

Oracle Insurance Insight

**Oracle Insurance Insight
Implementation Guide**

version 7.0.2

Part Number: E24031-01

October 2011

Oracle Insurance Insight Implementation Guide

Version 7.0.2

Part Number: E24031-01

October 2011

Copyright © 2011, Oracle. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Oracle, JD Edwards, and PeopleSoft are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

THIRD PARTY SOFTWARE NOTICES

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The Apache Software License, Version 2.0 Copyright (c) 2004 The Apache Software Foundation. All rights reserved.

The Apache Software License, Version 1.1 Copyright (c) 1999-2003 The Apache Software Foundation. All rights reserved.

CONTENTS

Preface

- v **Version**
- v **Purpose & Audience**
- vi **List of Acronyms and Abbreviations**
- vii **Naming Conventions**
- vii **Related Documents**
- viii **Customer Support**

Section I - Strategic Goals 1

Chapter 1: Introduction

- 3 **What Lines of Business Are Supported?**
- 3 **What Is Included in the Core Data Model?**
- 4 **What Are Dimensions?**
- 5 **What is the Adaptive Data Model?**
- 5 **What Are Slowly-Changing Dimensions?**
- 6 **What Are Metrics?**
- 7 **What Is ODI (Data Loading)?**
- 7 **What Is OBIEE (Data Visualization)?**

Section II - Project Planning 9

Chapter 2: Implementation Lifecycle

- 11 **Standard OII Project Stages**
- 12 **Standard OII Implementation Process**
 - 12 Data Mapping
 - 13 Source-to-Staging ETL Development

- 14 Data Mart Load
- 14 Customize Oll Data Views

Section III - Implementation 15

Chapter 3: Program Flow

17 Staging Schema Concepts

- 18 Party Model
- 18 Contact
- 19 Postal Address
- 19 Natural Keys
- 21 Row Number
- 21 Relationship Model

22 Normalization and Denormalization

22 User-Defined Fields

23 Concepts in Practice

24 Flow Diagrams

- 24 Loading to Staging
- 26 Staging to Warehouse
- 28 Warehouse to Data Mart

Chapter 4: Source-to-Staging ETL

33 Master Data Management Strategy

- 33 Metadata
- 35 Required Tables

36 Data Validation

43 Data Relationships

- 43 Unique Keys
- 43 Common/Shared Keys

44 Data Loading Considerations

44 Personal Auto Premiums

Chapter 5: Data Loading

45 System Codes

46 Source Code Translation and Description

46 Claim Transaction Codes

47 Policy Transaction Codes

48 Load Configuration

48 Analysis Options

48 Currencies

48 Default Begin Date

49 Default Date Format

49 Default Date/Time Format

49 Default End Date

49 Default Numeric Value

49 Default String Value

49 Incurred Loss Calculation Method

50 Incurred Loss with Expense Calculation Method

50 Late Arriving Methodology

50 Log Name/Path

51 Currency Configuration

51 Default Currency Conversion Rates

53 Multiple Currencies and Conversion Rates

55 Load Execution

55 Complete Load

56 Warehouse Load

57 Data Mart Load

58 Dimensions and Transactional Facts Load

59 Monthly Snapshot Facts Load

59 Load Scheduling

60 Clearing the OBIEE Cache at the End of the Data Load

Chapter 6: Data Visualization

61 OBIEE Repository

62 OBIEE Repository Linkage

62 Step 1: Open the OBIEE Repository

63 Step 2: Configure the Line of Business Subject Areas

67 Step 3: Configure the Insight700Config Subject Area

68 Step 4: Check in the Changes to the Physical Layer

69 Step 5: Restart the Services

71 OBIEE Repository Linkage for New or Updated LOB

71 Step 1: Perform an Import

73 Step 2: Regenerate the Metadata Dictionary

76 Step 3: Deploy the MetaData Dictionary

77 Step 4: Perform OBIEE Repository Linkage Steps

Chapter 7: Balancing Reports

79 Overview

79 Components

80 Deploying the Balancing Reports

80 Prerequisites

81 Step 1: Create a User for BI Publisher

85 Step 2: Create a JDBC Data Source

89 Balancing Report Descriptions

89 Claims

89 Policy

89 Premium A

89 Premium B

89 Premium and Exposure

90 Report Components

92 Data Model

92 List of Values

93 Parameters

- 93 Templates
- 93 Layouts
- 95 Viewing Reports**
- 96 Modifying Reports**
 - 96 Changing a Template
 - 97 Online Template Modification
 - 99 Offline Template Modification
 - 100 Modifying a Template
 - 103 Using the Wizard to Generate Tables and Fields
 - 106 Viewing the Output
 - 108 Customizing the Report
- 113 Appendix A: Relationship Codes**
 - 113 Party-Level Relationship Codes**
 - 113 Type-Level Relationship Codes**
 - 114 Role-Level Relationship Codes**
 - 115 Contact Codes**
- 117 – Index**

Preface

VERSION

This manual corresponds to Oracle Insurance Insight (OII) 7.0.2.

PURPOSE & AUDIENCE

The *OII Implementation Guide* is written for both a non-technical and a technical audience. The chapters are grouped into three content categories or sections:

Section I - Strategic Goals (Big Picture Ideas)

- **Chapter 1 - Introduction**
 - Presents the business objectives that OII was designed to address.

Section II - Project Planning (Managing Timelines)

- **Audience** – Project managers
- **Chapter 2 - Implementation Lifecycle**
 - A standardized approach to implementing the OII system from scratch.

Section III - Implementation (Technical Details)

- **Audience** – Implementation team
- **Chapter 3 - Program Flow**
 - Familiarizes the user with the OII data architecture.
- **Chapter 4 - Source-to-Staging ETL**
 - Outlines the key goals and best practices for building an Extraction-Translation-Loading (ETL) layer that feeds OII.
- **Chapter 5 - Data Loading**
 - A step-by-step guide to loading data from Staging to Warehouse to Data Marts.
- **Chapter 6 - Data Visualization**
 - A step-by-step guide to updating and linking the Oracle Business Intelligence Enterprise Edition (OBIEE) Repository.

- **Chapter7 - Balancing Reports**
 - Balancing Reports are a set of reports that can be opened and edited in BI Publisher to verify data flow between schemas in OII. This chapter describes how to deploy and manage these reports for use in BI Publisher.

For the purpose of the guide, it is assumed that all components have been installed and all environments have been setup. This guide walks the user through the planning and execution of an OII implementation.

For better understanding, the Implementation chapters assume knowledge of:

- Database management systems (DBMS), specifically Oracle 11g,
- Programming Language Structured Query Language (PL/SQL) and
- Oracle Data Integrator (ODI) 11g

LIST OF ACRONYMS AND ABBREVIATIONS

To make using this guide easier, all the full name is used the first time an acronym appears. In addition, all acronyms used in the *OII Implementation Guide* are listed below:

- CSI – Customer Support Identifier
- DBMS – Database Management System
- ETL – Extraction, Translation and Loading
- LOB – Line of Business
- OBIEE – Oracle Business Intelligence Enterprise Edition
- ODI – Oracle Data Integrator
- OII_DM – OII Data Mart schema
- OII_ST – OII Staging schema
- OII_WH – OII Warehouse schema
- OII – Oracle Insurance Insight
- OII Core – OII Core Data Model
- OLTP – Online Transaction Processing
- PL/SQL – Programming Language/Structured Query Language
- SCD – Slowly-Changing Dimension
- UAT – User-Acceptance Testing
- UDF – User-Defined Field

NAMING CONVENTIONS

To make using this guide easier, database objects are capitalized and italicized throughout the *OII Implementation Guide*:

- Example: *OII_ST.PLCY_TRANS* (*Policy Transaction*)
 - *OII_ST* – The relational (i.e. database) schema name
 - *PLCY_TRANS* – The relational (i.e. database) table name
 - *Policy Transaction* – The logical table name

RELATED DOCUMENTS

For more information, refer to the following documents:

- *Oracle Insurance Insight Release Notes* - This document describes the latest enhancements and updates to OII as well as issues that have been resolved in this version.
- *Oracle Insurance Insight Installation Guide* - This manual describes the steps for configuring and installing OII.
- *Oracle Insurance Insight Administration Guide* - This manual describes how to create and configure the required OII users and groups within OBIEE.
- *Oracle Insurance Insight Warehouse Palette User Guide* - This manual describes how to use the Warehouse Palette, an application that provides users with an easy-to-use interface to create and configure a Line of Business (LOB) and “publish” the LOB for incorporation into the OII system.
- *Oracle Insurance Insight User Guide* - OII uses OBIEE 11g as its front end interface, providing a set of dashboards, reports, and query building tools to use to analyze the OII data. This manual describes how to configure and run the OII reports as well as use the OBIEE analytic features to build custom queries to run against the OII data.
- *Oracle Insurance Insight Data Dictionary* - The data dictionary for the OII system.

CUSTOMER SUPPORT

If you need assistance with OII, you can log a Service Request at My Oracle Support (<https://support.oracle.com>). You will need your Customer Service Identifier (CSI) to register.

For community support, you can login to Oracle Mix (<http://mix.oracle.com>) and join the Insurance Business Intelligence and Data Warehousing group (<https://mix.oracle.com/groups/18191>) to review posts and/or create your own post to request assistance.

Address any additional inquiries to:

Oracle Corporation

World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:

Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Section I - Strategic Goals

Chapter 1

Introduction

The Introduction answers high-level “What” questions. Subsequent sections go into more detail and answer “How” questions. The Introduction serves two purposes:

- For an executive audience, this chapter answers the key questions about what OII can do and what problems it solves. This audience may not need to read beyond the Introduction.
- For a technical audience, this section gives perspective on what is meant to be accomplished in an OII implementation. It gives relevance to the technical details discussed throughout the rest of the guide.

WHAT LINES OF BUSINESS ARE SUPPORTED?

Six pre-defined Lines of Business (LOBs) are included with OII out of the box:

- Personal
 - Dwelling Fire
 - Homeowners
 - Personal Auto
 - Personal Umbrella
- Commercial
 - Commercial Auto
 - Commercial Property

Additional LOBs can be created from scratch or by using one of the six pre-defined LOBs as a starting template. Through the flexibility of the OII Adaptive Data Mode, any LOB can be edited in the Warehouse Palette. See the *OII Warehouse Palette User's Guide* for further information.

WHAT IS INCLUDED IN THE CORE DATA MODEL?

Many details about claim and policy transactions are common across different carriers and LOBs. The OII Core Data Model (OII Core) represents that part of the data architecture that is shared by all LOBs. All metrics and universal claim and policy transaction details for all LOBs reside in the OII Core and are exposed in the OII Corporate data marts:

- Monthly Snapshot Marts
 - Claim
 - Combined Policy & Claim
- Transactional Marts
 - Claim
 - Policy

Metrics are also presented with rich, LOB-specific dimensions in an additional data mart per each implemented LOB:

- LOB Combined Policy & Claim Monthly Snapshot

The four Corporate and one LOB-specific data marts listed above represent the default data marts provided by OII out of the box. OII provides the underlying data elements (i.e. fact tables and dimensions) for up to six data marts for Corporate data and for each implemented LOB. With Corporate data and the six pre-configured LOBs supplied with OII, this would allow for a total of up to forty-two data marts:

- Monthly Snapshot Marts
 - Claim
 - Combined Policy & Claim
 - Policy
- Transactional Marts
 - Claim
 - Combined Policy & Claim
 - Policy

WHAT ARE DIMENSIONS?

The concepts and calculations surrounding premiums, exposures, and losses are contained within the OII Core. Loss Reserve calculations, for instance, remain the same regardless of LOB. The dimensions, or risk items, are what differentiate one LOB from another. A dimension can be a covered item such as a vehicle. A dimension can also be something that provides additional detail about the policy such as a garage on a Personal Auto policy. Each of the supported LOBs is a collection of dimensions that describe that LOB.

For example, the pre-defined dimensions for Personal Auto include Vehicle and Driver. Looking at a metric across more than one dimension such as Paid Loss by Territory and Month is referred to as multi-dimensional analysis. A practically infinite number dimensional views are available in OII.

Creating and viewing reports for OII in OBIEE is covered in-depth in the *OII User Guide*.

WHAT IS THE ADAPTIVE DATA MODEL?

In the OII Adaptive Data Model, LOB dimensions are controlled directly by business users through the Warehouse Palette. OII automates the technical creation and configuration of the warehouse components (e.g. database tables, interfaces, internal ETL). Data model design is abstracted from the underlying physical database. The IT department has complete flexibility to scale the application appropriately, removing traditional modeling constraints.

The Warehouse Palette is a web-based application that allows users to create an LOB and configure the dimensions and associated attributes that make up the LOB. Its purpose is to provide users with an easy-to-use interface that facilitates building and modifying an LOB's components without directly managing the related technical details. Once the user "publishes" the LOB, the physical database structures and loading mechanisms (i.e. the interfaces and the internal ETL) are automatically added to OII.

IMPORTANT Once an LOB is published using the Warehouse Palette it is locked and cannot be modified. A published LOB must be "unpublished" in order to make additional edits. The process of unpublishing deletes all data for that LOB and removes any LOB-specific structures from the database.

Instructions on how to extend the OII data model are contained in the *OII Warehouse Palette User Guide*.

WHAT ARE SLOWLY-CHANGING DIMENSIONS?

Dimensions come from two sources, the OII Core and Adaptive dimensions:

- Core
 - Dimensions and their attributes in the OII Core that relate to all LOBs (e.g. Class, Limit, Producer)
- Adaptive
 - LOB-specific risk items contained in the metadata supplied for a pre-defined LOB (e.g. Driver, Dwelling, Vehicle)
 - LOB-specific risk items (e.g. Watercraft) or other descriptive attributes (e.g. Boat Length or Boat Type) built in the Warehouse Palette

With such variability among dimensions, OII offers two widely-used Slowly Changing Dimensions (SCD) methodologies to manage changing dimensional data over time:

- Type 1 SCD
- Type 2 SCD

The Type 1 SCD methodology is a simple approach to dealing with dimensional attribute changes. Only the current state of the dimension is preserved. The fact table is unchanged. When a dimensional attribute changes, the previous value is overwritten with the current value.

Dimension bloat is minimized, and performance is optimized for a Type 1 SCD. The obvious disadvantage to this method is that no record of historical attribute values is kept.

A Type 1 SCD is a good choice if there is little or no analytical benefit in maintaining old attribute values. Type 1 may also be a good choice if the dimensional attributes are expected to change frequently and there is little value in maintaining history.

The Type 2 SCD methodology is the primary technique for tracking historical data. Given a Natural Key supplied in the OII Staging schema, Type 2 SCDs generate a new dimensional record whenever a new attribute value is loaded. These new records, identified by a surrogate key, are linked to the fact table. With each new dimensional record added, the fact table is automatically partitioned in time and permanently bound to the dimensional attributes that define that time slice.

Type 2 SCDs preserve unlimited history, allowing for powerful data analysis. As a consequence, storage of unlimited history and query performance across an expansive table may affect performance.

A Type 2 SCD is a good choice if historical analysis offers analytical value and the dimension truly is slowly-changing. Type 2 may not be

a good choice if the attribute in question is subject to frequent change.

Each individual dimension can be set as a Type 1 or Type 2 SCD in the Warehouse Palette. This setting affects how changing dimensional data is handled in the Data Mart schema. In the Warehouse schema, full transactional history is always preserved for detailed analysis.

IMPORTANT! A Natural Key must be supplied for ALL rows in a dimension table if Type 1 or Type 2 is selected in the Warehouse Palette. History cannot be maintained in OII without the presence of a Natural Key in every row of the dimension table.

IMPORTANT For the source data in the Staging schema, for a given Natural Key, each row should have a different Transaction Effective Date (i.e. the Transaction Effective Date should be at least different in seconds).

Otherwise, If duplicate Transaction Effective Dates are present, the row with the highest ROW_NUM sharing the same Natural Key and same Transaction Effective Date will be considered as the latest and only business attributes from this row to be recorded into the Warehouse. This would result in losing the attributes changing history.

WHAT ARE METRICS?

Metrics are quantifiable measures that represent a piece of business data as seen in relation to one or more dimensions. An example would be Earned Premium by Month. In this case, Earned Premium is the metric and the dimension would be time (i.e. Month). All metrics are contained in the OII Core. There are too many metrics in OII to name them all here. Examples of other metrics include:

- Claim Count
- Future Written Premium
- In Force Premium
- Incurred Loss
- Loss Reserve

See the *OII User Guide* for a complete list of metrics.

Earned Premium deserves a special note because OII can calculate it in a number of ways. If raw premiums are included for policy transactions, OII can calculate Earned Premium using one of these methods:

- Monthly (1/24th)
- Daily
- Seasonal (utilizes a pre-defined table with seasonal factors)
- Original Premium

Alternatively OII can also accept pre-calculated Earned Premium amounts and forgo internal calculations. Earned Exposure is also treated in this manner.

WHAT IS ODI (DATA LOADING)?

ODI is an ETL tool that is used to load the OII data model schemas from Staging to Warehouse and Warehouse to Data Marts. Custom dimensions built in the Warehouse Palette are published to the Oracle database and supplement the OII Core. In the publish process, ODI is updated to handle new dimensions just as it would the OII Core components. Using ODI to load OII is covered in-depth in *Chapter 5: Data Loading*.

WHAT IS OBIEE (DATA VISUALIZATION)?

OBIEE is a data analysis tool that sits on top of the OII data model, providing dashboards, reports and custom analytic queries that help bring out novel characteristics inherent in the data. OBIEE use and configuration is covered in-depth in the *OII User Guide*. In OBIEE, dimensions are known as “filters” because of their ability to focus analysis on limited, relevant data.

See the *OII User Guide* for a complete listing of pre-defined filters, including those supplied with the six pre-defined LOBs. Linking OBIEE to the OII data model is covered in-depth in *Chapter 6: Data Visualization*.

Section II - Project Planning

Chapter 2

Implementation Lifecycle

STANDARD OII PROJECT STAGES

The standard OII implementation process is separated into four project stages as follows:

1. Initiation
2. Planning
3. Execution
4. Deployment

Initiation

In the Initiation stage, the customer's current source system(s) and requirements are assessed. Based on the estimated transactional data volume and allowing for growth hardware sizing can be assessed to determine configurations needs.

Planning

In the Planning stage, the following occurs:

- A detailed data analysis is performed to determine which fields will be exposed within OII, where they come from in the client's source system(s) and establish valid values.
- Data Mapping is performed in order to map the source system(s) data to OII Staging tables. This involves mapping of policy and claims data.
- Source codes are defined and standardized by the client in order to accommodate the OII data model and to ensure data consistency over time.
- Entities, operation, distribution and other hierarchies are defined prior to Source-to-Staging ETL development in order to enforce the correct parent-child relationships.
- The client's Earned Premium, Earned Exposure (if implemented) and Incurred Loss calculation methods are determined in order to configure OII for the selected pre-defined calculation methods.
- Signoff Metrics/Tolerances are established in order to commit to criteria for successful testing.
- Hardware is procured and configured. Required software is installed.

Execution

In the Execution stage the following occurs:

- The Source-to-Staging ETL is also designed and developed.
- Testing/balancing is done from the source system(s) data to the Staging tables.
- The Source-to-Staging ETL is re-tooled, as necessary. No calculations are performed on the Staging tables, so balancing from the source to the Staging tables is an iterative process of loading, testing and re-tooling the Source-to-Staging ETL.
- Warehouse and Data Mart loads are run followed by balancing.
- Routine data loads are configured in ODI. The loads must be configured to ensure there are no data gaps or duplications and that the process can integrate with the source system, month-end processing cycles, etc.
- User Acceptance Testing (UAT) is performed. After the OII's numbers match the agreed tolerances, OII is presented to the end users.
- Based on end-user feedback and request from the end users the OII/OBIEE data views can be customized.

Deployment

In the Deployment stage migration to production, administrative training, end-user training and transition to support takes place.

STANDARD OII IMPLEMENTATION PROCESS

The timelines below are estimates for a standard OII implementation of a single LOB (e.g. Commercial Auto or Homeowners) that includes policies and claims from a single source system. The time ranges could increase or decrease based upon the number of LOBs, the number of source systems and the complexity of data extraction from the source system. Coordinating multiple source systems and multiple LOBs in an OII environment often requires coordinated data mapping and ETL design so that a consistent view can be presented across all LOBs.

DATA MAPPING

- **Scope:** During this phase, the required and desired fields to be exposed within OII and the corresponding fields in the source system(s) are identified. Customizations needed for the Adaptive Data Model are identified. Sign-off metrics and tolerances are established during this period. Allocation of server resources and storage space are arranged prior to the ETL Development phase. Accommodations are also made to store the staging data.
- **Client Resources:**
 - One or more BAs who are familiar with the data from an insurance and end-user perspective
 - One technical resource who is familiar with accessing data and data architecture at a high level in the source system(s)

- **Assumptions:**
 - N/A
- **Risks:**
 - Multiple source systems increase the complexity and required time for this phase.

SOURCE-TO-STAGING ETL DEVELOPMENT

- **Scope:** During this phase, the OII Warehouse Palette is used to customize the Adaptive Data Model. From here, the Source-to-Staging ETL is designed, developed and re-tooled in order to perform any necessary data cleansing before moving data into the Staging tables (a.k.a. templates). Once transactions are loaded to the Staging schema, descriptions and translations should be configured in OII. These descriptions and translations are applied to customer source codes when OII loads from Staging to the Warehouse. To ensure that all expected transactions were extracted from the source, loaded and processed by OII, balancing should be performed.
- **Client Resources:**
 - One Business Analyst who can validate the relevance and appropriateness of the data being extracted from the source system(s) and sign off on the data reconciliation metrics between source and staging
 - One technical resource who is familiar with accessing data and data architecture at a high level in the source system(s)
 - One technical resource who can code and test the Source-to-Staging ETL using SQL and/or an ETL tool and execute the data load to Staging
- **Assumptions:**
 - The Business Analyst resource should be readily available to answer questions concerning the appropriateness of the data being extracted from the source system(s)
 - The two technical resources may be combined into one job role if a single resource has the breadth of knowledge of the data architecture and the skill set to code the Source-to-Staging ETL
- **Risks:**
 - Poor data quality will require that more time be spent cleaning the data. Ideally, data cleansing will be done in the source system(s) so that the source is consistent with what is loaded into OII and so that data manipulation is reduced in the Source-to-Staging ETL.
 - Multiple source systems increase the time needed to develop separate loading mechanisms and convert different code values from various systems into one consistent representation of the data
 - If the required transactional fields were not properly identified and addressed in the Source-to-Staging ETL, then it will take additional time to re-tool the Source-to-Staging ETL or investigate ways to accommodate the missing data in OII

DATA MART LOAD

- **Scope:** The transactional and monthly snapshot Data Marts are loaded, and the OBIEE Repository is customized to accommodate changes per the Adaptive Data Model.
- **Client Resources:**
 - One BA who can communicate with the end users and sign off on the data reconciliation metrics between the Warehouse and Data Mart schemas.
 - One technical resource who is familiar with accessing data and data architecture at a high level in the source system(s)
 - One technical resource who can re-tool the Source-to-Staging ETL using SQL and/or an ETL tool and execute the data load beyond the warehouse.
- **Assumptions:**
 - OII reconciles to data in the templates and not necessarily to historical customer reports.
- **Risks:**
 - Validation of Data Mart metrics (i.e. Earned Premium, Incurred Loss etc.) often involves a process of discovery where missing data is identified, the Source-to-Staging ETL code is updated and data is reloaded in OII (Staging, Warehouse and Data Marts). Focus on data reconciliation at the Warehouse level will identify issues earlier when it takes less time to reload. Waiting until the Data Mart Load to perform detailed data analysis may lead to additional iterations of this phase.
 - There can be confusion around calculated Data Mart metrics because customer reports may represent a different view of the same data. Both views may be correct and true representations. The measure of success for OII is being able to reconcile the Data Mart metrics to the Warehouse and Staging tables not necessarily to a customer report. Any differences that cannot be reconciled may ultimately be due to incomplete Staging data (via the Source-to-Staging ETL).

CUSTOMIZE OII DATA VIEWS

- **Scope:** In this phase, user requests are translated into custom OBIEE reports and data views.
- **Client Resources:**
 - End-user feedback (dashboard/report requests) and availability for training

Section III - Implementation

Chapter 3

Program Flow

The focus of this chapter is an overview of the flow of data between the processing stages in OII. Each of the six pre-defined LOBs provided with OII represent a reference implementation that is adaptable and customizable. An understanding of program flow is required to properly plan and implement an OII solution..

Note Keep in mind that the content presented herein is representative of the *reference implementation* of OII. The reference implementation is a model upon which an actual implementation is based. As such, actual implementation details may differ from the model.

The OII program flow utilizes three different schemas for processing: Staging, Warehouse and Data Mart. Generally speaking, data is slightly denormalized in the Staging schema. Depending on the source system data structures, the loading process may or may not include additional denormalization. During the flow of data from Staging into Warehouse, some renormalization occurs. Finally, during the flow of data from Warehouse into Data Mart, data undergoes denormalization. The purpose of the de- and renormalization is to reduce redundancies, flatten hierarchies, and prepare for the final load into fact and dimension tables in the Data Mart.

STAGING SCHEMA CONCEPTS

The Staging schema introduces several components that are important to understand:

- *Party Model*
- *Contact*
- *Postal Address*
- *Natural Keys*
- *Row Number*
- *Relationship Model*
- *Normalization and Denormalization*
- *User-Defined Fields*
- *Concepts in Practice*

These concepts are used to allow for flexibility and reuse in the system while maintaining a high level of performance in the Data Mart schema.

PARTY MODEL

The Party Model is used to establish roles and relationships for entities referenced in transactional data. The *OII_ST.Party (Party)* table is at the top of the Party Model hierarchy and contains fields shared by all Party Types. In this context, Party Type can refer to a person or an organization, while a Party Role describes the entity with respect to the transaction. This concept is illustrated in the following example:

John Smith is a personal auto policy holder for Alamere Insurance Company. For data relating to his personal auto policy, John is a PERSON party type, and holds the party role of INSD (INSURED) for the Personal Auto LOB. Similarly, Alamere Insurance Company is an ORG party type, and holds the role of INSUR_CO (INSURANCE COMPANY) in this LOB with respect to this transaction.

The Party Model allows for an entity to hold more than one role or relationship within the system and allows for the relationship to be established across LOB boundaries. This results in reduced data volume and enhanced reporting capabilities. This concept is illustrated by expanding on the previous example:

John Smith calls his agent and, during usual customer service interaction, learns that Alamere Insurance Company also offers Homeowners insurance policies. John requests a quote. In OII, John is already established as a PERSON party type. In the Homeowners LOB, he now holds the role of PROSPECT. His PERSON data is common to both LOBs.

Finally, the Party Model introduces context and hierarchies for entities and roles within the system. The Party-Level Relationship Code of the party establishes an entity's relationship to the transaction or to other parties. See the concept illustrated below, built upon the previous examples:

John Smith's insurance agency is established as an ORG party type. Similarly, the agent is a PERSON entity. With respect to transactions involving John's policy, the agency holds the PRODR (PRODUCER) party role while the agent holds the SUB_PRODR (SUBPRODUCER) party role. The hierarchy of PRODUCER and SUBPRODUCER is expressed via the Role-Level Relationship Code in the Party Model.

The actual relationship code values used in OII at the Party, Type and Role level match the database table name for that entity. When a new party dimension is created in Warehouse Palette, the table name becomes the new relationship code representing that entity. The pre-defined OII relationship codes are listed in Appendix A: Chapter A: Relationship Codes.

CONTACT

The *OII_ST.CNTCT (Contact)* table provides flexibility for defining contact endpoints for physical entities such as people or office locations. Currently, the Contact table extends only a physical address as a contact endpoint, or entity type. This information is housed in the *OII_ST.PSTL_ADDR (Postal Address)* table. In the future, additional entity types can be added, such as phone numbers and email addresses. A row in the Contact table is associated with a row in the Party table. A row in the Party table can be associated with one or more rows in the Contact table. A row in the Contact table is also associated with a row in the Postal Address table where the location information is stored.

John Smith is a Personal Auto policy holder with Alamere Insurance Company. He pays a monthly premium for his policy. When his monthly payment transactions are loaded to the OII system, John's home address is listed as a row in the Postal Address table. This row is then referenced by a row in the Contact table. The row in the Contact table is then referenced in the Party table, which contains a reference to John's entry in the Person table.

Note Since Contact is related to Party, it is important to note that if there are no rows in the Party table, there will be no rows in the Contact table.

POSTAL ADDRESS

The *Postal Address* table is used to store address locations. During the migration process to the Warehouse schema, the geographic boundaries present in the address are moved into a separate *OII_WH.GEOG_BNDRY (Geographic Boundary)* table. A reference is created between the *Postal Address* and *Geographic Boundary* tables. There are multiple types of geographic boundaries:

- State or Province
- Region
- Country

Regionalization of data by geographic boundary provides the OII user with a sufficiently granular level of detail while not burdening the dimensions in the Data Mart schema with street information (i.e. Address 1 and Address 2 fields) that is typically more useful in an operational environment (e.g. the generation of mailing labels). Once data for Address 1 and Address 2 are migrated to the Warehouse, it undergoes a regionalization transformation. All data is preserved at the Warehouse level, while only the geographic boundary information is moved to the dimensions in the Data Mart schema.

Note The basic installation for OII is a *model*, so if requirements dictate a need to maintain street address information in the Data Mart schema, the implementation team can readily accommodate this request easily.

The *Postal Address* table is used by the *Contact* table when storing address information related to the *Party* table. In addition, the *Postal Address* table is also used to store address information related to vehicles and claims.

NATURAL KEYS

Natural Keys are unique data identifiers of specific elements within the Staging schema. Natural Keys are typically associated with several types of data elements including Entities, Parties, Claims and Policies. A Natural Key serves multiple purposes, one of which is the ability to associate rows in one table with one or more rows in one or more other tables. Natural Keys are one method to keep data loads reduced which increase the performance of a data warehouse system by eliminating data duplication. Additionally, Natural Keys are a good method to use for ensuring the correlation between one or more tables for reporting purposes.

For example, a Natural Key is used in conjunction with the Party Model to allow association of a single entity with more than one role. Without the Party Model and Natural Keys, data duplication would occur – over time, this duplication results in reduced system performance and increased overhead. The Natural Key ensures that the correlation between an entity and multiple roles is valid. It is important in reporting to identify when roles are held by the same person or organization.

During the loading process, if a Natural Key is not supplied by the source system, a Natural Key is generated as a checksum of the *identifying columns* – that is, the collection of columns that uniquely identify a row – and is assigned to each transaction that is loaded from the customer data. If one or more rows of incoming data have the same values in the identifying columns, the generated checksum, the Natural Key, will be the same. Thus, it is important to ensure that the identifying columns are indeed serving the purpose to identify unique rows. The alternative is for the customer to supply a Natural Key.

The use of Natural Keys in OII has several caveats that are important to note for implementation teams:

- Every column with the exception of Natural Keys in the Staging model tables and Natural Keys used for counts in the monthly snapshot tables within the marts must be populated with data – that is, not NULL. Special notes on NULL values – with respect to fields with data type of string, a single character, default a single space is used to represent a lack of data for a column that does not allow NULLs. This single character can be configured in the *OII_SYS_CONFIG* table.
- Customers can supply their own Natural Keys from source systems provided the unique string requirement is met.
- Supplying Natural Keys must be performed in an all-or-nothing approach per entity; either all rows within the entity will have a supplied Natural Key or none of them will.
- If Natural Keys are empty, OII will compute Natural Keys in the Staging-to-Warehouse data population routines.
- If Natural Keys are computed, there must be enough sufficient non-empty data to establish a unique distinction between data rows. Otherwise, logical errors will occur.

If Natural Keys are computed, Slowly Changing Dimension (SCD) support is not available – it is impossible to implement Type 1 or Type 2 SCD. An SCD is a dimension that, as the name implies, may change over long periods of time. If a Natural Key is not supplied, then it is computed based on a collection of fields that uniquely identify a row in a table. Since it is then possible for one or more fields in the collection to change, the computed Natural Key would then be different when data changes. As such, there is no method to allow correlation of the changed data to its original state.

These concepts are illustrated in the following example:

Alamere Insurance Company designs a data structure that defines “person” entities. The PERSON table that houses data rows for these entities consists of only two columns, FirstName and LastName. Alamere decides to use First Name and Last Name rather than tax identification numbers as keys to identifying entities in the person table for legal reasons. Alamere is not currently computing natural keys. During data load to OII, the natural key is computed for each entity (row) in the table. If the “person” table contains John Smith and Jane Smith, the computed natural key will suffice to uniquely identify the two individuals. However, if a separate John Smith is entered in the “person” table, a conflict will result in computing natural keys – there are not enough columns to uniquely identify each row.

Alamere has a “location” table which gives a physical location to a row in the “person” table. If Jane Smith changes her last name to Jones, which results in an update to the “person” table, the ability to correlate the location data will be gone – the computed natural key for Jane Jones will be different from the computed natural key for Jane Smith.

The solution then is to either add more columns to the “person” table which guarantee uniqueness, or to have the source system generate a natural key that is added to each row prior to import into OII.

IMPORTANT For the source data in the Staging schema, for a given Natural Key, each row should have a different Transaction Effective Date (i.e. the Transaction Effective Date should be at least different in seconds).

Otherwise, If duplicate Transaction Effective Dates are present, the row with the highest ROW_NUM sharing the same Natural Key and same Transaction Effective Date will be considered as the latest and only business attributes from this row to be recorded into the Warehouse. This would result in losing the attributes changing history.

ROW NUMBER

In a typical relational database schema, a primary key is used to uniquely identify a row of data in a table. Such keys are then used to provide references between different tables. For example, a primary key in an Insured table would be referenced as a foreign key in a Policy table, indicating a relationship between data in the Policy and Insured tables where the foreign key matches the primary key.

Because the database schema is particular to the company and in many cases the business units therein, there is the potential to have as many different table definitions as there are business units and companies.

To simplify creation and maintenance of ODI Interfaces which populate the Staging model from customer data, the interfaces have been programmed to modify imported data by removing primary and foreign keys which results in destruction of referential integrity.

However, there still exists a need to be able to generate a holistic view of a transaction across the various tables in the model, and as such the data in Staging has introduced a unique identifier column: row number (*ROW_NUM*). The row number is always present for every staging load, regardless of how natural keys are provided or generated. The row number is simply a unique number, very similar to an identity column in SQL Server, that increments per row (or rows, when relationship codes are needed for uniqueness) within a given staging load to uniquely identify a transaction.

The ROW_NUM is used in each table in the Staging environment for all rows applicable to that unique business transaction. As such, a *virtual transaction* can be built from the tables in the Staging model using only the ROW_NUM.

RELATIONSHIP MODEL

In OII the relationship model is used to provide context and hierarchy in which multiple rows in a given table can be related. Relationship codes are maintained in the *OII_SYS* schema. The relationship code is expressed in tables within the Staging schema using a column named like *object_RLAT_CD*, where *object* is the type of entity contained in the table. This column is used to provide a part of the primary key of the table in addition to context and hierarchy. An example is shown below:

The OII_ST.PRODR table contains Producer entities, and features the PRODR_RLAT_CD column. In practical applications, a Producer may have Sub-producer that is related. The Sub-producer would also have a row entry in the PRODR table, and the relationship between the two rows for the Producer and Sub-producer is represented in the values contained in the PRODR_RLAT_CD.

The relationship model works in conjunction with the entities in the Party model to establish context between parties and transactions, and establish relationships between parties with respect to transactions.

NORMALIZATION AND DENORMALIZATION

Relational Databases used in *online transaction processing (OLTP)* are generally designed in a systematic way that ensures the data structures are useful for querying and possess performance characteristics that support OLTP. OLTP databases are characterized by a high volume of small transactions, such as payment processing for policies.

Normalization describes the design pattern in which the data structures meet the needs used by OLTP databases. For the purposes of OII, the main normalization characteristics that are import to understand are table relationships built by keys, and the use of joined tables. Normalization attempts to optimize database performance for inserting, updating or deleting records by storing different but related data in separate tables.

By contrast, *denormalization* attempts to organize database structures in such a way as to provide optimized performance for reading data. Databases used for *online analytical processing (OLAP)* are usually designed with denormalized database structures. Since OLAP applications are used to analyze trends in historical data over long periods of time, the sheer volume of accumulated data requires either a high-performance database or a database structure that is built for fast reading. An OII implementation provides both of these.

OII is an OLAP system, so the end-result of loading and program flow will result in denormalized data in the data mart.

USER-DEFINED FIELDS

Since the data model provided by OII is a reference implementation, it is expected that the schemas will be customized for each implementation. As such, most tables in the Staging and Warehouse schemas feature five fields that can be used to house data not contained in other areas. These User-Defined Fields (UDFs) allow for the data model to accommodate minor additions without having to unpublish, edit and republish an LOB. The naming of these fields will always contain “UDF” in the field name (e.g. PARTY_UDF_CD_1). The UDFs are normalized in Staging and Warehouse.

During the migration to Data Mart, the UDFs are denormalized in the dimension tables. As such, the dimension tables in the Data Mart may contain up to 30 UDF columns to accommodate the 1-5 UDF columns in each entity table. None of the UDF’s are included in the presentation layer. This means that the end user will not see anything in the OBIEE user interface for those UDF’s that have been utilized unless they are given a meaningful name, data is loaded to them and they are added to the presentation layer in the OBIEE Repository to make them available to the end user. *Chapter 6: Data Visualization* gives detailed instructions on how to enable custom dimensions, fields and UDFs in OBIEE.

CONCEPTS IN PRACTICE

The Party Model works in concert with Natural Keys to support the concept of multiple roles per entity. In the Staging model, there are no Natural Keys established for roles (e.g. INSURED or INSURANCE CO) or entities (e.g. PERSON or ORG). Instead, Natural Keys exist for each PARTY row. The relationship between the ENTITY and ROLE is established by ROW_NUM, and the hierarchy/context of the relationship is established by the relationship code (PARTY_RLAT_CD). In this manner, the Natural Keys for PARTY need not be reloaded for each ROLE, since an ENTITY may have more than one ROLE. The end result of the application of these concepts in OII:

- Less data is duplicated
- An ENTITY can hold more than one ROLE, and can be utilized across LOBs and subject areas (Policy Transactions, Claims Transactions, Quote Transactions, etc.)
- Business transactions can be presented holistically despite denormalization

The importance of Natural Keys for identifying rows cannot be understated – in order to ensure that the initial data load and subsequent loads have consistency, the Natural Key process must be defined accurately. This means that the customer must supply a Natural Key for each unique entity or must identify the collection of uniquely-identifying rows. The Natural Key identifies an entity in the data mart that is unique in its lifecycle. This concept is illustrated in the following example:

John Smith holds a Personal Auto policy with Alamere Insurance Company. Alamere chose to generate Natural Keys during the load to the Staging environment. When John's policies are loaded to the Warehouse, each Policy entity and related items (e.g. Vehicle entities, Driver entities) are related by the Natural Keys associated with these items. This includes the John Smith PERSON entity.

Subsequent to the historical load, Alamere issues a new Homeowner's policy to John. During the next load to the data mart, the Homeowner's Policy and related entities will be loaded, with their computed Natural Keys. John's PERSON entity, however, will use its existing Natural Key, and as such, will not be duplicated during the data load. Instead, the new Policy and related entities will be linked to the existing PERSON entity, thereby reducing data redundancy.

The sample example applies if Alamere chose to use identifying columns to compute a Natural Key.

FLOW DIAGRAMS

The overall program flow uses Oracle Data Integrator (ODI) as the ETL tool for moving data between schemas. The OII installation includes several packages in ODI which are used to load Warehouse and Data Mart schemas. For the reference implementation, there is an additional package used to load the Staging schema, however in practical application, the implementation team or customer is responsible for designing and implementing the Staging load process. ODI is a natural choice for this process; however the customer may have existing ETL tools and procedures that can be used as well.

It is important to note that while ETL tools provide *translation* capabilities as part of their functionality, it is not recommended to perform translation on data loaded to Staging unless absolutely necessary. If translation becomes necessary, the customer and implementation team are advised to consider applying a similar translation to the source data so there are no discrepancies between the source data and the data contained in OII.

The tables and processes described in the following program flow diagrams are illustrative of the overall process. The actual program flow includes many more tables and denormalization procedures.

LOADING TO STAGING

Data from customer systems is generally maintained in one or more database management systems, and is composed of tables that are related by one or more keys. This arrangement of data is well-suited for maintaining transactional data for daily operations but is regarded as less than optimal for data warehousing. As such, the initial data load to the staging environment begins the process of denormalizing relational data. The following diagram illustrates a typical denormalization of a customer’s relational data during the load to the staging environment.

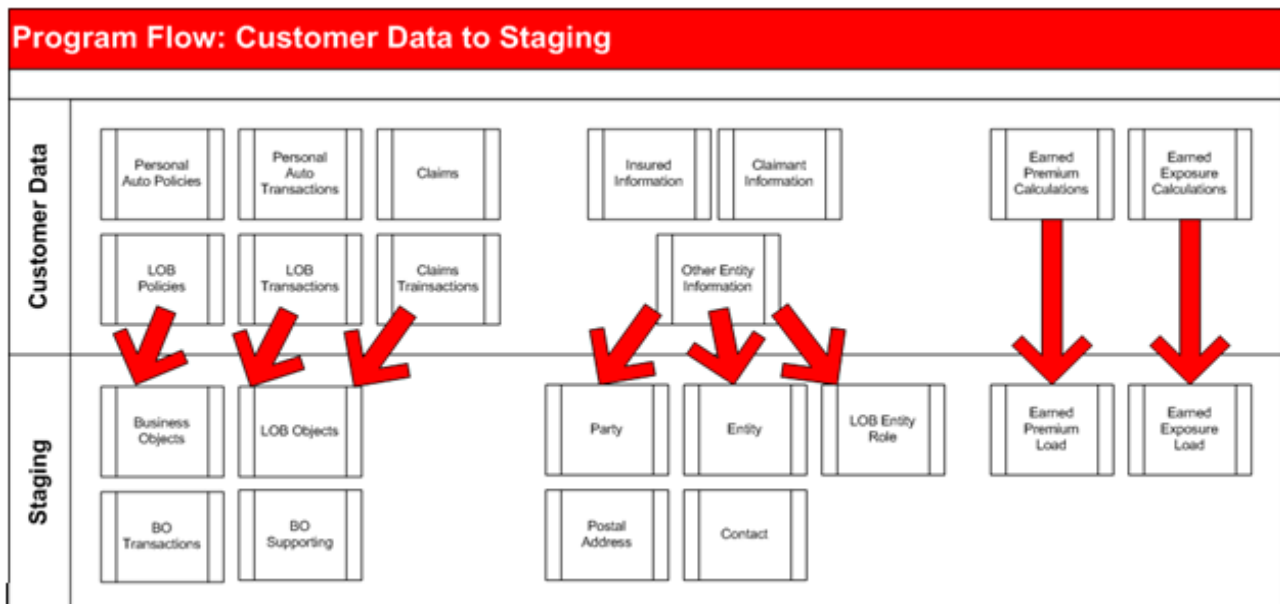


Figure 1: Loading to Staging Flow

The Customer Data row at the top of the diagram above illustrates multiple tables in a customer’s enterprise data store. The leftmost cluster in this swim lane represents typical tables used for LOB-specific transactions and business data:

- **Business Data Tables** – data related LOB-specific business objects.
 - **Personal Auto Policies** – exemplifies an LOB-specific table housing business data.
 - **LOB Policies** – a representative placeholder object that is used to denote other groupings of LOB-specific tables (e.g. Homeowners Policies, Commercial Auto Policies, Umbrella Policies, etc).
 - **Claims** – data related to claims.

Note Claims are a special case since a claim may not be related to a policy, so claims are separate from policies.

- **Transactional Data Tables** – transactional data related to LOB-specific business objects.
 - **Personal Auto Transactions** – transactions for the Personal Auto LOB.
 - **LOB Transactions** – a representative placeholder that denotes other groupings of LOB-specific transaction tables.
 - **Claims Transactions** – transactions for Claims.

The center cluster in the top swim lane illustrates multiple tables containing information about *referenced entities* in the customer's data. In this context, an *entity* is an object which is referenced in LOB-specific data. The entities may or may not be LOB-specific in this case. Typical entities are:

- External Customers (Insureds, Policyholders, Claimants)
- Internal customers (Producers, Underwriters)
- Businesses (Insurers, Reinsurers, Suppliers)

Finally, the rightmost cluster illustrates the Earned Premium and Earned Exposure information that may exist in a customer's enterprise data structure.

As mentioned previous, in typical customer data structures the information contained therein is linked with enforced key relationships to provide referential integrity between the tables. A transaction in the LOB-Specific Policy Transaction table is related to a transaction in the Insured Information table, as well as the Policy Information table. These tables are typically wide, meaning they contain many columns to house the entity-specific data needed, such as name, address, contact information or additional attributes of the policy related to the transaction. This describes the typical relational database design that most customer data systems employ.

During the *ETL* process which loads the Staging environment, the key relationships are removed as data is denormalized into a collection of tables. Policy data that is universal across LOBs is segmented into a Policy Data table, while LOB-specific data is moved into separate tables. Data that references parties such as Insureds and Insurance Companies, is denormalized according to the party type (person or organization) and the role that entity plays in the transaction (Insured or Insurance Company). Additionally, address and contact information is denormalized into postal address information.

Of special importance is the handling of Earned Premium and Earned Exposure data. The latest release of OII includes the ability to import these types of transactions directly to the Staging tables. There is no denormalization that occurs during the load to Staging; instead the reporting data for Earned Premium and Earned Exposure is loaded according to the amount and reporting time period. It is not necessary to populate these tables – if they are left empty, OII will perform the necessary calculations during the Staging to Warehouse flow step.

STAGING TO WAREHOUSE

During load into the Staging schema, data is denormalized from the original references. This process uses the aforementioned ROW_NUM construct to maintain a consistency across the tables. When applying the denormalization across LOBs, a reduction in data redundancy can be achieved, and hierarchical data structures can be flattened.

When data is loaded from staging into the warehouse re-normalization is applied to the data in preparation for the final load into the fact and dimension tables in the data mart. The following diagram illustrates a typical renormalization flow from staging to warehouse.

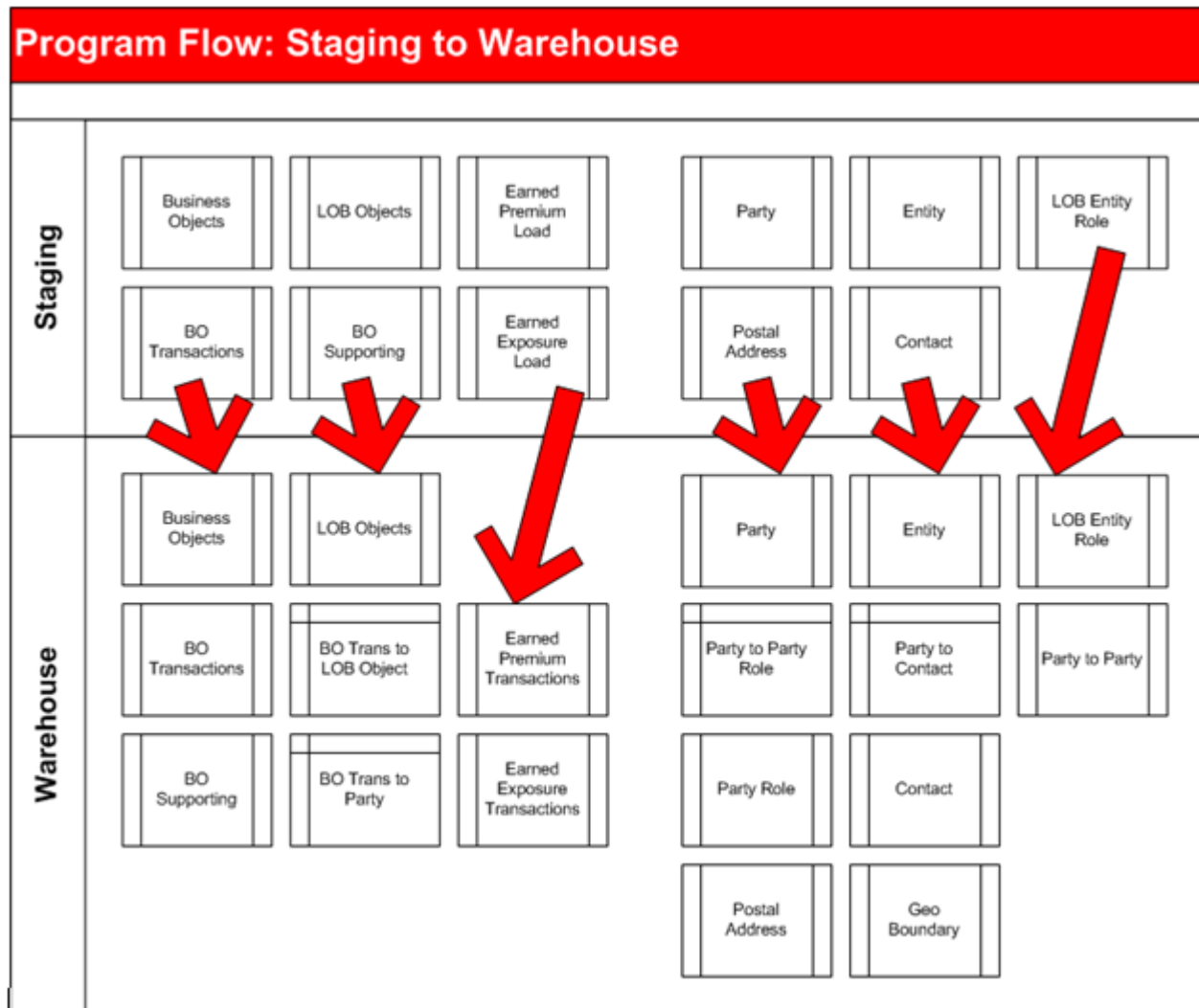


Figure 2: Staging to Warehouse Flow

The top row in the diagram above shows the representative tables present in the Staging schema, which is migrated to tables in the lower swim lane, the Warehouse schema. During the migration, the previously denormalized data is renormalized into OII data structures. This process includes several important steps:

- Creating *Bridge Tables*
- Introduction of ID columns

- Earned Exposure/Earned Premium Calculation
- Translation of relationship code
- Separation of Addresses

Bridge tables (also called many-to-many tables) provide a relationship between two tables, and in some cases provide additional data about the relationship. In the diagram, bridge tables are represented by the boxes with top and left borders, whereas data tables have left and right borders. By convention the bridge tables are generally named according to the two tables that are being joined in the relationship. In OII, bridge tables that have “_TO_” in the name. The following example illustrates a relationship-only bridge table:

The OII_WH.CLM_TRANS table contains a list of claims transactions. The OII_WH.PA_VEH table contains a list vehicles referenced by Personal Auto policies. The OII_WH.CLM_TRANS_TO_PA_VEH bridge table relates one or more vehicles in the OII_WH.PA_VEH table to a claim transaction in OII_WH.CLM_TRANS.

The following example illustrates a bridge table that includes a relationship and additional data about the relationship:

The OII_WH.PARTY table contains a list of entities. The OII_WH.PARTY_ROLE table contains a list of roles that an entity may possess. The OII_WH.PARTY_TO_PARTY_ROLE bridge table supplies the relationship between a OII_WH.PARTY and a OII_WH.PARTY_ROLE, while also including a time span that indicates the duration that the entity held the role.

In the load to the Staging schema, a ROW_NUM was used to provide a virtual transaction across the denormalized tables. When data is loaded to the Warehouse, ROW_NUM is replaced by an ID column which is then used as a reference for rows in other tables that refer to that specific entry.

The OII_WH.PARTY table contains a list of entities. As new rows are inserted into the OII_WH.PARTY table, OII checks that the new row is not a duplicate of an existing row. Each row in the OII_WH.PARTY table must be unique, a characteristic enforced by the PARTY_ID primary key column. By using this method, data redundancy is eliminated – each entity in the OII_WH.PARTY table is unique, and as new data is loaded, OII will reference existing entities rather than creating new rows.

When OII_WH.PARTY is referenced in the OII_WH.PARTY_TO_PARTY_ROLE bridge table, the reference is made using the PARTY_ID column as a foreign key in the bridge table.

If the Earned Premium and Earned Exposure tables were populated in the Staging schema, the information is simply carried over to the Warehouse and Data Mart schemas. Otherwise, OII will calculate the Earned Premium and Earned Exposure values for use during the Warehouse and Data Mart loads. OII supports multiple methods of calculating Earned Premium and Earned Exposure, such as Daily, Monthly, Seasonal and Original Premium. See *Metadata* on page 33 in *Chapter 4: Source-to-Staging ETL* for more information on the Earned Premium and Earned Exposure settings.

Note Transactions for Earned Premium and Earned Exposure have granularity to months only. This applies to calculated and loaded Earned Premium and Earned Exposure transactions.

When the migration to Warehouse occurs, the relationship codes in the **_RLAT_CD* columns are translated into *type codes*, represented as columns named like **_TYP_CD*. The same hierarchical relationships and context present in Staging are preserved during the migration to Warehouse.

In the Staging schema, addressing information is stored in the *OII_ST.PSTL_ADDR* table. During the migration process to the Warehouse schema, the geographic boundaries present in the address are moved into a separate table called *OII_WH.GEOG_BNDRY*. A reference is created between the *OII_WH.PSTL_ADDR* and *OII_WH.GEOG_BNDRY* tables using the ID scheme mentioned previously.

WAREHOUSE TO DATA MART

The Data Mart schema is designed as a star-schema architecture with two types of tables: *Facts* and *Dimensions*. A *fact* is generally recognized to be a transaction. A *dimension* is an attribute of a fact. Put simply, if a transaction occurs today at noon for \$5, the fact is the transaction amount (\$5) and the dimension is the date and time (today at noon).

The data model for the data mart has been designed using the *normalized star schema* pattern, as opposed to the *snowflake pattern*. In practice, this means that the base dimension tables are linked directly to base fact tables and are normalized to reduce redundant data. The star schema is used to realize performance benefits offered by the database and query platform.

Note The snowflake pattern includes dimension tables that have related dimension tables, rather than all dimensions relating directly to the fact tables.

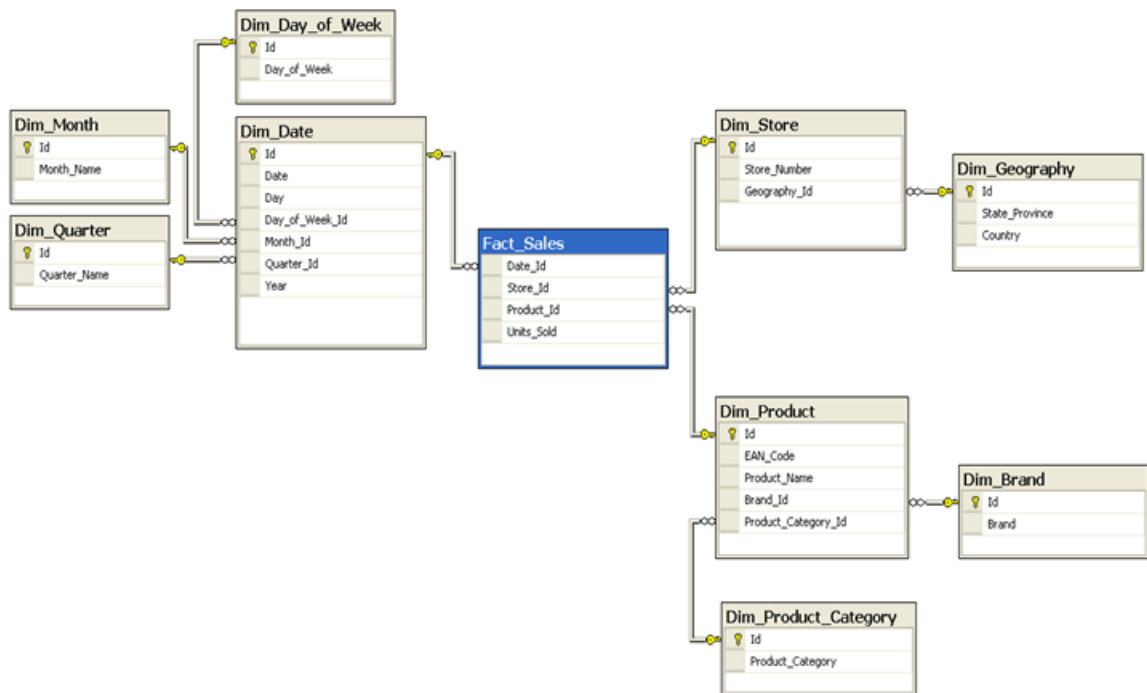


Figure 3: Snowflake Schema Example

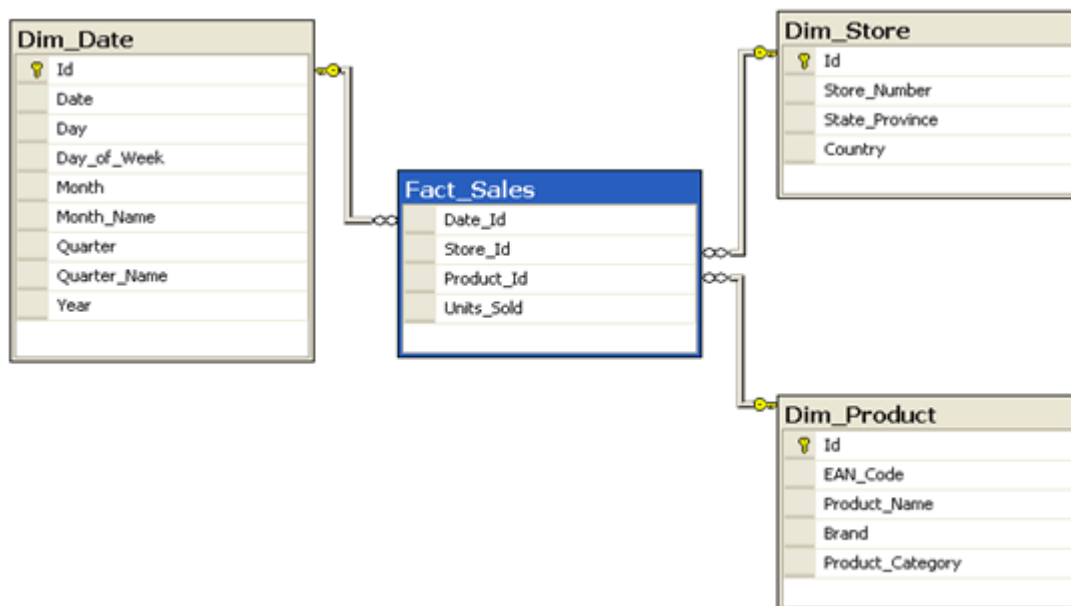


Figure 4: Star Schema Example

The Data Mart schema includes four core fact tables. Of these, two tables contain transactions for policies and claims: *OII_DM.PLCY_TRANS_FACT* and *OII_DM.CLM_TRANS_FACT*. The remaining two core tables contain the monthly snapshots of transaction data for policies and claims: *OII_DM.PLCY_MTH_FACT* and *OII_DM.CLM_MTH_FACT*. Within these four core tables are all the metrics for all LOBs. Since all metrics are part of the core tables and do not vary, the same interface can be used to load the transaction fact tables regardless of LOB.

Through the use of LOB-specific *extension* fact tables, the four core fact tables are kept abstract and generic, and LOB-specific facts are stored in the extension tables. As such, core fact tables contain the common facts OF the related core common dimensions. This means much more efficient loading of adapted content, ease in creation of LOBs, and allows for significantly better performance during loads. An additional benefit is reduced storage needs, since there are no empty fact keys for unused LOBs. Keep in mind, of course, that this design is the reference implementation only and can be changed.

Fact-to-fact join keys are used to give a 1-to-1 extension of base fact tables to deal with LOB-specific dimension tables. For example, the *OII_DM.PLCY_TRANS_FACT* table has an *LOB_FACT_ID* column. This column is used to relate the *OII_DM.PLCY_TRANS_FACT* table to the LOB-specific table, *OII_DM.PA_FACT* via the *PA_FACT_ID* column. The relationship between the core fact table and the LOB-specific extension fact table is expressed by $LOB_FACT_ID = PA_FACT_ID$.

The relationship between the LOB-specific extension fact table and LOB-specific Dimension table is expressed in the *PA_VEH_ID* and *PA_DRV_ID* columns. The *PA_FACT_ID* column relates back to the core fact table *OII_DM.PLCY_TRANS_FACT* which provides a consistent LOB-specific view of transactions that is accommodated by the adaptive data model.

In practice, OII uses *views* to present data in the dashboards. *Corporate* views are based on the core fact and dimension tables – as illustrated above, the core tables are not LOB-specific and provide a holistic view of the business without the details specific to any LOB. By contrast, the LOB-specific views provide a holistic view of an LOB such that the core and extended fact tables appear as one LOB-specific fact table, thus allowing a drill-down from high-level cross-LOB metrics into specific metrics for a designated LOB.

Data loaded from the Warehouse is denormalized when loaded into the Data Mart. The following diagram illustrates a typical denormalization flow from Warehouse to Data Mart:

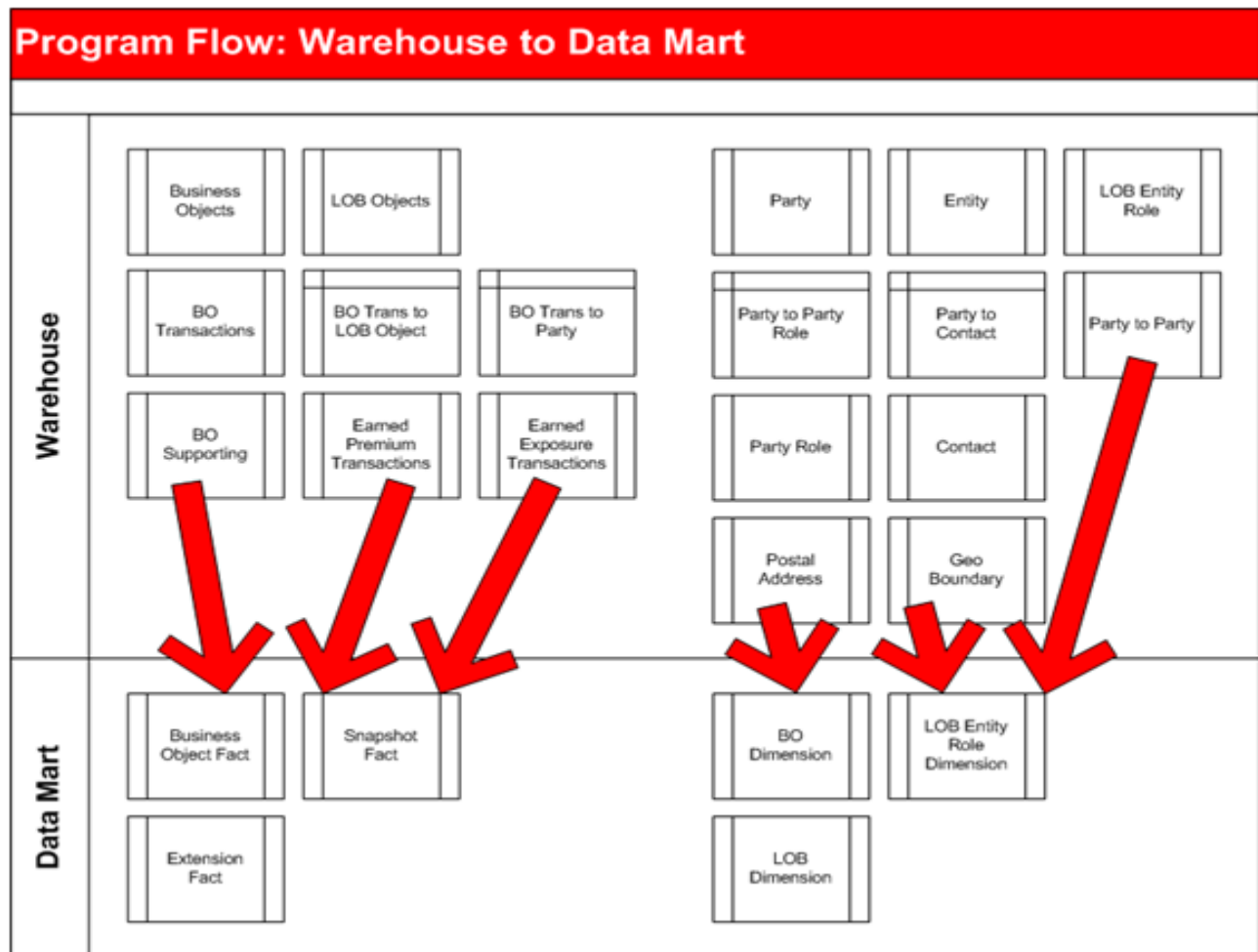


Figure 5: Warehouse to Data Mart Flow

As mentioned previously, transactional information from the warehouse schema is contained in multiple normalized tables, represented in the diagram above as:

- Business Objects (BO)
 - BO Transactions
 - BO Supporting
- LOB Objects
 - Related BO Transactions to LOB Object

- Related BO Transactions to Party
- Earned Premium Transactions
- Earned Exposure Transactions

The transactions in the Warehouse are denormalized into the data mart schema, using the methodology to extend core metrics described previously, into the following types of tables:

- Business Object Fact
- Snapshot Fact
- Extension Fact

Similarly, normalized entity tables in the warehouse schema are denormalized into the data mart schema as dimension tables. The denormalization process is based upon the *role* held by the Party – as shown in the Party to Party Role bridge table – and as such, the parties are created in the appropriate role-based dimension table, illustrated in the example below:

A person in the warehouse PARTY table that is related to the role of Producer will be expressed in the PRODR_DIM table in data mart.

Additional normalization occurs with PSTL_ADDR and GEOG_BNDRY tables. As mentioned previously the localized address information is discarded upon migration to the data mart. Regional information is maintained and written in to the role-based dimension table. Building on the previous example:

A person in the warehouse PARTY table has referenced CONTACT information and is also related to addressing information in PSTL_ADDR and GEOG_BNDRY. During normalization into the data mart, the resulting row in the PRODR_DIM contains the contact and regionalization information for the person.

Dates are also normalized into dimension tables using the following method. The tables which contain date information are DT_DIM, MTH_DIM and TIME_DIM.

When a date value is normalized from the warehouse, the various attributes of that date are created or referenced. All required attributes of the date, month and time are created. In the DT_DIM table, the primary key is the date itself, with the additional columns used to provide attributes of the date such as the context of the date with respect to calendar year, month and week. The same method applies to month and time. For example:

A transaction is migrated from warehouse to data mart. The transaction effective date is expressed as 1/1/2010 1:30:00 PM. The following information is contained in the DT_DIM table:

- *DT_ID = 1/1/2010*
 - *DT_FULL_DESC = Friday, January 1, 2010*
 - *DAY_NUM_IN_WK = 6*
 - *DAY_NM = Fri*
 - *DAY_LONG_NM = Friday*
- *DAY_NUM_IN_CAL_MTH = 1*
 - *DAY_NUM_IN_CAL_YR = 1*

- *Etc...*
- *MTH_ID = 1*
 - *CAL_MTH_NM = Jan*
 - *CAL_MTH_LONG_NM = January*
 - *CAL_MTH_NUM_IN_YR = 1*
 - *Etc...*
- *TIME_ID = 1:30:00 PM*
 - *HR_12_NM = One o'clock*
 - *HR_24_NM = 13 hrs*
 - *HR_12_SHORT_NM = 1*
 - *HR_12_NUM = 1*
 - *HR_24_NUM = 13*
 - *AM_PM_CD = PM*
 - *Etc...*

Currently, columns in Data Mart tables that end in “_DT_ID” relate back to the DT_DIM tables.

Source-to-Staging ETL

MASTER DATA MANAGEMENT STRATEGY

The task associated with mapping data from the client source system(s) to the OII Staging tables is one of the most critical steps in a successful implementation. Listed below are some best practices to use as guidance during this process.

- Data quality analysis or data profiling should be performed on the source data to ensure the data quality and completeness for business requirements.
- Logical data mapping describes the source elements, target elements (i.e. fields in the OII Staging tables) and any transformations that need to occur between the source and target. Transformation rules should be stated explicitly for each target field and address any necessary data conversion. When converting a source date to a target date column, validate that the source date is in the expected format.
- Reconcile the count of records at the end of the loading step against the records at the end of the transform step. This check ensures that all transformed records are accounted for and nothing is omitted during the loading phase.

Note NULL values retrieved from the source must be set to a default value before loading the Staging tables. The Natural Key column in each staging table is the only exception to the “No NULLs” rule. A Nature Key must be supplied in order to support history (i.e. Type 1 or Type 2 SCDs).

METADATA

In OII, metadata refers to the data (e.g. currency conversion rates) and settings (e.g. the Earned Premium calculation method for an LOB) that is necessary to process the transactional and dimensional data loaded in Staging. The metadata tables include:

- Configuration (*OII_SYS* schema)
 - *OII_SYS.SYS_CONFIG* – This table contains several user-configurable settings for the OII system.
 - *OII_SYS.SYS_CURR_CNV* – OII loads this system table with the currency conversion data provided by the user in the *OII_ST.CURR_CNV* Staging table.
 - *OII_SYS.SYS_ERND_PREM_CONFIG* – OII loads this system table with the Earned Premium method chosen by the user per LOB. There are four methods to choose from:

- **D'** – Daily method. This method is standardized and formula-based. This is the default method for OII.
- **'M'** – Monthly (1/24) method. This method is standardized and formula-based.
- **'O'** – Original Premium method. This method is standardized and formula-based.
- **'S'** – Seasonal method. This method uses metadata that is loaded in the *OII_SYS.SYS_SEASONAL_CALC_CONFIG* table. The *FORMULA_CD* column in the *OII_SYS.SYS_ERND_PREM_CONFIG* table will need to be loaded with one of the two provided standardized seasonal methods ('1' or '2'). The *FORMULA_CD* column should contain '%' for all other Earned Premium Methods, (i.e. Daily, Monthly and Original).
- *OII_SYS.SYS_ERND_EXPO_CONFIG* – This table is pre-loaded with metadata that is used for the different Earned Exposure calculation methods. These methods are standardized and formula-based.
 - **TYP_CD** options are:
 - 'W' for Written Exposure.
 - 'E' for Earned Exposure.
 - **LOB_CD**: Provided to configure exposure per Line of Business.
 - **EXPO_BASIS_CD**: This code is optional and if it is not provided it should have a value of '%'
 - **FORMULA_CD**: For Written Exposure (TYP_CD 'W') the options are:
 - '1' – Original Written Exposure amount.
 - '2' – Original Written Exposure amount divided by 100
 - '3' – Original Written Exposure amount divided by 1000
 - '4' – Number of days between the Transaction Effective Date and the Coverage Expiration Date.
 - **FORMULA_CD**: For Earned Exposure (TYP_CD 'E') there is only one option:
 - '1' – The Earned Exposure amount is calculated based on the Written Exposure amount from above.
- *OII_SYS.SYS_RNG* – This table contains system-wide range lookup metadata.
- *OII_SYS.SYS_SEASONAL_CALC_CONFIG* – This table is pre-loaded with seasonal-calculation metadata that is used when the Seasonal Earned Premium method is selected.
- *OII_SYS.SYS_XLT_NM* – OII System Code Columns are pre-loaded in this table. Customer Code Columns that need corresponding descriptions are defined by the user in this table.
- *OII_SYS.SYS_XLT_VAL* – Descriptions for OII System Codes are pre-loaded in this table. Translations and descriptions for source codes are configured by the user in this table.
- Warehouse (OII_WH schema)
 - *OII_WH.PARTY_ROLE* – This table is pre-loaded with the system-defined Party Roles. For more information, see Appendix A: Relationship Codes.

- Dimensions (OII_DM schema)
 - *OII_DM.DT_DIM* – This table is pre-loaded with dates from 1900 – 2099 plus dummy dates.
 - *OII_DM.MTH_DIM* – OII loads this table with month information generated from the DT_DIM table.
 - *OII_DM.TIME_DIM* – This table is pre-loaded with hour information.

All Warehouse entities and data mart dimensions have a default row loaded with a dummy ID of zero (0). These entries should remain in place once data is loaded.

REQUIRED TABLES

Listed below are the tables that are required in order for the dimension tables to be loaded correctly.

Required Staging Tables for a Policy, Earned Exposure or Earned Premium Transaction:

- *OII_ST.BILL_ACCT* (Billing Account)
- *OII_ST.CLASS* (Class)
- *OII_ST.COVRG* (Coverage)
- *OII_ST.COVRG_STATUS* (Coverage Status)
- *OII_ST.DC* (Distribution Channel)
- *OII_ST.DED* (Deductible)
- *OII_ST.GRP_PRG* (Group Program)
- *OII_ST.INSUR_CO* (Insurance Company)
- *OII_ST.INSUR_CO* (Insurance Company)
- *OII_ST.INSUR_CO* (Insurance Company)
- *OII_ST.INSUR_CO* (Insurance Company)
- *OII_ST.INSUR_CO* (Insurance Company)
- *OII_ST.LMT* (Limit)
- *OII_ST.LOB* (Line of Business)
- *OII_ST.PLCY* (Policy)
- *OII_ST.PRODR* (Producer)
- *OII_ST.REINS_CNTRCT* (Reinsurance Contract)
- *OII_ST.REINS_CO* (Reinsurance Company)
- *OII_ST.REINS_CO* (Reinsurance Company)
- *OII_ST.UW* (Underwriter)

Required Staging Tables for a Claims Transaction:

- *OII_ST.BILL_ACCT* (Billing Account)
- *OII_ST.CAT* (Catastrophe)
- *OII_ST.CLASS* (Class)
- *OII_ST.CLMT* (Claimant)
- *OII_ST.CLMT_ADJ* (Claim Adjuster)

- *OII_ST.CLM_EXAMR (Claim Examiner)*
- *OII_ST.CLM_MGR (Claim Manager)*
- *OII_ST.CLM (Claim)*
- *OII_ST.COVRG (Coverage)*
- *OII_ST.COVRG_STATUS (Coverage Status)*
- *OII_ST.DC (Distribution Channel)*
- *OII_ST.DED (Deductible)*
- *OII_ST.FEAT (Feature)*
- *OII_ST.FEAT_STATUS (Feature Status)*
- *OII_ST.GRP_PRG (Group Program)*
- *OII_ST.INSUR_CO (Insurance Company)*
- *OII_ST.LITG (Litigation)*
- *OII_ST.LMT (Limit)*
- *OII_ST.LOB (Line of Business)*
- *OII_ST.PLCY (Policy)*
- *OII_ST.PRODR (Producer)*
- *OII_ST.REINS_CNTRCT (Reinsurance Contract)*
- *OII_ST.REINS_CO (Reinsurance Company)*
- *OII_ST.SUPPLIER (Supplier)*
- *OII_ST.UW (Underwriter)*

DATA VALIDATION

OII supports data validation in line with the ETL processes. OII supports two types of data validation: Code Lookup Validations and SQL Validations. Data validation is performed in ODI prior to loading any data to the Warehouse or Data Mart schemas. This process is critical to ensure the data quality and completeness for business requirements. The data validations listed below are pre-loaded in OII.

- **Code Lookup Validations:**
 - LOB_CD
 - PLCY_TRANS_CD
 - CLM_TRANS_CD
- **SQL Validations:**
 - BOOK_DT must be greater than or equal to SRC_SYS_PROC_DT.

- BOOK_DT must be greater than or equal to TRANS_EFF_DT.
- PLCY_EFF_DT must be less than or equal to PLCY_EXPR_DT.
- COVRG_EFF_DT must be less than or equal to PLCY_EXPR_DT.
- COVRG_EXPR_DT must be equal to PLCY_EXPR_DT.
- COVRG_EFF_DT must be greater than or equal to PLCY_EFF_DT.

There are several OII_SYS tables that are used to facilitate the Data Validation:

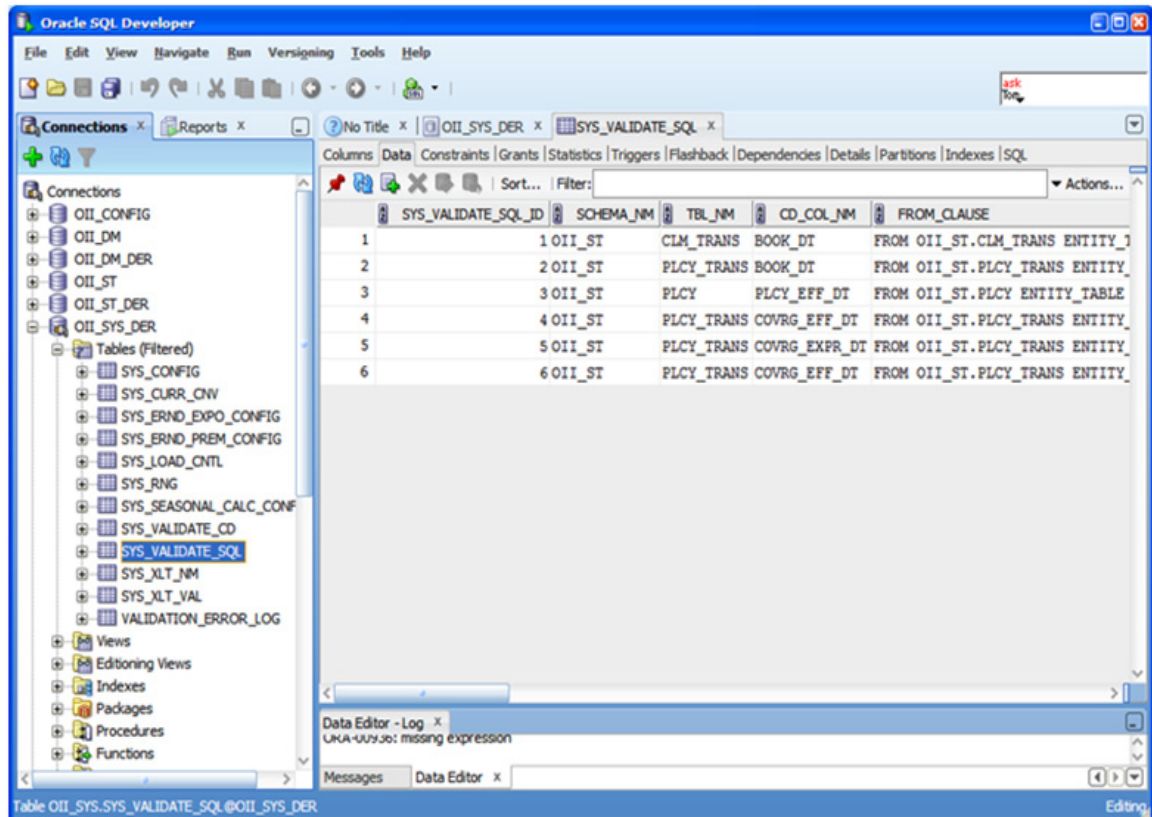
- **OII_SYS.SYS_VALIDATE_CD:**
 - This table defines all code-based validations.
 - The PROC_TYP_FLG column is used to enable/disable each validation. A value of 'V' indicates a validation is enabled and a value of 'X' indicates a validation is disabled.

The screenshot shows the Oracle SQL Developer interface with the OII_SYS.SYS_VALIDATE_CD table data displayed in the Data Editor. The table has six columns: SYS_VALIDATE_SQL_ID, SCHEMA_NM, TBL_NM, CD_COL_NM, and FROM_CLAUSE. The data is as follows:

SYS_VALIDATE_SQL_ID	SCHEMA_NM	TBL_NM	CD_COL_NM	FROM_CLAUSE
1	OII_ST	CLM_TRANS	BOOK_DT	FROM OII_ST.CLM_TRANS ENTITY_1
2	OII_ST	PLCY_TRANS	BOOK_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
3	OII_ST	PLCY	PLCY_EFF_DT	FROM OII_ST.PLCY ENTITY_TABLE
4	OII_ST	PLCY_TRANS	COVRG_EFF_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
5	OII_ST	PLCY_TRANS	COVRG_EXPR_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
6	OII_ST	PLCY_TRANS	COVRG_EFF_DT	FROM OII_ST.PLCY_TRANS ENTITY_1

Figure 6: OII_SYS.SYS_VALIDATE_CD

- **OII_SYS.SYS_VALIDATE_SQL:**
 - This table defines all SQL based validations.
 - The PROC_TYP_FLG column is used to enable/disable each validation. A value of ‘V’ indicates a validation is enabled and a value of ‘X’ indicates a validation is disabled.



The screenshot shows the Oracle SQL Developer interface. The left pane displays a tree view of the database schema, with the OII_SYS_DER schema expanded and the SYS_VALIDATE_SQL table selected. The main pane shows the table's structure with the following columns:

SYS_VALIDATE_SQL_ID	SCHEMA_NM	TBL_NM	CD_COL_NM	FROM_CLAUSE
1	OII_ST	CLM_TRANS	BOOK_DT	FROM OII_ST.CLM_TRANS ENTITY_1
2	OII_ST	PLCY_TRANS	BOOK_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
3	OII_ST	PLCY	PLCY_EFF_DT	FROM OII_ST.PLCY ENTITY_TABLE
4	OII_ST	PLCY_TRANS	COVRG_EFF_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
5	OII_ST	PLCY_TRANS	COVRG_EXPR_DT	FROM OII_ST.PLCY_TRANS ENTITY_1
6	OII_ST	PLCY_TRANS	COVRG_EFF_DT	FROM OII_ST.PLCY_TRANS ENTITY_1

Figure 7: OII_SYS.SYS_VALIDATE_SQL

The Data Validation can be executed from a pre-defined ODI Package or it can be executed directly from ODI:

- Data Validation is integrated into the ODI Package – **Warehouse Load Conditional**.

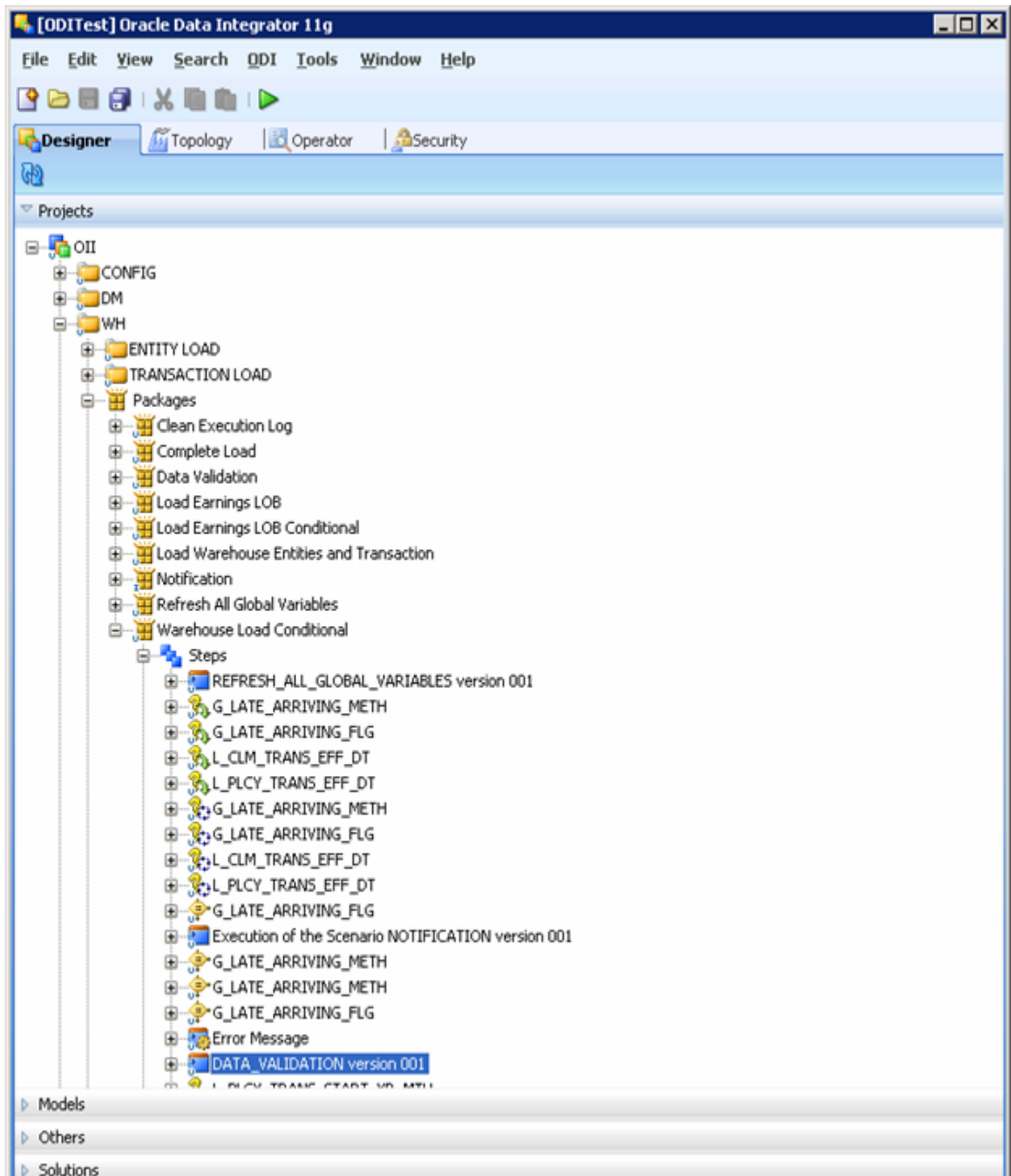


Figure 8: Data Validation

- To directly execute the Data Validation from ODI, locate the DATA_VALIDATION Scenario, right mouse click and select **Execute** from the menu.

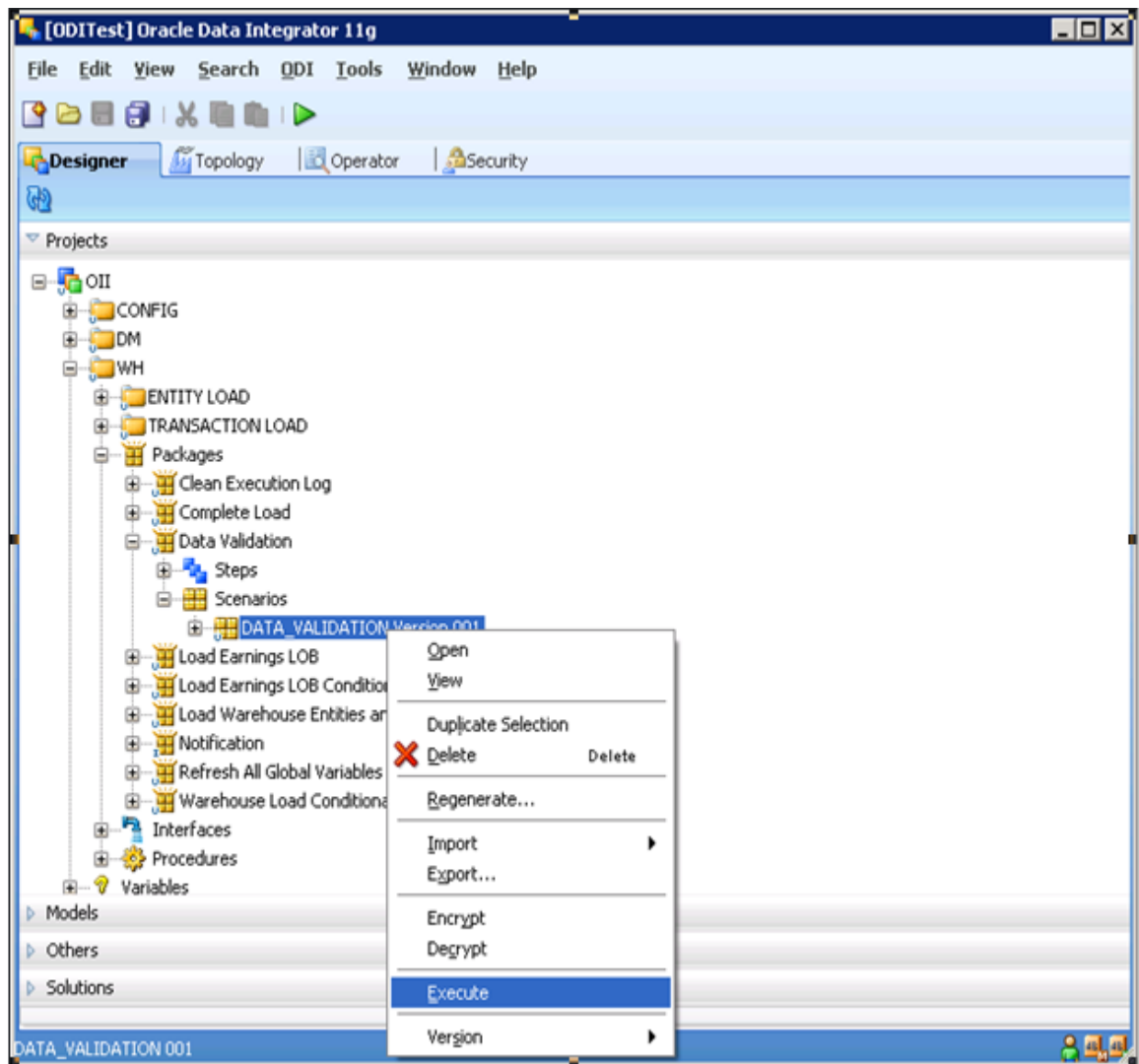


Figure 9: Execute Data Validation from ODI

If there are data validation errors, the Operator console will show the error as shown in the figure below.

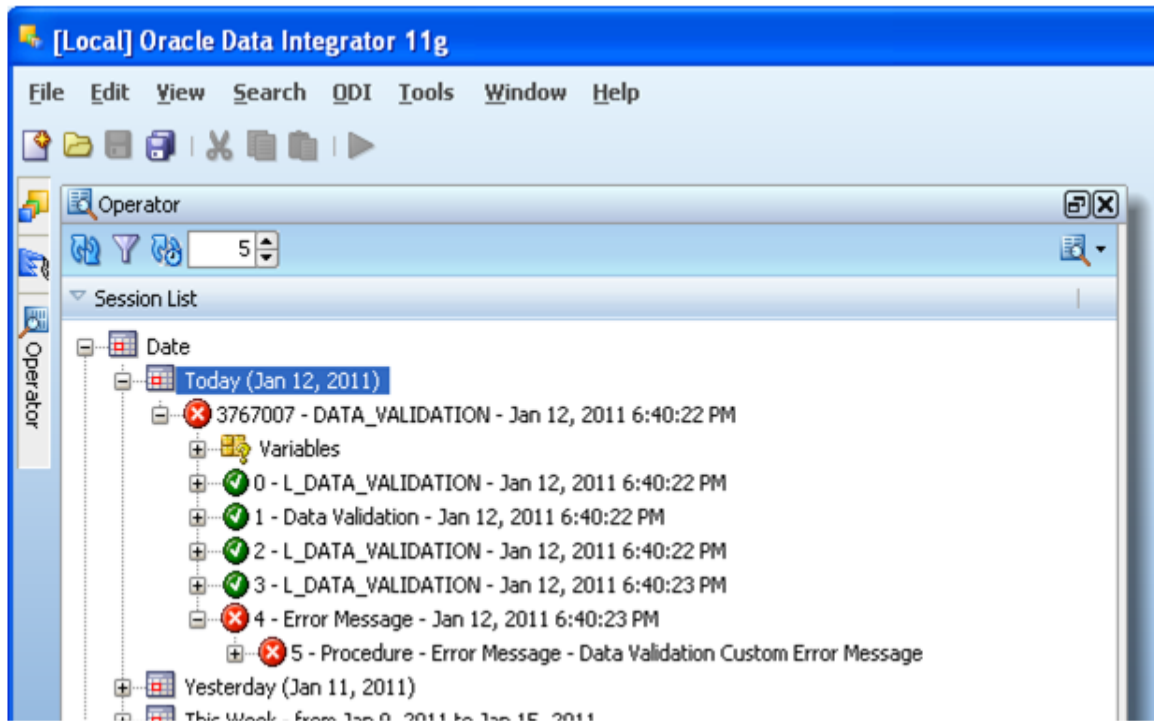


Figure 10: Data Validation Errors

The table, OII_SYS.VALIDATION_ERROR_LOG, contains details of validations errors. Using this information, further analysis on the staging data can be performed.

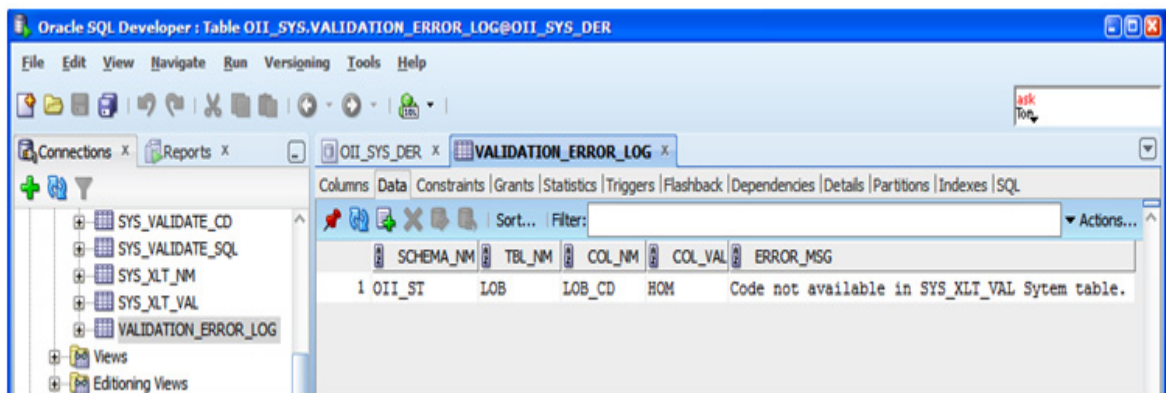


Figure 11: OII_SYS.VALIDATION_ERROR_LOG

With this example the validation error indicates a failure with translating an LOB_CD value of ‘HOM’. With this type of error, you will need to verify that the staging data contains a code value that is represented in the OII_SYS.SYS_XLT_VAL table.

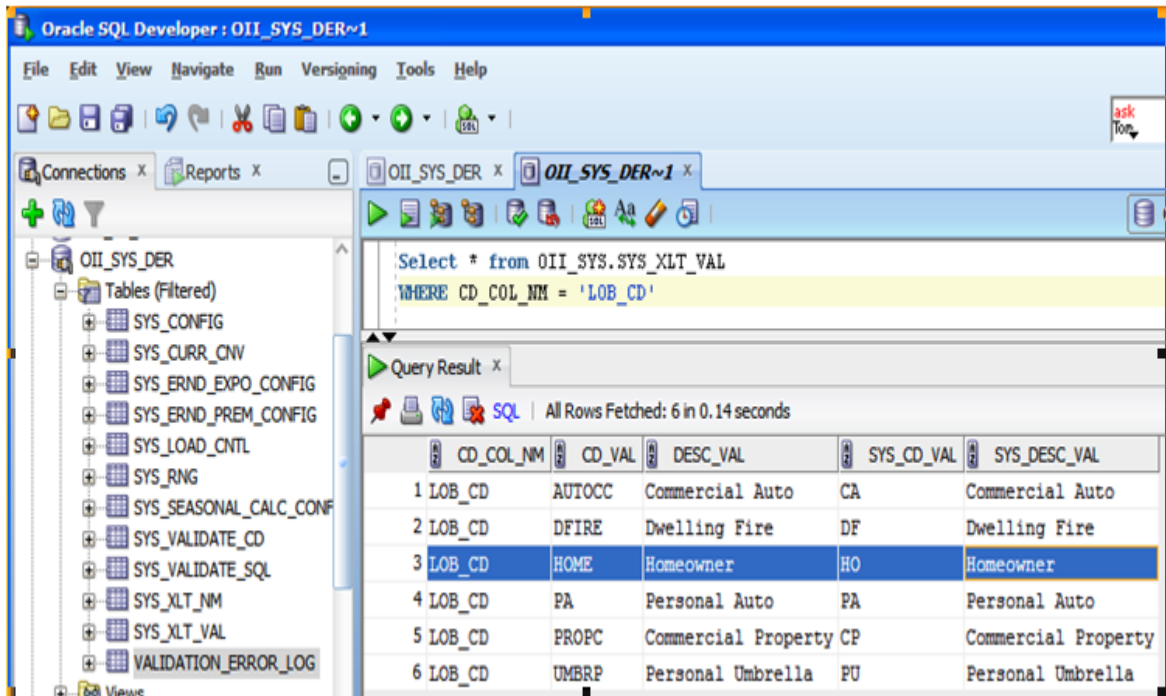


Figure 12: Failure in Translating the LOB_CD Value

With this example, the validation process has identified at least one row of data in the *OII_ST.LOB* table with an invalid LOB_CD value of ‘HOM’. This value should be ‘HOME’ which translates to the ‘Homeowner’ line of business with a SYS_CD_VAL of ‘HO’. This error must be corrected in staging before the data is allowed to be loaded into the warehouse.

DATA RELATIONSHIPS

UNIQUE KEYS

In the staging tables, there are specific transactions for Policies, Claims, Earned Premiums and Earned Exposures. For *OII_ST.PLCY* and *OII_ST.CLM* tables natural keys are used to uniquely identify a policy and a claim. Natural keys are discussed in detail in *Chapter 3: Program Flow*. Unique keys are used to connect Policy, Claim, Earned Premium and Earned Exposure transactions within the warehouse tables.

- *OII_WH.PLCY_TRANS*
 - Includes `PLCY_TRANS_ID` to uniquely identify a specific policy transaction.
- *OII_WH.CLM_TRANS*
 - Includes `CLM_TRANS_ID` to uniquely identify a specific claim transaction.
 - Includes `PLCY_ID` to establish a relationship between a specific claim transaction and its matching policy.
- *OII_WH.ERND_PREM_TRANS*
 - Includes `ERND_PREM_TRANS_ID` as well as `PLCY_TRANS_ID` to create the relationship between a specific Earned Premium transaction and its matching policy transaction.
- *OII_WH.ERND_EXPO_TRANS*
 - Includes `ERND_EXPO_TRANS_ID` as well as `PLCY_TRANS_ID` to create the relationship between a specific Earned Exposure transaction and its matching policy transaction.

COMMON/SHARED KEYS

In the staging environment, there are four possible types of transactions that can be loaded from the customer data: Policy Transactions, Claim Transactions, Earned Premium Transactions and Earned Exposure Transactions. The staging table that is critical in making a connection between all of these transactions is the *OII_ST.PLCY*. For each of the previously mentioned transaction types, a unique policy should be associated with that transaction.

When the data from the table *OII_ST.PLCY* is loaded to the warehouse, a `POLICY_ID` and a `POLICY_TRANS_ID` is generated in the table *OII_WH.PLCY_TRANS*. The `POLICY_ID` is shared between the tables *OII_WH.PLCY_TRANS* and *OII_WH.CLM_TRANS*, creating a connection between a specific claim and its matching policy. The `POLICY_TRANS_ID` is shared between the tables *OII_WH.PLCY_TRANS* and *OII_WH.ERND_PREMIUM* as well as *OII_WH.ERND_EXPOSURE*, creating a connection between a specific Earned Premium and Earned Exposure transaction and its matching policy.

DATA LOADING CONSIDERATIONS

In coding the Source-to-Staging ETL, it is important to know how OII expects the transactions to relate to one another and the policy as a whole. One of these considerations is how to distribute premium in a Personal Auto Policy.

PERSONAL AUTO PREMIUMS

In the Personal Auto LOB, each transaction has one associated driver. If that transaction includes any Written Premium, it will be assigned to the vehicle and/or driver associated with that transaction.

In order to perform premium or loss analysis in Personal Auto at the Vehicle and/or Driver level (i.e. a sub-policy level), the premium should be apportioned across multiple transactions. The premium assigned to that transaction will be associated with the particular vehicle and/or driver for that transaction when the data is loaded into the Staging tables. Transactions loaded with zero premium will not assign any premium to the vehicle and/or driver associated with that transaction, so sub-policy level analysis will not be available.

Chapter 5

Data Loading

After the Source-to-Staging ETL has loaded the transactional data, the OII system is ready to be configured to handle the customer data, loaded and prepared for presentation to the end user. The Data Loading chapter introduces the OII System Codes that control processing of transactions, load configuration options, currency configuration, load execution and OBIEE Repository linkage.

SYSTEM CODES

System Codes are functional codes in OII that tell the system how to route and process transactions. System Codes are defined in the OII_SYS.SYS_XLT_NM table. System code translations and description are contained in the OII_SYS.SYS_XLT_VAL table. Four categories are covered:

- **Claim Transaction Code (CLM_TRANS_CD)** – This system code represents what type of loss activity occurred in the claim transaction. The functional codes supplied with OII cannot be changed. They are described below. Additional Claim Transaction Codes can be added by the user to group related transaction types, but these transactions will never be included in the monthly snapshot data marts.
- **Line Of Business Code (LOB_CD)** – This system code identifies the LOB classification. By default, there is a single code defined for each of the six pre-defined LOBs. Additional Line of Business Codes can be set by the user in Warehouse Palette. When an LOB is published, its code is added to the OII_SYS.SYS_XLT_VAL table.
- **Policy Transaction Code (PLCY_TRANS_CD)** – This system code identifies the policy transaction. The functional codes supplied with OII cannot be changed. They are described below. Additional Policy Transaction Codes can be added by the user to group related transaction types, but these transactions will never be included in the monthly snapshot data marts.

SOURCE CODE TRANSLATION AND DESCRIPTION

In addition to providing descriptions for the pre-defined OII System Codes, the OII_SYS.SYS_XLT_VAL table translates client source codes present in the Staging data into OII System Codes. No additional configuration is needed in the OII_SYS.SYS_XLT_NM table if a source code is being translated to one of the three System Codes defined above.

If descriptions are needed for source codes outside of the three categories above, an entry is needed in the OII_SYS.SYS_XLT_NM table to define the staging table and column that are being translated. The target code and description columns are also defined here. Source codes are translated and described in the OII_SYS.SYS_XLT_VAL table exactly as it is done for the System Codes.

IMPORTANT Translations and descriptions for client source codes must be provided before loading Staging data to the Warehouse. If transactions have already been loaded to the Warehouse, changes to the translations and descriptions will only affect transactions loaded after those changes have been made.

CLAIM TRANSACTION CODES

Claim Transaction Codes are required to process claim transactions correctly. These codes also allow for the proper display of claim transaction types in OBIEE. There are currently 19 Claim Transaction Codes.

These consist of three basic categories of transactions:

- **Reserve** – The initial reserve set up on a claim, including the initial reserve on a reopened claim.
- **Change** – Any change in reserves on a claim, including the final reserve transaction (eliminating the loss reserve) in a closed or reclosed claim.
- **Paid/Recovered** – Any paid amounts on a claim, including any financial adjustments or modifications to paid losses. This also includes any recoveries made for Salvage, Subrogation or Deductible.

For each category, there are six claim transaction types:

- **Loss** – Amounts reserved, paid or recovered in relation to the settlement of the claim.
- **Allocated Adjustment Expense (ALAE)** – Amounts reserved, paid or recovered in relation to adjustment expenses specifically allotted to a claim.
- **Unallocated Adjustment Expense (ULAE)** – Amounts reserved, paid or recovered in relation to adjustment expenses that are not specifically allocated to a claim. These amounts are not usually used on a claim transaction basis, but rather on a financial accounting basis.
- **Salvage** – Amounts reserved, paid or recovered in relation to property collected as salvage in the settlement of a claim.
- **Subrogation** – Amounts reserved, paid or recovered in relation to reimbursement from a third party responsible for damage or liability in the settlement of a claim.
- **Deductible** – Amounts reserved, paid or recovered in relation to reimbursement of a deductible in the settlement of a claim.

In addition to these eighteen codes, there is a special code that does not fit into the categories above.

- **Non Financial** – The Non Financial code allows for a transaction to be entered without affecting the reserve amount. When loaded, the transaction is available in the transaction data marts, but it will never be included in the monthly snapshot data marts.

The 19 functional Claim Transaction Codes are grouped in the three categories (Reserve, Change and Paid\Recovered) along with the special Non Financial code below:

Claim Transaction Type	Code	Description
Loss Reserve	LR	Initial or reopened Expense Reserve
Allocated Expense Reserve	AR	Initial or reopened ALAE Reserve
Unallocated Expense Reserve	UR	Initial or reopened ULAE Reserve
Salvage Reserve	SR	Initial or reopened Salvage Reserve
Subrogation Reserve	RS	Initial or reopened Subrogation Reserve
Loss Deductible Recovery Reserve	DR	Initial or reopened Deductible Reserve
Loss Change	LC	Change in Loss Reserve
Allocated Expense Change	AC	Change in ALAE Reserve
Unallocated Expense Change	UC	Change in ULAE Reserve
Salvage Change	SC	Change in Salvage Reserve
Subrogation Change	RC	Change in Subrogation Reserve
Loss Deductible Change	DC	Change in Deductible Reserve
Paid Loss	PL	Paid Loss Amount
Allocated Expenses Paid	AE	Paid ALAE Amount
Unallocated Expenses Paid	UE	Paid ULAE Amount
Salvage Recovered	SL	Recovered Salvage Amount
Subrogation Recovered	SB	Recovered Subrogation Amount
Loss Deductible Recovered	DE	Recovered Deductible Amount
Non Financial	NF	Non-Financial Amount (may be zero or non-zero dollar amount)

POLICY TRANSACTION CODES

Policy Transaction Codes are required to process premiums and policy transactions correctly. Like the claim codes, the policy codes allow for the proper display of policy transaction types in OBIEE. There are currently 9 Policy Transaction Codes.

Policy Transaction Type	Code
Additional Premium Audit	AP
Cancelled Flat	CF
Cancellation	CN
Cancelled Other	CO
New Business	NB
Non-Premium Endorsement	NP
Renewal Business	RB
Reinstatement Business	RE
Return Premium Audit	RP

LOAD CONFIGURATION

The load parameters listed below are found in the OII_SYS.SYS_CONFIG table. These values should be set prior to executing the data load in ODI.

ANALYSIS OPTIONS

- **Code Names:** ANALYZE_WH, ANALYZE_DM
- **Description:** These parameters determine if the warehouse and/or data mart need to be analyzed as part of internal loading (ETL) process. A “Y” value (default) will invoke the analysis process (dbms_stats) as part of ETL. Oracle recommends running stats on a regularly scheduled basis. The analysis option is provided as a way to automate this feature. The analysis options are used in the packages below:
- Load Warehouse Entities and Transactions
- Load Monthly Snapshot Facts
- Load Data Mart Dimensions and Transactional Facts

CURRENCIES

- **Code Names:** LOCAL_CURR, GLOBAL_CURR_1, GLOBAL_CURR_2, and GLOBAL_CURR_3
- **Description:** These parameters define the Local Reporting Currency (i.e. the default currency; LOCAL_CURR) and the optional Global Reporting Currencies. Up to three Global Reporting Currencies can be defined (i.e. GLOBAL_CURR_1, GLOBAL_CURR_2 and GLOBAL_CURR_3). Transaction amounts are input to the OII system in as Document Currency. The Local Reporting Currency and the Global Reporting Currencies can be utilized along with the currency conversion rate data (see *Currency Configuration* on page 51) to convert the Document Currency to the corresponding Reporting Currency equivalents.
- **Example Value:** “USD”

DEFAULT BEGIN DATE

- **Code Name:** DFLT_BEGIN_DT
- **Description:** This date represents the lowest date value that the system can support. This value will also serve as a default date value if an input date is NULL. The date format “MM-DD-YYYY” is recommended. The format of the value needs to match the DFLT_DT_FORMAT value.
- **Example Value:** “01-01-0001”

DEFAULT DATE FORMAT

- **Code Name:** DFLT_DT_FORMAT
- **Description:** Default date format for the system. The date format “MM-DD-YYYY” is recommended.

DEFAULT DATE/TIME FORMAT

- **Code Name:** DFLT_DT_TIME_FORMAT
- **Description:** Default date time format. The format “MM-DD-YYYY HH24:MI:SS” is recommended.

DEFAULT END DATE

- **Code Name:** DFLT_END_DT
- **Description:** This is the default end date for the system. This date will be used to mark the end date of active record in case of slowly changing record. The format of the value needs to match the DFLT_DT_FORMAT value.
- **Example Value:** “12-31-9999”

DEFAULT NUMERIC VALUE

- **Code Name:** NUM_DFLT_VAL
- **Description:** The Default Numeric Value should be supplied for any numeric field in Staging when no data is available from the source system. If the value supplied matches the Default Numeric Value, OII knows to treat this field as having no data. NUM_DFLT_VAL must be a single digit number.
- **Example Value:** “0”

DEFAULT STRING VALUE

- **Code Name:** STR_DFLT_VAL
- **Description:** The Default String Value should be supplied for any string field in Staging when no data is available from the source system. If the value supplied matches the Default Numeric Value, OII knows to treat this field as having no data. STR_DFLT_VAL must be a single character and cannot be a single quote character.
- **Example Value:** “ ”

INCURRED LOSS CALCULATION METHOD

- **Code Name:** INCRD_CALC_METH is the Incurred Loss Calculation Method, value 1 – 5, default is 1
- **Description:** The following options are available.

- Method 1: Loss Reserve + Paid Loss
- Method 2: Loss Reserve + Paid Loss + Salvage + Salvage Reserve + Subrogation + Subrogation Reserve
- Method 3: Loss Reserve + Paid Loss + Salvage + Salvage Reserve + Subrogation + Subrogation Reserve + Deductible + Deductible Reserve
- Method 4: Loss Reserve + Paid Loss - Salvage + Salvage Reserve - Subrogation + Subrogation Reserve + Deductible + Deductible Reserve
- Method 5: Loss Reserve + Paid Loss + Salvage + Subrogation + Deductible

INCURRED LOSS WITH EXPENSE CALCULATION METHOD

- **Code Name:** INCRD_WEXPNS_CALC_METH is the Incurred Loss with Expense Calculation Method, value 1 – 2, default is 1
- **Description:** The following options are available.
 - Method 1: Incurred Loss + Allocated Expense + Allocated Expense Reserve + Unallocated Expense + Unallocated Expense Reserve
 - Method 2: Incurred Loss + Allocated Expense + Allocated Expense Reserve

LATE ARRIVING METHODOLOGY

- **Code Name:** LATE_ARRIVING_METH
- **Description:** This parameter determines the loading method to be used. The loading methods listed below are supported.
 - Method 1: Ignore Late-Arriving Transaction. Load the regular transactions.
 - Method 2: If there are Late-Arriving Transactions, stop the load and wait for user intervention to continue. Otherwise, continue as normal.
 - Method 3: Load Late-Arriving Transactions but do not delete/reload Earned Premium/ Earned Exposure/MTH_SNAP_SHOT for the affected period.
 - Method 4: Load Late Arriving Transactions and automatically reload Earned Premium/ Earned Exposure/MTH_SNAP_SHOT for the affected period.

LOG NAME/PATH

- **Code Name:** LOG_FILE_NM_PATH
- **Description:** Path and File name of the OS log file to be generated in case of late arriving notification. File will be created on the machine hosting the ODI AGENT, hence path should be for the machine hosting agent responsible for invoke the processes.

CURRENCY CONFIGURATION

OII supports loading transactional amounts in an unlimited number of Document Currencies and reporting in up to four Reporting Currencies.

Transactions are always loaded in denominations of the Document Currency that is defined at the transactional level. For claim transactions, a code for the Document Currency is entered in the DOC_CURR_CD column of the OII_ST.CLM_TRANS (Claim Transaction) table. If a policy transaction, Earned Exposure transaction or Earned Premium transaction is loaded, a code for the Document Currency must be entered in the DOC_CURR_CD column of the OII_ST.PLCY_TRANS (Policy Transaction) table.

OII can support up to four Reporting Currencies: one Local Reporting Currency (LOCAL_CURR) and up to three additional Global Reporting Currencies (GLOBAL_CURR_1, GLOBAL_CURR_2 and GLOBAL_CURR_3), as defined above under “Currencies.”

A daily exchange rate must be entered for the Document Currency and each Reporting Currency that is defined. This exchange rate data is transferred to the OII_SYS.CURR_CNV (Currency Conversion) table whenever the warehouse tables are loaded. See *Load Execution* on page 55 for definitions of which scenarios involve loading warehouse tables.

DEFAULT CURRENCY CONVERSION RATES

Before loading the warehouse tables for the first time, the OII_ST.LOAD_DEFAULT_EXCHANGE_RATE procedure should be executed for any currency that is both a Document Currency and a Reporting Currency. This will populate the necessary rows in the OII_ST.CURR_CNV table so that conversion rates will not need to be loaded to convert a currency to itself. Executing this procedure sets the self-to-self conversion rate at 1 for the specified time span.

Connect to the OII_ST schema in Oracle SQL Developer. Under Procedures in OII_ST, find the LOAD_DEFAULT_EXCHANGE_RATE procedure, as shown in Figure 13 on page 52.

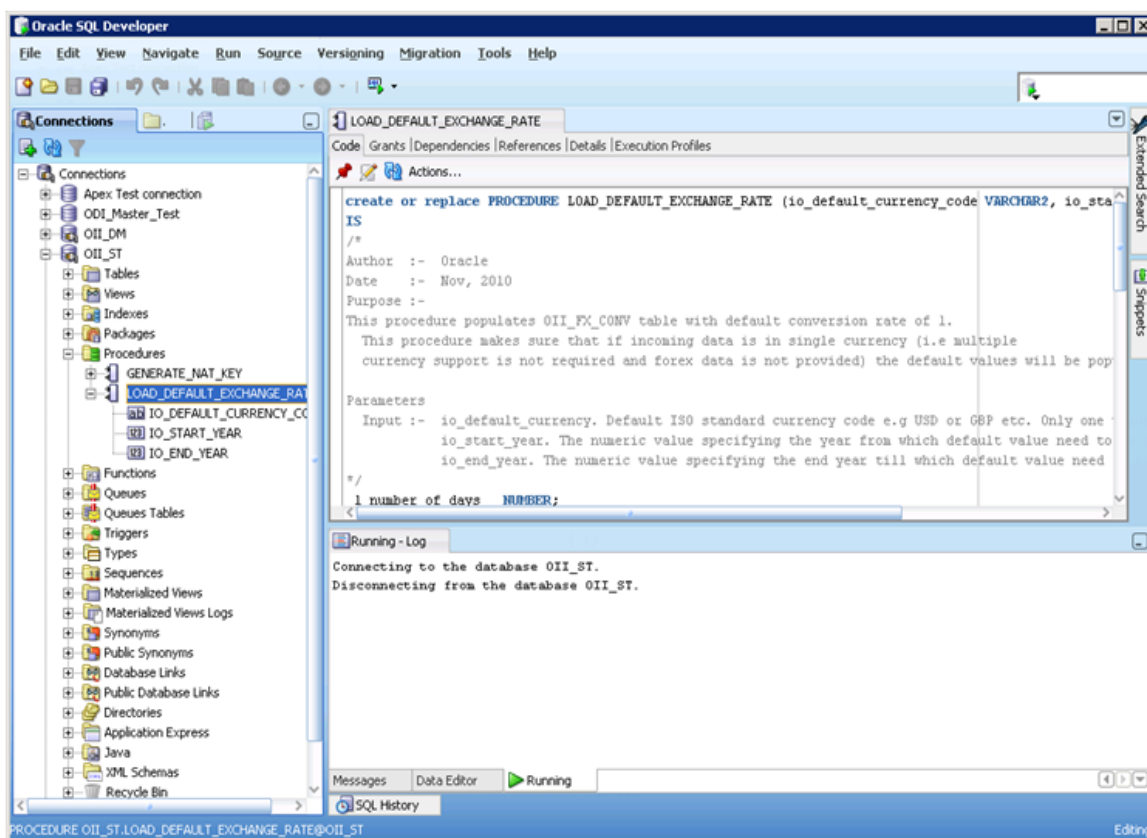


Figure 13: The `LOAD_DEFAULT_EXCHANGE_RATE` Procedure

The inputs are:

- **IO_DEFAULT_CURRENCY_CODE** – This is the Reporting Currency Code (e.g. ‘USD’ or ‘GBP’) as defined in the `OII_SYS.SYS_CONFIG` table.
- **IO_START_YEAR** – This numeric value specifies the year from which the default conversion rate needs to be generated.
- **IO_END_YEAR** – This numeric value specifies the year to which the default conversion rate needs to be generated.

When specifying a start year and end year, select years that encompass the entire historical and future reporting period. This will populate all necessary conversion rate data for the currency so that this procedure only needs to be run once.

Right click on the procedure, `LOAD_DEFAULT_EXCHANGE_RATE`, and select **Run**. The **Run PL/SQL** dialog box will appear.

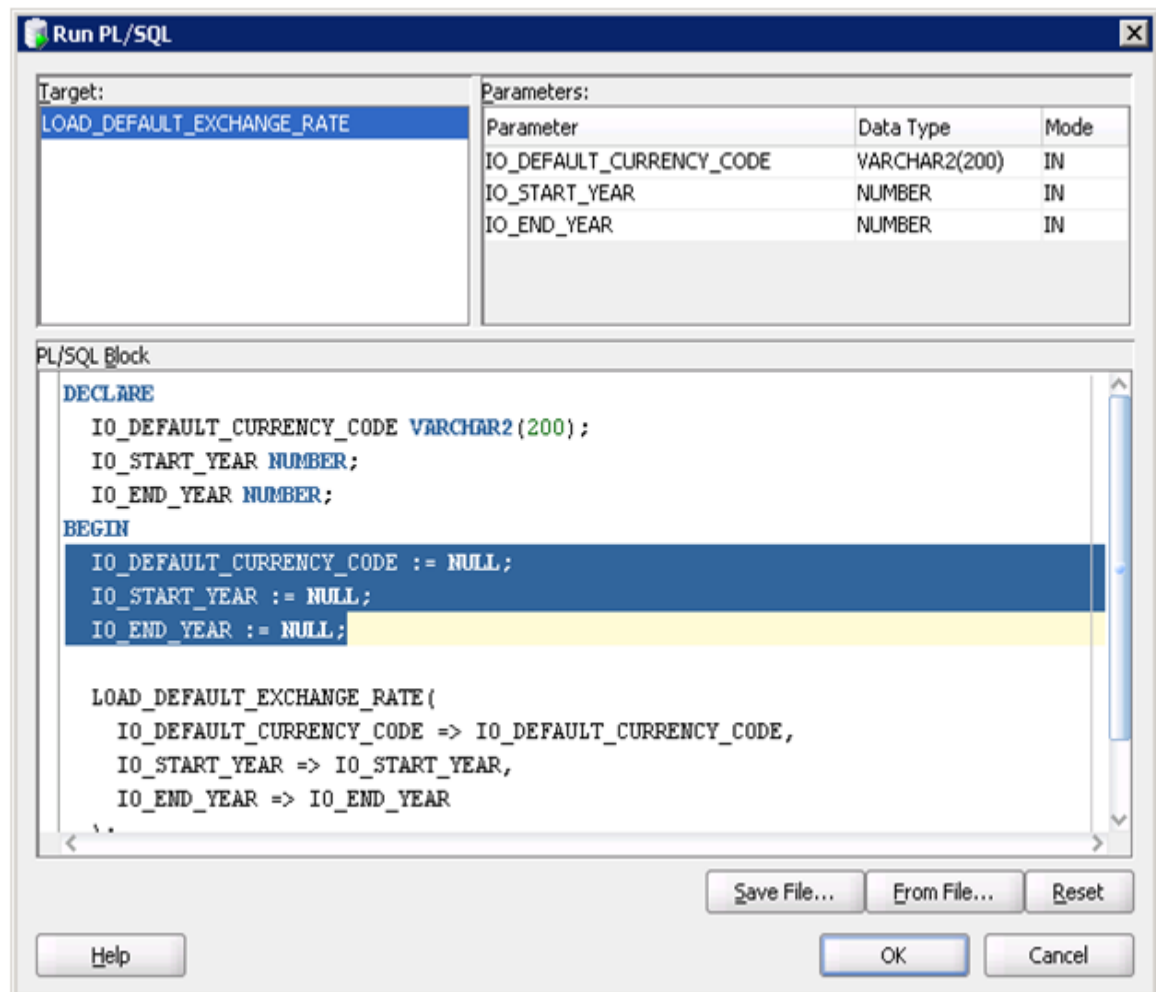


Figure 14: Run PL/SQL Dialog Box

Replace the NULL values with the desired values, as shown in the highlighted inputs above. Click the “OK” button to run. Note that the `IO_DEFAULT_CURRENCY_CODE` input value will need single quotes (e.g. ‘USD’) because it is a `VARCHAR2` data type.

MULTIPLE CURRENCIES AND CONVERSION RATES

If a single currency is used for the Document Currency and the Reporting Currencies, loading the default currency conversion rate is sufficient. If transactional amounts will be reported in currencies other than the currency used for loading, daily conversion rate data will need to be provided for each Document-Currency-to-Reporting-Currency pair.

If needed, daily currency conversion rate data is loaded in the `OII_ST.SYS_CURR_CNV` (Currency Conversion) table. The columns in the `OII_ST.SYS_CURR_CNV` table are listed below:

- `CURR_CNV_DT` – This is the date for which the exchange rate applies.

- **CURR_CNV_FROM_CD** – This is the code for the Document Currency (e.g. USD or GBP). This code will be defined in the OII_SYS.SYS_CONFIG table if it is also a Reporting Currency. Otherwise, Document Currency codes will only appear in the DOC_CURR_CD column of the appropriate transaction table and here, in the OII_ST.SYS_CURR_CNV table.
- **CURR_CNV_TO_CD** – This is the code for the appropriate Reporting Currency as defined in the OII_SYS.SYS_CONFIG table (e.g. EUR or INR).
- **CURR_CNV_RATE** – This numeric value represents the coefficient multiplied by the Document Currency (CURR_CNV_FROM_CD) to produce an equal value in terms of the Reporting Currency (CURR_CNV_TO_CD).

Loading of the OII_ST.SYS_CURR_CNV table with daily currency conversion rates should be incorporated into or alongside the Source-to-Staging ETL process that loads transactional data.

SYS_ID	CURR_CNV_FROM	CURR_CNV_TO	CURR_CNV_RATE		
*1	49237	31-DEC-99	USD	EUR	1.36992
*2	5627	30-DEC-99	USD	EUR	1.36992
*3	65958	29-DEC-99	USD	EUR	1.36992
*4	24199	28-DEC-99	USD	EUR	1.36992
*5	65957	27-DEC-99	USD	EUR	1.36992
*6	14777	26-DEC-99	USD	EUR	1.36992
*7	53426	25-DEC-99	USD	EUR	1.36992
*8	24198	24-DEC-99	USD	EUR	1.36992
*9	24197	23-DEC-99	USD	EUR	1.36992
*10	24196	22-DEC-99	USD	EUR	1.36992
*11	53425	21-DEC-99	USD	EUR	1.36992
*12	14776	20-DEC-99	USD	EUR	1.36992
*13	49236	19-DEC-99	USD	EUR	1.36992
*14	24195	18-DEC-99	USD	EUR	1.36992
*15	53424	17-DEC-99	USD	EUR	1.36992
*16	5626	16-DEC-99	USD	EUR	1.36992
*17	49235	15-DEC-99	USD	EUR	1.36992
*18	53423	14-DEC-99	USD	EUR	1.36992
*19	24194	13-DEC-99	USD	EUR	1.36992

Figure 15: OII_SYS.SYS_CURR_CNV Table

LOAD EXECUTION

Several execution options are available to load OII. The scenarios listed below can be executed to load the appropriate layers of the OII data model.

COMPLETE LOAD

The COMPLETE_LOAD scenario loads the Warehouse, Dimensions, Transactional Fact tables and the Monthly Snapshot Fact tables.

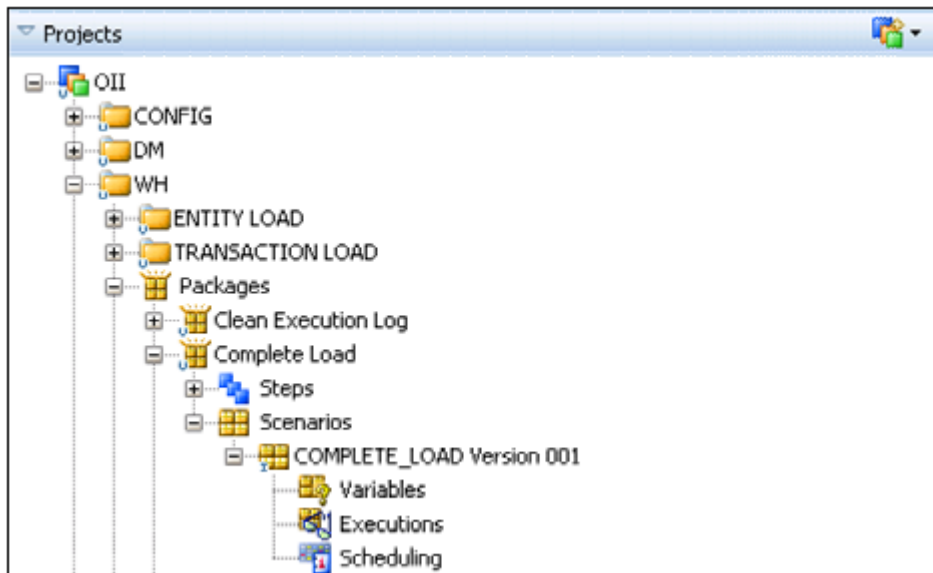


Figure 16: Complete Load

WAREHOUSE LOAD

The WAREHOUSE_LOAD_CONDITIONAL scenario loads the Warehouse tables. This scenario is useful in debugging the data produced from the Source-to-Staging ETL..

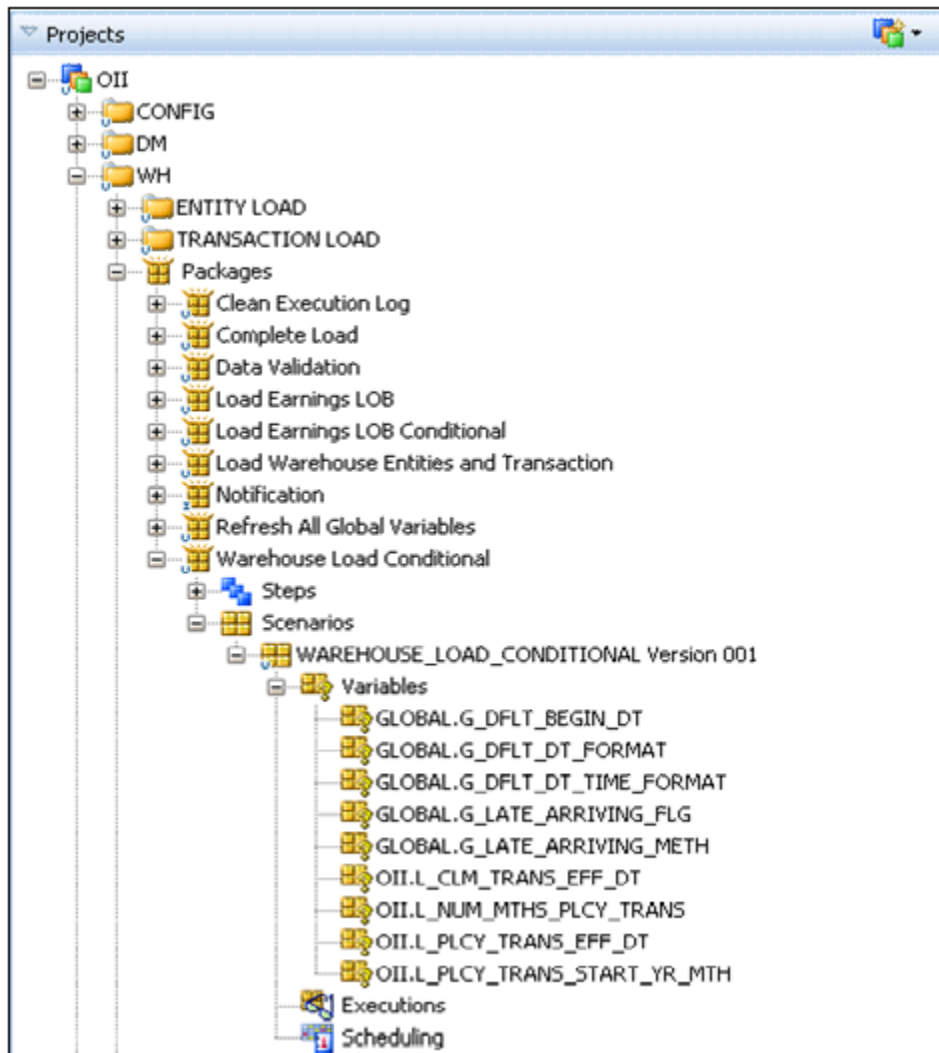


Figure 17: Warehouse Load

DATA MART LOAD

The LOAD_DATA_MART_DIMENSIONS_AND_FACTS scenario loads the Dimensions, Transactional Fact tables and the Monthly Snapshot Fact tables. This scenario can be used to complete the loading after the Warehouse Load has been executed.

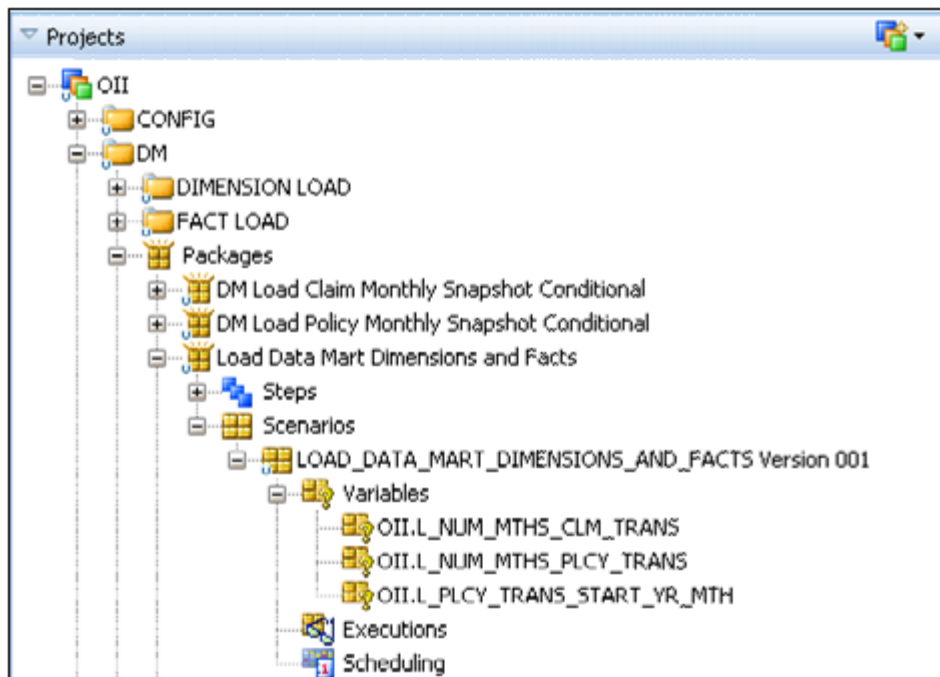


Figure 18: Data Mart Load

DIMENSIONS AND TRANSACTIONAL FACTS LOAD

The LOAD_DATA_MART_DIMENSIONS_AND_FACTS_DAILY scenario loads the Dimensions and the Transactional Fact tables. This scenario can be used after the Warehouse Load_ whenever the Monthly Snapshot Fact tables are not ready to be loaded. This may be the case if routine loads are performed multiple times within a given month.

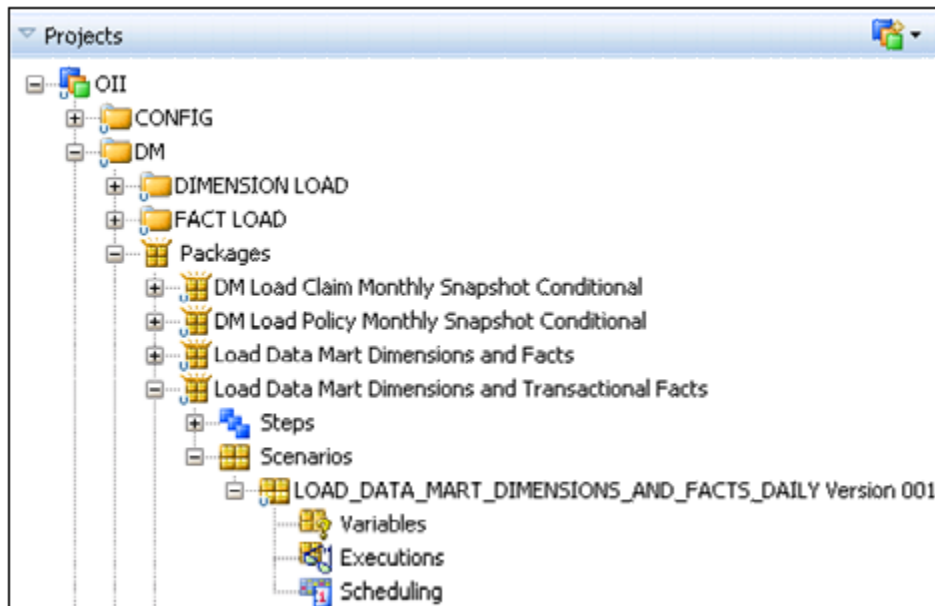


Figure 19: Dimensions and Transactional Facts Load

MONTHLY SNAPSHOT FACTS LOAD

The LOAD_MONTHLY_SNAPSHOT scenario loads the Monthly Snapshot Fact tables. This scenario can be used to populate the Monthly Snapshot Fact tables after the Warehouse, Dimensions and Transactional Fact tables have been loaded.

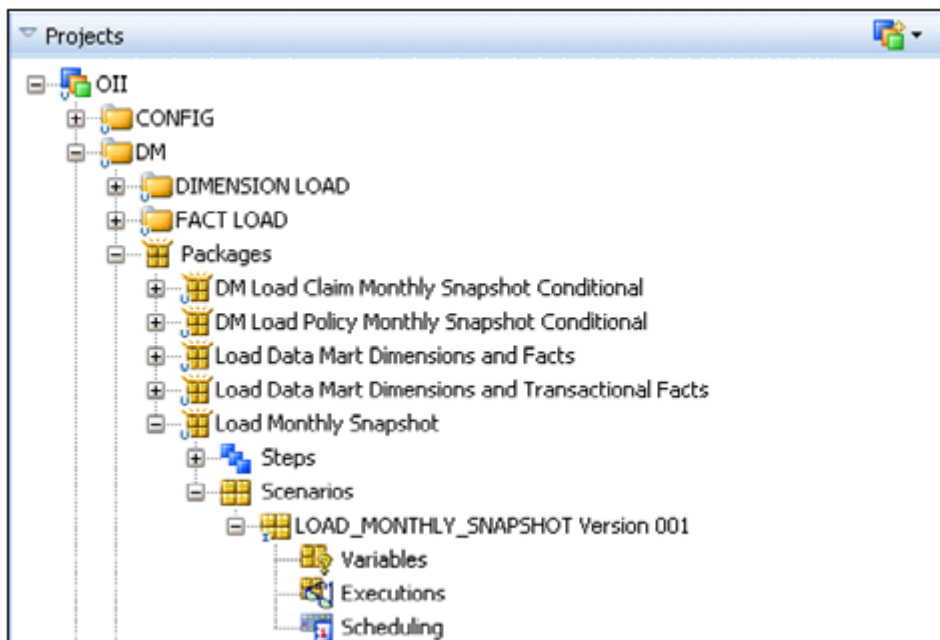


Figure 20: Monthly Snapshot Facts Load

LOAD SCHEDULING

Each of the above mentioned Scenarios can be scheduled. A schedule for a scenario will have to be defined and configured in Oracle Data Integrator. Within the help section of Oracle Data Integrator there are detailed instructions on how to define and configure a schedule. Schedules are attached to specific scenarios in Oracle Data Integrator. These schedules are managed by the scheduler built-in run-time agent. Each Schedule allows a start date and a repetition cycle to be specified.

CLEARING THE OBIEE CACHE AT THE END OF THE DATA LOAD

Once you have completed the load process you should purge the OBIEE cache to improve performance. To do so, follow the steps below.

1. Create a file, (e.g. *purgecache.txt*) which contains the following command:

```
Call SAPurgeAllCache()
```

2. Run **bi-init.cmd** (or **bi-init.sh** on UNIX) from the following location:

```
<MW_HOME>/instances/<instance number>/bifoundation/OracleBIApplication/coreapplication/setup
```

3. Run the **nqcmd** utility to purge the OBIEE cache:

```
nqcmd -d my_dsn -u my_username [-p my_password] -s sql_input_file
```

Where:

- **my_dsn** - is the ODBC connection name for the BI Server.
- **my_username** - is the WebLogic admin user name.
- **my_password** - is the password.
- **sql_input_file** - is the full path of the file you created in step 1 (e.g. *purgecache.txt*)..

For example:

In Windows:

```
nqcmd -d AnalyticsWeb -u <weblogic_admin user> -p <weblogic_admin password> -s purgecache.txt
```

In Unix:

```
./nqcmd -d AnalyticsWeb -u <weblogic_admin user> -p <weblogic_admin password> -s purgecache.txt
```


Data Visualization

OBIEE REPOSITORY

OII offers pre-defined OBIEE subject areas that match each LOB reference implementation in Warehouse Palette. Any modifications to the reference implementation must be manually updated in the OBIEE Repository.

Any UDFs that are utilized and contain data will need to be added to the presentation layer in the OBIEE Repository in order for them to be available to the end user in the reports and dashboards.

Please reference the following manuals in the OBIEE documentation set:

- *Oracle Business Intelligence Server Administration Guide*
- *Oracle Business Intelligence Presentation Services Administration Guide*
- *Oracle Business Intelligence Answers, Delivers, and Interactive Dashboards User Guide*

OBIEE REPOSITORY LINKAGE

Once data has been loaded to the Data Mart schema, the OII data model must be linked to the OBIEE Repository before logging into OBIEE Analytics to view the data. This section details those steps.

IMPORTANT In the case that an LOB was created that was not included in the six that are available out of the box there are a number of steps that need to be performed in order to have that LOB included in the AnalyticsWeb repository.

If this is the case then skip to *OBIEE Repository Linkage for New or Updated LOB* on page 71. Complete the steps in that section before executing the steps in this section.

Step 1: Open the OBIEE Repository

1. Open the OBIEE Repository Administration Tool by selecting:
Start>All Programs>Oracle Business Intelligence>Administration
2. When the Administration Tool opens, select **File>Open>Online** to display the Online Repository dialog box.
3. In the login dialog box that appears, enter the Repository Password (the default password is *welcome1*) and click on **OK**.
4. Enter a valid User Name and Password. The default user name is *weblogic* and whatever password you selected when you installed OBIEE.

The following Repository screen will open:

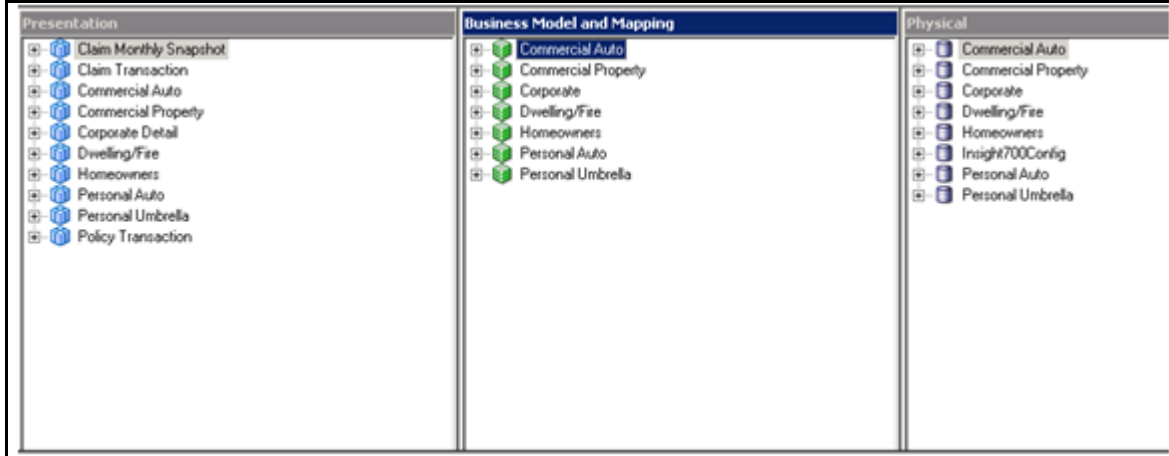


Figure 21: Layers of the OBIEE Repository for OII

Step 2: Configure the Line of Business Subject Areas

1. Go to the Physical Layer pane. This pane contains the Subject Areas for each Line of Business (Commercial Auto, Homeowners, etc.) and an additional Subject Area called **Insight700Config**. You will need to configure the settings under each of the Line of Business Subject Areas and the **Insight700Config** Subject Area.

Important The following configuration steps apply only to the Subject Areas for each Line of Business. The steps to configure the **Insight700Config** Subject Area are on page 67.

2. Expand each of the Line of Business Subject Areas under the Physical Layer pane. The Connection Pool icon will appear underneath each Subject Area:

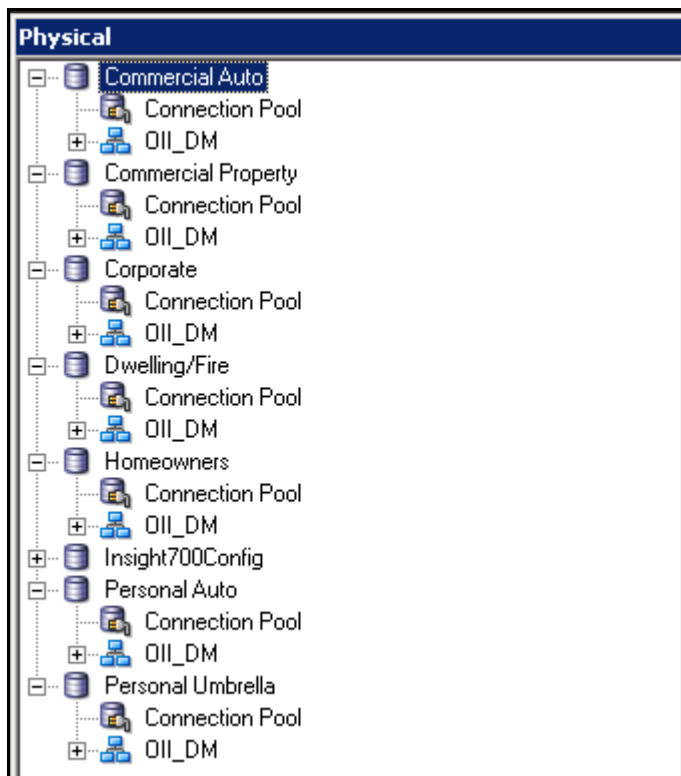


Figure 22: Subject Areas under the Physical Layer

3. Starting with the Commercial Auto Subject Area, click on the Connection Pool icon.

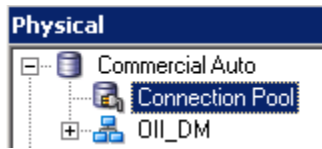


Figure 23: Connection Pool Icon

4. A message box will appear and ask you if you want to check out the Connection Pool.
5. Select **Yes**. The Connection Pool screen opens:

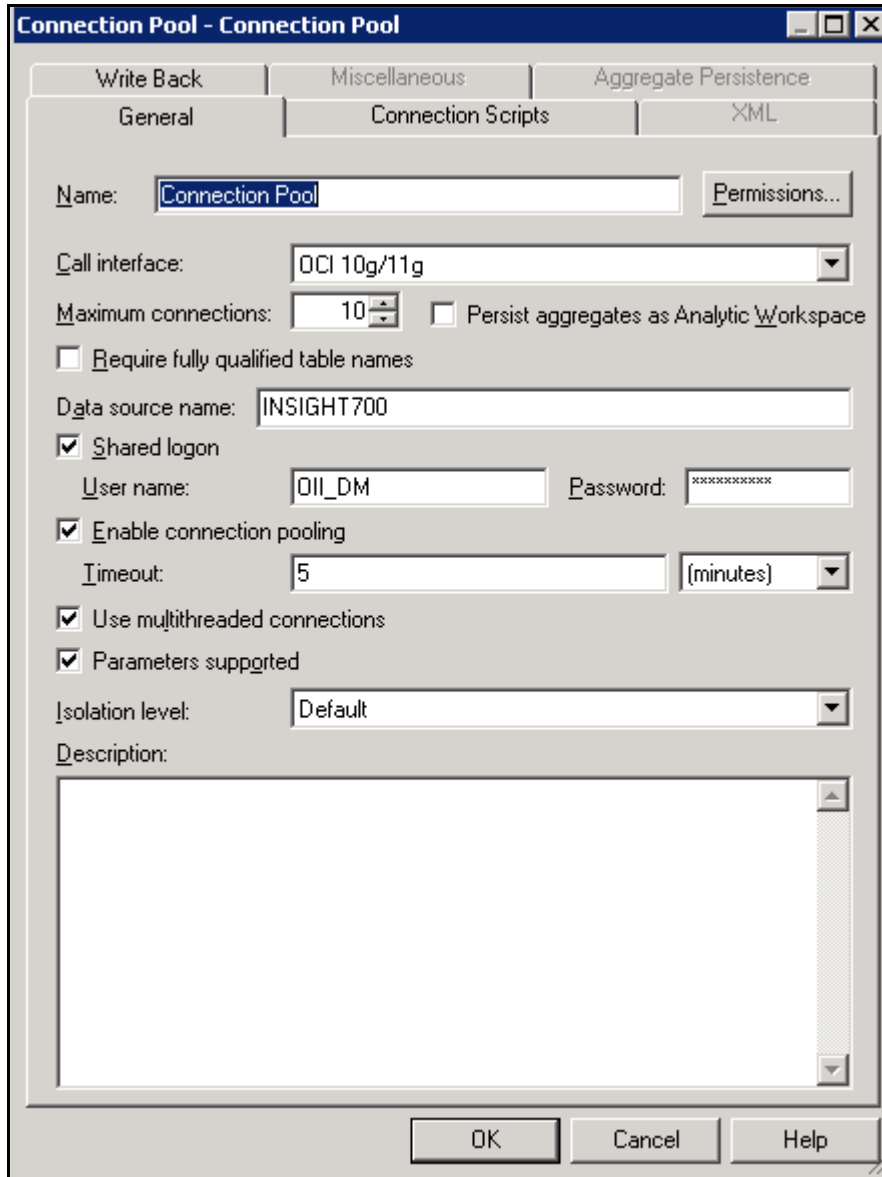


Figure 24: The Connection Pool Screen

6. Make sure that the **Require fully qualified table names** option is unchecked.
7. Under the “Shared Login” section, enter the same user name and password that you entered on the **Schema Configuration Parameters** screen for the Data Mart Schema (see the *Oracle Insurance Insight Installation Guide*).

The screenshot shows the 'Specify Insight Schema Configuration Parameters' dialog box for Oracle Insurance Insight 7.0.0. The 'Data Mart Schema' row is highlighted with a red rectangle, showing the username 'OII_DM' and a masked password. Other rows include Staging Schema (OII_ST), Warehouse Schema (OII_WH), System Configuration Schema (OII_SYS), Work Schema (OII_WRK), and Warehouse Palette Schema (OII_WP), each with its respective username and a masked password field.

Figure 25: Data Mart Schema User Name and Password on the Schema Configuration Parameters Screen

Note The default user name is OII_DM. If you entered a different user name on the Schema Configuration Parameters screen, use that name here.

8. Leave all of the other settings as is and click **OK** to close the screen.
9. To ensure the user name and password entered in step 7 above is correct:
 - a. Open any table under one of the Subject Areas.
 - b. Right click on the table you selected and choose ‘Update Row Count’. You will be prompted to check out the table’s objects-

c. Click 'Yes'.

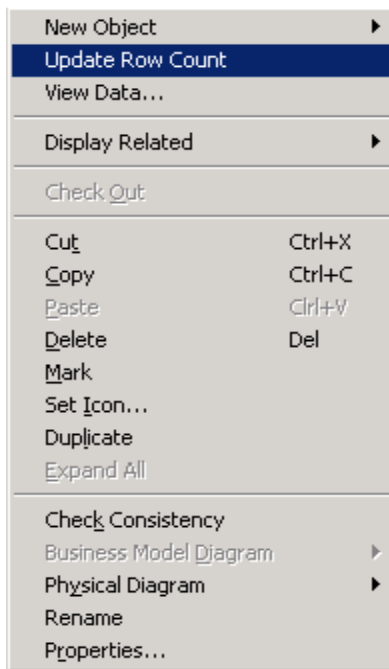


Figure 26: Update Row Count

d. If an error occurs it means that the user name/password is incorrect or the ODBC connection information supplied is incorrect.

10. Repeat steps 1-9 for each of the remaining Line of Business Subject Areas:

- Commercial Property
- Corporate
- Dwelling Fire
- Homeowners
- Personal Auto
- Personal Umbrella

Step 3: Configure the Insight700Config Subject Area

1. Click on the Connection Pool icon under the **Insight700Config** Subject Area to open the Connection Pool window.

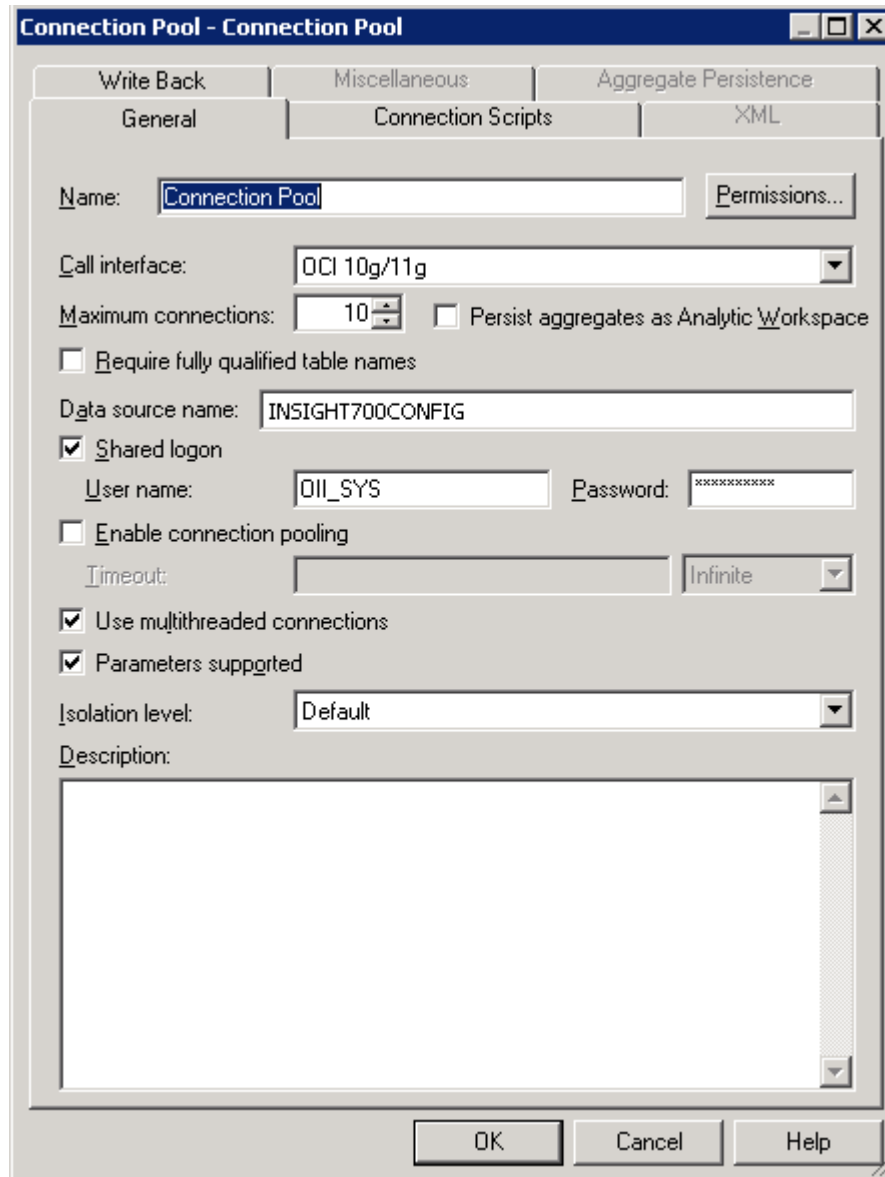


Figure 27: Insight700Config Connection Pool Window

2. Make sure the data source name is **Insight700Config**.
3. Make sure that the **Require fully qualified table names** option is unchecked.

- Under the “Shared Login” section, enter the same user name and password that you entered on the **Schema Configuration Parameters** screen for the System Configuration Schema (see the *Oracle Insurance Insight Installation Guide*).

Schema Name	User Name	Password
Staging Schema :	OIL_ST	*****
Warehouse Schema :	OIL_WH	*****
Data Mart Schema :	OIL_DM	*****
System Configuration Schema :	OIL_SYS	*****
Work Schema :	OIL_WRK	*****
Warehouse Palette Schema :	OIL_WP	*****

Figure 28: System Configuration Schema User Name and Password on the Schema Configuration Parameters Screen

Note The default user name is OIL_SYS. If you entered a different user name on the Schema Configuration Parameters screen, use that name here.

- Leave all of the other settings as is and click **OK** to close the screen.

Step 4: Check in the Changes to the Physical Layer

- Check in all the changes by selecting the “Check-in” button at the top tool bar.
- Click the **Save** button.
- Click the **Close** and **Exit** buttons to exit the OBIEE Administration Tool.

Step 5: Restart the Services

1. Launch the Enterprise Manager by entering the following URL in your browser:

http://<hostname>:7001/em

Note In the above URL, <hostname> can be the server name or IP address where you installed OBIEE.

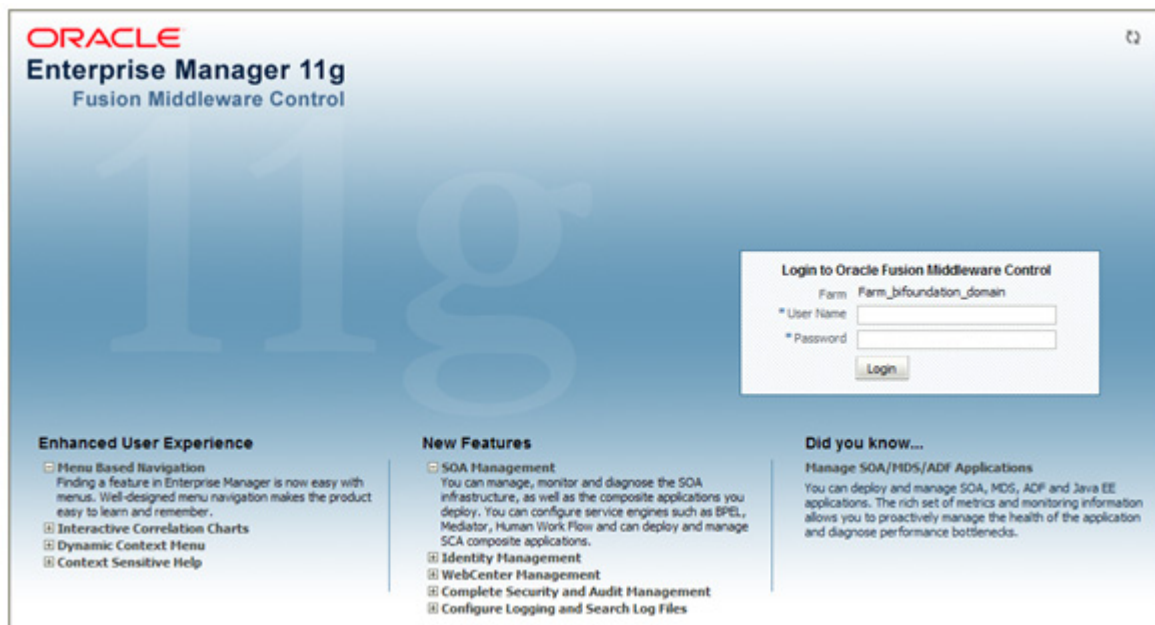


Figure 29: Enterprise Manager Login Screen

2. Log into Enterprise Manager using the administrator name and password that was specified during the OBIEE installation. The Oracle Fusion Middleware Control will open.

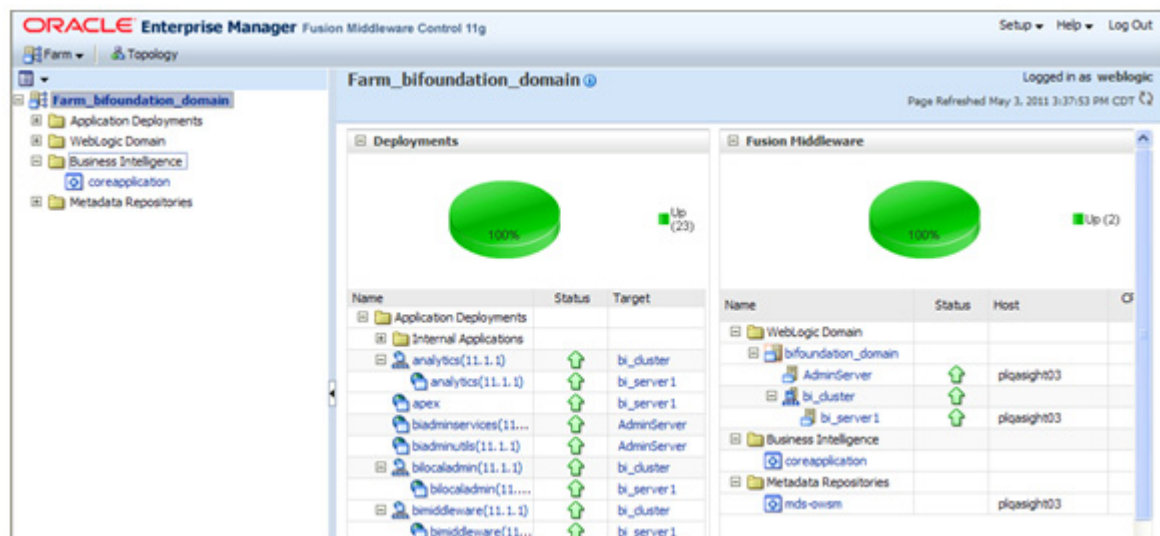


Figure 30: Oracle Fusion Middleware Control

3. From the navigation tree on the left, select **Farm_<domain_name>Business Intelligence>coreapplication**.
4. In the screen on the right, click on the **Capacity Management** tab and then the **Availability** tab.
5. Click on the **Restart All** button to restart the services. A dialog box will appear and ask you to confirm your action to restart the services.

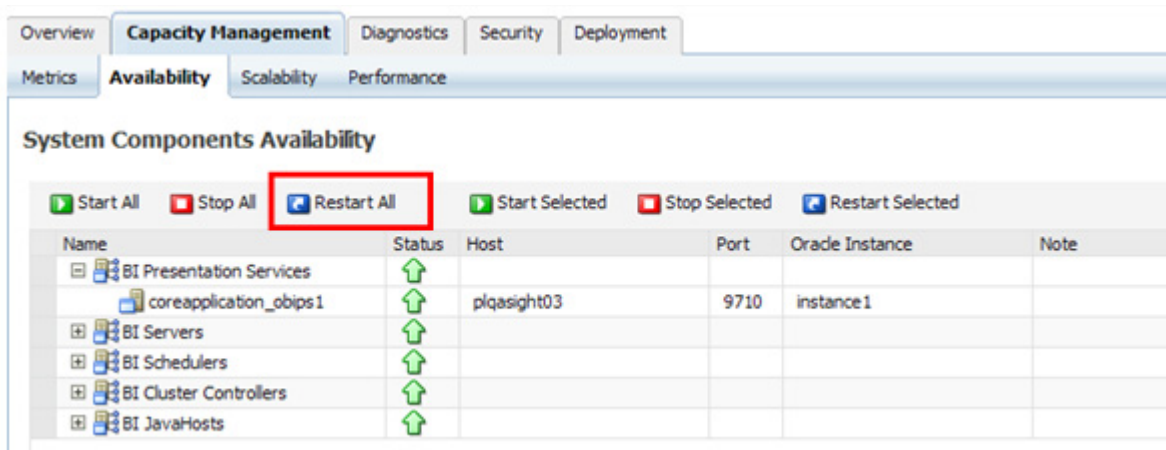


Figure 31: Select the "Restart All" Button

6. Click the **Yes** button in the dialog box to proceed with the restart. A message will appear to confirm that all services have restarted.

OBIEE REPOSITORY LINKAGE FOR NEW OR UPDATED LOB

Before you perform these steps it is assumed that the following tasks have been performed in the Warehouse Palette:

- a new LOB has been created or an existing LOB has been modified,
- all Risk Entities for the LOB have been defined,
- all Attributes for the Risk Entities have been defined, and
- the LOB has been successfully published.

For instructions on how to do this please refer to the *Oracle Insurance Insight Warehouse Palette User Guide*.

STEP 1: PERFORM AN IMPORT

1. Open the OBIEE Repository Administration Tool by selecting:
Start>All Programs>Oracle Business Intelligence>Administration
2. When the Administration Tool opens, select **File>Open>Online** to display the Online Repository dialog box.
3. In the login dialog box that appears, enter the repository password (the default password is *welcome1*) and click on **OK**.
4. Enter a valid user name and password. The default user name is *weblogic* and whatever password you selected when you installed OBIEE.

The following Repository screen will open:

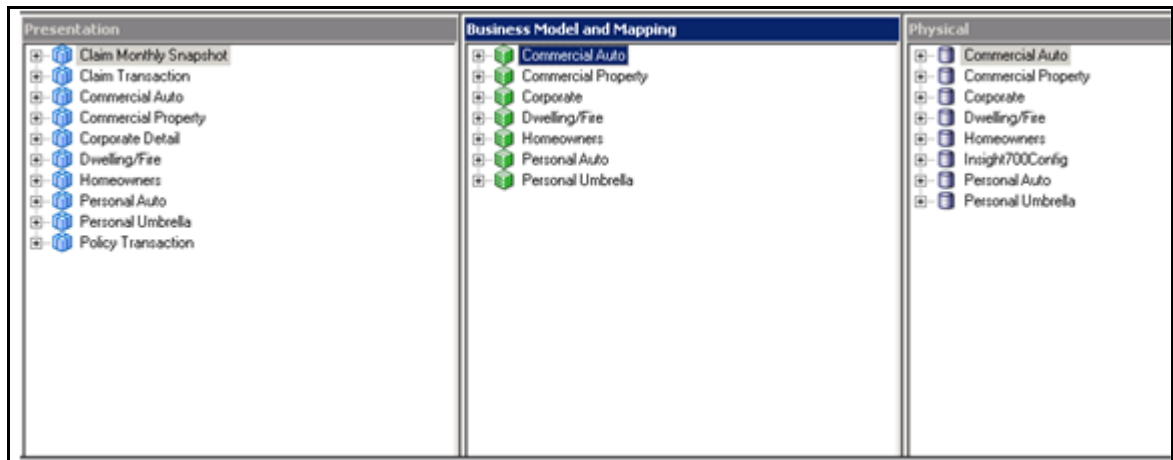


Figure 32: Layers of the OBIEE Repository for OII

5. Select **File>Import Metadata**. The Import Metadata Wizard will open.

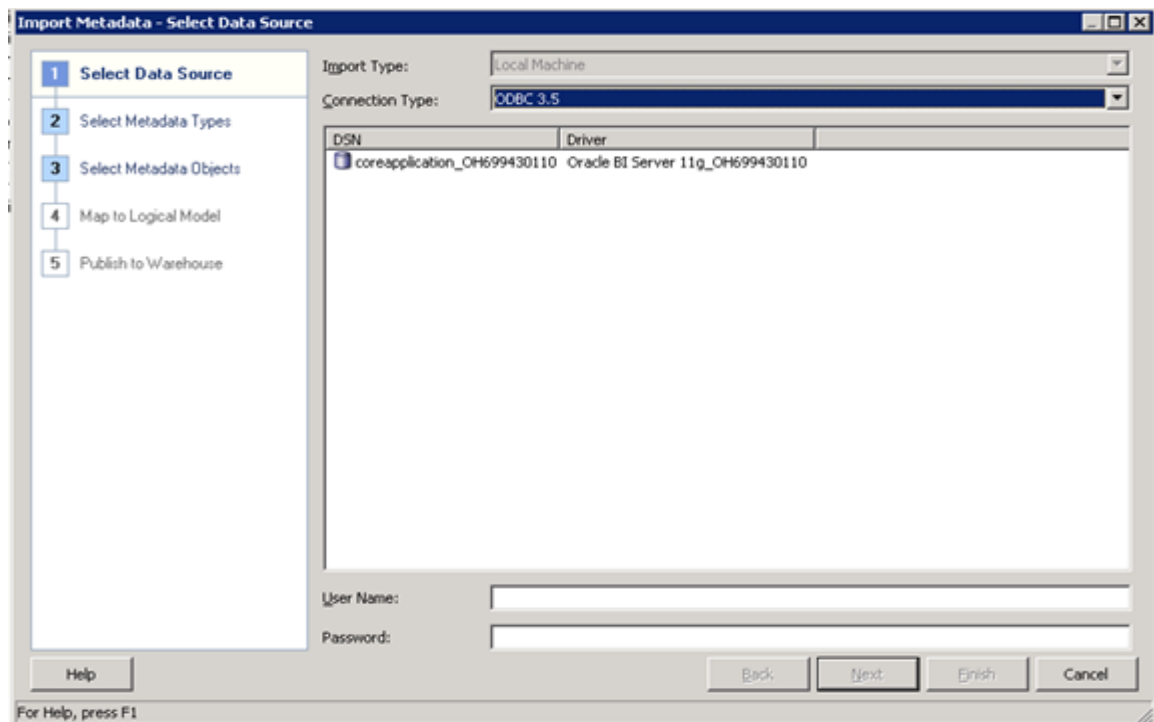


Figure 33: Import Metadata Wizard

6. From the **Connection Type** list select: **OCI 10g/11g**. The Data Source screen will change to the one below:

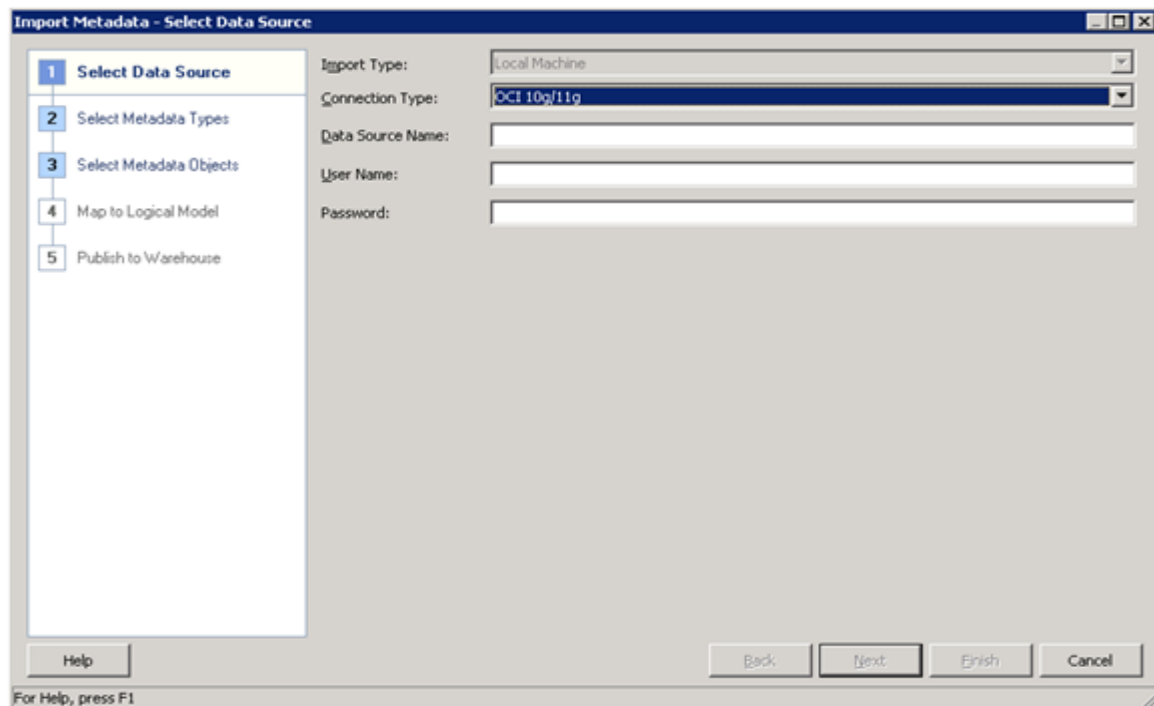


Figure 34: Data Source Screen with the “OCI 10g/11g” Connection Type Selected

7. At the remaining fields on this screen you will need to enter the name and user/password of the **Insight700** database connection that you created for the OII installation as described in the *Oracle Insurance Insight Installation Guide*.
 - **Data Source Name:** Enter **Insight700** as the data source name. This is the name of the database connection that you created for the OII installation as described in the *Oracle Insurance Insight Installation Guide*.
 - **User Name/Password:** Enter the same **user name** and **password** that you used when you created the **Insight700** database connection that you created for the OII installation as described in the *Oracle Insurance Insight Installation Guide*.
8. Click **Next**. The **Select Metadata Objects** screen appears.
9. To complete the remaining import steps refer to the *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition*.

STEP 2: REGENERATE THE METADATA DICTIONARY

1. Go to the directory: `<OII_ROOT>\install\obiee\metadata_dic`

For example:

C:\Oracle\Insight_Home\Insurance\oii\7.0.0\install\obiee\metadata_dic

2. In order to preserve the original “out-of-the-box” metadata dictionary, create a new folder into which you will place the regenerated metadata dictionary. In this example we will name this folder: **Insight_702_Custom**.

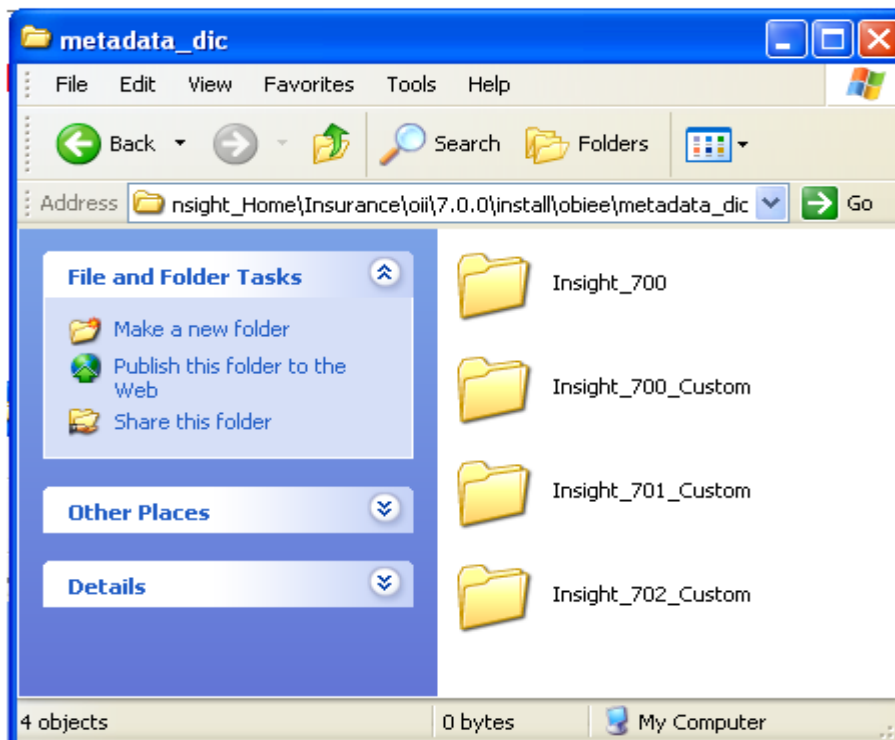


Figure 35: Create the `Insight_702_Custom` Folder

3. From the Oracle BI Administration Tool menu select **File>Open>Offline**. The **Open Dialog** box appears.

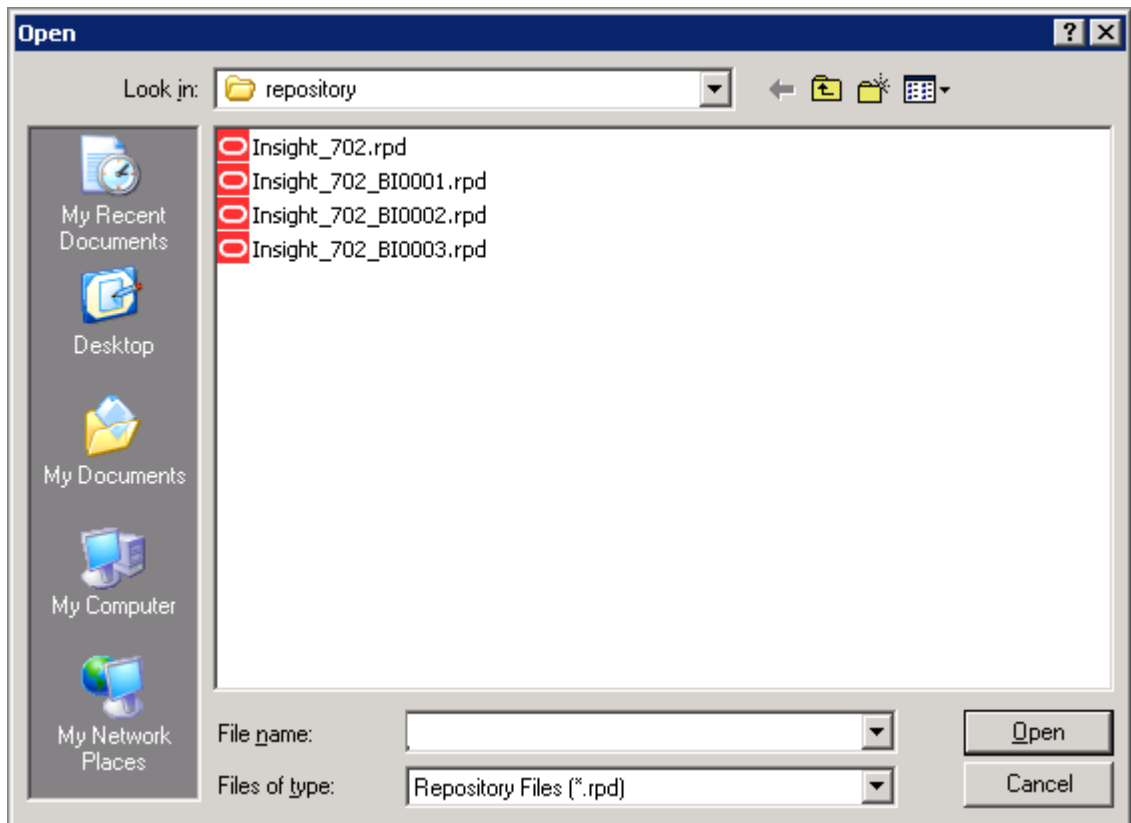


Figure 36: Select *Insight_702.rpd*

4. In the Open Dialog box select the **Insight_702.rpd** repository file and click **Open**.
5. An information dialog box will appear and states that the repository file can only be opened as Read-Only.
6. Click **Yes**. You will be prompted for a password (the default password is *welcome1*).

- Enter “welcome1” as the password and click **OK**. The Oracle BI Administration Tool screen will open.

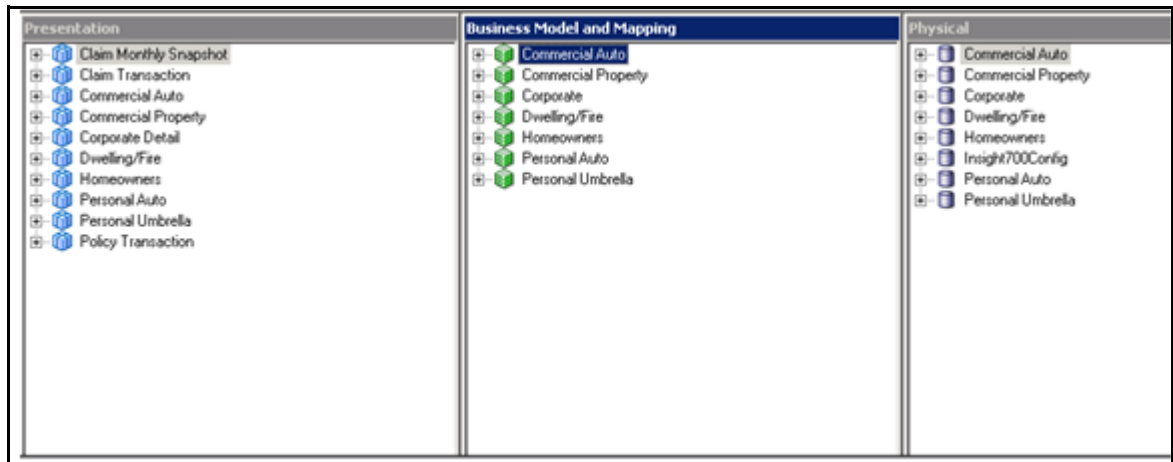


Figure 37: Layers of the OBIEE Repository for OII

- Select **Tools>Utilities**. The **Utilities** dialog box will open.

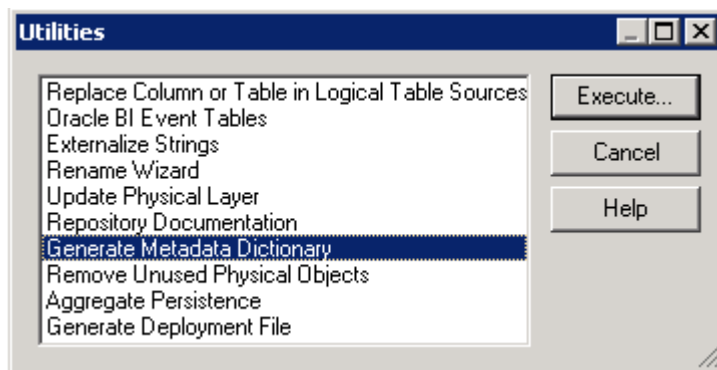


Figure 38: Select “Generate Metadata Dictionary”

- Select **Generate Metadata Dictionary** and click the **Execute** button. The **Choose Directory** dialog box will open.

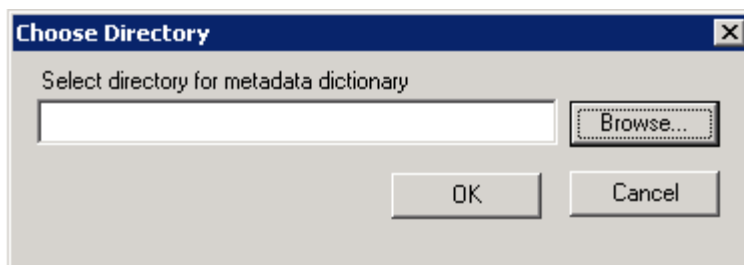


Figure 39: Choose Directory Dialog Box

- Use the **Browse** button to select the **Insight_702_Custom** directory that you created at the beginning of this section.

- Using the manual, *Oracle Fusion Middleware Metadata Repository Builder's Guide for Oracle Business Intelligence Enterprise Edition*, define the proper logical layer and presentation layer object for the newly added or modified LOBs.

STEP 3: DEPLOY THE METADATA DICTIONARY

- Go to the directory where you created the **Insight_702_Custom** folder in *Step 2: Regenerate the Metadata Dictionary* on page 73:

:<OII_ROOT>\install\obiee\metadata_dic\Insight_702_Custom

For example:

C:\Oracle\Insight_Home\Insurance\oii\7.0.0\install\obiee\metadata_dic\
Insight_702_Custom

- Copy the entire **Insight_702_Custom** folder to:

<MW_HOME>\analyticsRes

For example:

C:\Oracle\Middleware\analyticsRes

The folder, **Insight_702_Custom**, now sits at the base of the **analyticsRes** folder. For example:

C:\Oracle\Middleware\analyticsRes\Insight_702_Custom

Important Once the copy is complete, go into the **Insight_702_Custom** folder and make sure you have at least 36,000 files and over 1600 folders in total for this folder. If not extracted properly, some files may become lost during the transfer due to the long path/file name.

- .Rename the **Insight_702_Custom** folder to match the name of the repository file you initially deployed as described in the *Oracle Insurance Insight Installation Guide 7.0.2* (e.g., **Insight_702_BI0003**).

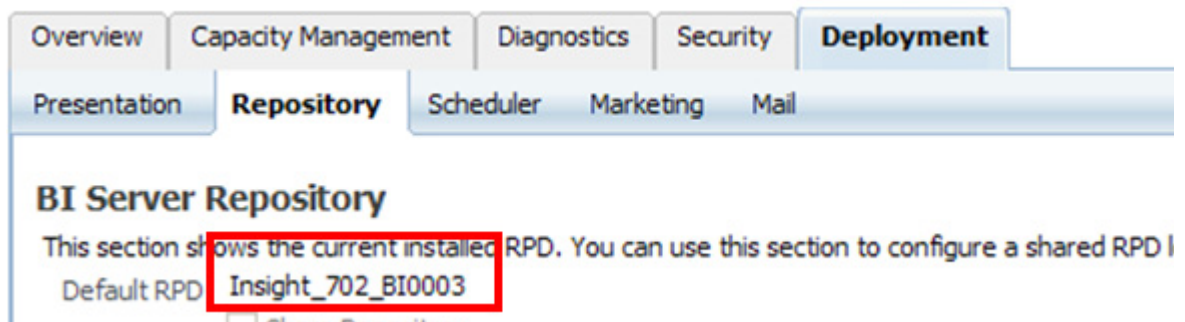


Figure 40: Rename the *Insight_702_Custom* Folder to Match the Deployed Repository File

For example:

C:\Oracle\Middleware\analyticsRes\Insight_702_BI0003

4. Login back into Enterprise Manager.
5. From the navigation tree on the left, expand **Farm_<domain_name>Business Intelligence>coreapplication**.
6. Select the **Capacity Management** tab in the screen on the right.
7. Select the **Restart All** button.

STEP 4: PERFORM OBIEE REPOSITORY LINKAGE STEPS

Return to the section, *OBIEE Repository Linkage* on page 62, and complete the steps in that section.

Balancing Reports

OVERVIEW

Balancing Reports are one method to verify data flow between schemas in the Oracle Insurance Insight (OII) system. A typical OII implementation contains three logical schemas: *Stage*, *Warehouse*, and *Data Mart*. These schemas represent the stages of customer data as it is processed in OII:

- *Stage* – data in this schema is directly imported or fed by external systems. The data has not been manipulated to fit into any particular data model other than what the external systems have defined. This is the input to the following schema.
- *Warehouse* – data in this schema has been processed through the first level of manipulation in OII. Transactions will be categorized and parsed into the appropriate model tables, in preparation for migration to the following schema.
- *Data Mart* – data in this schema has been aggregated into the appropriate model for the customer’s lines of business. This is the end point in the process of the dataflow.

Since data is manipulated between the Stage-to-Warehouse and Warehouse-to-Data Mart flows, it is important to validate that the transformation and migration logic for those flows is correct. The Balancing Reports provide this capability in a convenient, easy-to-use format that provides reuse, portability and flexibility.

COMPONENTS

OII is an Insurance-specific add-on to Oracle Business Intelligence Enterprise Edition (OBIEE). OBIEE includes an additional component called Oracle Business Intelligence Publisher (BI Publisher) that is used for generating reports, and as you might guess, the Balancing Reports are authored using BI Publisher.

Additionally, the Balancing Reports rely on some additional data aggregation that occurs by way of *views* built into the OII database structures. These views allow a consolidated picture of the data in each schema that requires less complexity in the reports used in BI Publisher.

DEPLOYING THE BALANCING REPORTS

Balancing Reports are a combination of *templates*, saved as RTF documents, and BI Publisher XDO files, which contain the data mappings, parameters and all the other information that goes into the generation of a report.

Follow the instructions in this section to setup and deploy the Balancing Reports.

PREREQUISITES

Prior to installing the Balancing Reports, make sure that the following prerequisites have been met:

- BI Publisher is installed and running on your server.
- The OII Reports catalog has been deployed along with the rest of the OII catalog files. All OII catalogs should have been deployed when you configured OBIEE as part of the OII post-installation process. Refer to the *Oracle Insurance Insight Installation Guide* for further information.
- All Balancing Report related views have been created on OII_SYS (the system configuration schema you specified when you installed OII) schema. Go to the OII_SYS schema and verify that the following views are present: CLAIMS_DM, CLAIMS_ST, CLAIMS_WH, EP_ST, EP_WH, POLICY_DM, POLICY_ST, POLICY_WH, XP_ST, XP_WH.

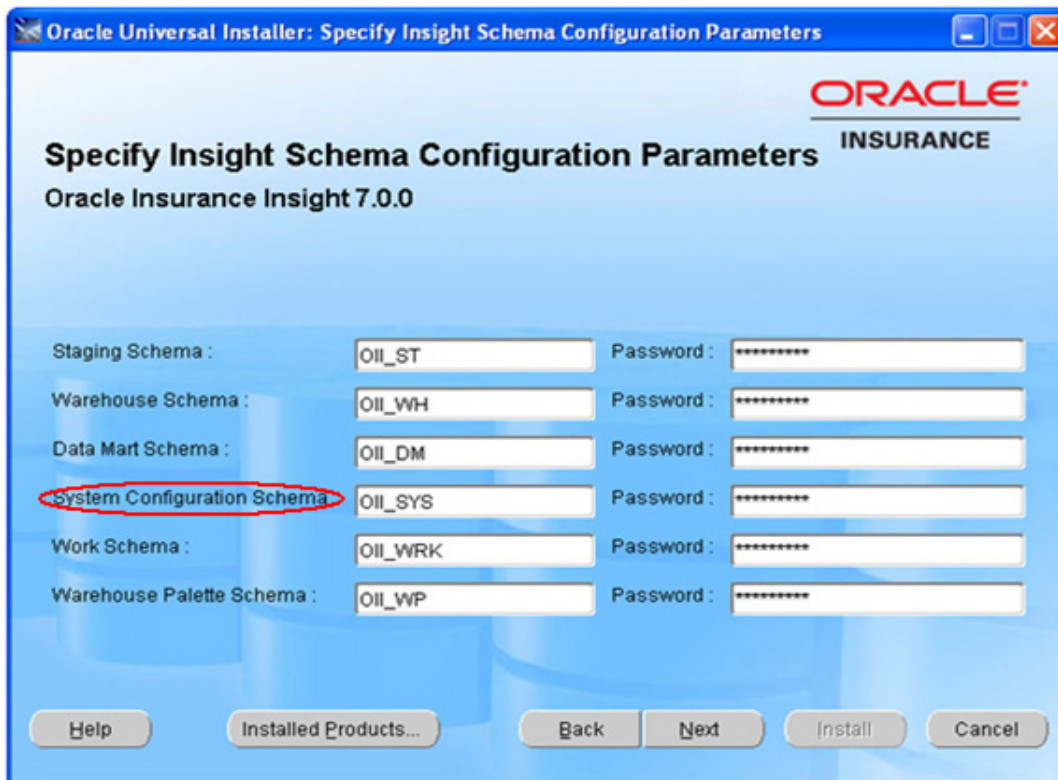


Figure 41: OII_SYS - System Configuration Schema

STEP 1: CREATE A USER FOR BI PUBLISHER

This step requires you to create a BI Publisher user with Administrator privileges.

1. Open the WebLogic Administration Console by entering the administrator user name and password for the server domain that was specified during the OBIEE installation process.

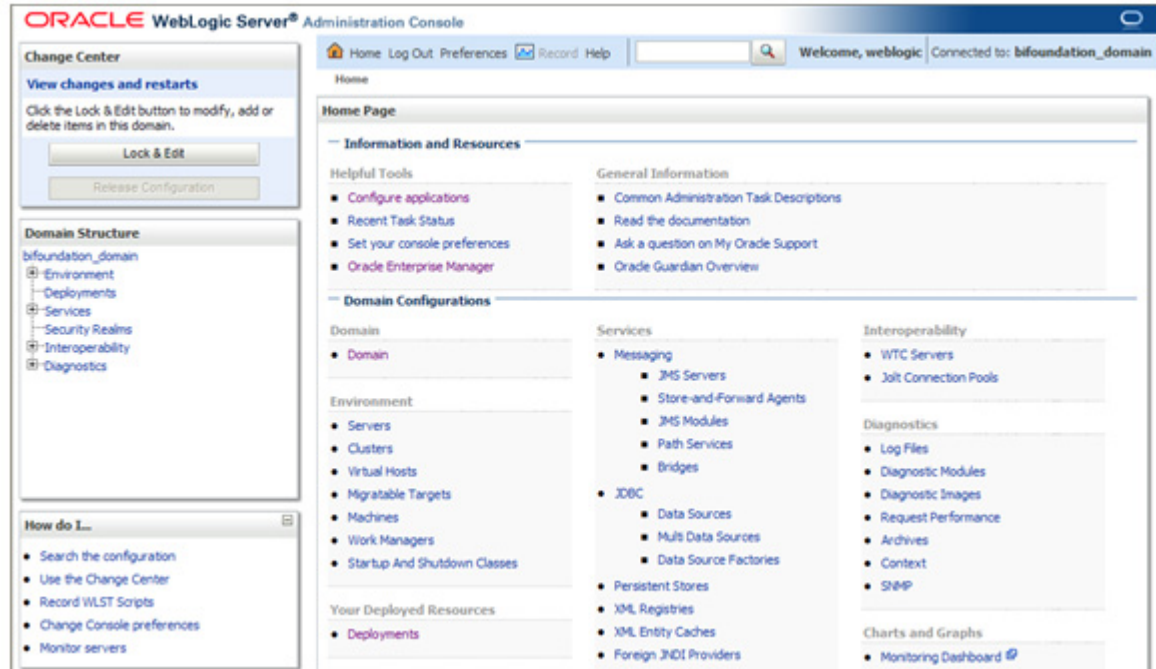


Figure 42: WebLogic Server Administration Console

2. Select **Security Realms** in the left pane. The **Summary of Security Realms** page will appear on the right.

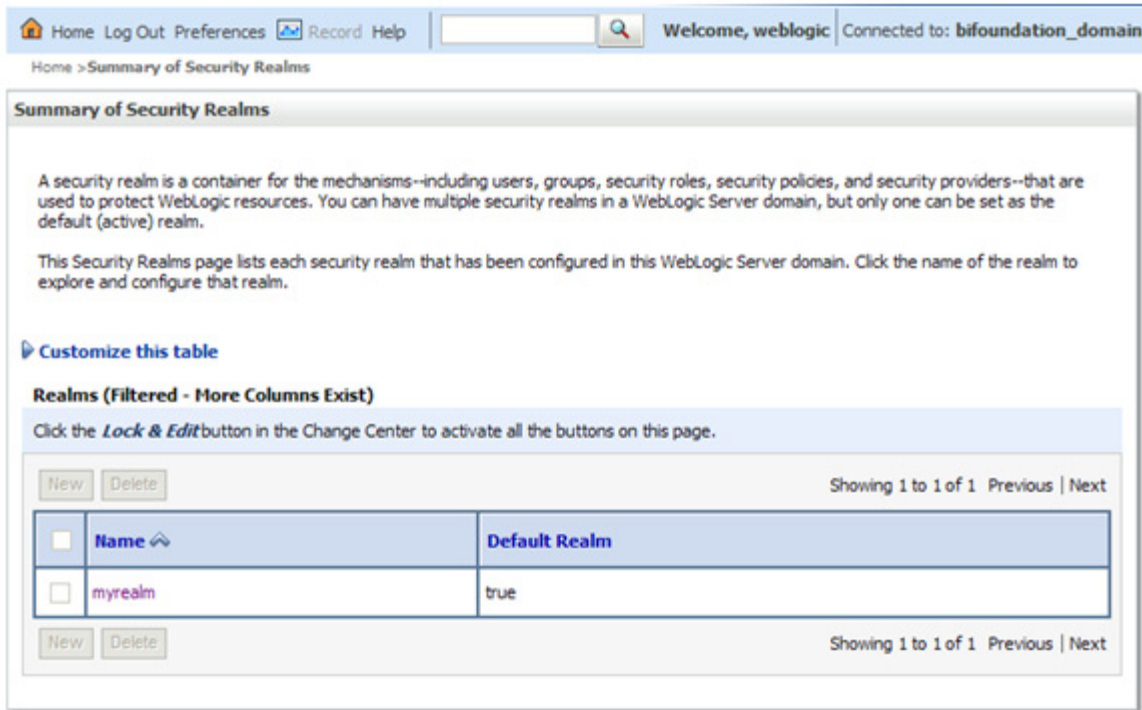


Figure 43: Summary of Security Realms Page

3. Click on **myrealm**. The **Settings for myrealm** page will open.

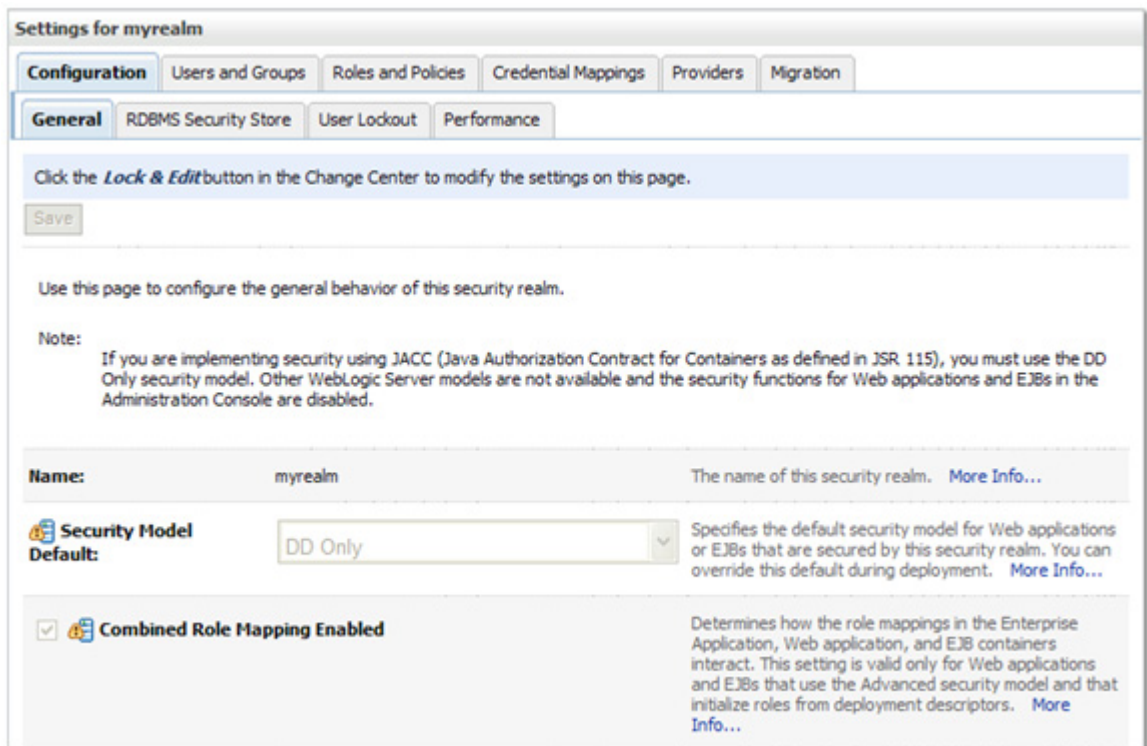


Figure 44: Settings for myrealm Page

- Click on the **Users and Groups** tab at the top of the page, and then click on the **Users** tab to display a list of configured users in **myrealm**.

Users

Showing 1 to 10 of 10 Previous | Next

<input type="checkbox"/>	Name ↕	Description	Provider
<input type="checkbox"/>	adminlistener	APEX Listener administrator user	DefaultAuthenticator
<input type="checkbox"/>	BISystemUser	BI System User	DefaultAuthenticator
<input type="checkbox"/>	managerlistener	APEX Listener manager user	DefaultAuthenticator
<input type="checkbox"/>	OracleSystemUser	Oracle application software system user.	DefaultAuthenticator

Figure 45: List of Users

- Click on the **New** button at the top of this table. The **Create a New User Page** opens.

Create a New User

User Properties

The following properties will be used to identify your new User.
* Indicates required fields

What would you like to name your new User?

* **Name:**

How would you like to describe the new User?

Description:

Please choose a provider for the user.

Provider:

The password is associated with the login name for the new User.

* **Password:**

* **Confirm Password:**

Figure 46: Create a New User Page

- Enter the following information on this page:
 - Name** - Enter a name for the user. For example, **bipadmin**.
 - (Optional) Description** - Enter a description
 - Provider** - Accept the default provider: **DefaultAuthenticator**.

- **Password** - Enter a password that is at least 8 characters long.
 - **Confirm Password** - Re-enter the password.
7. Click **OK**. You will be returned to the **Users** tab where the user you just created will appear in the list of users.
 8. If you are not already there, return to the **Users** page to display the table listing the configured users.
 9. Click on the name of the user you just created, **bipadmin**. A settings page for the selected user will open.

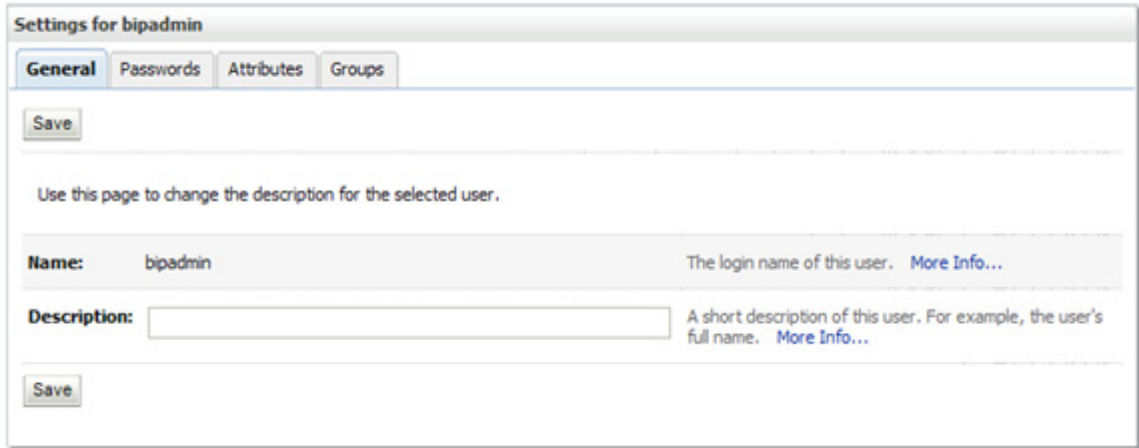


Figure 47: Settings Page for the Selected User

10. Select the **Groups** tab. The **Groups** page opens:

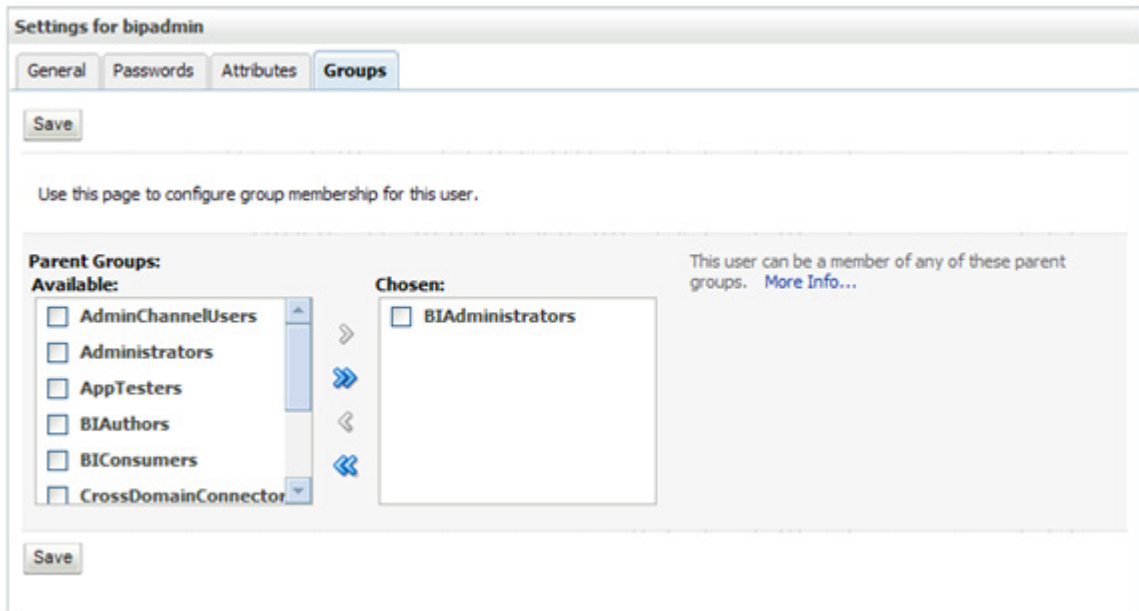


Figure 48: Groups Page

11. Add the **bipadmin** user to the **BIAdministrators** group by selecting **BIAdministrators** in the **Available** list box and using the arrow button to move it to the **Chosen** list on the right.
12. Click the **Save** button.
13. Exit the WebLogic Administration Console.

STEP 2: CREATE A JDBC DATA SOURCE

Now that you have created a user you must next create a JDBC data source within BI Publisher.

1. Open a browser and navigate to your BI Publisher installation. The URL for the installation should be of the form:

http://<hostname>:<port>/xmlpserver/

Where:

- **hostname** - The hostname is the IP address or machine name of the server where BI Publisher is installed.
 - **port** - The *port* setting will vary depending on the installation method used for BI Publisher. If you installed a standalone BI Publisher setup, the port is usually 9704. Verify with the console window where BI Publisher is running. If you installed BI Publisher under WebLogic, the port is usually 7001. Again, verify with the console window where WebLogic is running.
2. You will be presented with a login screen.



Figure 49: BI Publisher Login Screen

3. Login as the user that you created in the previous section (e.g., **bipadmin**). The BI Publisher home page will appear.

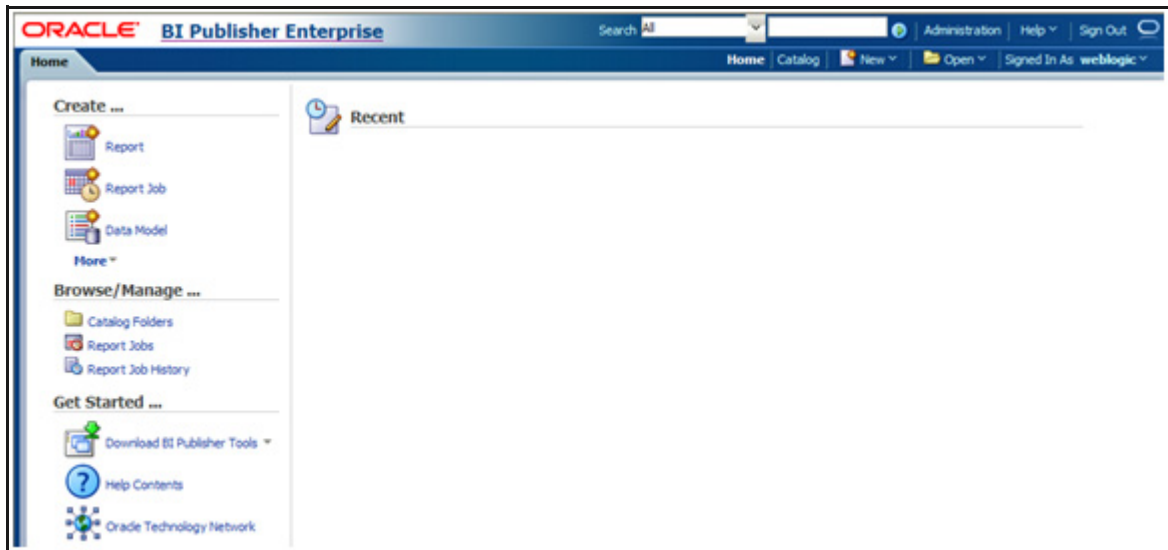


Figure 50: BI Publisher Home Page

4. Click on the **Administration** link on the top right corner. The screen below opens:

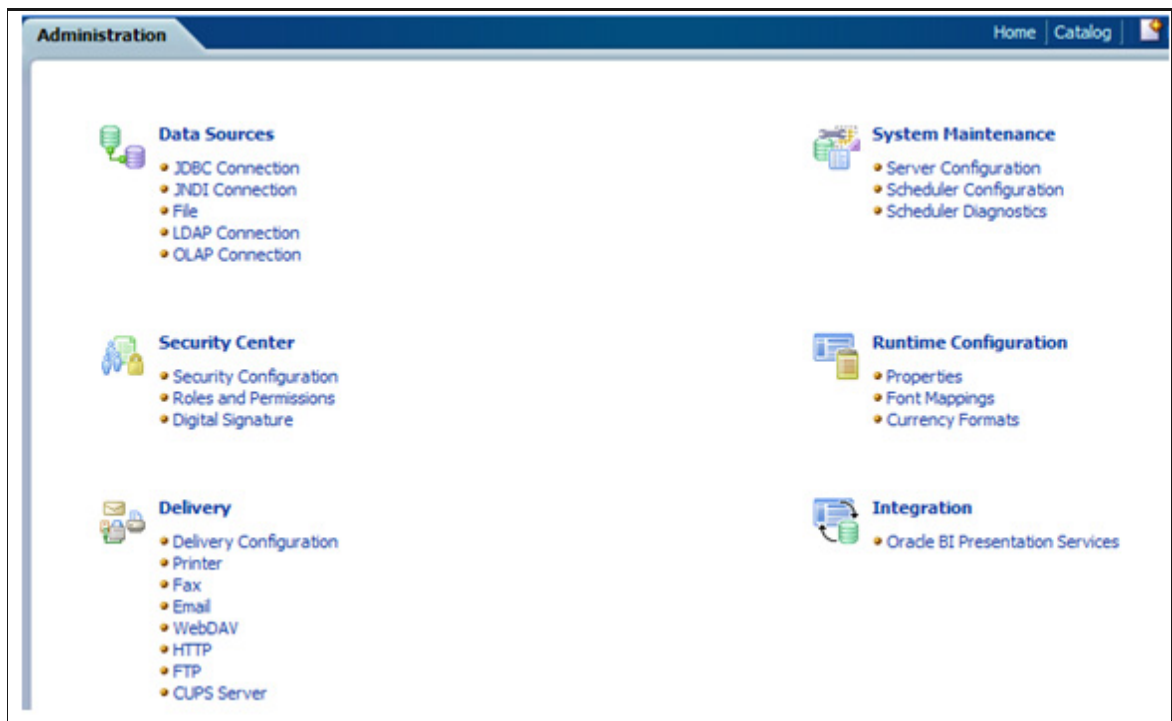


Figure 51: BI Publisher Administrative Screen

- Click on **JDBC Connection** under the **Data Sources** section. The following screen opens.

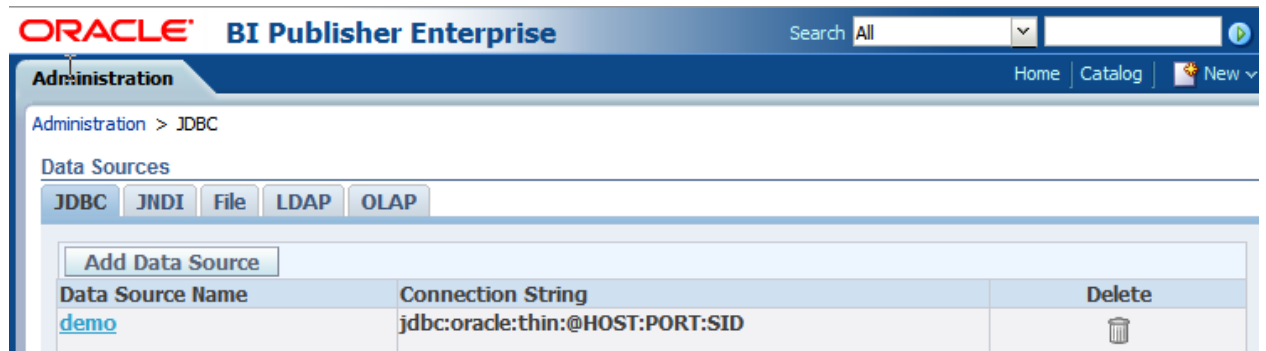


Figure 52: JDBC Data Source Screen

- Click on **Add Data Source**. The following screen opens.

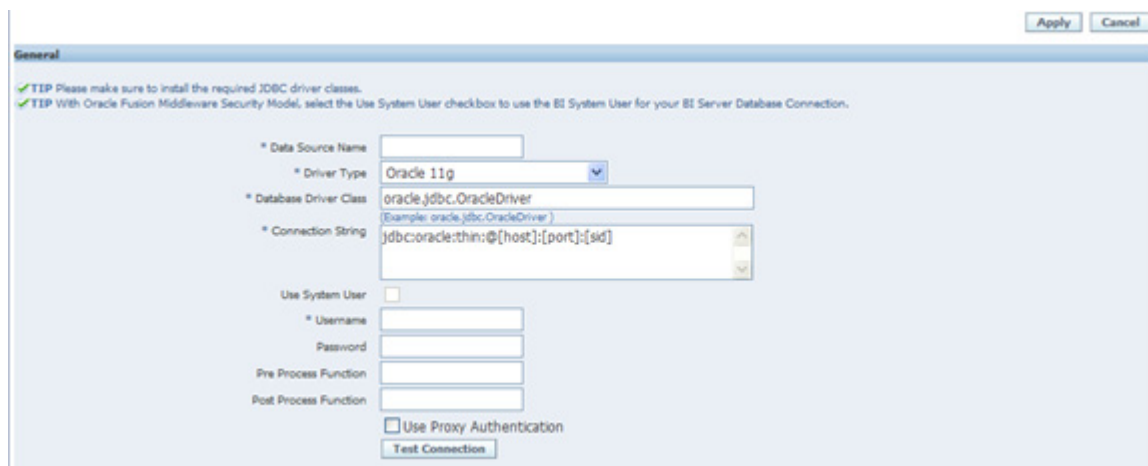


Figure 53: Add Data Source Screen

- Enter the following values:
 - Data Source Name** - *Insight7*
 - Username** - *OII_SYS*
- Fill in the appropriate connection information as it applies to your system.
- Click on the **Test Connection** button to confirm that the connection is successful.
- Click on the **Apply** button to save the settings.

11. Click on the **Catalog** link in the menu at the top of the screen. The following screen opens.

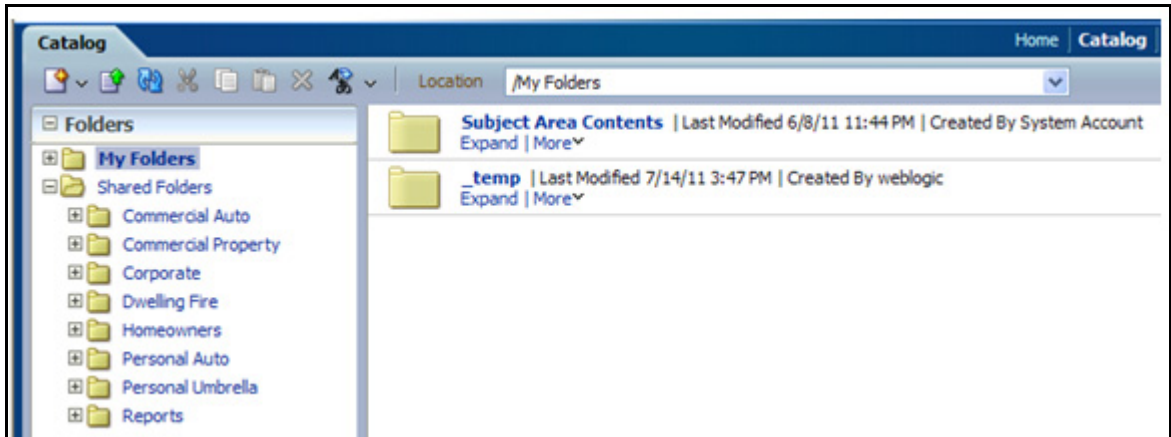


Figure 54: Catalog Screen

12. Under the **Folders** tree, select **Shared Folders>Reports>Balancing Reports**. The Balancing Reports will appear in the pane on the right once the data is loaded.

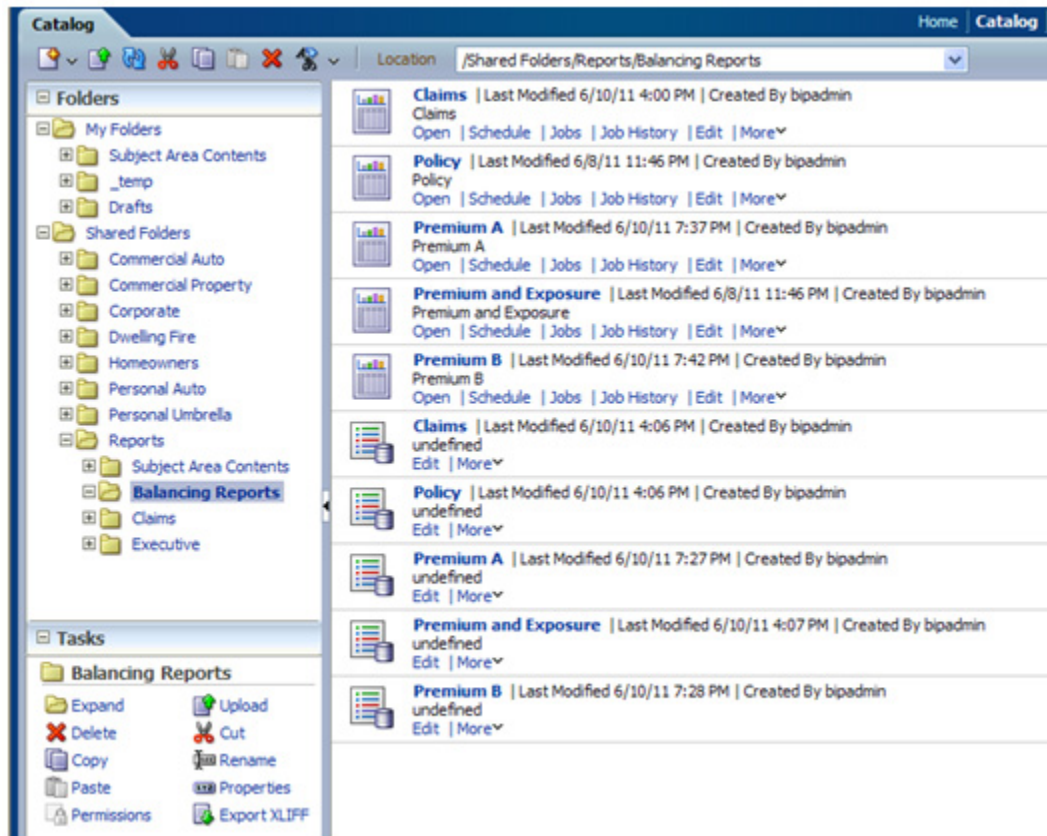


Figure 55: Balancing Reports

BALANCING REPORT DESCRIPTIONS

OII provides the following five Balancing Reports,

CLAIMS

For each Line of Business, Claim Year, and Schema (e.g. Staging, Warehouse, and Data Mart), the number of Claim transactions are given as well as the sum of the corresponding Claim Transaction Amounts. Filters may be applied on the Claim Transaction Code, Transaction Year, and Line of Business.

POLICY

For each Line of Business, Transaction Year, and Schema (e.g. Staging, Warehouse, and Data Mart), the number of Policy transactions are given as well as the sum of the corresponding Policy Transaction Amounts. Filters may be applied on the Policy Transaction Code, Transaction Year, and Line of Business.

PREMIUM A

The Premium A report provides the total Written, Earned, and Unearned Premium amounts among the Staging, Warehouse, and Data Mart layers. Filters may be applied on the Year and Month.

PREMIUM B

The Premium B report provides the total Written, Earned, and Unearned Premium amounts between the Policy Transactions table of the Warehouse and the Policy Transaction and Policy Monthly Snapshot fact tables of the Data Mart. Filters may be applied on the Year and Month.

PREMIUM AND EXPOSURE

The Premium and Exposure report provides the total Earned and Unearned Premium amounts as well as the total Earned and Unearned Exposure amounts among the Staging and Warehouse layers by Line of Business, Year, and Month. Filters may be applied on the Line of Business, Year, and Month.

REPORT COMPONENTS

The following sections will familiarize you with some of the more common aspects of reports within BI Publisher. For a detailed description of the BI Publisher report capabilities, refer to the following manuals:

- *Oracle Fusion Middleware User’s Guide for Oracle Business Intelligence Publisher*
- *Oracle Fusion Middleware Report Designer’s Guide for Oracle Business Intelligence Publisher*
- *Oracle Fusion Middleware Data Modeling Guide for Oracle Business Intelligence Publisher*

The complete BI Publisher documentation set can be found at the following link:

<http://www.oracle.com/technetwork/middleware/bi-publisher/documentation/index.html>

A report is made from the following components:

- **Data Model** – the source(s) of data used as input to the report. Various data model types are available, but the Balancing Reports use only SQL statements to query the views and tables in the OII model.
- **List of Values** – a List of Values (LoV) can be used as a source for a parameter list. LoVs can come from a SQL query or a fixed list. The Balancing Reports use both.
- **Parameters** – user-selectable values that control the data displayed in the report.
- **Layouts** – Templates built around RTF documents which integrate data from the Data Model using Parameters as filtering criteria. A report can have many layouts from which the user can choose.

To view the components:

Each Balancing Report in the right pane has a corresponding Data Model. The components for a Balancing Report can be viewed by selecting the Data Model for a report.

1. Login to BI Publisher and navigate to **Shared Folders>Reports>Balancing Reports**.
2. Click the “Open” link for a Balancing Reports Data Model (e.g., ‘Claims’).



Figure 56: Click on Edit Link to Display the Report’s Components

You will see a display of all the components in the report.

The left pane displays a tree view of the report components. Clicking a component will display the properties there of in the right pane and can also enable/disable the New, Delete and Ordering Arrows displayed under the Report tab. In most cases you should not need to modify any of the components unless you need to create a new report or layout.

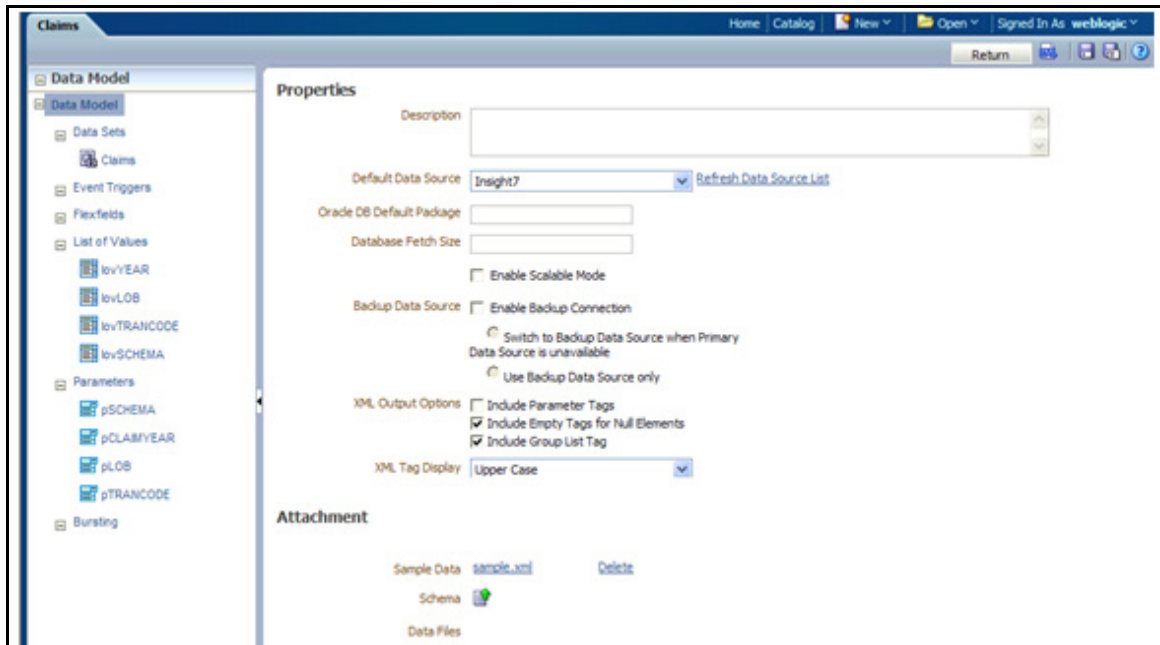


Figure 57: Report Components

DATA MODEL

Most reports feature a single data model, which contains a detail SQL query to pull the data for the report. Do not modify the data model unless you have reason to do so. You can also specify the data source (defined in the installation steps) for the Data Model.

The output from the data model is XML data that is used as input to the report engine.

LIST OF VALUES

As mentioned previously, Lists of Values (LoVs) are used as sources for parameters applied to reports. The Balancing Reports each define similar sets of LoVs depending on the data model and required output in the report. An example is shown below:

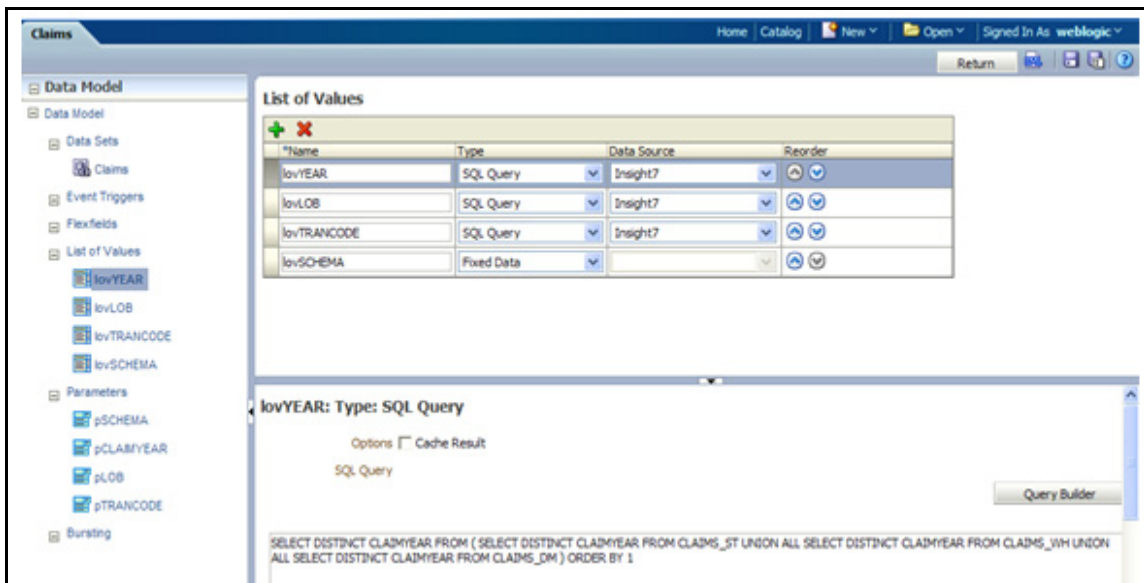


Figure 58: List of Values

The source for the “lovYEAR” is an SQL statement. A LoV may also contain a list of hard-coded values.

PARAMETERS

Parameters are used as filtering criteria for a report, and are referenced in the SQL queries in the report's Data Model. Parameters have several different types, however the Balancing Reports only use the "Menu" type, which means that the parameters are displayed as Drop-down selectors with data from LoVs. The parameters are defined to allow the user to select multiple or all values, as shown below:

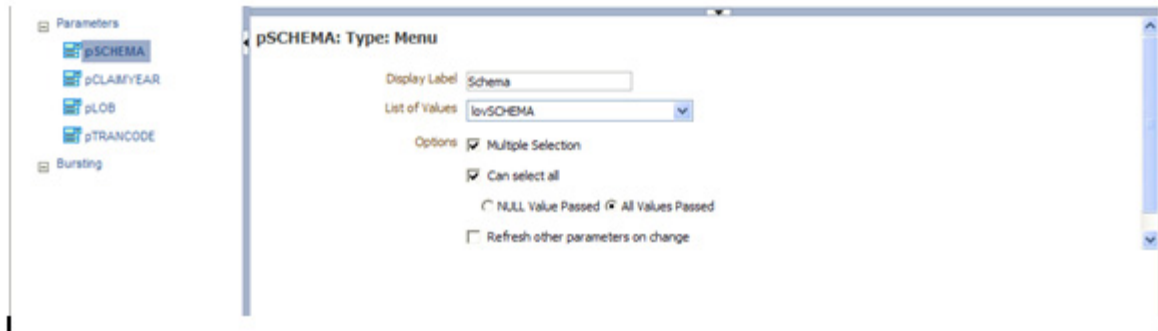


Figure 59: Parameters

TEMPLATES

A Template is a document which defines the fields, grouping, aggregation and display of data in a report. BI Publisher has many available template types; however Balancing Reports are all created as RTF templates. You may use Template Builder for Word-in to author templates.

LAYOUTS

A Layout is a correlation between a template and a set of allowed output types. The Balancing Reports allow all output types. You can view available templates by clicking on the **Edit** link for a Balancing Report.

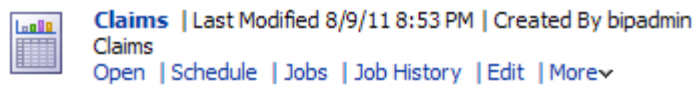


Figure 60: Select the Edit Link to View Templates

A list of available Layout templates will appear.

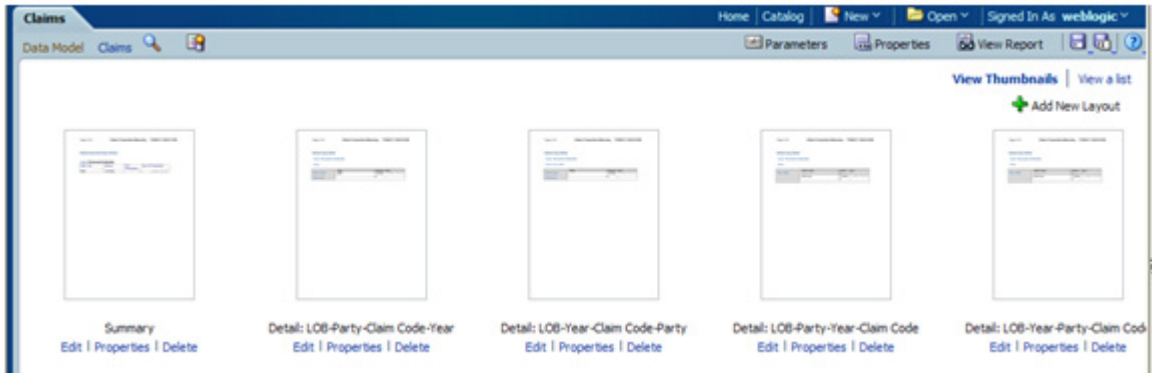


Figure 61: List of Layouts

You can download a layout by clicking on the **Edit** link and saving it to your machine.

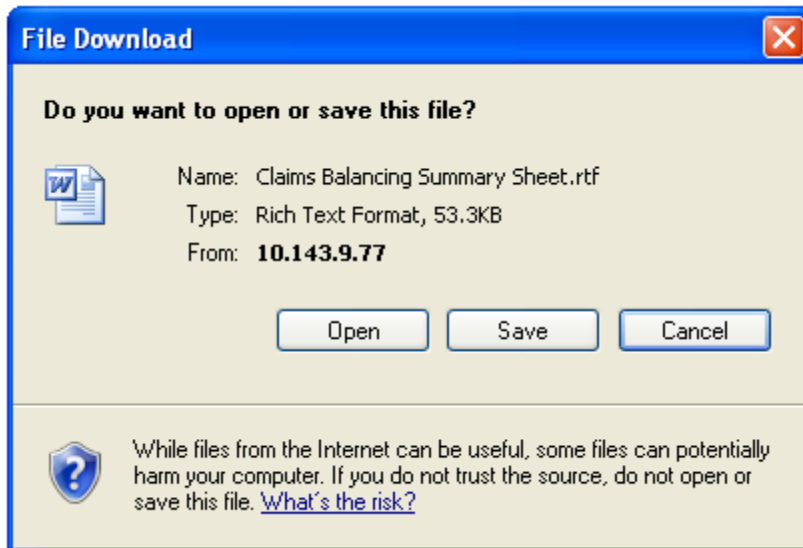


Figure 62: Download a Template

If you need to make changes to a template, simply download the RTF template, make edits, and then upload again, at this screen, using the same file name. You will be prompted to overwrite the existing template. You can also download sample data that is provided with the reports. You can use this sample data with the BI Publisher Add-in so reports can be modeled off-line from the BI Publisher server and data sources.

VIEWING REPORTS

To view a report:

1. Under the **Folders** tree, select **Shared Folders>Reports>Balancing Reports**
2. Click on the report name (e.g. “Claims”) to view the report.

The reports are configured with various drop-down selectors that allow customization of the parameters to the report. These selectors enable reporting on different slices of data that correspond to key data points in a typical business application and lifecycle:

- The default layout for the Report is displayed automatically when the report is initially displayed.
- You can change the input parameters to the report and click the **Refresh** button to update the display.
- If you want to include multiple selections, hold down the CTRL key while you click your selections in the dropdown.
- You can select “All” at the top of the dropdown lists, to include all values for that selector.

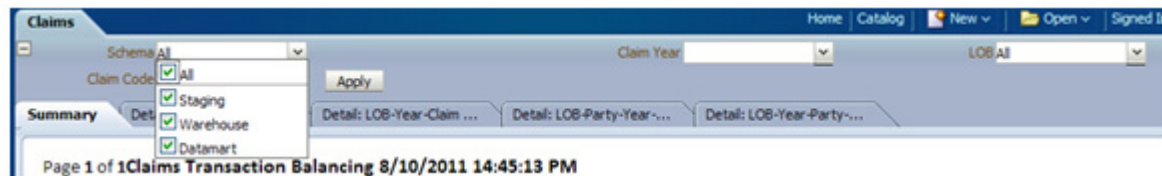


Figure 63: Viewing a Report

The tabs across the top of the screen allow you to change the Template to display data in according to the different templates that are assigned to that report. The Balancing Reports typically offer a Summary and Detail template, which presents two ways to view balancing data.

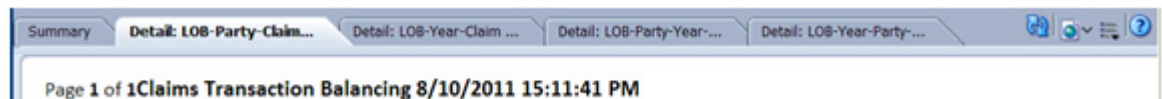


Figure 64: Balancing Detail

Some of the Reports offer multiple Templates to change the aggregation points of the data, which presents the report in a slightly different format depending on your requirements and balancing techniques.

You can also change the output format using the appropriate dropdown. By default, the output is displayed as an HTML version of the template. If you want to save a report, you can click the **View Report** button to download it to your local machine.

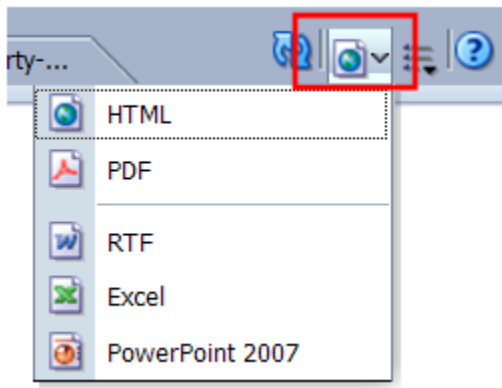


Figure 65: Selecting a New Output Format

MODIFYING REPORTS

Modifying the reports may become necessary over time if OII data models change, or as reporting requirements evolve. Since reports are built from several different components, modifying a report depends on what you need to accomplish. Data for reports is provided by way of SQL Views built into the database. The views are queried by *Data Models* in BI Publisher, using *Parameters*, which are sourced from *Lists of Values* as filters. Therefore, you must determine at what level the change(s) must be made. The simplest changes are to the templates themselves.

CHANGING A TEMPLATE

In order to modify templates, you must have the Template Builder for Word installed. You can download the Template Builder for Word while in BI Publisher from the link under the Get Started section on the BI Publisher Home Page.

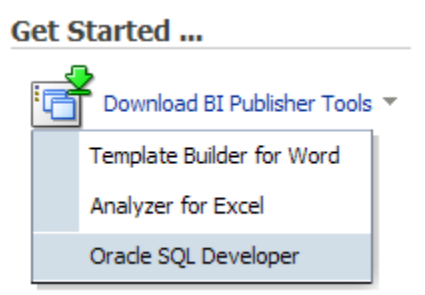


Figure 66: Selecting Template Builder

- Use the Template Builder for Word to access BI Publisher server – this option allows you to interact with the BI Publisher server online using live data.

- Download templates from BI Publisher manually – this option allows you to download templates and data for offline modification. You will need to upload the template back to BI Publisher once you have completed your changes.

Online Template Modification

1. Start Microsoft Word, and open a new blank document.
2. On the Add-Ins tab, select **Oracle BI Publisher**→**Log On**.
3. Enter your credentials and ensure the URL to the report server is correct, then click the **Login** button.
4. Once you have logged on, navigate to the appropriate folder containing your report.
5. Click the folder on the left tree, and the available reports for the folder will be displayed in the top right pane.

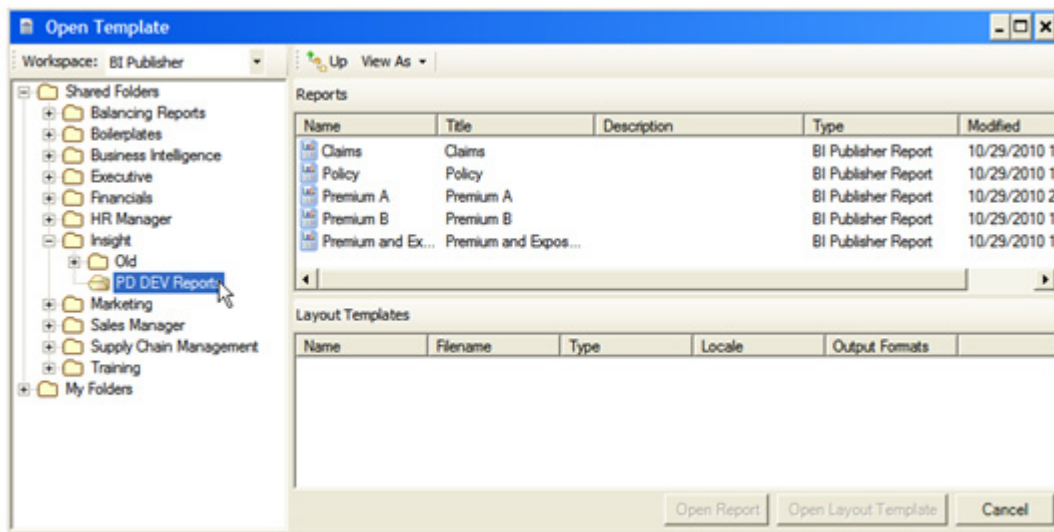


Figure 67: Available Reports

6. Once you click an available report, the layout templates for that report will be displayed in the lower right pane. From here, you can create a new layout template, or edit one of the existing templates.

To Edit an Existing Template:

1. Click the template and then click “Open Layout Template”.

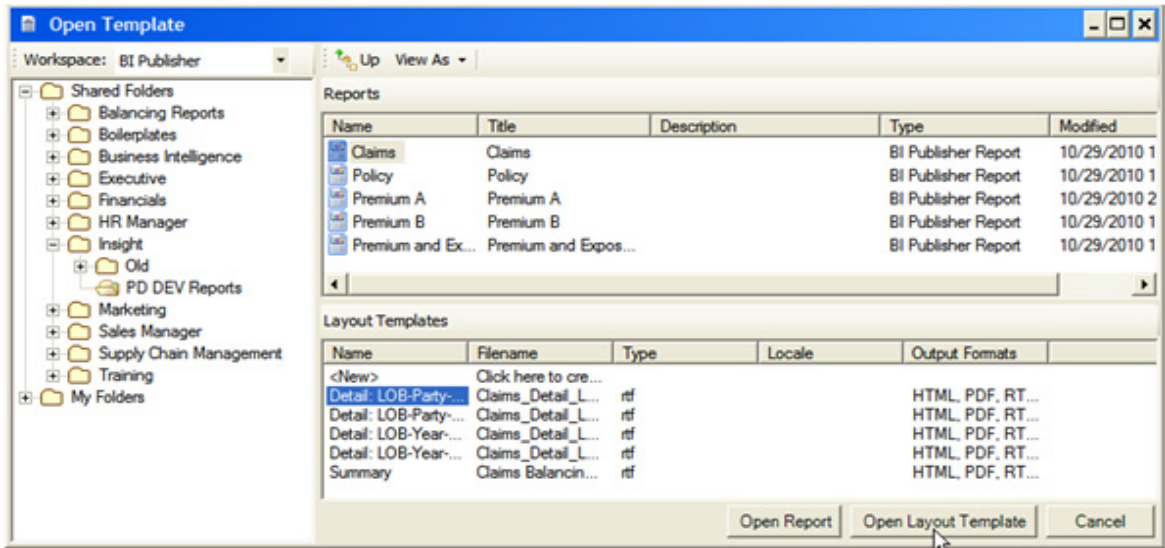


Figure 68: Select “Open Layout Template”

2. When you have completed your editing session, you can upload your changes directly into the existing layout template, or you can create a new template.

To Update the Existing Template:

- Go to the Add-Ins tab and select Oracle BI Publisher → Upload Template.

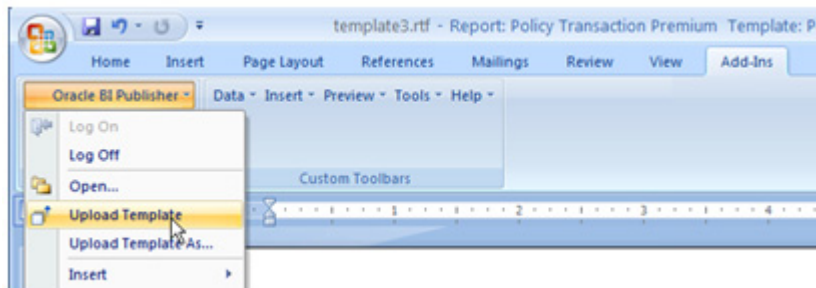


Figure 69: Select Upload Template

To Create a New Template:

- Go to the Add-Ins tab and select Oracle BI Publisher→Upload Template As... and then enter the new template name when prompted. You'll see a confirmation dialog following successful upload.

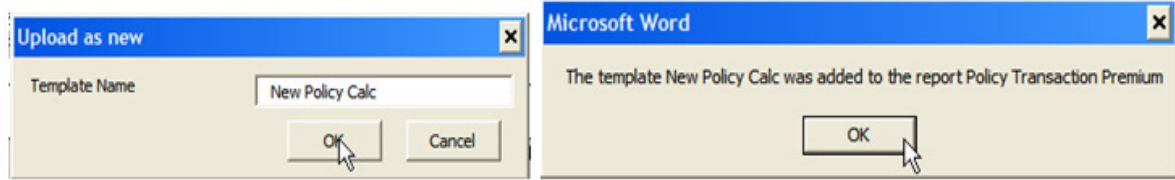


Figure 70: Create a New Template

Offline Template Modification

- Login to BI Publisher and navigate to **Shared Folders>Reports>Balancing Reports**.
- Locate the Balancing Report whose template you plan to edit and select the **Edit** link.



Figure 71: Select the Edit Link to View Templates

A screen of available Layout templates for that Balancing Report will appear.

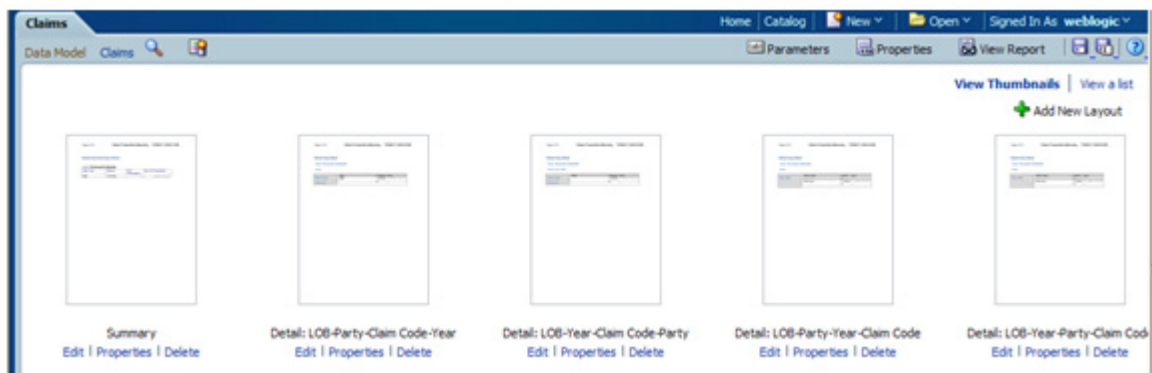


Figure 72: List of Layouts

3. Locate the template you wish to download and click on the **Edit** link to save it to your machine.

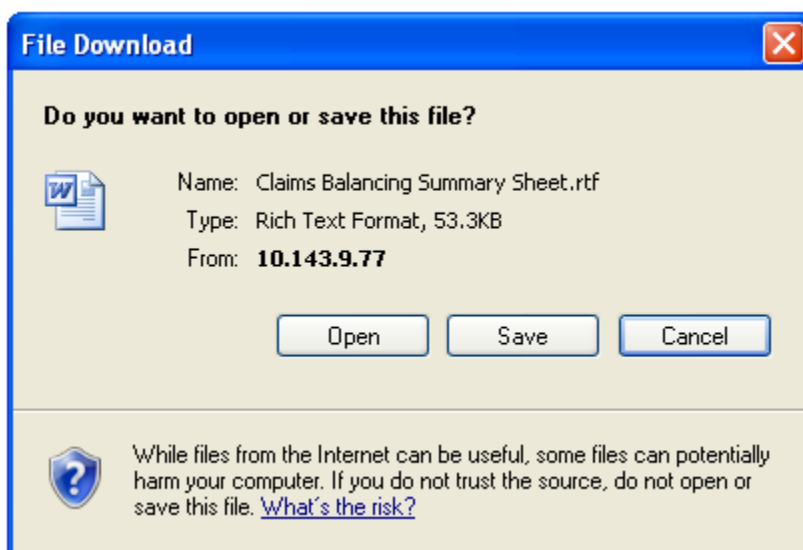


Figure 73: Download a Template

4. Click the template you wish to modify. The template will be downloaded to your local machine.
5. In the BI Publisher catalog, navigate to the report and click the **Open** link.
6. Click the **Actions** icon, then click **Export**, and then click **Data**. You will be prompted to save the XML file for the template.
7. Save the file to a local directory on your machine.
8. Use the **Load Sample XML** feature below to load the saved XML file.
9. Open Microsoft Word and open the template you just downloaded.
10. You can now proceed to edit the template.

MODIFYING A TEMPLATE

Before any template modification can begin, data and schema for the template must be loaded. If you are working online, this has been done for you automatically. If you are working offline, you will need to perform the following instructions:

1. On the Add-Ins tab, click “Data→Load XML Data...” and select the downloaded XML file. You will see a dialog box indicating successful load of the data set.

Note The BI Publisher Add-In data load applies to all instances of Microsoft Word that you may have open, so if you are editing multiple templates that do not use the same data set you will have to load the appropriate data for the template before performing operations with the BI Publisher Add-In.

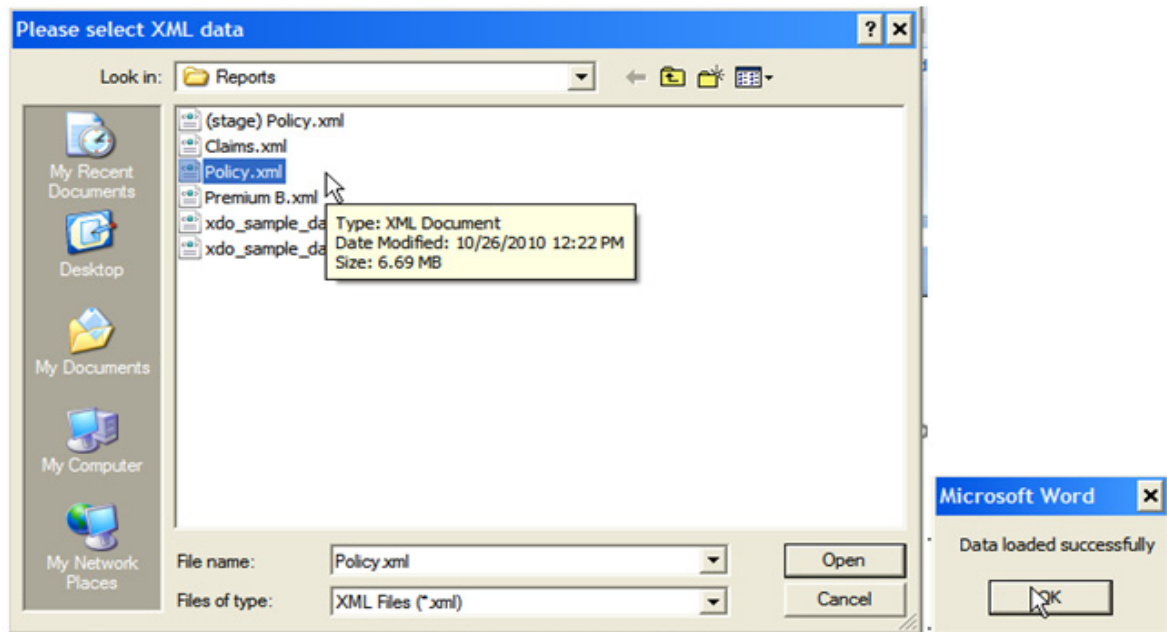


Figure 74: Download Data

Note You can still load data if you are working online, however any data load you perform will override the live data provided by BI Publisher. If you want to revert to the live data, you will have to log off and log on to BI Publisher. To do this, go to the Add-Ins tab and select Oracle BI Publisher -> Log Off, then Oracle BI Publisher -> Log On.

2. Now you are ready to modify the template. The easiest method to insert a field is to use the Field Chooser.

3. Display the Field Chooser by selecting the Add-Ins tab, and then Insert→Field...

The Field Chooser will display the available fields in the loaded XML data set. You can apply calculations such as SUM or AVG on a field by selecting the appropriate item from the dropdown.

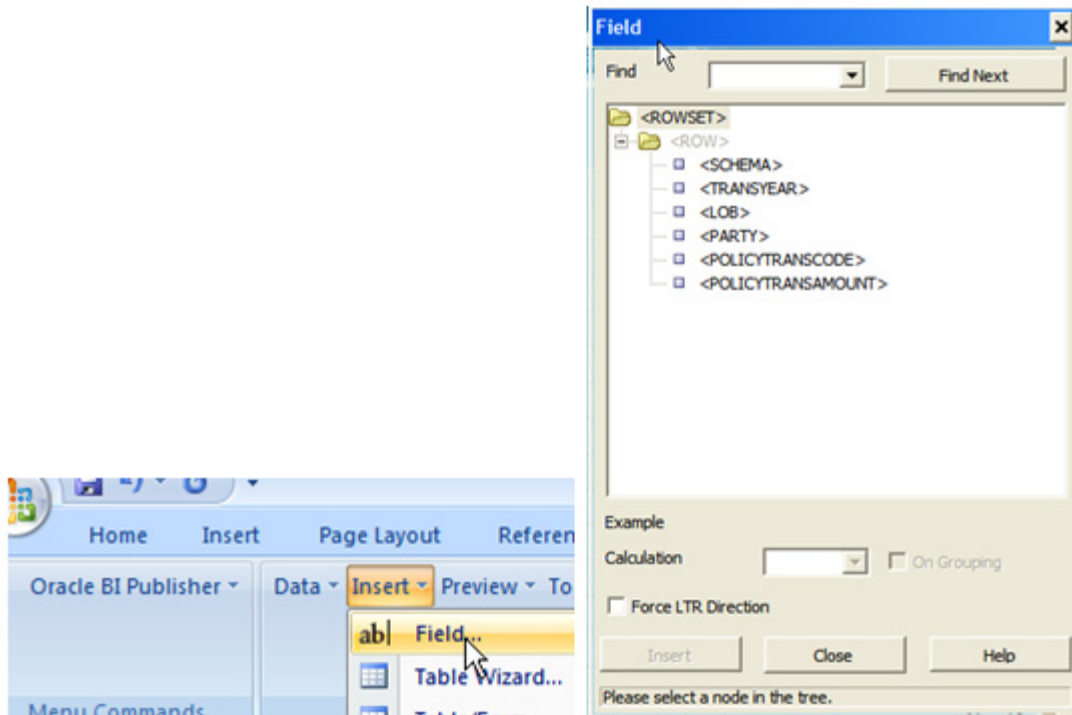


Figure 75: Field Chooser Dialog Box

4. To insert a field, highlight it and click Insert. The field will be inserted into the template.

Using the Wizard to Generate Tables and Fields

You can also use a wizard to generate tables and fields for you based on the data in the XML file. To run the Wizard:

1. Select Insert→Form Wizard... from the Add-Ins tab.

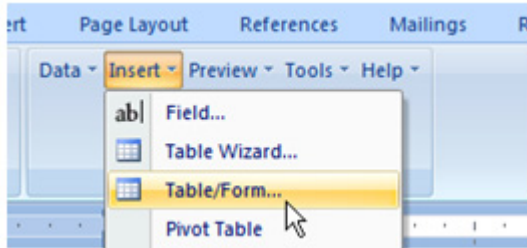


Figure 76: Start the Wizard

2. Select the Table format for your report.

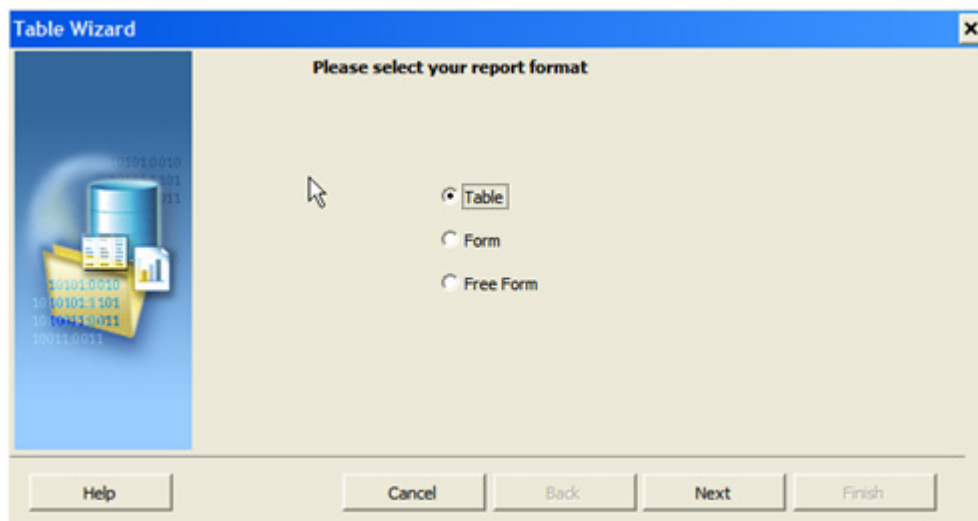


Figure 77: Select Table Format

If your report contains multiple data sets (e.g. the report has more than one Data Model defined in BI Publisher) you can select which set you would like to see in the report. BI Publisher also offers the ability to combine multiple Data Models into a single data set, in which case you would see only one choice, as shown below:

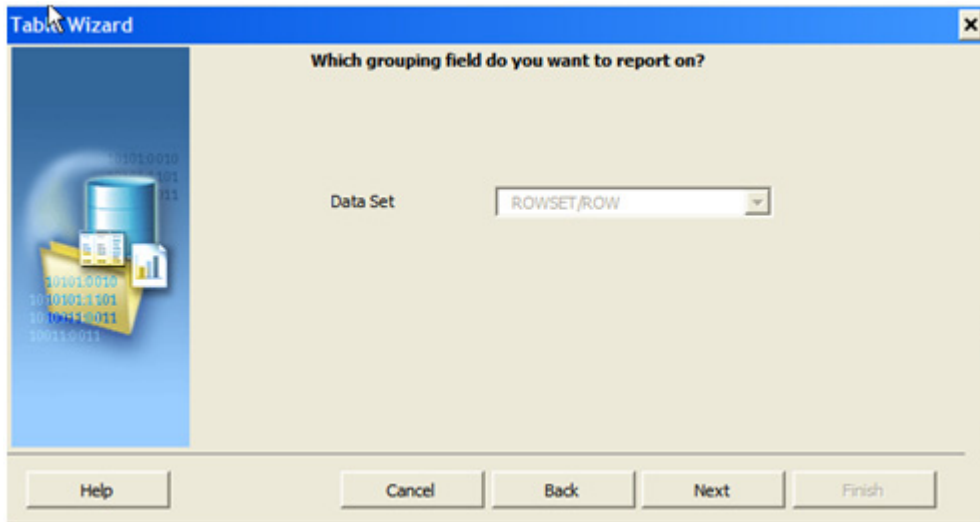


Figure 78: Select Data Set

3. Select the fields you want to appear in your report. The order of the fields below is not important at this time.



Figure 79: Select the Fields for the Report

- For the purposes of balancing data, grouping is very important. Most of the reports have different methods of grouping data depending on how the balancing information must be viewed. At this screen you can configure the order of grouping.

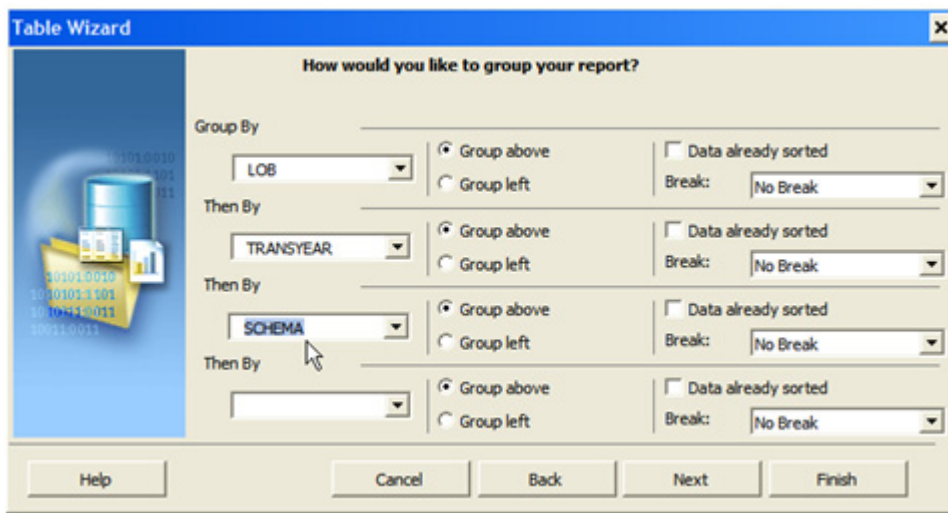


Figure 80: Configure Order of Grouping

- To enable sorting of data, select the field, direction, and data type for sorting.

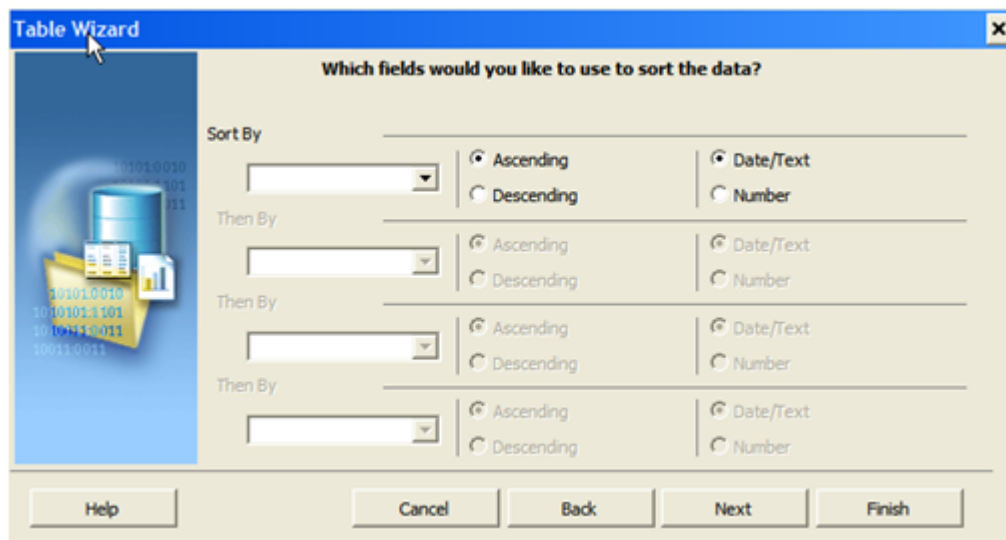


Figure 81: Enable Data Sorting

- 6. You can set up labels for any fields that are not used in grouping or sorting. These can be changed later.

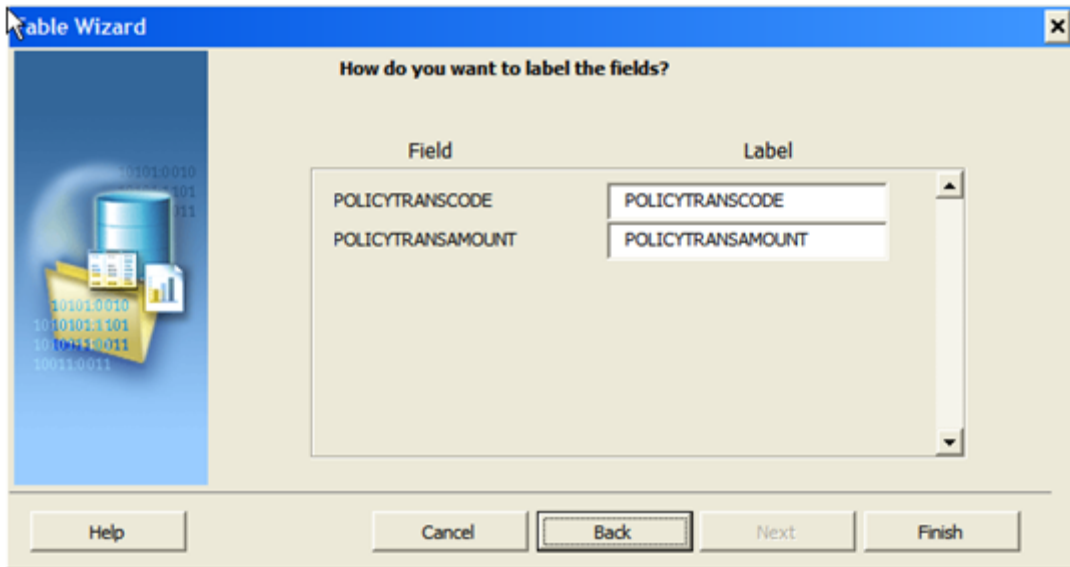


Figure 82: Setup Labels

Viewing the Output

After the wizard is completed, you will see the generated output. The items with gray highlight indicate special fields placed in the template. You can rearrange the generated output, create tables and modify fields just as you would a normal Word document.

Output Generated by Wizard:

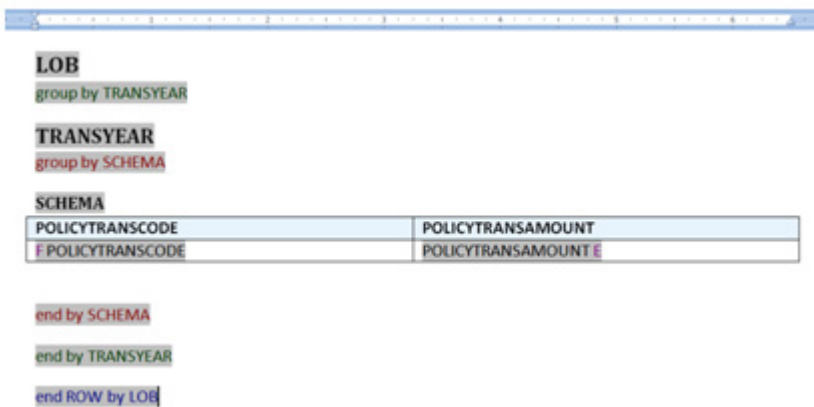


Figure 83: Output Generated by Wizard

Reorganized Output:

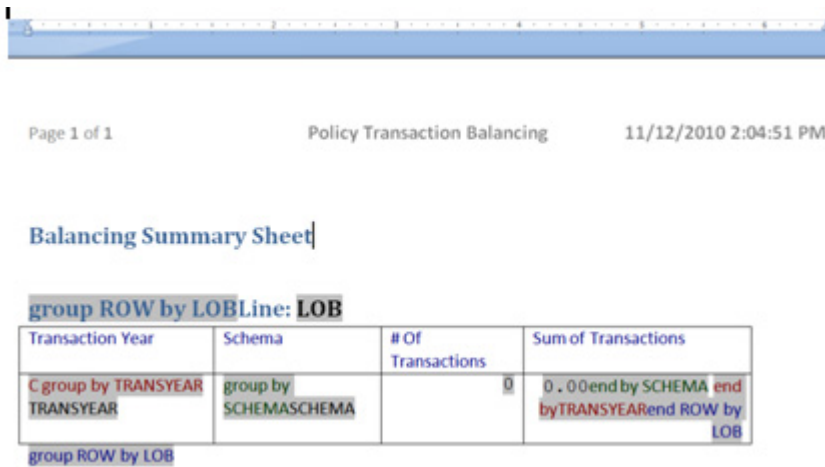


Figure 84: Reorganized Output

To preview the report with data, click Preview on the Add-Ins tab, and select the appropriate output type from the dropdown. The report will be generated using the loaded XML data.

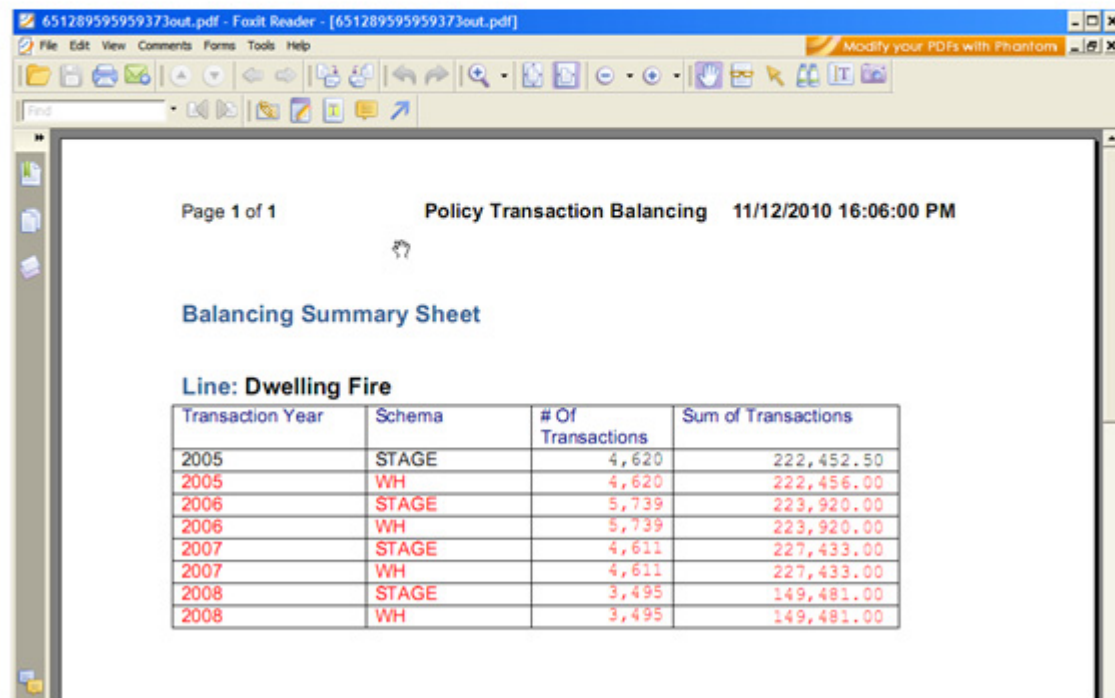
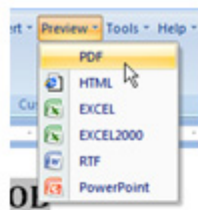


Figure 85: Preview a Report

Customizing the Report

You can further customize the report by changing the **grouping** and **calculations**.

Changing Grouping

An example of a grouping is shown below. Access the grouping by double-clicking on a grouping field, or by highlighting some text or fields in the document and selecting Insert→Repeating Group... from the Add-Ins tab.

The first tab, Properties, shows the correlation between the data input and the grouping selection. The For Each selection corresponds to a data row element in the XML, as shown here:

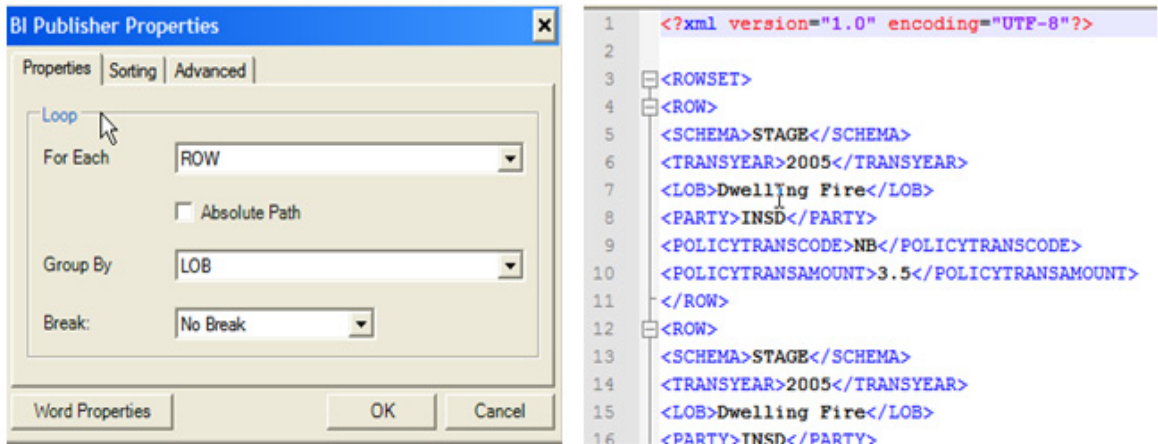


Figure 86: Properties Tab

The Group By selection corresponds to the selection made in the Wizard, however this can be changed using the dropdown. You can also change the Sorting properties as well:

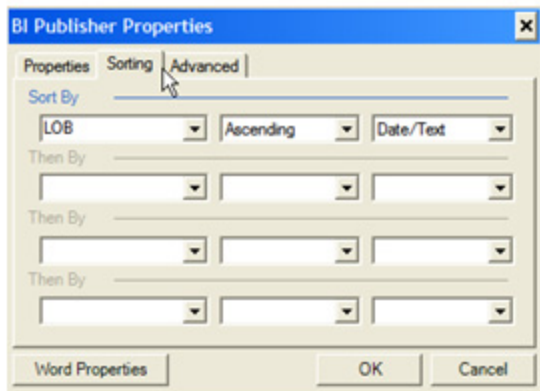


Figure 87: Sort Properties

The Advanced tab shows the underlying field codes which will be inserted in the template, and allows you to change the text that is displayed as a place holder in the field.

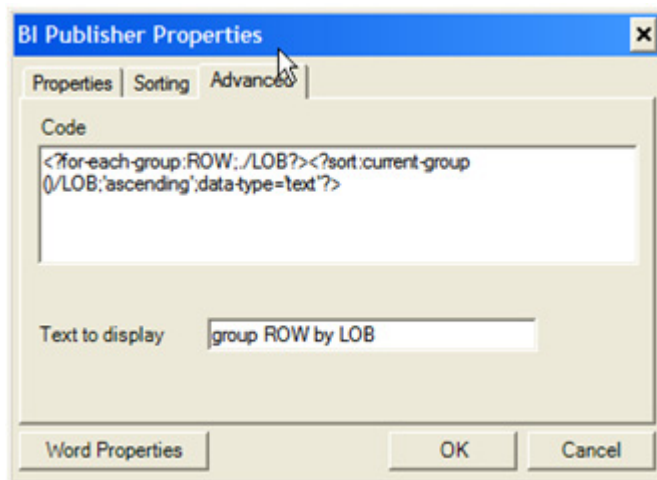


Figure 88: Advanced Tab

Once you have become accustomed to editing the field code directly, you may not need the various tabs offered in the field properties and instead you may edit the properties in the Code box as shown above.

Changing Calculations

Another common activity with Balancing Reports is to apply a calculation to a field. In most of the reports, there is at least one field that includes a SUM or COUNT calculation. Double-click a field to see its properties:

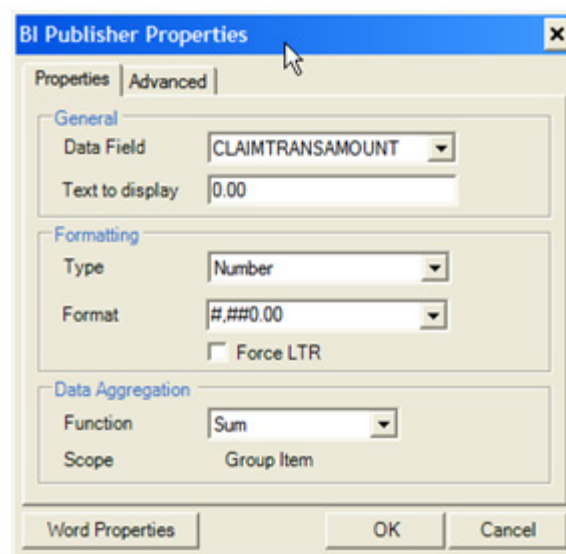


Figure 89: Applying a Calculation to a Field

The field shown below has a Sum calculation that is scoped to all CLAIMTRANSAMOUNT fields within the grouping. Additionally, the value is formatted as a number that has precision to two decimal places, and optionally includes comma formatting if the value requires. You can change the calculation using the Function dropdown if the field is already in place on the report, or you can set the calculation when adding the field to the report.

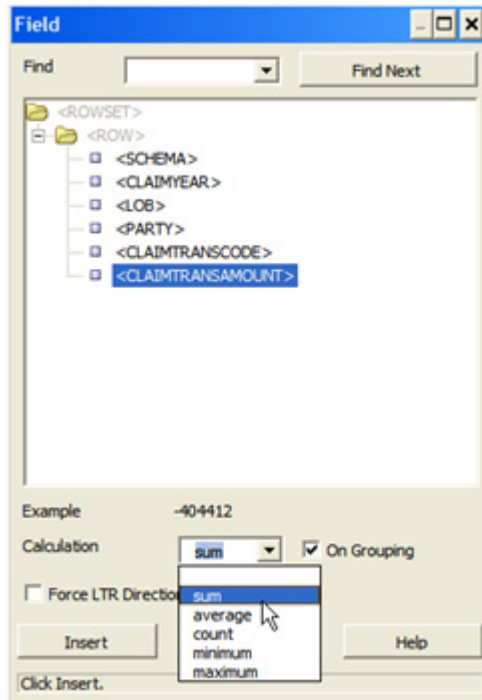


Figure 90: Select a New Function from Drop-Down List

Be sure to check the On Grouping control when adding fields to a report, and verify that your grouping is set up appropriately. Occasionally grouping can be confusing so it helps to view the groups in the report itself:

group ROW by LOBLine: LOB

Claim Year	Schema	# Of Transactions	Sum of Transactions
group by CLAIMYEAR CLAIMYEAR	group by SCHEMASchema	t	0.00 end by SCHEMA end by CLAIMYEAR end ROW by LOB

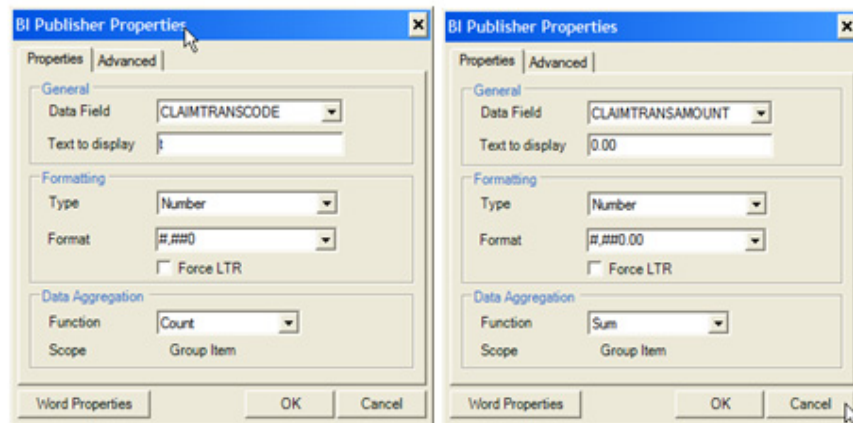


Figure 91: Viewing Grouping in Report

In the report shown above:

- The top-level grouping is by LOB. That means that the table will be reproduced for each distinct LOB group.
- The next levels of grouping are CLAIMYEAR and SCHEMA, and you can see that the field values for each are output directly after the grouping field.
- Next is a field which represents the number of transactions at the grouping of LOB, CLAIMYEAR and SCHEMA. That number is shown as field *t*, and uses the COUNT function scoped to the current group.
- Finally, the Sum of transactions field is shown, represented by *0.00*, which indicates that the field has a Number formatting attribute set. This field has the SUM function scoped to the current group. This report might be expressed in plain English as “display the count and sum of transactions, grouped by LOB, CLAIMYEAR and SCHEMA.”

An example of the output is shown below:

Balancing Summary Sheet

Line: AUTOCC

Claim Year	Schema	# Of Transactions	Sum of Transactions
2005	STAGE	1,272	1,104,530.24
2005	WH	1,272	1,104,530.24
2006	STAGE	1,822	1,950,058.88
2006	WH	1,822	1,950,058.88
2007	STAGE	727	820,009.71
2007	WH	727	820,009.71
2008	STAGE	238	46,658.68
2008	WH	238	46,658.68

Line: DFIRE

Claim Year	Schema	# Of Transactions	Sum of Transactions
2005	STAGE	790	1,264,502.62
2005	WH	790	1,264,502.62
2006	STAGE	1,379	5,476,350.48
2006	WH	1,379	5,476,350.48
2007	STAGE	746	859,666.38
2007	WH	746	859,666.38

Figure 92: Sample Summary Report

Relationship Codes

PARTY-LEVEL RELATIONSHIP CODES

OII_ST.PARTY (Party)

- CA_DRV
- CLMT
- CLM_ADJ
- CLM_EXAMR
- CLM_MGR
- INSD
- INSUR_CO
- PAR_INSUR_CO
- PA_DRV
- PRODR
- REINS_CO
- SUB_PRODR
- SUPPLIER
- UW
- UW_ASST

TYPE-LEVEL RELATIONSHIP CODES

OII_ST.ORG (Organization)

- CLMT
- CLM_ADJ
- INSD
- INSUR_CO

- PAR_INSUR_CO
- PRODR
- REINS_CO
- SUB_PRODR
- SUPPLIER

OII_ST.PERS (Person)

- CA_DRV
- CLMT
- CLM_ADJ
- CLM_EXAMR
- CLM_MGR
- INSD
- PA_DRV
- PRODR
- SUB_PRODR
- SUPPLIER
- UW
- UW_ASST

ROLE-LEVEL RELATIONSHIP CODES

OII_ST.INSUR_CO (Insurance Company)

- INSUR_CO
- PAR_INSUR_CO

OII_ST.PRODR (Producer)

- PRODR
- SUB_PRODR

OII_ST.UW (Underwriter)

- UW
- UW_ASST

CONTACT CODES

OII_ST.CNTCT (Contact)

- CA_DRV
- CLMT
- CLM_ADJ
- CLM_EXAMR
- CLM_MGR
- INSD
- INSUR_CO
- PAR_INSUR_CO
- PA_DRV
- PRODR
- REINS_CO
- SUB_PRODR
- SUPPLIER
- UW
- UW_ASST

OII_ST. PSTL_ADDR (Postal Address)

- CA_DRV
- CA_VEH
- CLM
- CLMT
- CLM_ADJ
- CLM_EXAMR
- CLM_MGR
- CP_BLDG
- DF_DWLNG
- HO_DWLNG
- INSD
- INSUR_CO
- PAR_INSUR_CO
- PA_DRV
- PA_VEH
- PRODR
- REINS_CO
- SUB_PRODR
- SUPPLIER
- UW
- UW_ASST

INDEX

A

Adaptive Data Model, 5
Allocated Adjustment Expense (ALAE), 46

B

Balancing Report Components
 Data Model, 92
 Layouts, 93
 List of Values, 92
 Parameters, 93
 Templates, 93
Balancing Reports, 79
 Components, 90
 Editing, 96
 Installing, 80
 Viewing, 95
Bridge Tables, 26

C

Cache
 Purging OBIEE Cache, 60
Claim Transaction Categories, 46
Claim Transaction Codes, 46
Claim Transaction Types, 46
Contact, 18
CONTACT table, 18
Core Data Model, 3
Corporate Views, 30
Customize OII Data Views Phases, 14
 Client Resources, 14

D

Data Loading, 45
Data Mapping Phases, 12
 Assumptions, 13
 Client Resources, 12
 Risks, 13
 Scope, 12
Data Mart Load Phases, 14
 Assumptions, 14
 Client Resources, 14
 Risks, 14
 Scope, 14
Data Model
 Fact Tables, 29
Data Relationships, 43
Data Validation, 36
Deductible, 46, 47
Denormalization, 22

Dimensions, 4

E

ETL Development Phases, 13
 Assumptions, 13
 Client Resources, 13
 Risks, 13
 Scope, 13

F

Fact Tables
 Data Mart, 29
Facts, 28
Flow Diagrams, 24

G

geographic boundaries, 19

K

Keys
 Common/Shared, 43

L

Line of Business, 3
 Commercial Auto, 3
 Commercial Property, 3
 Dwelling Fire, 3
 Homeowners, 3
 Personal Auto, 3
 Personal Umbrella, 3
Load Options, 48, 51, 55
Loading to Staging, 24
Loss, 46

M

Master Data Management Strategy, 33
Metadata, 33
Monthly Snapshot Marts
 Combined Policy & Claim, 4

N

Natural Keys, 19
Normalization, 22

O

OII Implementation Timeline, 12
OII Project Stages
 Deployment, 11
 Execution, 11

- Initiation, 11
- Planning, 11
- OII Project Stages, 11
- online analytical processing, 22
- online transaction processing, 22

P

- Party Model, 18
- Personal Auto Premiums, 44
- Policy Transaction Codes, 47
- Postal Address, 19
- Program Flow, 17
- PSTL_ADDR table, 18
- Purge OBIEE Cache, 60

R

- referenced entities, 25
- Relationship Codes, 113
- Relationship Model, 21
- Required Tables, 35

S

- Salvage, 46
- schema
 - normalized star schema, 28
 - snowflake pattern, 28
- Slowly-Changing Dimensions, 5
- snowflake schema, 28
- Source-to-Staging ETL, 33
- Staging Model Concepts, 17
 - Contact, 18
 - Party Model, 18
 - Postal Address, 19
 - Relationship Model, 21
- Staging to Warehouse, 26
- star schema, 28
- Subrogation, 46
- System Codes
 - Claim Transaction Code, 46

T

- Transactional Marts
 - Combined Policy & Claim, 4

U

- Unallocated Adjustment Expense (ULAE), 46
- Unique Keys, 43

V

- Views, 30
 - Corporate Views, 30

W

- Warehouse to Data Mart, 28