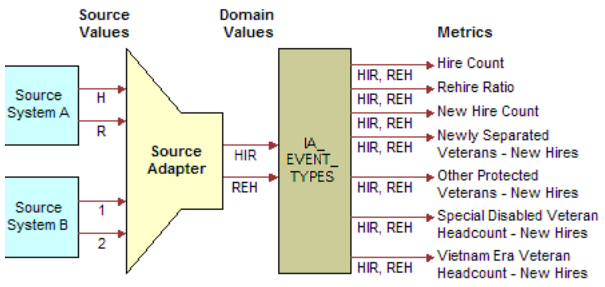
7.10.2.7 Vietnam Era Veteran Head count - New Hires

This metric counts regular full-time and part-time employees who belong to this category of veterans and were hired during the previous 12 months. The calculation is:

```
SUM(CASE WHEN CMMNEMPD.VETERAN_STAT_CODE = '2' AND CMMNEMPT.EMP_CAT_CODE = 'R' AND
(CMMNEVTP.W_EVENT_GRP_CODE = 'HIR' OR CMMNEVTP.W_EVENT_GRP_CODE = 'REH') AND
EVNT.EVENT_DK >= (CMMNDATE.DATE_KEY - 365) AND EVNT.EVENT_DK <= CMMNDATE.DATE_KEY
THEN EVNT.EVENT_CNT ELSE 0 END)
```

Each of these metric calculations is based on the conformed domain values HIR and REH. All records whose source values are converted to one of these conformed domain values are included in the metric calculations, as shown in Figure 7–15.

Figure 7–15 Metric Values From HIR and REH Domain Values



Index