

Oracle® Business Intelligence

Standard Edition One Tutorial

Release 10g (10.1.3.2.1)

E10312-01

May 2007

Oracle Business Intelligence Standard Edition One Tutorial, Release 10g (10.1.3.2.1)

E10312-01

Copyright © 1979, 2007, Oracle. All rights reserved.

Primary Author: Jenny Tsai

Contributing Authors: Kasturi Shekhar, Jennie Vasquez

Contributors: Kurt Wolff, Pedro Arnal, Jon Ainsworth, Mike Donohue, Richard Green, Crystal Burek, Gianluca Nostro, Srinivas Putrevu, Yi L. Lu, Marla Azriel, Carolyn Bruse, Roberto Falcinelli

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.

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

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.

Contents

Preface	ix
Audience	ix
Documentation Accessibility	ix
Related Documents	x
Conventions	x
1 Introducing Oracle BI Standard Edition One	
1.1 Oracle Database 10g Standard Edition One	1-1
1.1.1 Oracle Warehouse Builder	1-2
1.2 Oracle BI Server	1-2
1.3 Oracle BI Answers	1-2
1.4 Oracle BI Interactive Dashboards	1-2
1.5 Oracle BI Publisher	1-3
1.6 Oracle BI Standard Edition One Usage Considerations	1-3
2 Using the Tutorial	
2.1 Who Should Read This Tutorial	2-1
2.2 How This Tutorial Is Organized	2-1
2.2.1 Case Study—The Global Electronics Corporation	2-2
2.2.2 Database Structures for the GEC Case Study	2-3
2.2.2.1 BISE1_SALES Schema	2-3
2.2.2.2 BISE1_SALESWH Schema	2-3
2.2.2.3 BISE1_TUTORIALWH Schema	2-4
2.2.3 Accounts Used in the Tutorial	2-4
2.2.4 Tutorial Files and Directory Structure	2-6
3 Set Up the Data Mart	
3.1 About Setting Up the Data Mart	3-1
3.2 Examining a Completed Warehouse Builder Project	3-2
3.2.1 Access the OWB Repository Using the OWB Design Center	3-2
3.2.2 Explore the Completed OWB Project	3-3
3.3 Creating the Dimensions and the Cube	3-7
3.3.1 Create the CHANNELS Dimension Using the Wizard	3-8
3.3.2 Create the PRODUCTS Dimension Using the Editor	3-12
3.3.3 Create the TIMES Dimension Using the Time Wizard	3-17

3.3.4	Create the SALES Cube Using the Editor	3-20
3.4	Creating the Mappings.....	3-22
3.4.1	Design the Mapping to Load the PRODUCTS Dimension.....	3-23
3.4.2	Use Tcl Scripts to Create Mappings for the CHANNELS Dimension and the SALES Cube 3-37	
3.5	Generating the Mappings	3-38
3.6	Designing a Process Flow	3-40
3.6.1	Create a New Process Flow	3-40
3.6.2	Add a Fork Activity.....	3-42
3.6.3	Add the Mappings.....	3-43
3.6.4	Add an AND and Control Activities	3-45
3.6.5	Add Transitions	3-46
3.7	Validating and Generating the Process Flow	3-48
3.8	Deploying Mappings and Loading Data	3-49
3.8.1	Register the Locations	3-49
3.8.2	Deploy Sequences	3-50
3.8.3	Deploy Tables.....	3-52
3.8.4	Deploy the Mappings.....	3-52
3.8.5	Deploy the Process Flow.....	3-53
3.8.6	Execute the Process Flow.....	3-53
3.8.7	View the SALES Cube Data	3-56

4 Build the BI Repository

4.1	About the Oracle BI Administration Tool	4-1
4.2	Building the Physical Layer.....	4-2
4.2.1	Verify an ODBC Data Source Exists.....	4-2
4.2.2	Start Oracle BI Services	4-3
4.2.3	Import Tables from the Database Schema	4-3
4.3	Building the Business Model and Mapping Layer	4-5
4.3.1	Create a Business Model.....	4-6
4.3.2	Populate the Business Model	4-7
4.3.3	Rename Business Model Objects	4-8
4.3.4	Delete Unnecessary Business Model Objects.....	4-11
4.3.5	Build Dimension Hierarchies.....	4-12
4.4	Building the Presentation Layer	4-20
4.5	Testing and Validating a BI Repository	4-21
4.5.1	Run a Consistency Check	4-21
4.5.2	Enable Query Logging	4-22
4.5.3	Use Oracle BI Answers to Execute Queries	4-23
4.5.4	Use the Query Log to Verify Queries	4-25
4.6	Creating Calculation Measures.....	4-26
4.6.1	Create a New Measure.....	4-27
4.6.2	Create a Calculation Measure Using Logical Columns	4-27
4.6.3	Create a Calculation Measure Using Physical Columns.....	4-31
4.6.4	Create a Calculation Measure Using the Calculation Wizard	4-34
4.7	Using Initialization Blocks and Variables - Advanced.....	4-36
4.7.1	Explore an Initialization Block for Session Variables	4-37

4.7.2	Test the Initialization Block and Session Variables	4-39
4.7.3	Create a Dynamic Repository Variable	4-42
4.8	Executing Direct Database Requests - Advanced	4-45
4.9	Using Aggregates - Advanced	4-46
4.9.1	Use a Direct Database Request to Create Aggregate Tables	4-47
4.9.2	Import the Aggregate Tables	4-47
4.9.3	Create Physical Joins to the Aggregate Tables	4-48
4.9.4	Map Logical Columns to Aggregate Columns.....	4-49
4.9.5	Test the Aggregation Using Oracle BI Answers.....	4-51
4.10	Creating Time Series Measures - Advanced	4-51
4.10.1	Identify a Dimension as a Time Dimension.....	4-52
4.10.2	Create a Month Ago Measure	4-52
4.10.3	Create a Change Month Ago Measure	4-54
4.10.4	Create a ToDate Measure	4-55
4.10.5	Test the Time Series Measures	4-55
4.11	Adding Multiple Sources - Advanced	4-57
4.11.1	Create an ODBC DSN	4-57
4.11.2	Import the Excel Data Source Into the BI Repository.....	4-58
4.11.3	Create a Physical Key	4-59
4.11.4	Create the Physical Joins.....	4-59
4.11.5	Set the Dimension Key	4-61
4.11.6	Map the Logical Dimension Columns	4-62
4.11.7	Create the Quota Measures	4-63
4.11.8	Test the Quota Measures	4-66

5 Analyze the Data

5.1	Prerequisites	5-1
5.2	Creating a Query and a Chart	5-2
5.2.1	Create an Answers Query	5-2
5.2.2	Add Filters on Columns.....	5-4
5.2.3	Create Totals and Format Results	5-6
5.2.4	Create a Chart.....	5-9
5.3	Working with Pivot Tables	5-14
5.4	Creating Column Selectors	5-17
5.5	Creating a Narrative View	5-19
5.6	Creating View Selectors	5-21
5.7	Creating Conditional Formats, Gauges, and Navigation	5-22
5.7.1	Create a Report with Conditional Formats.....	5-22
5.7.2	Add Gauges to the Report.....	5-23
5.7.3	Add Navigation to the Report	5-25

6 Publish Data on Dashboards

6.1	Prerequisites	6-1
6.2	Creating an Interactive Dashboard	6-2
6.3	Using Dashboard Prompts and Presentation Variables.....	6-6
6.3.1	Create a Dashboard Prompt.....	6-6

6.3.2	Use a Presentation Variable to Populate a Title	6-8
-------	---	-----

7 Create and Publish Reports

7.1	Logging in to Oracle BI Publisher and Setting Preferences	7-1
7.2	Creating a Report Based on the Operational Data	7-2
7.2.1	Create a JDBC Connection to the Data Source	7-3
7.2.2	Define a New Report	7-3
7.2.2.1	Create the New Report	7-3
7.2.2.2	Create a Data Model for the Report	7-3
7.2.2.3	Create Lists of Values and Parameters	7-5
7.2.2.4	Create a Layout	7-6
7.2.3	Create an RTF Template for the Report	7-6
7.2.3.1	Install Oracle BI Publisher Desktop	7-6
7.2.3.2	Open the Report Layout Template	7-7
7.2.3.3	Insert Report Fields Into the Template	7-7
7.2.3.4	Apply Format Masks to the Fields	7-10
7.2.4	View the Report	7-11
7.3	Automatically Delivering BI Publisher Reports	7-12
7.3.1	Configure Oracle BI Publisher Scheduler Report Delivery Options	7-12
7.3.2	Schedule FTP Delivery of a Report	7-13
7.4	Publishing a BI Publisher Report in BI Dashboards	7-14
7.4.1	Create an Oracle BI Publisher Report from an Oracle BI Answers Request	7-14
7.4.2	Publish the Oracle BI Publisher Report on Oracle BI Interactive Dashboards	7-18
7.5	Creating an Oracle BI Publisher Report Using BI Server Metadata	7-19
7.5.1	Create an Oracle BI Publisher Report from BI Server Metadata	7-20
7.5.2	Publish the Template to View the Report Data in Oracle BI Publisher	7-21

8 Reset the Tutorial Exercise Environment

8.1	Reset the Oracle Warehouse Builder Environment	8-1
8.2	Reset the BI Server Metadata Repository	8-2
8.3	Reset the BI Presentation Services Web Catalog	8-2

9 Next Steps

9.1	Enhancing the Security of Your Oracle BI Environment	9-1
9.2	Administering the Database	9-2
9.3	Using Oracle Warehouse Builder	9-2
9.4	Administering Oracle Warehouse Builder	9-3
9.5	Administering Oracle BI Server	9-4
9.6	Using Oracle BI Answers and Dashboards	9-4
9.7	Administering Oracle BI Answers and Dashboards	9-5
9.8	Using Oracle BI Publisher	9-5
9.9	Administering Oracle BI Publisher	9-5

A Data Mart Concepts

A.1	What Is a Data Mart?	A-1
A.2	How Is It Different from a Data Warehouse?	A-1

A.3	Dependent and Independent Data Marts.....	A-2
A.4	What Are the Steps in Implementing a Data Mart?.....	A-2
A.4.1	Designing	A-3
A.4.2	Constructing	A-3
A.4.3	Populating.....	A-3
A.4.4	Accessing.....	A-3
A.4.5	Managing	A-4

B Design the Data Mart

B.1	Defining the Scope of the Data Mart Project.....	B-1
B.2	Defining the Requirements for the Data Mart.....	B-2
B.2.1	Define Business Requirements	B-2
B.2.2	Identify Technical Requirements.....	B-3
B.2.3	How Do You Know If You Have Done It Right?	B-3
B.3	Data Mart Design.....	B-3
B.3.1	Creating a Logical Design.....	B-4
B.3.2	Creating a Wish List of Data	B-5
B.3.3	Identifying Sources.....	B-5
B.3.4	Classifying Data for the Data Mart Schema.....	B-6
B.3.4.1	Dimensions	B-6
B.3.4.2	Facts	B-6
B.3.4.3	Granularity	B-7
B.3.5	Designing the Star Schema	B-7
B.3.6	Moving from Logical to Physical Design	B-7
B.3.6.1	Estimating the Size of the Data Mart	B-8
B.3.6.2	What Is Metadata?	B-9

Preface

Oracle Business Intelligence Standard Edition One is a comprehensive business intelligence (BI) offering designed for small-to-midsize businesses or departments in larger organizations. It is a complete BI solution that includes interactive dashboards, highly formatted reporting, ad hoc query and analysis, Extract-Transform-Load (ETL), data modeling, and server administration—all in an easy-to-install package with minimal configuration.

Audience

This document is intended for company-level or department-level IT administrators, or for data warehousing or business intelligence specialists.

Documentation Accessibility

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

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

In addition to this tutorial, Oracle Business Intelligence Standard Edition One has the following documentation:

- *Oracle Business Intelligence Standard Edition One Installation Guide*
- Oracle Business Intelligence Standard Edition One component-level Online Help (for example, Oracle BI Answers Help, Oracle BI Administration Tool Help)

You can also refer to the following:

- The Oracle Business Intelligence Standard Edition One Documentation Web site:
http://www.oracle.com/technology/documentation/bi_se1.html
- The Oracle Business Intelligence Standard Edition One Product Web site:
<http://www.oracle.com/technology/products/bi/standard-edition-one.html>
- The latest information on supported versions is on the Certify application at:
<http://metalink.oracle.com>
- The Oracle Business Intelligence Standard Edition One Metalink note 429373.1 is updated periodically as new information becomes available. For the latest information, go to the Oracle MetaLink site at:
<http://metalink.oracle.com>

Conventions

The following text conventions are used in this document:

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

Introducing Oracle BI Standard Edition One

Selecting a business intelligence product to suit all your needs is a challenging task. There are too many products, too many vendors, and too much confusion. Often, you must act as a systems integrator for products that are designed, implemented, sold, and supported by different vendors.

By purchasing Oracle Business Intelligence Standard Edition One (Oracle BI Standard Edition One), you now have in one place all of the products that you need to extract the business intelligence concealed in your business data so that you can more easily make the right business decisions at the right time. Oracle BI Standard Edition One has been developed on one key premise: *Better Decisions Made Simple*.

Using Oracle BI Standard Edition One, you can immediately reap the benefits of being able to quickly distill operational information into business reports and to deliver them automatically through a number of channels, including Web dashboards. Taking it a step further, you can readily transform operational data into data marts, which enable you to further classify and aggregate information so that you can perform in-depth analysis through any number of dimensions (such as time) and hierarchies (for example, drill down from year to quarter to month).

The components of Oracle BI Standard Edition One are:

- Oracle Database 10g Standard Edition One, which includes Oracle Warehouse Builder
- Oracle BI Server
- Oracle BI Answers
- Oracle BI Interactive Dashboards
- Oracle BI Publisher

This chapter contains the following topics:

- [Section 1.1, "Oracle Database 10g Standard Edition One"](#)
- [Section 1.2, "Oracle BI Server"](#)
- [Section 1.3, "Oracle BI Answers"](#)
- [Section 1.4, "Oracle BI Interactive Dashboards"](#)
- [Section 1.5, "Oracle BI Publisher"](#)

1.1 Oracle Database 10g Standard Edition One

Oracle Database 10g is the latest release of the leading relational database for business intelligence and data warehousing today. Oracle is most often chosen because of its

success in satisfying the core requirements for data management: performance, scalability, and manageability.

1.1.1 Oracle Warehouse Builder

Oracle Warehouse Builder, also known as Warehouse Builder or OWB, is a business intelligence tool that provides an integrated solution for designing and deploying enterprise data warehouses, data marts, and business intelligence applications.

Warehouse Builder is more than an extract, transform, and load (ETL) tool. It supports the complete information management life cycle:

- **Design:** Importing enterprise data models, graphical modeling of multidimensional schemas, and intuitive mapping of the source to the target
- **Build:** Generation and population of the warehouse and marts by using Oracle Database features
- **Extract, transform, and load data:** Extracting relational sources, flat file sources, and enterprise resource planning (ERP) sources, such as Oracle E-Business Suite, PeopleSoft, and SAP
- **Integrate:** Integrating readily with the Oracle Database (Oracle Real Application Clusters, Oracle Online Analytical Processing, Oracle Spatial, Oracle Enterprise Manager, and so on), Oracle E-Business Suite, and Oracle Business Intelligence
- **Maintain:** Maintaining the warehouse and refreshing data

1.2 Oracle BI Server

This dedicated BI server provides the common business model and abstraction layer for the analytics applications. Oracle BI server supports access to multiple data sources, including relational databases (Oracle, SQL Server, and so on), OLAP sources (Analysis Services, Oracle OLAP) and offline sources (Excel, XML, and so on), among others.

The platform supports a full complement of access, analysis, and information delivery options, all in one fully integrated Web environment. Each of these components serves different audiences in the organization who have different appetites for the same underlying data, but need to access it in different ways. But unlike other BI tools, all components are integrated by a common architecture, enabling a seamless and intuitive user experience.

1.3 Oracle BI Answers

Oracle BI Answers provides true end user ad hoc capabilities in a pure Web architecture. Users interact with a logical view of the information—completely hidden from data structure complexity while simultaneously preventing runaway queries—and can easily create charts, pivot tables, reports, and visually appealing dashboards, all of which are fully interactive and drillable and can be saved, shared, modified, formatted, or embedded in the user's personalized Oracle BI Intelligence Dashboards. The results are new levels of business user self-sufficiency in an environment that is fully secure and controlled by IT.

1.4 Oracle BI Interactive Dashboards

Oracle BI Interactive Dashboards provide any knowledge worker with intuitive, interactive access to information that is actionable and dynamically personalized

based on the individual's role and identity. In the Oracle BI Interactive Dashboards environment, the end user is working with live reports, prompts, charts, tables, pivot tables, graphics, and tickers in a pure Web architecture. The user has full capability for drilling, navigating, modifying, and interacting with these results. Oracle BI Interactive Dashboards can also aggregate content from a wide variety of other sources, including the Internet, shared file servers, and document repositories.

1.5 Oracle BI Publisher

Oracle BI Publisher (formerly known as XML Publisher) offers the most efficient, scalable reporting solution available for complex, distributed environments. It provides a central architecture for generating and delivering information to employees, customers, and business partners—both securely and in the right format. Oracle BI Publisher report formats can be designed using Microsoft Word or Adobe Acrobat, tools most users are already familiar with. Oracle BI Publisher also enables you to bring data in from multiple data sources into a single output document. You can deliver reports by printer, e-mail, or fax. You can publish your report to a portal. You can allow users to collaboratively edit and manage reports on Web-based Distributed Authoring and Versioning (WebDav) Web servers. As part of Oracle BI, Oracle BI Publisher leverages common dashboarding, metadata, security, calculation, caching, and intelligent request generation services.

1.6 Oracle BI Standard Edition One Usage Considerations

One of the key benefits of Oracle BI Standard Edition One is that you can begin by implementing only those components that are needed to satisfy your immediate requirements, and then evolve the solution architecture as your business requirements grow.

The following example demonstrates how you might begin using Oracle BI Standard Edition One, and then extend the architecture to meet your growing requirements:

1. Start with Oracle BI Publisher by connecting it to your transactional system (such as an order entry application) to generate operational business reports and documents.
2. You now have multiple data sources, an Excel spreadsheet containing sales projections, and the order entry system. Using these data sources, you need to report on actuals versus targets. You can add Oracle BI Server into the architecture to logically tie together these two data sources.
3. The volume of order entry data has grown significantly, and reporting directly off the transactional system poses a performance issue. You need to aggregate and stage the historical transactional data by creating a data mart. You can use Oracle Warehouse Builder to build the data mart.
4. Both the reporting requirements and the user population have increased, and have become more sophisticated. You can add Oracle BI Answers and Oracle BI Interactive Dashboards to deliver ad hoc querying and self-service dashboards to your users.

Using the Tutorial

This tutorial presents an overview of the basics and a guided tour of how to use the products in Oracle BI Standard Edition One to quickly develop BI reports, build dashboards, and create data marts. This book does not emphasize formal definition of concepts or detailed discussion of the underlying issues. For details on each of the products and its use, refer to the product documentation available online. In addition, several instructor-led and online courses on these products are available through Oracle University (<http://education.oracle.com>).

This chapter contains the following topics:

- [Section 2.1, "Who Should Read This Tutorial"](#)
- [Section 2.2, "How This Tutorial Is Organized"](#)

2.1 Who Should Read This Tutorial

This book is intended for anyone involved in extracting business intelligence using Oracle BI Standard Edition One. Given the cross-section of organizational units and individuals involved in building a data mart, you could fit any of the following roles:

- Information technology (IT) professional
- Business analyst
- Database administrator
- Data mart designer
- Application developer
- Systems integrator

In general, this book defines technical terms and concepts when they are introduced. However, it assumes that you know the underlying operating system and are familiar with basic system administration tasks. It also assumes some knowledge of relational database systems.

2.2 How This Tutorial Is Organized

The organization of the tutorial follows the process flow of building and using a data mart. At each step in the process, this book provides some technical background on the products used and then uses a case study to walk you through the implementation steps.

The installation of Oracle BI Standard Edition One provides you with a database instance, BISE1DB, which holds all data referenced in the case study. The installation

takes care of all of the setup needed to use any of the Oracle BI Standard Edition One components to access this database.

Oracle Corporation recommends that you use this BISE1DB database as the starting point for building your data mart, rather than creating a new database. That is the approach taken in this book.

If you want to work through the tutorial chapters in a different order, to suit your immediate needs, you can do so as long as you work through the steps in sequence within each chapter. For example, if you want to begin by learning how to create and publish reports based either on your operational data, or an existing data mart, you can proceed directly to [Chapter 7, "Create and Publish Reports"](#), read the introductory and prerequisites sections, and then begin with step 1 in that chapter.

To help you determine your best entry point into the tutorial, here is an overview of what each of the subsequent chapters cover:

- In [Chapter 3, "Set Up the Data Mart"](#), you use Oracle Warehouse Builder to create and populate a data mart using an operational data source (an order entry system) as your starting point.
- In [Chapter 4, "Build the BI Repository"](#), you use the Oracle BI Server Administration Tool to build a BI metadata repository from a data mart schema as well as from a Microsoft Excel spreadsheet. The BI metadata repository presents a view of the data that can be easily understood and readily accessed using client tools such as Oracle BI Answers.
- In [Chapter 5, "Analyze the Data"](#), you use Oracle BI Answers to query the data mart and display the query results in various ways, including cross-tabs and charts.
- In [Chapter 6, "Publish Data on Dashboards"](#), you use Oracle BI Interactive Dashboards to publish the BI Answers results so that users can view them through a dashboard.
- In [Chapter 7, "Create and Publish Reports"](#), you use Oracle BI Publisher to create and publish reports based on the operational data, and the data mart (through BI Answers queries).

This section contains the following topics:

- [Section 2.2.1, "Case Study—The Global Electronics Corporation"](#)
- [Section 2.2.2, "Database Structures for the GEC Case Study"](#)
- [Section 2.2.3, "Accounts Used in the Tutorial"](#)
- [Section 2.2.4, "Tutorial Files and Directory Structure"](#)

2.2.1 Case Study—The Global Electronics Corporation

This tutorial presents a case study to illustrate the use of Oracle BI Standard Edition One in building and managing a data mart. The case study represents activities at a fictitious organization, Global Electronics Corporation (GEC).

GEC, established in 2003, distributes computer hardware and software components to customers all over the world. GEC wants to analyze which components of its business are profitable, which could become more profitable, and which sales channels are most effective.

2.2.2 Database Structures for the GEC Case Study

There are several schemas in the BISE1DB database that you use throughout the tutorial. Database schemas are logical containers for objects in the database.

The operational data source is the order entry schema, called **BISE1_SALES**. This schema contains the source data for the prebuilt data mart schema, called **BISE1_SALESWH**.

A second data mart schema called **BISE1_TUTORIALWH** initially contains no objects. If you work through the tutorial in the given chapter sequence, you will complete the design, construction, and population of this data mart. If you choose to skip [Chapter 3, "Set Up the Data Mart"](#), and launch into any of the other chapters first, you will need to populate this schema by running a script.

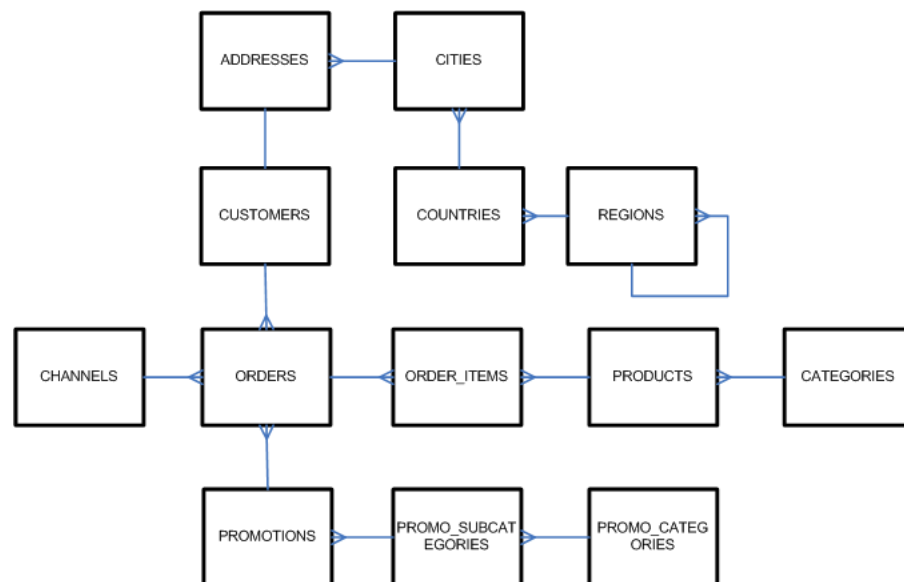
This section contains the following topics:

- [Section 2.2.2.1, "BISE1_SALES Schema"](#)
- [Section 2.2.2.2, "BISE1_SALESWH Schema"](#)
- [Section 2.2.2.3, "BISE1_TUTORIALWH Schema"](#)

2.2.2.1 BISE1_SALES Schema

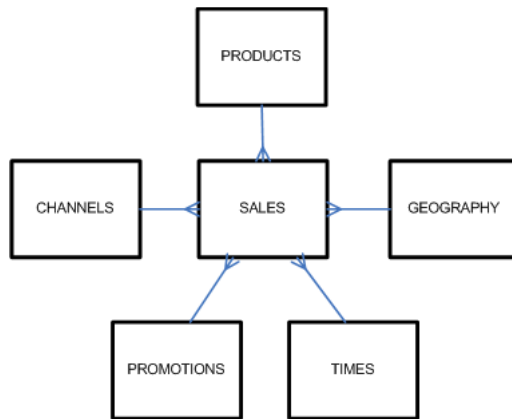
The BISE1_SALES schema contains tables that store the data captured by the order entry application. The entity relationship diagram (ERD) shown in [Figure 2-1](#) represents how the tables in this schema are related. This is the source schema for the prebuilt data mart schema, BISE1_SALESWH. This is also the source schema you will use to build the BISE1_TUTORIALWH data mart schema.

Figure 2-1 Entity Relationship Diagram for BISE1_SALES Schema



2.2.2.2 BISE1_SALESWH Schema

The BISE1_SALESWH schema contains tables that comprise the data mart star schema. The entity relationship diagram (ERD) shown in [Figure 2-2](#) represents how the tables in this schema are related. Note that there are five dimensions (**PRODUCTS**, **GEOGRAPHY**, **TIMES**, **PROMOTIONS**, and **CHANNELS**), and one cube (**SALES**).

Figure 2–2 Entity Relationship Diagram for BISE1_SALESWH Schema

2.2.2.3 BISE1_TUTORIALWH Schema

The BISE1_TUTORIALWH schema contains no objects initially. Upon successful completion of [Chapter 3, "Set Up the Data Mart"](#), this schema will be populated with the tables that comprise the data mart star schema and be equivalent to the BISE1_SALESWH schema.

If you are skipping [Chapter 3, "Set Up the Data Mart"](#), before working on any of the other chapters, you will need to populate the BISE1_TUTORIALWH schema by performing these steps:

1. Open a command prompt window.
2. Navigate to the directory where you installed Oracle BI Standard Edition One.
3. Navigate to the `tutorial\owb` subdirectory.
4. Run the `reset_bise1_tutorialwh.bat` script.
5. Navigate to the `tutorial\bi_ad` subdirectory.
6. Run the `imp_bise1tutorialwh.bat` script. Note that the script assumes the password for BISE1_TUTORIALWH is **welcome1**. If you specified a different password, edit the `imp_bise1tutorialwh.bat` file and replace `welcome1` with the correct password.

2.2.3 Accounts Used in the Tutorial

Several accounts are created as part of the Oracle BI Standard Edition One installation. [Table 2–1](#) lists those accounts you will use directly as you work through the tutorial. It will be useful for you to determine the passwords for these accounts before you proceed with the tutorial.

Table 2–1 Accounts Used in the Tutorial

Account Name	Account Type	Password Set During Installation?	Description
BISE1_SALES	Database	Yes	This database account owns the schema that contains tables which store the data captured by GEC's order entry application. This schema serves as the source for both the prebuilt data mart (BISE1_SALESWH) as well as the data mart you will build (BISE1_TUTORIALWH).
BISE1_SALESWH	Database	Yes	This database account owns the schema that contains the prebuilt data mart objects.
BISE1_TUTORIALWH	Database	Yes	This database account owns the schema that will be used as the data mart target schema for Oracle Warehouse Builder. Once this schema is populated with the data mart objects, it is used by Oracle BI as the data source for the repository definition, and so on, as detailed in the tutorial.
OWBREPOS_OWNER	Database	Yes	This database account owns the pre-created Oracle Warehouse Builder repository.
OWBREPOS_USER	Database	Yes	This database account has the privilege to access the Oracle Warehouse Builder repository, and is used to connect to it through the Oracle Warehouse Builder Design Center tool.
OWF_MGR	Database	Yes	This is the Oracle Workflow manager database account. It is used in Oracle Warehouse Builder to submit the deployment process, which populates the data mart objects, such as the fact and dimension tables.
SYS	Database	Yes	This is the database "super user" account. It has access to all database objects, and can perform any task in the database.

Table 2–1 (Cont.) Accounts Used in the Tutorial

Account Name	Account Type	Password Set During Installation?	Description
Administrator	BI Server (also used to authenticate BI Presentation Services users)	No	This account is pre-created in the BI Server metadata repository. The password is <code>Administrator</code> (case-sensitive). It is used to open the default BI metadata repository (AnalyticsWeb) through the BI Administration tool. This same account and password is also used to login to BI Presentation Services (<code>http://localhost:9704/analytics</code>) to access the pre-created BI dashboard and Answers requests.
Administrator	BI Publisher	No	This account is pre-created in BI Publisher (stored in the <code>datasources.xml</code> file). The password is <code>Administrator</code> . This account is used to login to BI Publisher (<code>http://localhost:9704/xmlpserver</code>) and must be the same as the BI Presentation Services account for the predefined integration points between BI Publisher and BI Presentation Services to work.

2.2.4 Tutorial Files and Directory Structure

The tutorial files are located in a directory called **tutorial**, under the Oracle BI Standard Edition One installation home directory, which you specified at installation time. Under the **tutorial** directory, there is a file and several subdirectories:

- The **tutorial.pdf** file is this book, in Adobe PDF format.
- The three subdirectories (**owb**, **bi_pub**, and **bi_ad**) contain files that are required as you work through the tutorial. They also contain files that can be used to reset the tutorial exercise environment back to the starting point, or to a known state. See [Chapter 8, "Reset the Tutorial Exercise Environment"](#) for more information.

Set Up the Data Mart

In this chapter, you use Oracle Warehouse Builder (OWB) to create the logical and physical design of the data mart.

Prerequisites: There are no prerequisites for this chapter.

This chapter contains the following topics:

- [Section 3.1, "About Setting Up the Data Mart"](#)
- [Section 3.2, "Examining a Completed Warehouse Builder Project"](#)
- [Section 3.3, "Creating the Dimensions and the Cube"](#)
- [Section 3.4, "Creating the Mappings"](#)
- [Section 3.5, "Generating the Mappings"](#)
- [Section 3.6, "Designing a Process Flow"](#)
- [Section 3.7, "Validating and Generating the Process Flow"](#)
- [Section 3.8, "Deploying Mappings and Loading Data"](#)

3.1 About Setting Up the Data Mart

To set up the data mart, you use OWB components to:

1. Create the logical design for the data mart star schema.
2. Map the logical design to a physical design.
3. Generate code to create the objects for the data mart.
4. Create a process flow for populating the data mart.
5. Execute the process flow to populate the data mart.

In addition to a powerful graphical interface, OWB provides a metadata repository that holds detailed design information about your databases. The repository is implemented as a set of tables in an Oracle Database. The data you enter in the repository is available to any user who has at least read access to the repository application system.

To access the OWB repository, you use an OWB component called Design Center, the main OWB client in which you design sources, targets, ETL mapping and transformations. Using the OWB Design Center, you import data source definitions, define target structures, validate the structures, generate and deploy the code to implement the structures, and execute the process flows to run the ETL mappings that load data into the target structures. Once deployed, Warehouse Builder assists in the daily maintenance as well as monitoring of the deployed system.

3.2 Examining a Completed Warehouse Builder Project

Let us begin by taking a tour of the completed data warehouse project called GEC_DW (Global Electronics Corporation Data Warehouse). This is a data warehouse using a star schema, with one cube (SALES) and five dimensions (CHANNELS, GEOGRAPHY, PRODUCTS, PROMOTIONS, and TIME).

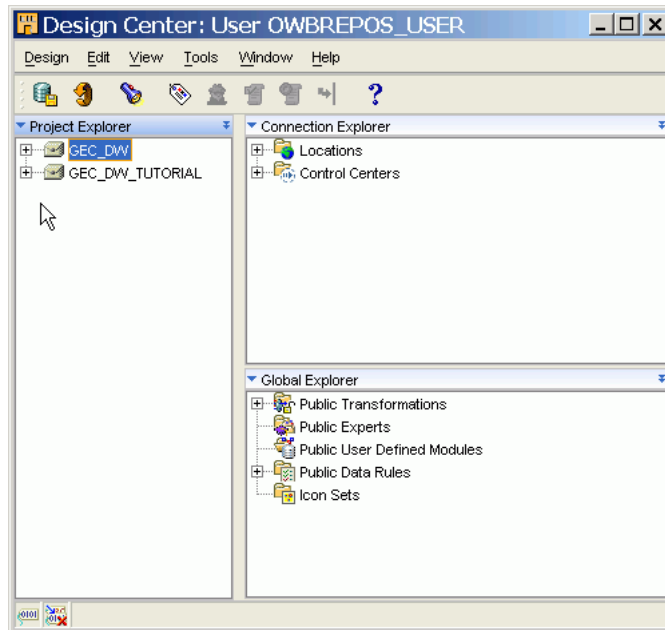
This section contains the following topics:

- [Section 3.2.1, "Access the OWB Repository Using the OWB Design Center"](#)
- [Section 3.2.2, "Explore the Completed OWB Project"](#)

3.2.1 Access the OWB Repository Using the OWB Design Center

To access the OWB repository using the OWB Design Center, follow these steps:

1. From the Windows Start menu, select **Programs > Oracle - BISE1Home1_ WarehouseBuilder > Warehouse Builder > Design Center** to invoke Warehouse Builder. The Design Center Logon window takes a few seconds to appear.
2. Before logging on with the username and password, you must establish connect information. Click **Show Details**.
3. The Design Center Logon window expands. Provide the host name, port number, and service name that were defined during installation:
 - **Host:** localhost
 - **Port:** 1521
 - **Service Name:** bise1db
4. In the top portion of the Design Center Logon window, enter the OWB repository user name and password.
 - **User Name:** owbrepos_user
 - **Password:** welcome1 (or, use whatever password you specified during BISE1 installation)
5. Click **OK** in the Design Center Logon window. The Design Center appears.

Figure 3–1 OWB Design Center

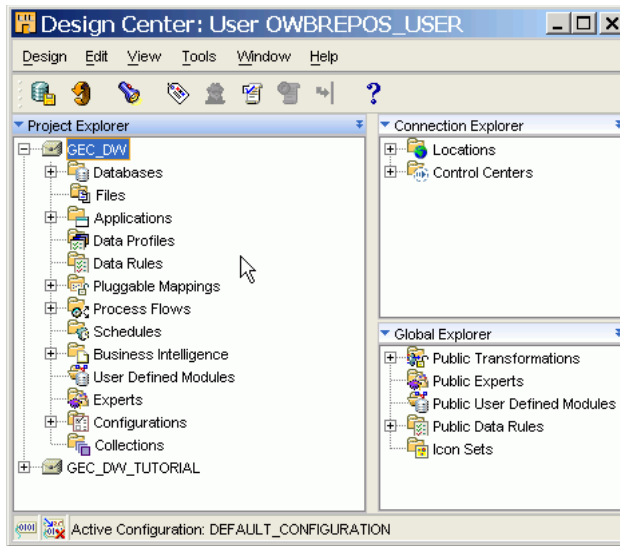
3.2.2 Explore the Completed OWB Project

The Design Center is divided into three windows: Project Explorer, Connection Explorer, and Global Explorer. In Project Explorer, you see two projects, **GEC_DW** (Global Electronics Corporation Data Warehouse), the completed project for your use as a reference, and **GEC_DW_TUTORIAL**, the project you will be modifying as you work through the guided steps in this tutorial.

To explore the completed OWB project:

1. Select and expand the **GEC_DW** project.
2. A project is a container to manage your design work. Warehouse Builder contains wizards, object editors, property sheets, and object finding tools that assist you in designing your business intelligence system. Under the **GEC_DW** project, various object types appear in the tree: Databases, Files, Applications, Data Profiles, and so on.

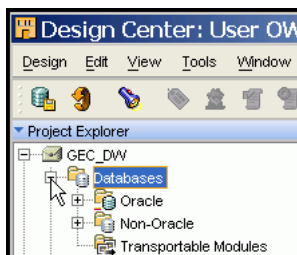
Figure 3–2 GEC_DW Project



In this tutorial, you will be working with the **Databases**, **Files**, and **Process Flows** object types.

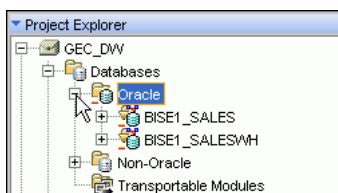
- Expand the **Databases** node. You can see various object types that can participate in your design: Oracle, Non-Oracle, and Transportable Modules.

Figure 3–3 Databases Node of GEC_DW Project



- Expand the **Oracle** node. You can see two modules defined in the GEC_DW project: **BISE1_SALESWH** and **BISE1_SALES**. Modules are logical groupings of source or target definitions. Source modules hold metadata describing source systems from which you extract data. Target modules hold metadata describing your target data warehouse or data marts. Every target module must be mapped to a target user schema. Make sure that the target module references this target schema by assigning an appropriate location.

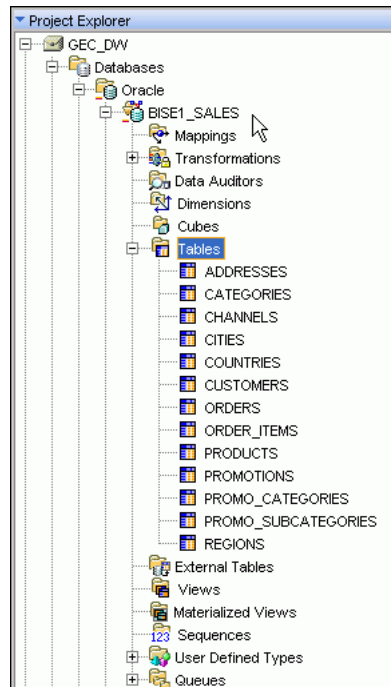
Figure 3–4 Databases > Oracle Node of the GEC_DW Project



- The **BISE1_SALES** module is the source module. Expand the **BISE1_SALES** module, then expand the **Tables** node. These are the operational data table

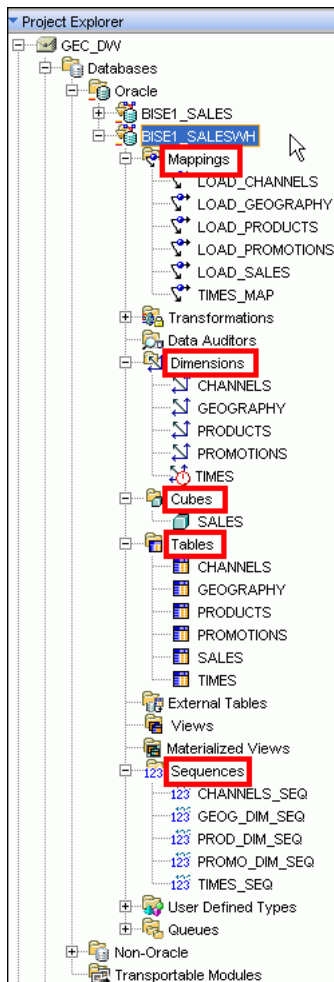
definitions used to construct the target data warehouse object definitions stored in the BISE1_SALESWH module.

Figure 3-5 BISE1_SALES > Tables Node of the GEC_DW Project

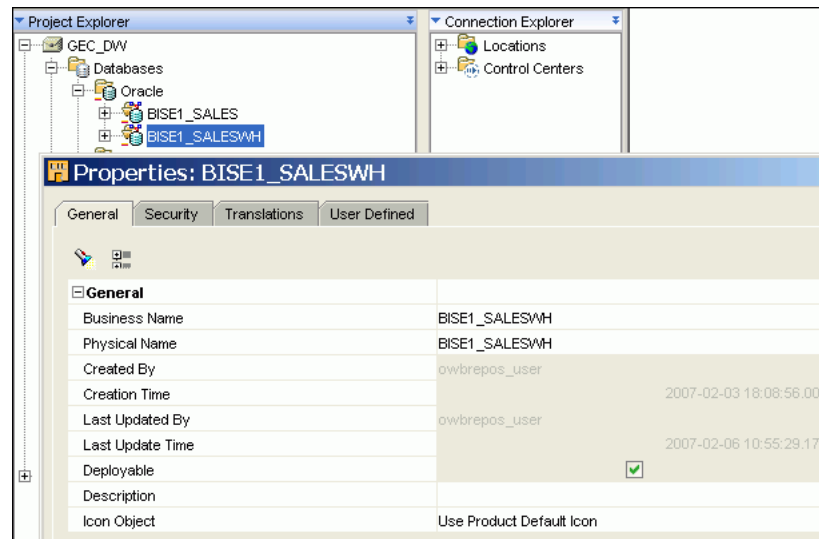


- The BISE1_SALESWH module is the target module. Expand the **BISE1_SALESWH** module, then expand the **Mappings**, **Dimensions**, **Cubes**, **Tables**, and **Sequences** nodes. These are the object definitions used to create and populate the data warehouse.

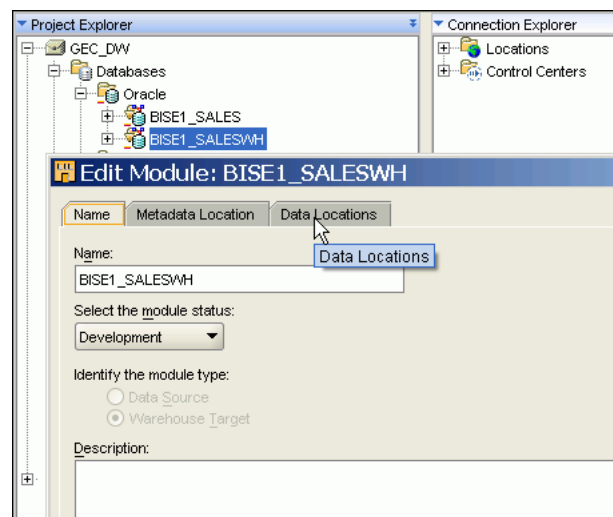
Figure 3-6 BISE1_SALESWH > Mappings Node of the GEC_DW Project



7. View the property settings for the BISE1_SALESWH module by right-clicking the module, then selecting **Properties** from the pop-up menu. Notice that objects have both a **Business Name** and a **Physical Name**, which can be different. Usually, you should synchronize them. Close the **Properties** window.

Figure 3–7 BISE1_SALESWH Properties

8. Double-click the **BISE1_SALESWH** module to invoke its Editor window. Almost all objects have both a Properties window, and an Editor window. You can set the object definitions and configurations using the Editor window. Close the Editor window.

Figure 3–8 BISE1_SALESWH Editor Window

Now that you have examined a completed data warehouse project, you are ready to complete your own data warehouse project called **GEC_DW_TUTORIAL**. The **GEC_DW_TUTORIAL** project will build the same data warehouse as the **GEC_DW** project. The only difference between these two projects is the data warehouse target. While the **GEC_DW** project is deployed to the **BISE1_SALESWH** target, you will deploy the **GEC_DW_TUTORIAL** project to the **BISE1_TUTORIALWH** target.

3.3 Creating the Dimensions and the Cube

In Warehouse Builder, you can define dimensions and cubes using a wizard or an editor. Use the wizard to create dimensional and cube objects easily. The wizard

creates a fully functional dimensional or cube object, including the implementation objects that store the dimensional or cube object data but with predefined defaults for certain features. Alternatively, you can use the editor to create or edit dimensional or cube objects. Use editors instead of wizards when you want to specify settings different from the default settings of the wizards, or to access additional settings not provided in the wizards.

This section contains the following topics:

- [Section 3.3.1, "Create the CHANNELS Dimension Using the Wizard"](#)
- [Section 3.3.2, "Create the PRODUCTS Dimension Using the Editor"](#)
- [Section 3.3.3, "Create the TIMES Dimension Using the Time Wizard"](#)
- [Section 3.3.4, "Create the SALES Cube Using the Editor"](#)

3.3.1 Create the CHANNELS Dimension Using the Wizard

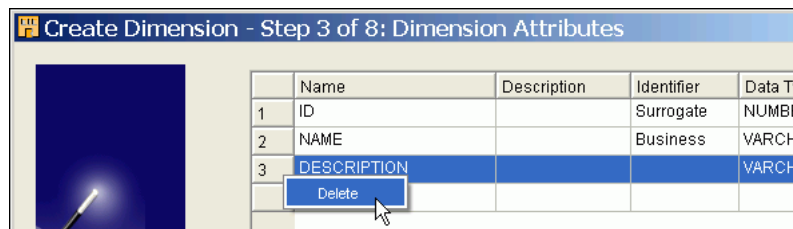
To create the CHANNELS dimension using the wizard:

1. In the Design Center Project Explorer, expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH**.
2. Right-click **Dimensions**, then select **New > Using Wizard**. The Create Dimension Wizard launches. Click **Next** on the Welcome page.
3. On the Name and Description page, enter the following:
 - **Name:** CHANNELS
 - **Description:** Channels Dimension
4. Click **Next**.
5. On the Storage Type page, choose **ROLAP: Relational storage** and click **Next**.

Note: The ROLAP storage type is best if you have high volumes of data combined with high refresh rates, or if you have detailed high-volume data. When you store data in a relational form, OWB by default uses the star schema implementation method.

6. On the Dimension Attributes page, you will find three predefined columns: **ID**, **NAME**, and **DESCRIPTION**.
7. Delete the **DESCRIPTION** column. To do this, right-click the number **3**, in the column to the left of **DESCRIPTION**, and select **Delete**, as shown in [Figure 3–9](#).

Figure 3–9 Dimension Attributes Page: Deleting the Description Column



8. For the **Name** attribute, select empty value from the **Identifier** drop-down list, and change Length to **60**.

Figure 3–10 Dimension Attributes Page: Changing the Length to 60

	Name	Description	Identifier	Data Type	Length
1	ID		Surrogate	NUMBER	
2	NAME			VARCHAR2	60

9. Add a new attribute, **SOURCE_ID** (Business, Varchar2, 40), as shown in [Figure 3–11](#), and click Next.

Figure 3–11 Dimension Attributes Page: Adding the SOURCE_ID Attribute

	Name	Description	Identifier	Data Type	Length
1	ID		Surrogate	NUMBER	
2	NAME			VARCHAR2	60
3	SOURCE_ID		Business	VARCHAR2	40

10. On the Levels page, you create the three levels as shown in [Figure 3–12](#).

Figure 3–12 Levels Page

Create Dimension - Step 4 of 8: Levels		
Specify the levels in the default hierarchy:		
	Name	Description
1	TOTAL	Total
2	CLASS	Class
3	CHANNEL	Channel

11. Click Next.
12. On the Level Attributes page, specify the attributes for each level, as follows:
 - For the TOTAL level, select attributes as shown in [Figure 3–13](#).

Figure 3–13 Attributes for TOTAL Level

Create Dimension - Step 5 of 8: Level Attributes			
Levels:			
	Name	Description	
1	TOTAL	Total	
2	CLASS	Class	
3	CHANNEL	Channel	

Level Attributes for TOTAL:			
	Dimension Attribute Na...	Applicable	Level Attribute Name
1	ID	<input checked="" type="checkbox"/>	ID
2	NAME	<input checked="" type="checkbox"/>	NAME
3	SOURCE_ID	<input checked="" type="checkbox"/>	SOURCE_ID

- For the CLASS level, select attributes as shown in [Figure 3–14](#).

Figure 3–14 Attributes for CLASS Level

Levels:		
	Name	Description
1	TOTAL	Total
2	CLASS	Class
3	CHANNEL	Channel

Level Attributes for CLASS:			
	Dimension Attribute Na...	Applicable	Level Attribute Name
1	ID	<input checked="" type="checkbox"/>	ID
2	NAME	<input checked="" type="checkbox"/>	NAME
3	SOURCE_ID	<input checked="" type="checkbox"/>	SOURCE_ID

- For the CHANNEL level, select attributes as shown in [Figure 3–15](#).

Figure 3–15 Attributes for CHANNEL Level

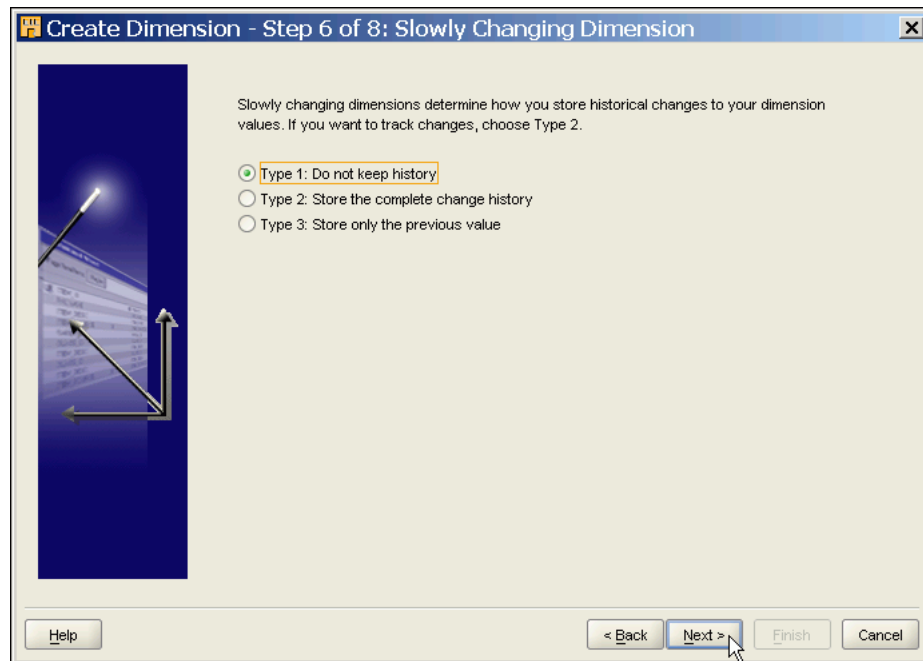
Levels:		
	Name	Description
1	TOTAL	Total
2	CLASS	Class
3	CHANNEL	Channel

Level Attributes for CHANNEL:			
	Dimension Attribute Na...	Applicable	Level Attribute Name
1	ID	<input checked="" type="checkbox"/>	ID
2	NAME	<input checked="" type="checkbox"/>	NAME
3	SOURCE_ID	<input checked="" type="checkbox"/>	SOURCE_ID

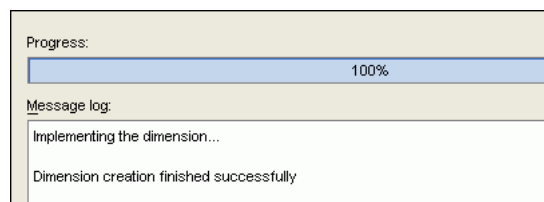
Note: For the lowest level, all the attributes are selected by default.

13. Click **Next**.
14. On the Data Policy page, accept the default choice of **Type 1: Do not keep history**, then click **Next**.

Note: A slowly changing dimension (SCD) is a dimension that stores and manages both current and historical data over time in a data warehouse. Examples of SCDs include product package sizes, package types, prices, and country names. If you need to track the evolutionary history of the data values, you can choose Type 2 or 3, which require a separate license of the Oracle Warehouse Builder Enterprise ETL Option.

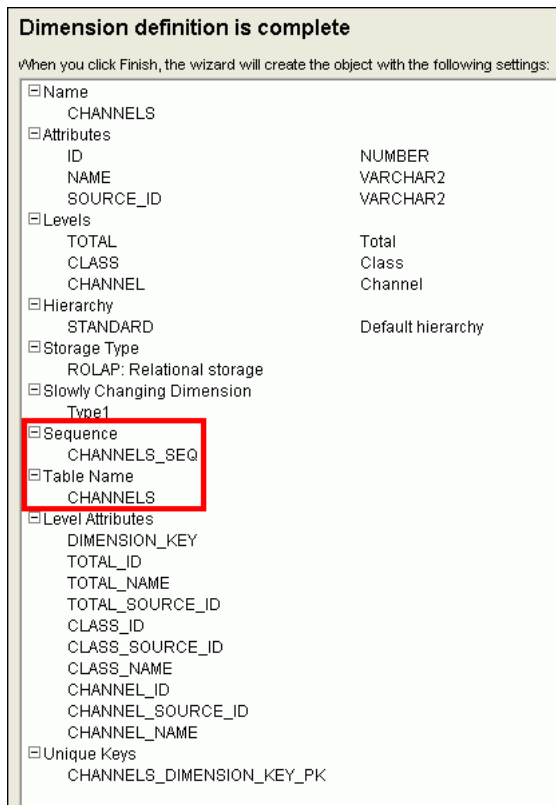
Figure 3–16 Data Policy Page

15. Examine the Pre Create Settings page and click **Next**.
16. The Dimension Creation Progress implements the dimension, and the dimension is successfully created. Click **Next**.

Figure 3–17 Dimension Creation Progress

17. The Summary page lists the dimension definition. Notice a database table and sequence will be created as well. Click **Finish**.

Figure 3–18 Summary Page



18. Save your work by clicking the **Save All** icon on the Design Center toolbar.

Figure 3–19 Save All Icon



19. Click **Yes** in the Warehouse Builder Warning dialog box.

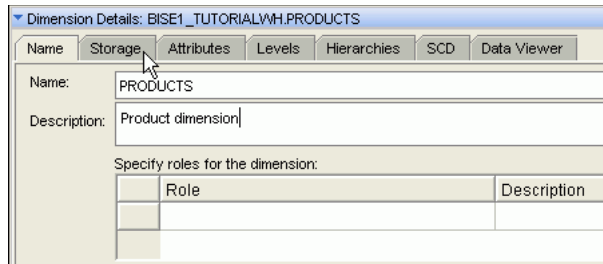
Note: The warning dialog box appears every time you click **Save All**.

3.3.2 Create the PRODUCTS Dimension Using the Editor

To create the PRODUCTS dimension using the editor:

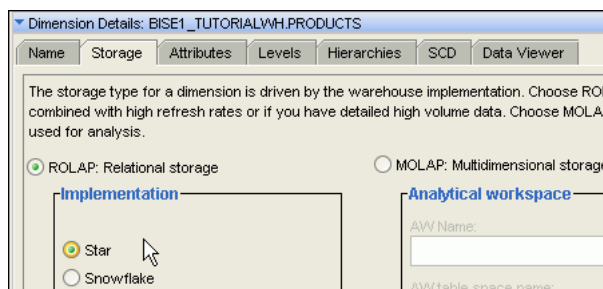
1. In the Design Center Project Explorer, expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH**, if it is not already expanded.
2. Right-click **Dimensions**, then select **New > Using Editor**. The Data Object Editor launches.
3. In the Dimension Details window, ensure that the **Name** tab is selected. In the **Name** field, change **DIMENSION_1** to **PRODUCTS**. In the **Description** field, enter **Product dimension**.

Figure 3–20 Dimension Details: Name Tab



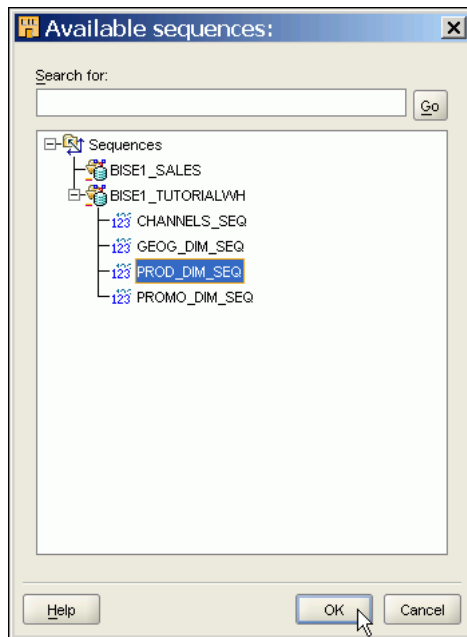
4. Click the **Storage** tab. Accept the default option of **ROLAP: Relational storage**. For **Implementation**, choose **Star**.

Figure 3–21 Dimension Details: Storage Tab



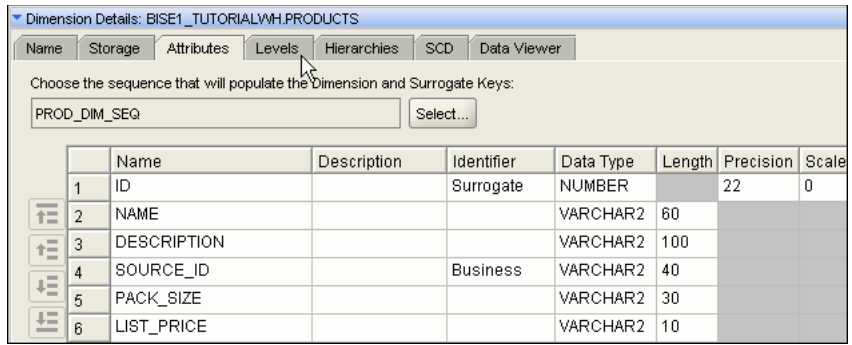
5. Click the **Attributes** tab. Click **Select**. In the Available Sequence dialog box, expand **BISE1_TUTORIALWH** and select **PROD_DIM_SEQ**. Click **OK**.

Figure 3–22 Available Sequences Dialog Box



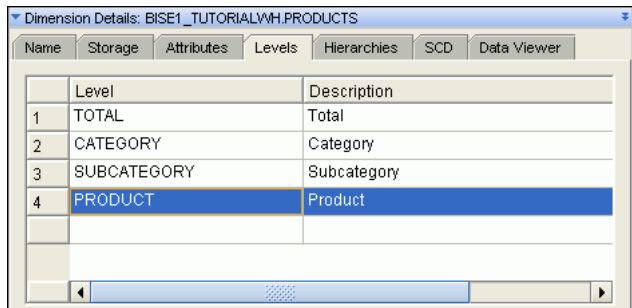
6. The sequence will populate the Dimension Key. Enter attribute information as shown in [Figure 3–23](#).

Figure 3–23 Dimension Details: Attributes Tab



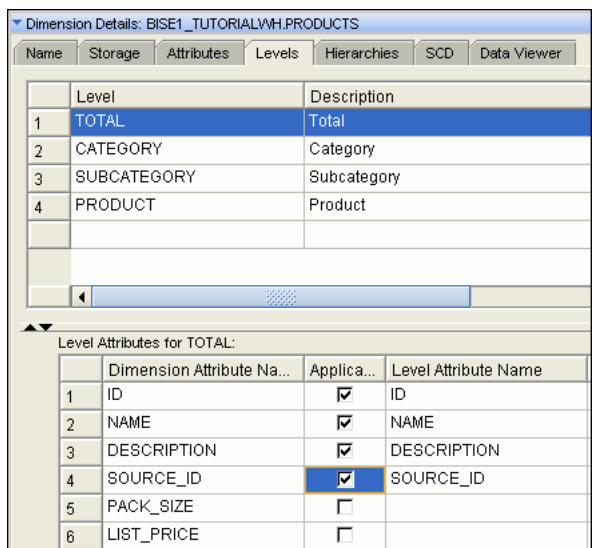
7. Click the **Levels** tab. This tab has two sections: Levels and Level Attributes for *level_name*.
 - a. In the **Levels** section, enter information as shown in [Figure 3–24](#).

Figure 3–24 Dimension Details: Levels Section of Levels Tab



- b. In the Level Attributes for each level, specify the following:
 - Set level attributes for **TOTAL** as shown in [Figure 3–25](#).

Figure 3–25 Dimension Details: Level Attributes for TOTAL



- Set level attributes for **CATEGORY** as shown in [Figure 3–26](#).

Figure 3–26 Dimension Details: Level Attributes for CATEGORY

Dimension Details: BISE1_TUTORIALVH.PRODUCTS			
Name		Storage	Attributes
	Level		Description
1	TOTAL		Total
2	CATEGORY		Category
3	SUBCATEGORY		Subcategory
4	PRODUCT		Product
Level Attributes for CATEGORY:			
	Dimension Attribute Na...	Applica...	Level Attribute Name
1	ID	<input checked="" type="checkbox"/>	ID
2	NAME	<input checked="" type="checkbox"/>	NAME
3	DESCRIPTION	<input checked="" type="checkbox"/>	DESCRIPTION
4	SOURCE_ID	<input checked="" type="checkbox"/>	SOURCE_ID
5	PACK_SIZE	<input type="checkbox"/>	
6	LIST_PRICE	<input type="checkbox"/>	

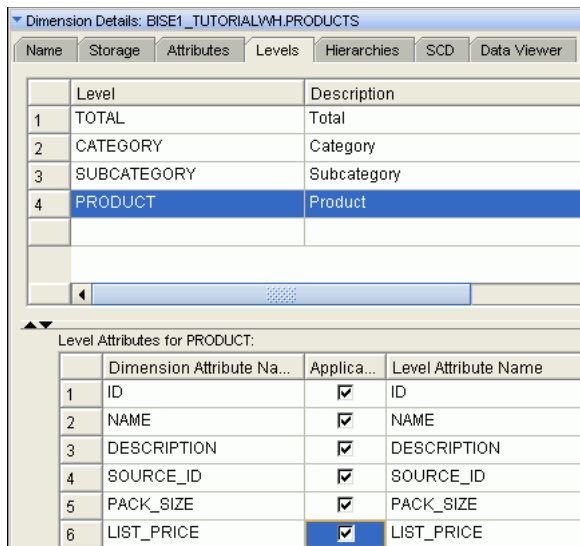
- Set level attributes for **SUBCATEGORY** as shown in [Figure 3–27](#).

Figure 3–27 Dimension Details: Level Attributes for SUBCATEGORY

Dimension Details: BISE1_TUTORIALVH.PRODUCTS			
Name		Storage	Attributes
	Level		Description
1	TOTAL		Total
2	CATEGORY		Category
3	SUBCATEGORY		Subcategory
4	PRODUCT		Product
Level Attributes for SUBCATEGORY:			
	Dimension Attribute Na...	Applica...	Level Attribute Name
1	ID	<input checked="" type="checkbox"/>	ID
2	NAME	<input checked="" type="checkbox"/>	NAME
3	DESCRIPTION	<input checked="" type="checkbox"/>	DESCRIPTION
4	SOURCE_ID	<input checked="" type="checkbox"/>	SOURCE_ID
5	PACK_SIZE	<input type="checkbox"/>	
6	LIST_PRICE	<input type="checkbox"/>	

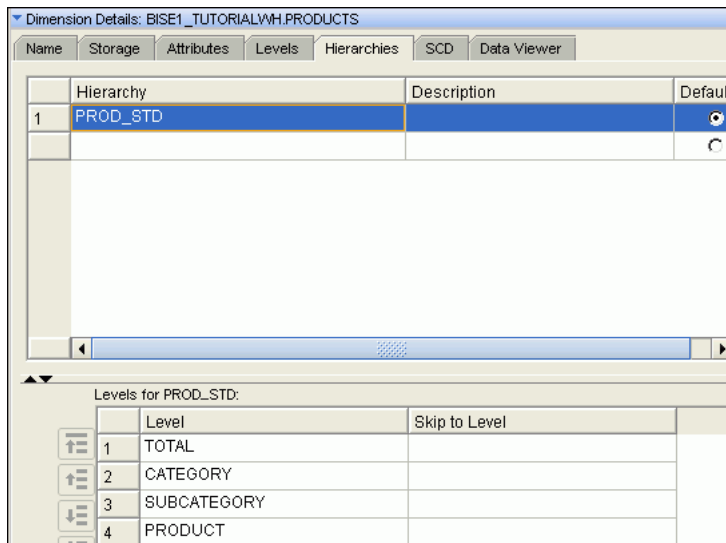
- Set level attributes for **PRODUCT** as shown in [Figure 3–28](#).

Figure 3–28 Dimension Details: Level Attributes for PRODUCT

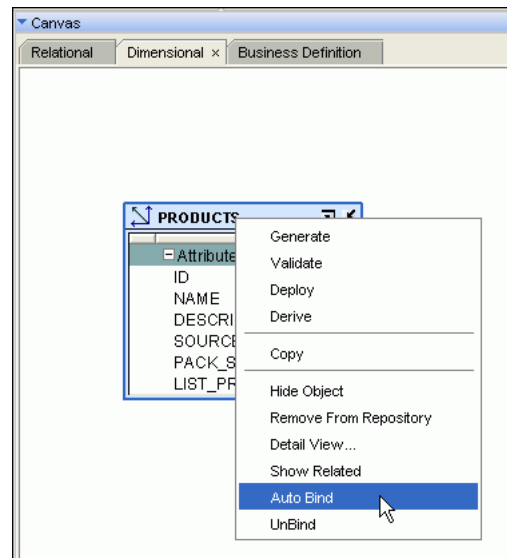


- Click the **Hierarchies** tab. Select Level values from the drop-down list as shown in [Figure 3–29](#).

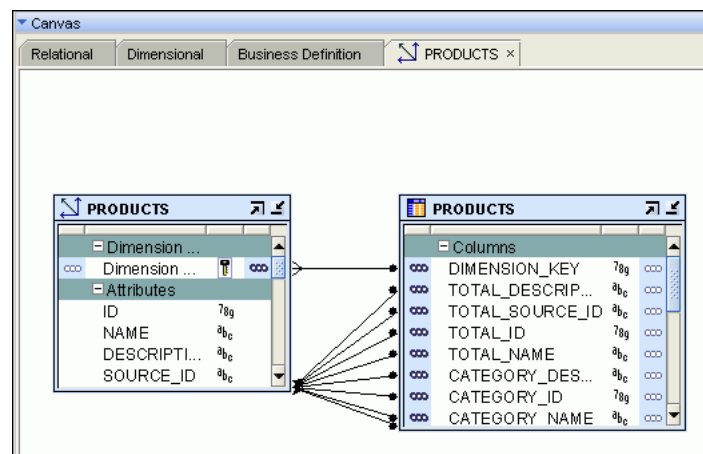
Figure 3–29 Dimension Details: Hierarchies Tab



- Click the **SCD** tab. Note the default selection, **Type 1: Do not keep history**.
- Create the dimension table and bind it to the repository. On the canvas, right-click the **PRODUCTS** dimension and select **Auto Bind**. When you perform auto binding, Warehouse Builder automatically maps the measures and dimension references of the cube to the database columns that store their data.

Figure 3–30 Auto Binding

11. You have now created the **PRODUCTS** dimension. [Figure 3–31](#) shows what you will see after you complete the binding process.

Figure 3–31 PRODUCTS Dimension

12. Save your work by clicking **Save All** on the Data Object Editor toolbar.
13. Close the Data Object Editor.

3.3.3 Create the TIMES Dimension Using the Time Wizard

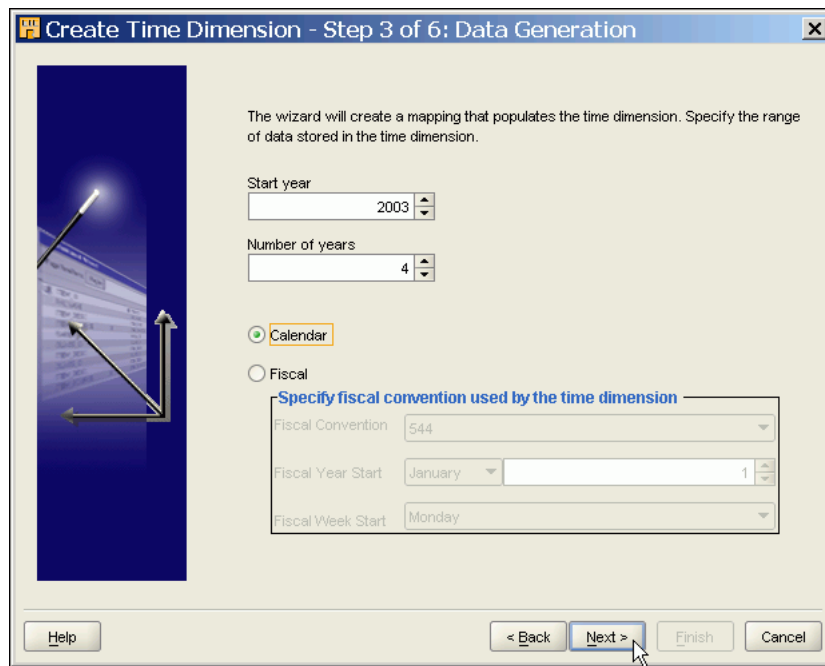
To create the TIMES dimension using the Time Wizard:

1. In Project Explorer in the Design Center, expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH**. Right-click **Dimensions**, then select **New > Using Time Wizard**. The Create Time Dimension Wizard launches. Click **Next** on the Welcome page.
2. On the Name and Description page, enter the following:
 - **Name:** TIMES
 - **Description:** Times Dimension

3. Click **Next**.
4. On the Storage Type page, accept the default **ROLAP: Relational storage** and click **Next**. The Data Generation page appears.
5. Enter the following information:
 - **Start year:** 2003
 - **Number of years:** 4
 - Ensure that the **Calendar** option is selected.

Note: On the Data Generation page, you specify the range of time data that is required for your warehouse. This information will be used to generate a mapping that will populate the time dimension. Within this mapping, the dates you enter are added as parameters, allowing you to rerun this mapping with dates at a later stage.

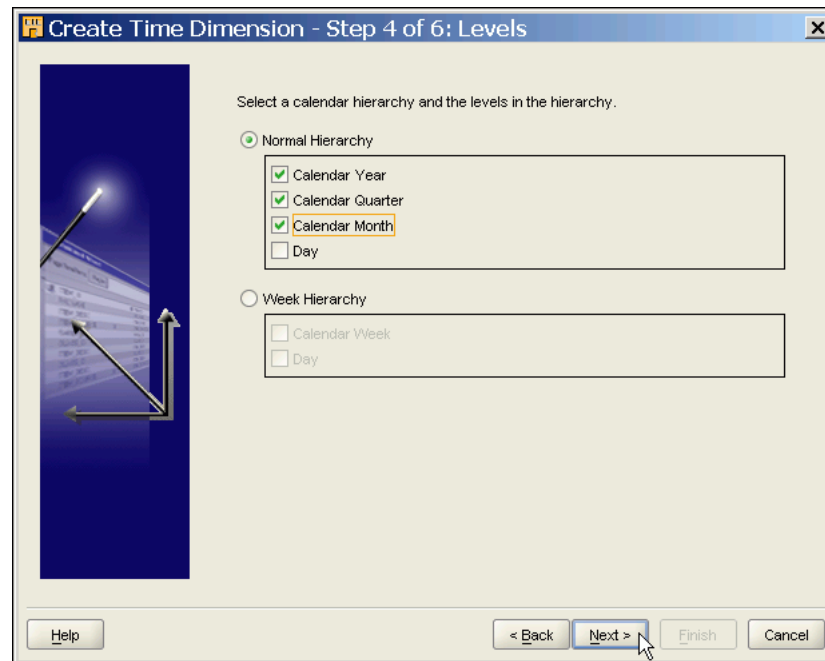
Figure 3–32 Data Generation Page



Click **Next**.

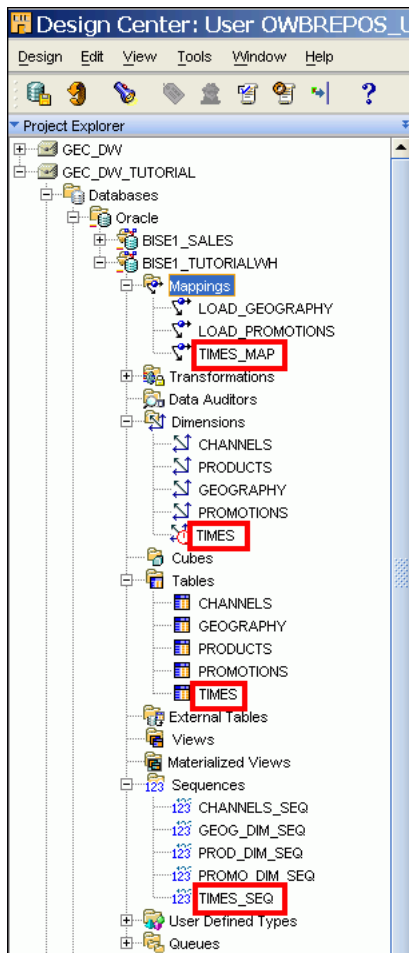
6. On the Levels page, select **Normal Hierarchy**, and select **Calendar Year**, **Calendar Quarter**, and **Calendar Month** as shown in the [Figure 3–33](#), then click **Next**.

Figure 3-33 Levels Page



7. On the Pre Create Settings page, examine the details and click **Next**. A Progress bar will show the progress as the wizard creates the objects.
8. On successful completion, click **Next**. On the Summary page, examine the details again and click **Finish**.
9. In Project Explorer, observe that the wizard has generated four objects necessary for a fully functional TIMES dimension:
 - a. Expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH > Dimensions**. You can see the **TIMES** dimension.
 - b. Expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH > Sequences**. You can see the **TIMES_SEQ** sequence that populates the surrogate ID of the time dimension levels.
 - c. Expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH > Tables**. You can see the **TIMES** table, which supports the relational implementation of the time dimension that will physically store the time data.
 - d. Expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH > Mappings**. You can see the **TIMES_MAP** mapping, which populates the time dimension.

Figure 3–34 Project Explorer: TIMES Dimension, TIMES_SEQ, TIMES Table, and TIMES_MAP



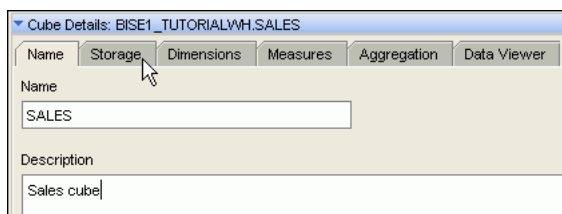
10. Click **Save All** to save your work.

3.3.4 Create the SALES Cube Using the Editor

To create the SALES cube using the editor:

1. From the Project Explorer in the Design Center, expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH**. Right-click **Cubes**, then select **New > Using Editor**. The Data Object Editor window is displayed.
2. In the Cube Details window, ensure that the **Name** tab is selected. In the **Name** field, change **CUBE_1** to **SALES**. In the Description field, enter **Sales cube**.

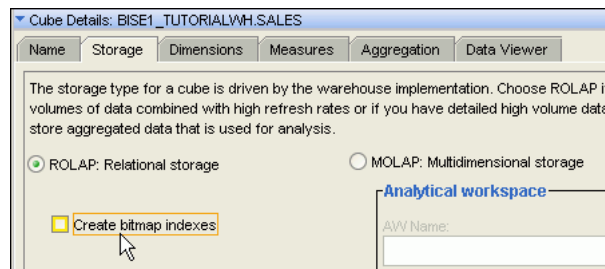
Figure 3–35 Cube Details: Name Tab



3. Click the **Storage** tab. Choose settings as follows:
 - Accept the default selection, **ROLAP: Relational data structures**. Using this option, the cube definition and its data are stored in the relational form in the database.
 - Deselect **Create bitmap indexes**.

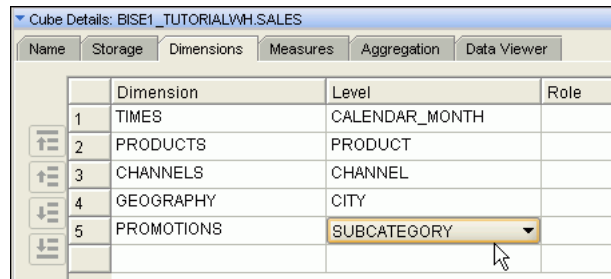
Note: Bitmap indexes can enhance performance. To make use of bitmap indexes, a separate license for Oracle Database Enterprise Edition is required.

Figure 3–36 Cube Details: Storage Tab



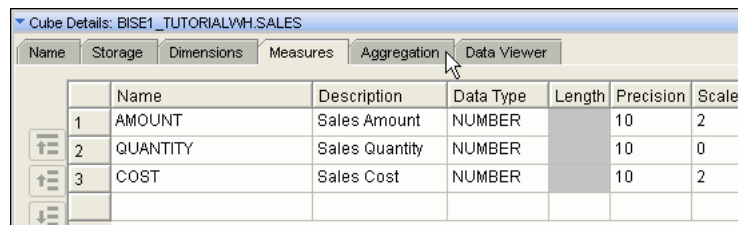
4. Click the **Dimensions** tab. Select the dimension from the drop-down list in the dimension column and the corresponding levels in the Level column, as shown in [Figure 3–37](#).

Figure 3–37 Cube Details: Dimensions Tab



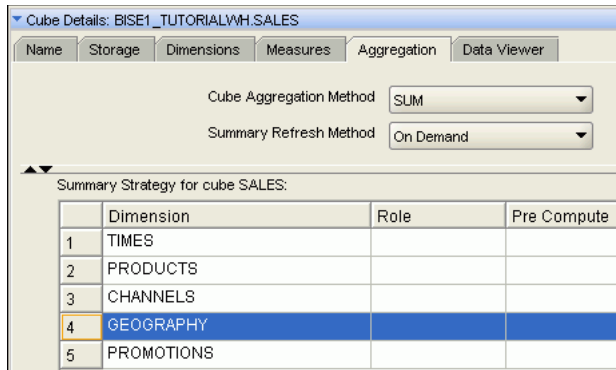
5. Click the **Measures** tab and enter the information as shown in [Figure 3–38](#).

Figure 3–38 Cube Details: Measures Tab



6. Click the **Aggregation** tab. The default Aggregation function **SUM** is already selected for all the dimensions. This is used to aggregate the cube data, and you can accept this default setting. As an example, [Figure 3–39](#) shows that the Cube Aggregation Method is SUM for the GEOGRAPHY dimension.

Figure 3–39 Cube Details: Aggregation Tab

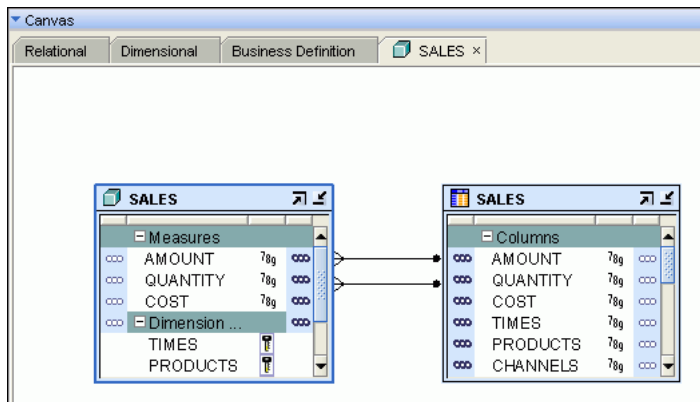


After defining the cube structure, you will now specify the details of the database tables or views that store the cube data.

7. On the canvas, in the Data Object Editor, right-click the **SALES** cube and select **Auto Bind**. When you perform auto binding, Warehouse Builder automatically maps the measures and dimension references of the cube to the database columns that store their data.

Figure 3–40 shows what you will see on the canvas after selecting Auto Bind.

Figure 3–40 Sales Cube



8. Save your work and close the Data Object Editor.

3.4 Creating the Mappings

In this section, you will perform the following tasks:

- Design the mapping to load the PRODUCTS dimension. Note that the GEOGRAPHY and PROMOTIONS mappings have been pre-created for you.
- Run the tcl script to create the mappings to load the CHANNELS dimension and SALES cube.
- The TIMES mapping was automatically created when you created the TIMES dimension using the time dimension wizard.

This section contains the following topics:

- [Section 3.4.1, "Design the Mapping to Load the PRODUCTS Dimension"](#)

- Section 3.4.2, "Use Tcl Scripts to Create Mappings for the CHANNELS Dimension and the SALES Cube"

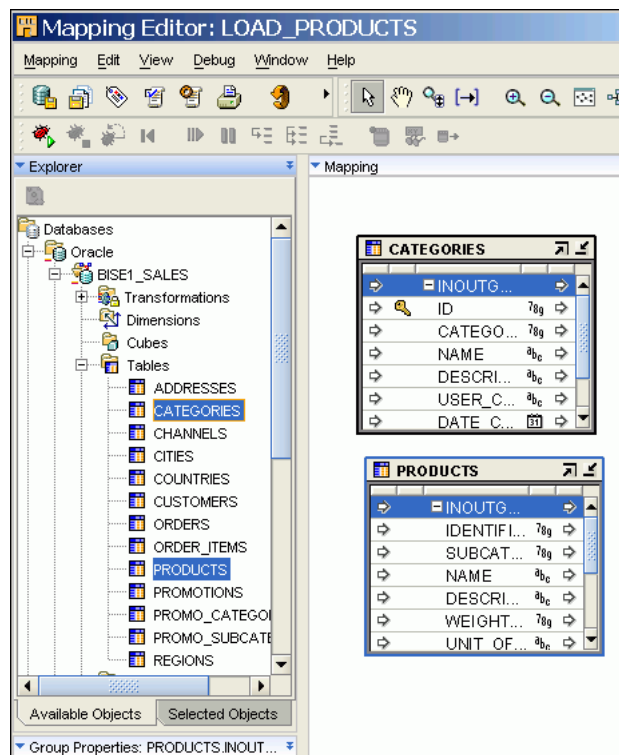
3.4.1 Design the Mapping to Load the PRODUCTS Dimension

To design the mapping to load the PRODUCTS dimension:

1. From the Project Explorer in the Design Center, expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH**. Right-click the **Mappings** node and select **New**. The Create Mapping window is displayed.
2. In the Create Mapping window, enter **LOAD_PRODUCTS** as the name of the mapping. Click **OK**. The Mapping Editor appears.
3. In the Explorer panel, ensure that the **Available Objects** tab is selected. Expand **Databases > Oracle > BISE1_SALES > Tables**. Drag the **CATEGORIES** and the **PRODUCTS** tables to the canvas.

Note: If you dragged both tables simultaneously to the canvas, you may initially see only one table displayed in the Mapping canvas. The second table is likely hidden behind the table you see. To see the second table, click and drag the visible table slightly to the top.

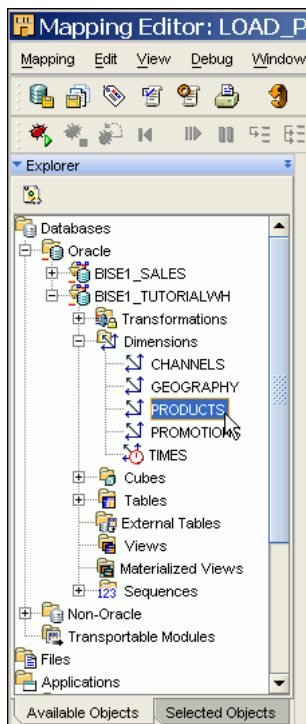
Figure 3–41 Mapping Editor: CATEGORIES and PRODUCTS



4. On the canvas, double-click the header area of the **PRODUCTS** table.
5. In the Table Editor, ensure the **Name** tab is selected. In the **Name** field, change **PRODUCTS** to **PRODUCTS_IN**. Click **OK**.
6. From the Explorer panel, collapse the **BISE1_SALES** module and expand **BISE1_TUTORIALWH > Dimension**.

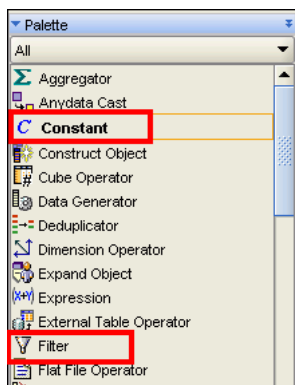
7. Drag the **PRODUCTS** dimension to the canvas.

Figure 3–42 Mapping Editor: PRODUCTS



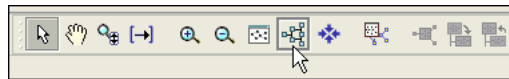
8. Double-click the header area of the **PRODUCTS** dimension. In the Table Editor, in the **Name** field, change **PRODUCTS** to **PRODUCTS_OUT** and click **OK**.
9. Add the **CONSTANT** and **FILTER** operators to the canvas. From the Palette, drag the Constant operator to the canvas. You need two filters, so drag the Filter operator to the canvas twice.

Figure 3–43 Mapping Editor: Constant and Filter



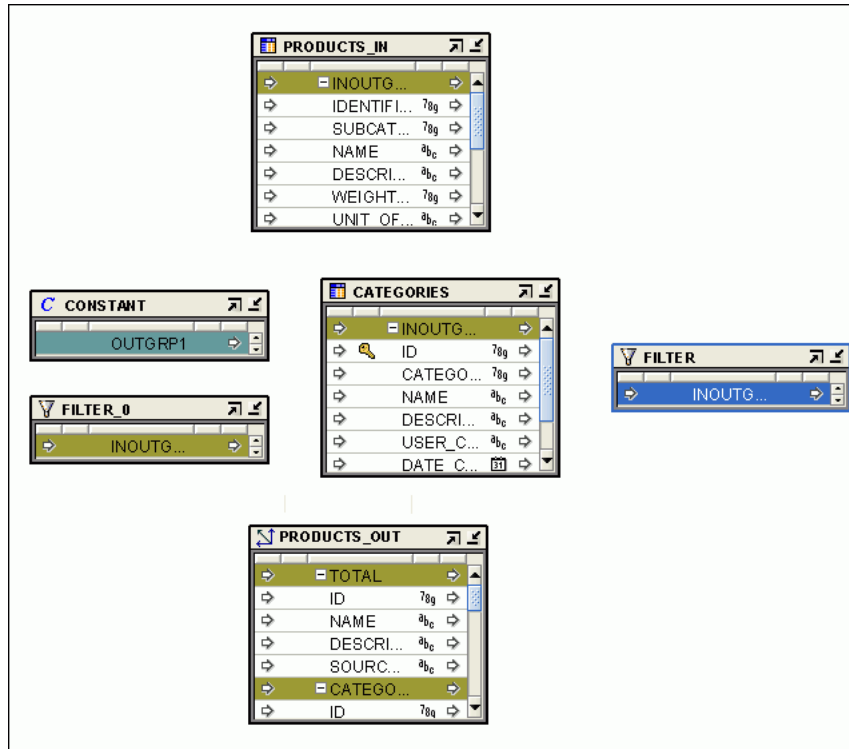
10. Click the **Auto Layout** icon on the Mapping Editor toolbar to arrange the objects on the canvas.

Figure 3–44 Mapping Editor Toolbar: Auto Layout Icon



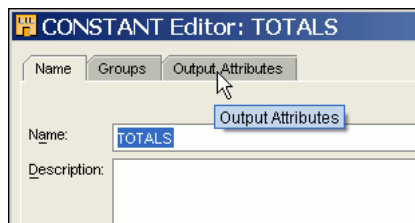
11. Your canvas now appears as shown in [Figure 3–45](#).

Figure 3–45 Canvas After Choosing Auto Layout



12. Double-click the **CONSTANT** operator’s header and change its name to **TOTALS**. Select the **Output Attributes** tab in the **CONSTANT** Editor.

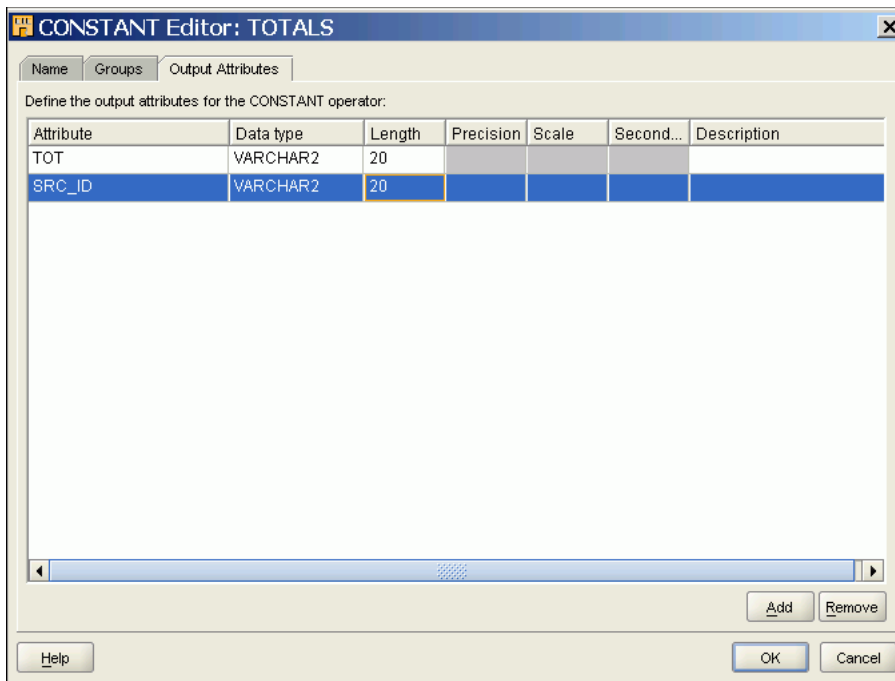
Figure 3–46 CONSTANT Editor: TOTALS



13. Click the **Add** button to add the following two attributes:

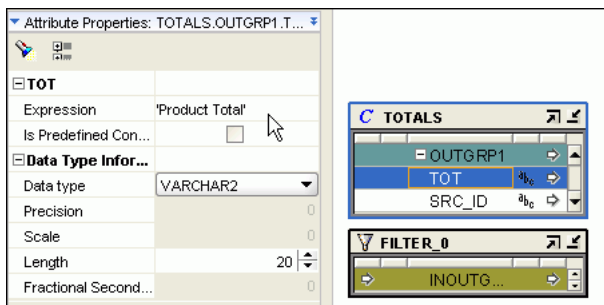
- **TOT - VARCHAR2(20)**
- **SRC_ID – VARCHAR2(20)**

Figure 3–47 CONSTANT Editor: Output Attributes



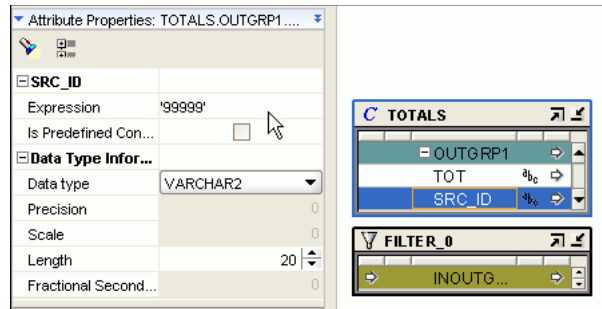
14. Click **OK** to close the CONSTANT Editor window.
15. On the Canvas, click the **TOTALS** constant operator and select the **TOT** attribute. In the Property Inspector window on the left, click the field next to **Expression** and enter **'Product Total'**.

Figure 3–48 TOT Attribute



16. On the Canvas, with the **TOTALS** constant operator selected, click the **SRC_ID** attribute. In the Property Inspector window, click the field next to **Expression** and enter **'99999'**.

Figure 3–49 SRC_ID Attribute



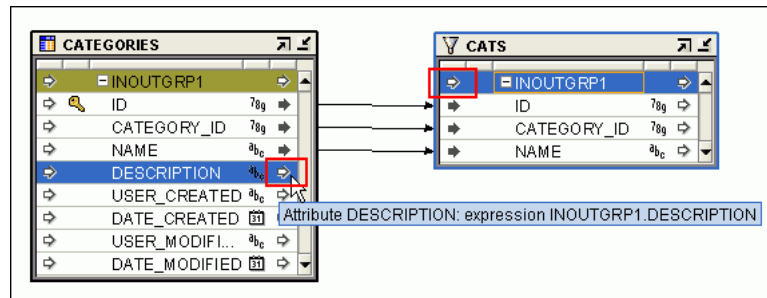
17. Double-click the **FILTER** operator header on the canvas. Change the name **FILTER** to **CATS**. Click **OK** to close the Filter Editor.
18. Maximize the **CATEGORIES** operator so that all the attributes are clearly visible.
19. Add a connection line from attributes in operator **CATEGORIES.INOUTGRP1** to **INOUTGRP1** of operator **CATS**, as shown in [Table 3–1](#).

Table 3–1 CATEGORIES.INOUTGRP1 to CATS Mapping

CATEGORIES.INOUTGRP1	CATS
ID	INOUTGRP1
NAME	INOUTGRP1
DESCRIPTION	INOUTGRP1
CATEGORY_ID	INOUTGRP1

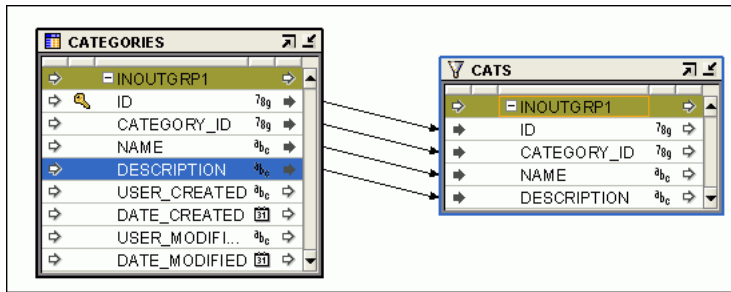
To draw the connection line, click the right arrow for each **CATEGORIES** attribute, hold down the mouse button, and drag the line to the left arrow of **CATS.INOUTGRP1**.

Figure 3–50 Connecting CATEGORIES.INOUTGRP1 to CATS



[Figure 3–51](#) shows what you should see when you are done adding the connections.

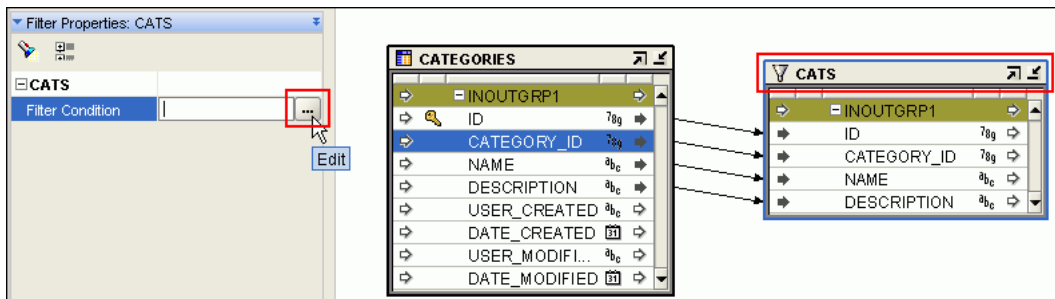
Figure 3–51 Completed Connections Between CATEGORIES.INOUTGRP1 and CATS



- Specify the filter condition for the CATS filter operator as: **INOUTGRP1.CATEGORY_ID IS NULL.**

To do this, click the header area of the CATS filter operator on the canvas. On the left-hand side, in the Filter Properties: CATS window, click the field next to **Filter Condition**. Click the ellipsis (...). The Expression Builder for the filter condition is launched. OWB enables you to build the condition, or lets you enter the filter condition.

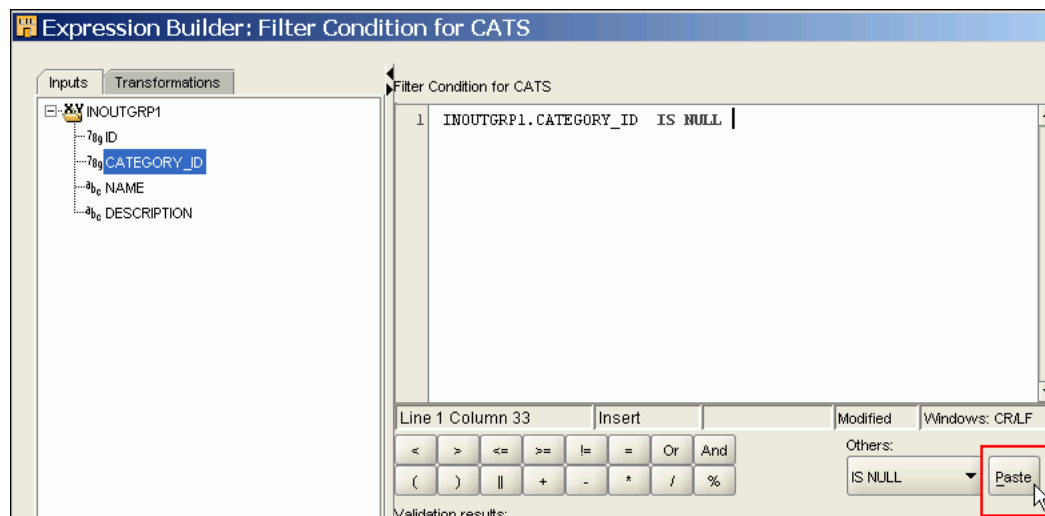
Figure 3–52 Specifying the Filter Condition



- In the Expression builder on the left-hand side, expand **INOUTGRP1**. Double-click **CATEGORY_ID**. You see that **INOUTGRP1.CATEGORY_ID** is pasted in the Filter condition field on the right-hand side.

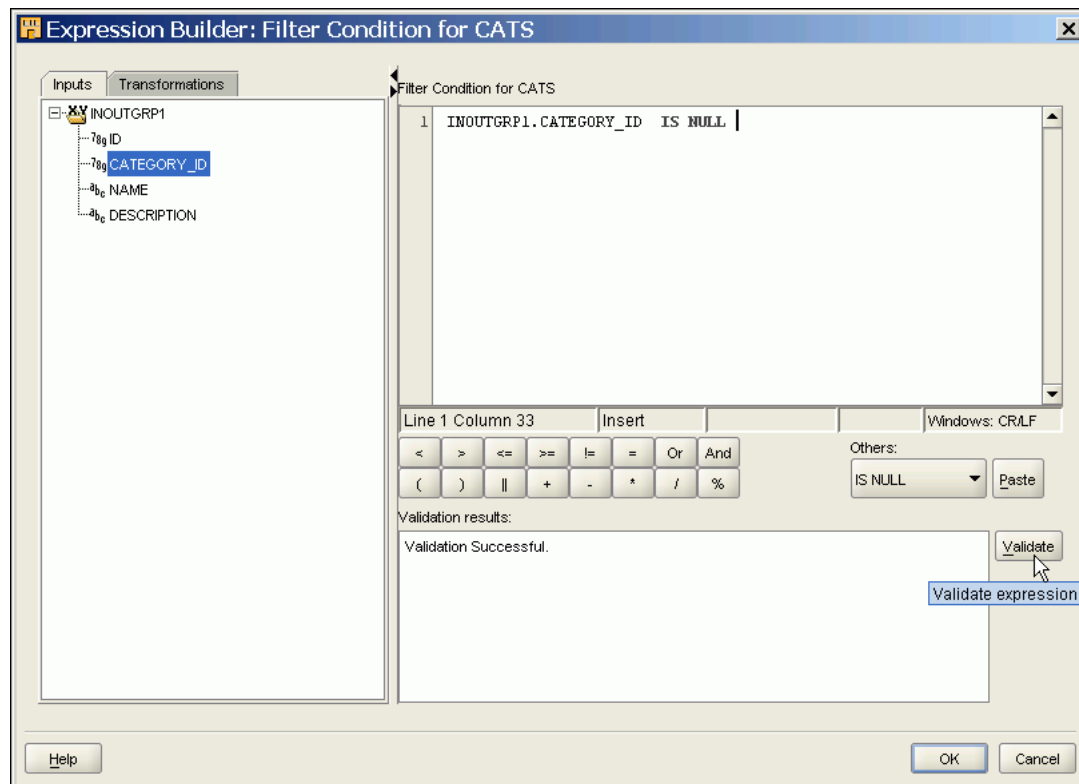
In the lower right area, from the **Others** drop-down list, select **IS NULL** and click **Paste**. You see the expression **INOUTGRP1.CATEGORY_ID IS NULL** in the **Filter Condition** for CATS field.

Figure 3–53 Filter Condition for CATS: Pasting IS NULL



22. Click **Validate**. You will see the result in the **Validation results** field. If the validation is successful, click **OK**.

Figure 3–54 Validating the Expression



23. Change the other **FILTER** operator to **SUBCATS**. Double-click the **FILTER** operator header on the canvas. Change the name **FILTER0** to **SUBCATS**. Click **OK** to close the Filter Editor.
24. Maximize the **CATEGORIES** operator so that all the attributes are clearly visible.

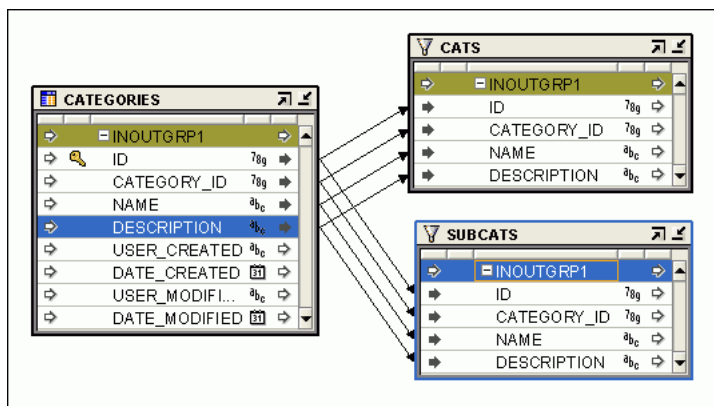
25. Add a connection line from the attributes in operator **CATEGORIES.INOUTGRP1** to **INOUTGRP1** of operator **SUBCATS** as shown in [Table 3–2](#).

Table 3–2 CATEGORIES.INOUTGRP1 to SUBCATS Mapping

CATEGORIES.INOUTGRP1	SUBCATS
ID	INOUTGRP1
NAME	INOUTGRP1
DESCRIPTION	INOUTGRP1
CATEGORY_ID	INOUTGRP1

[Figure 3–55](#) shows what you should see when you are done adding the connections.

Figure 3–55 Completed Connections Between CATEGORIES.INOUTGRP1 and SUBCATS



26. Specify the filter condition for the **SUBCATS** filter operator as **INOUTGRP1.CATEGORY_ID IS NOT NULL**.

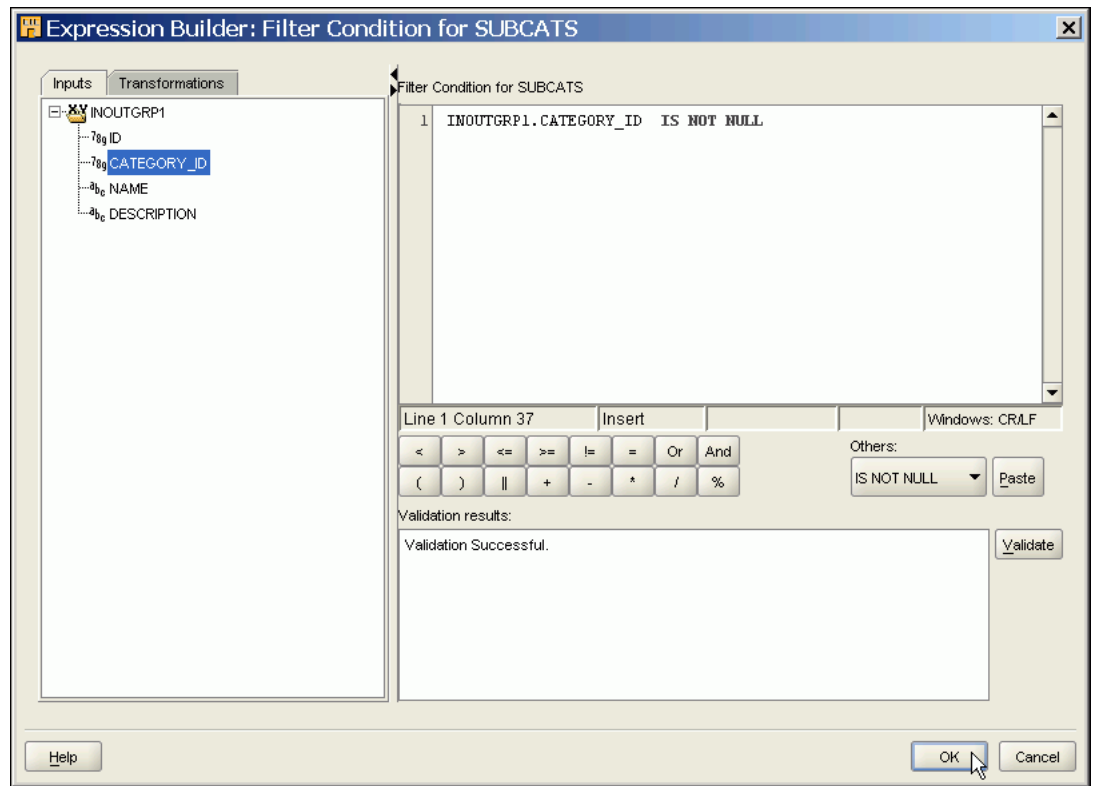
To do this, click the header area of the **SUBCATS** filter operator on the canvas. On the left-hand side, in the Filter Properties: SUBCATS window, click the field next to **Filter Condition**. Click the ellipsis (...). The Expression Builder for the filter condition is launched.

27. In the Expression builder on the left-hand side, expand **INOUTGRP1**. Double-click **CATEGORY_ID**. You see that **INOUTGRP1.CATEGORY_ID** is pasted in the Filter condition field on the right-hand side.

In the lower right area, from the **Others** drop-down list, select **IS NOT NULL** and click **Paste**. You see the expression **INOUTGRP1.CATEGORY_ID IS NOT NULL** in the **Filter Condition** for **SUBCATS** field.

28. Click **Validate**. You will see the result in the Validation results field. If the validation is successful, click **OK**.

Figure 3–56 Filter Condition for SUBCATS



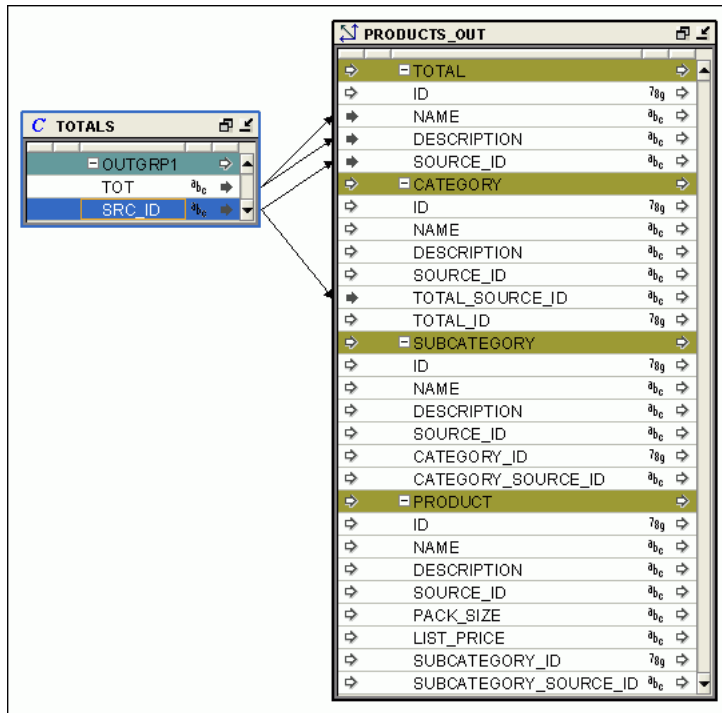
29. Draw connection lines from the attributes in **OUTGRP1** of **TOTALS** constant operator to attributes in groups in the **PRODUCTS_OUT** dimension operator, as shown in [Table 3–3](#).

Table 3–3 TOTALS.OUTGRP1 to PRODUCTS_OUT Mapping

TOTALS.OUTGRP1	PRODUCTS_OUT
TOT	TOTAL.NAME
TOT	TOTAL.DESCRPTION
SRC_ID	TOTAL.SOURCE_ID
SRC_ID	CATEGORY.TOTAL_SOURCE_ID

[Figure 3–57](#) shows what you should see when you are done adding the connections.

Figure 3–57 Completed Connections Between TOTALS.OUTGRP1 and PRODUCTS_OUT



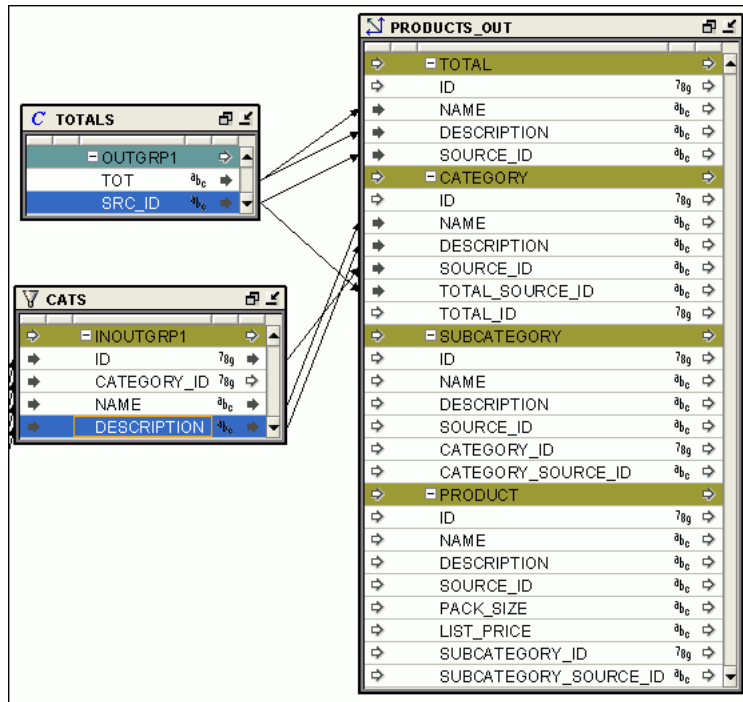
30. Draw connection lines from the attributes in **INOUTGRP1** of **CATS** filter operator to attributes in the **CATEGORY** group of dimension operator **PRODUCTS_OUT**, as shown in [Table 3–4](#).

Table 3–4 CATS.INOUTGRP1 to PRODUCTS_OUT Mapping

CATS.INOUTGRP1	PRODUCTS_OUT
ID	CATEGORY.SOURCE_ID
NAME	CATEGORY.NAME
DESCRIPTION	CATEGORY.DESCRPTION

[Figure 3–58](#) shows what you should see when you are done adding the connections.

Figure 3–58 Completed Connections Between CATS.INOUTGRP1 and PRODUCTS_OUT



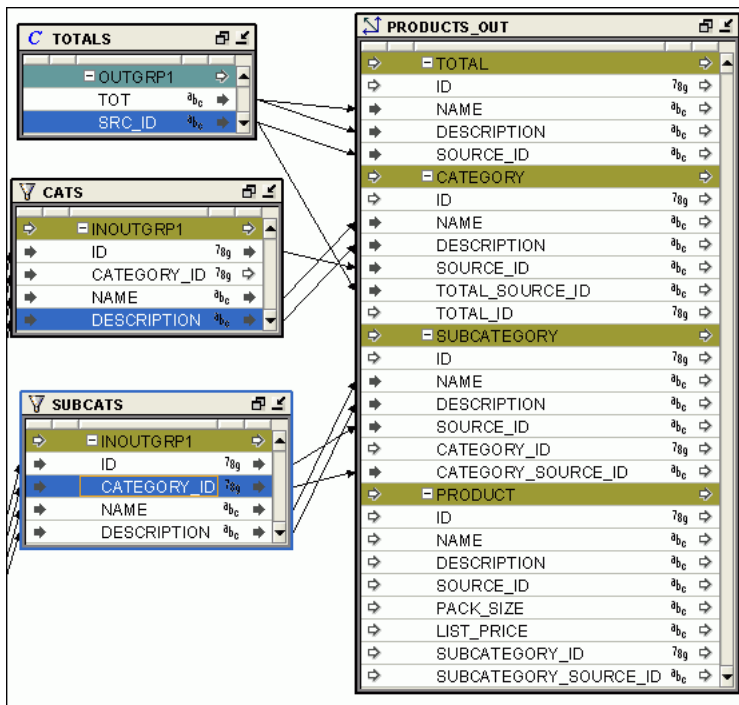
31. Draw connection lines from the attributes in **INOUTGRP1** of **SUBCATS** filter operator to attributes in the **SUBCATEGORY** group of dimension operator **PRODUCTS_OUT**, as shown in [Table 3–5](#).

Table 3–5 SUBCATS.INOUTGRP1 to PRODUCTS_OUT Mapping

SUBCATS.INOUTGRP1	PRODUCTS_OUT
CATEGORY_ID	SUBCATEGORY.CATEGORY_SOURCE_ID
ID	SUBCATEGORY.SOURCE_ID
NAME	SUBCATEGORY.NAME
DESCRIPTION	SUBCATEGORY.DESCRPTION

[Figure 3–59](#) shows what you should see when you are done adding the connections.

Figure 3–59 Completed Connections Between SUBCATS.INOUTGRP1 and PRODUCTS_OUT



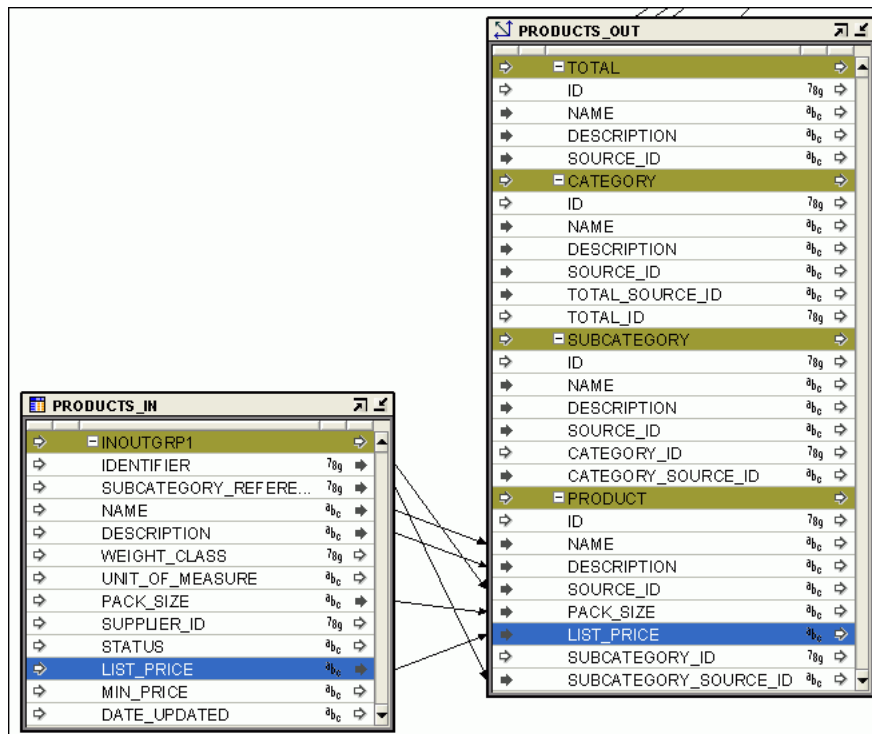
32. Draw connection lines from the attributes in **INOUTGRP1** of table operator **PRODUCTS_IN** to attributes in the **PRODUCT** group of dimension operator **PRODUCTS_OUT**, as shown in [Table 3–6](#).

Table 3–6 PRODUCTS_IN.OUTGRP1 to PRODUCTS_OUT Mapping

PRODUCTS_IN.OUTGRP1	PRODUCTS_OUT
SUBCATEGORY_REFERENCE	PRODUCT.SUBCATEGORY_SOURCE_ID
IDENTIFIER	PRODUCT.SOURCE_ID
NAME	PRODUCT.NAME
DESCRIPTION	PRODUCT.DESCRPTION
PACK_SIZE	PRODUCT.PACK_SIZE
LIST_PRICE	PRODUCT.LIST_PRICE

[Figure 3–60](#) shows what you should see when you are done adding the connections.

Figure 3–60 Completed Connections Between PRODUCTS_IN.OUTGRP1 and PRODUCTS_OUT

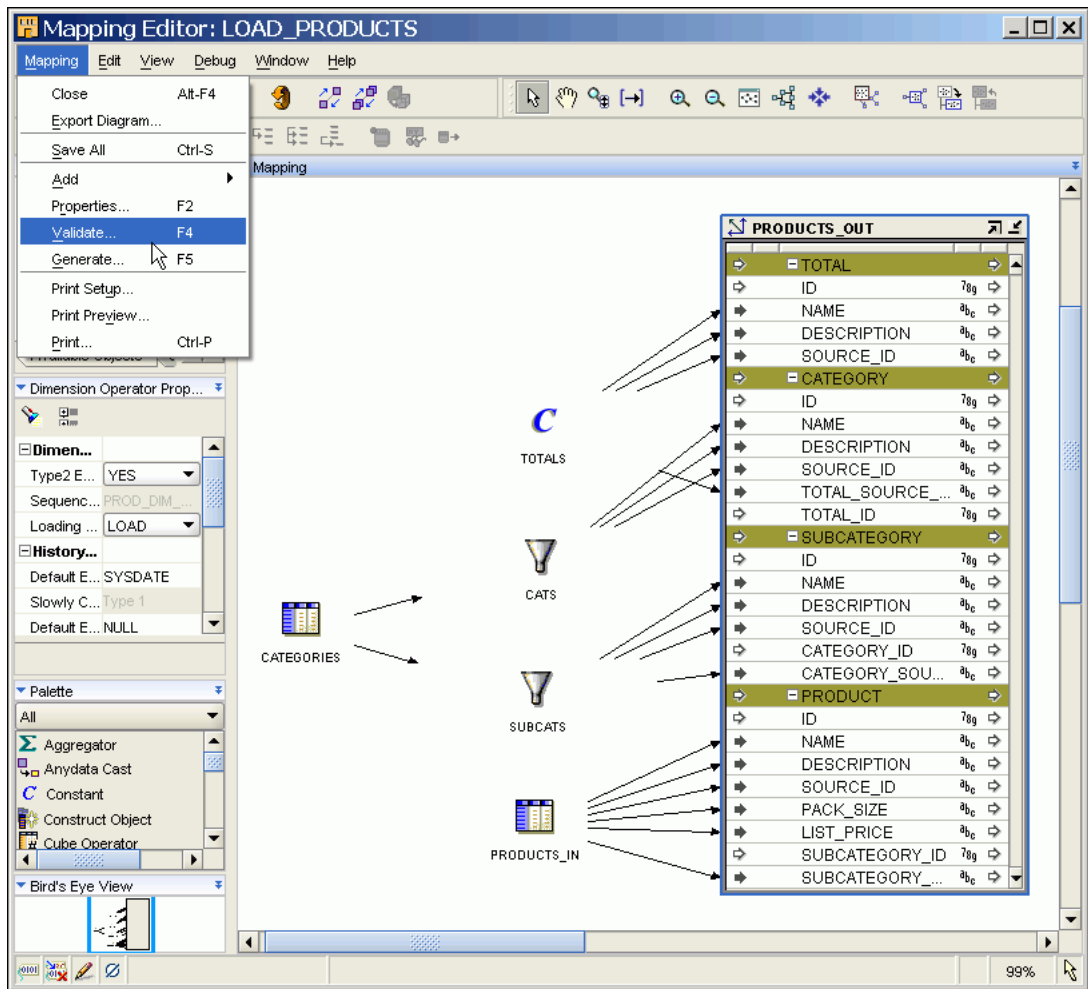


Note: You can collapse objects on the canvas by clicking the Minimize icon:



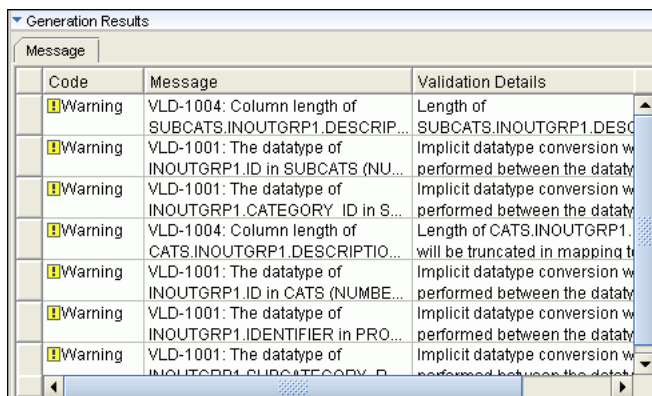
33. Validate the **LOAD_PRODUCTS** mapping by selecting **Mapping > Validate** from the menu.

Figure 3-61 Validating the LOAD_PRODUCTS Mapping



34. The Generation Results panel appears. The warnings on implicit datatype conversion can be ignored. The Oracle Database will perform automatic datatype conversion.

Figure 3-62 Generation Results Panel



35. Select **Generate** from the **Mapping** menu. This will generate the code required to implement the design. Ignore the warnings (7 warnings). Close the Generation Results window.
36. Save your work and close the Mapping Editor.

3.4.2 Use Tcl Scripts to Create Mappings for the CHANNELS Dimension and the SALES Cube

Tcl (Tool Command Language) is a scripting language that can automate and simplify tasks. Oracle Warehouse Builder includes a rich Tcl interface called OMB*Plus. OMB*Plus can be invoked from the command line, or from within the OWB Design Center.

To create mappings for the Channels Dimension and the Sales Cube:

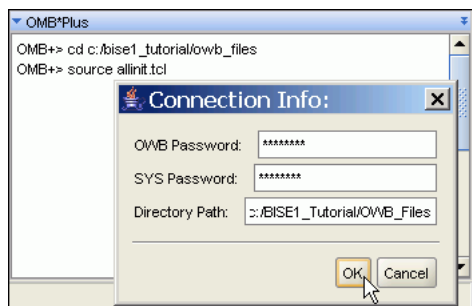
1. In the OWB Design Center, select **Window > OMB*Plus** from the menu.
2. In the OMB*Plus panel, at the OMB+> prompt, enter the following:

```
OMB+> cd BISE1_installation_location/tutorial/owb/tcl
OMB+> source allinit.tcl
```

Note: The slash (/) character is specific to the OMB*Plus TCL interpreter and must be typed in as specified.

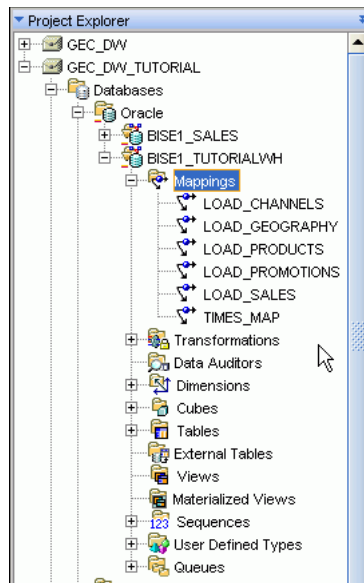
3. Enter the password for the **OWBREPOS_OWNER** and **SYS** database accounts. Enter the location of the tcl files (it should be `BISE1_installation_location/tutorial/owb/tcl`). Click **OK**.

Figure 3–63 Connection Info Dialog Box



4. At the OMB+> prompt, enter the following:


```
OMB+> source loadmaps.tcl
```
5. Using the Project Explorer, verify that the **LOAD_SALES** and **LOAD_CHANNELS** objects have been created.

Figure 3–64 Project Explorer: LOAD_CHANNELS and LOAD_SALES

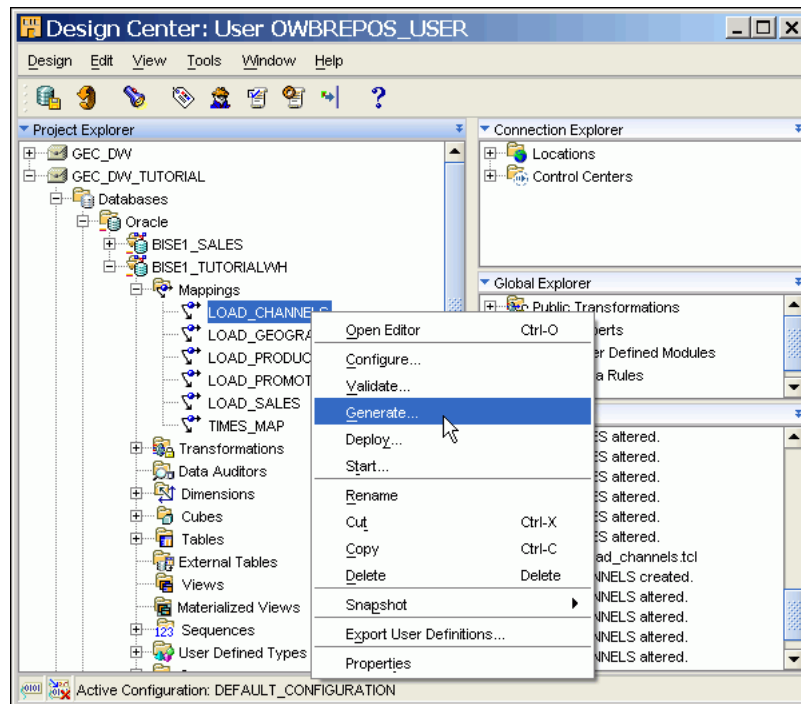
6. Save your work. Now, you have all the mappings needed to complete the design of the warehouse. In the next section, you will use the Generate option to generate the code to implement the design.

3.5 Generating the Mappings

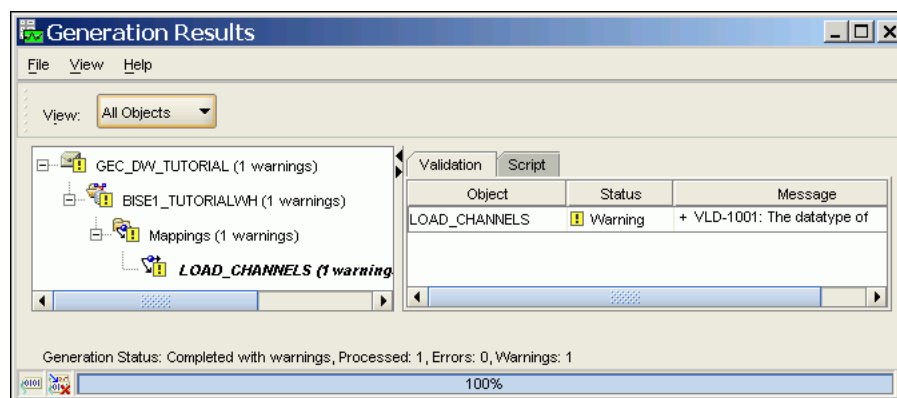
You have already designed and generated the LOAD_PRODUCTS mappings. Now, you need to generate mappings for the other dimensions, as well as the SALES cube.

To generate mappings for the other dimensions and SALES cube:

1. In the Design Center, expand **BISE1_SALESWH > Mappings**.
2. Right-click the **LOAD_CHANNELS** mapping and select **Generate**.

Figure 3–65 LOAD_CHANNELS Mapping: Generate

3. The Generation Results window displays the results. Ignore the warnings. There should not be any errors. Close the Generation Results window.

Figure 3–66 Generation Results Window

4. Repeat Steps 2 and 3 for each of the remaining mappings. You can ignore the warnings:
 - LOAD_PROMOTIONS returns 3 warnings
 - LOAD_GEOGRAPHY returns 9 warnings
 - LOAD_SALES returns 5 warnings
 - TIMES_MAP returns no warnings

3.6 Designing a Process Flow

Depending on the business intelligence needs, you can specify how you want to deploy your design. You can run individual mappings to populate target tables. However, you will more likely want to control dependencies and sequencing among multiple mappings by organizing multiple interdependent mappings within a process flow. When working with process flows, you will use an Oracle Workflow location. An Oracle Workflow location has been pre-defined for the purposes of this tutorial.

This section contains the following topics:

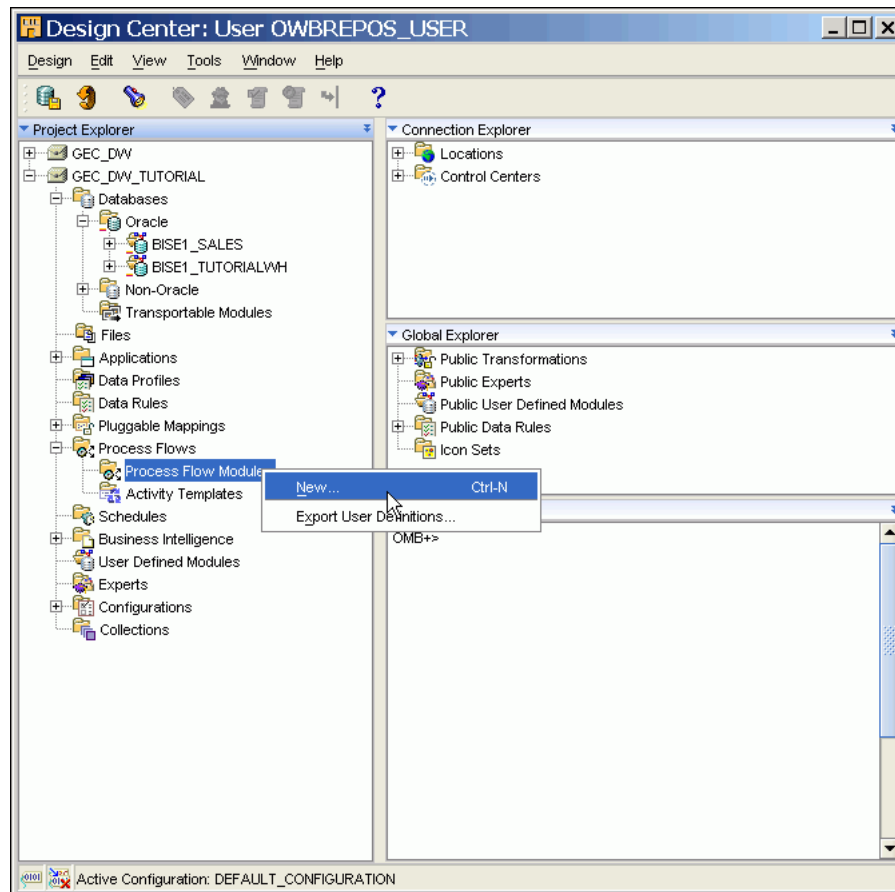
- [Section 3.6.1, "Create a New Process Flow"](#)
- [Section 3.6.2, "Add a Fork Activity"](#)
- [Section 3.6.3, "Add the Mappings"](#)
- [Section 3.6.4, "Add an AND and Control Activities"](#)
- [Section 3.6.5, "Add Transitions"](#)

3.6.1 Create a New Process Flow

To create a new process flow:

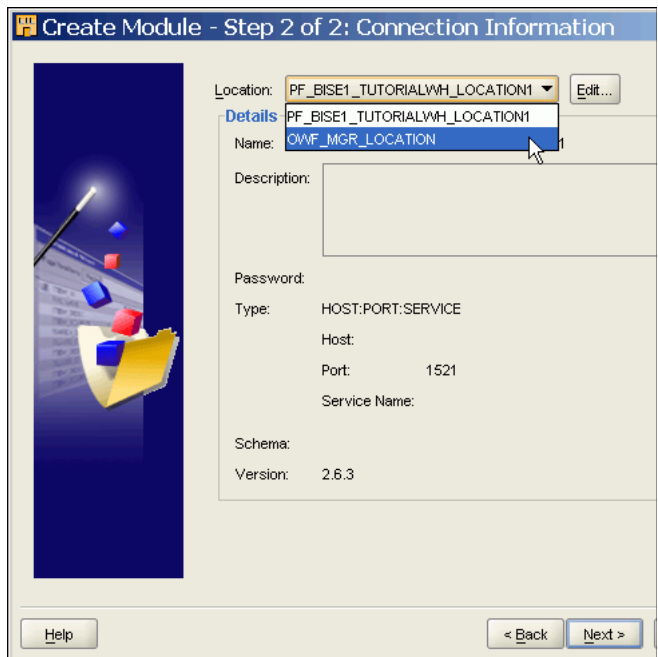
1. In the Design Center, from the Connection Explorer panel, expand **Locations > Process Flows and Schedules > Oracle Workflow**.
2. Double-click **OWF_MGR_LOCATION**. Enter the password for **OWF_MGR** (specified during installation). Click **Test Connection**. If successful, click **OK**.
3. In the Design Center, from the Project Explorer, expand **GEC_DW_TUTORIAL > Process Flows**. Right-click **Process Flow Modules** and select **New**. The Create Module Wizard is launched. Click **Next** on the Welcome page.

Figure 3-67 New Process Flow Module



4. On the Name and Description page, enter the following information:
 - **Name:** PF_BISE1_TUTORIALWH
 - **Description:** Process Flow module for BISE1_TUTORIALWH
 Then, click **Next**.
5. On the Connection Information page, select **OWF_MGR_LOCATION** from the **Location** drop-down list. Click **Next**.

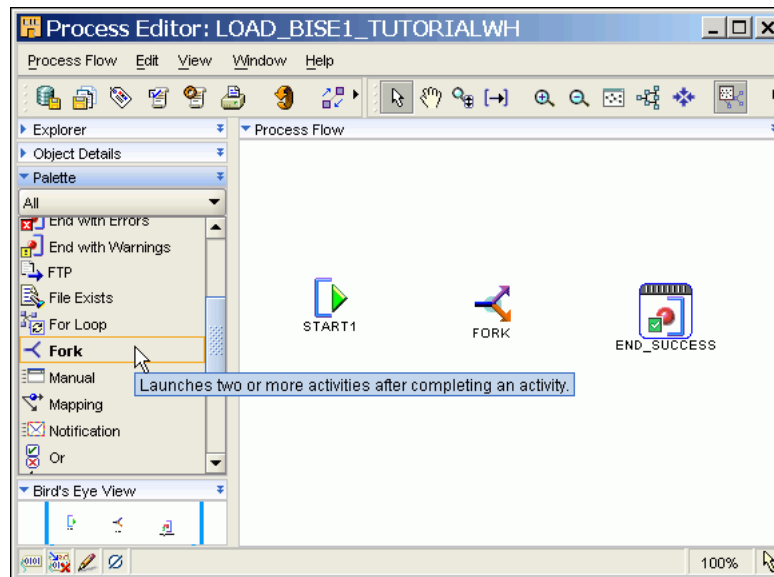
Figure 3–68 Create Module: Connection Information



6. On the Summary page, review the summary details and click **Finish**. The Create Process Flow Package dialog box appears.
7. In the Create Process Flow Package dialog box, enter **PK_SALES** as the name of the process flow package and click **OK**. The Create Process Flow dialog box appears.
8. In the Create Process Flow dialog box, enter **LOAD_BISE1_TUTORIALWH** as the name of the process flow and click **OK**. The process flow is created and the Process Editor is launched.

3.6.2 Add a Fork Activity

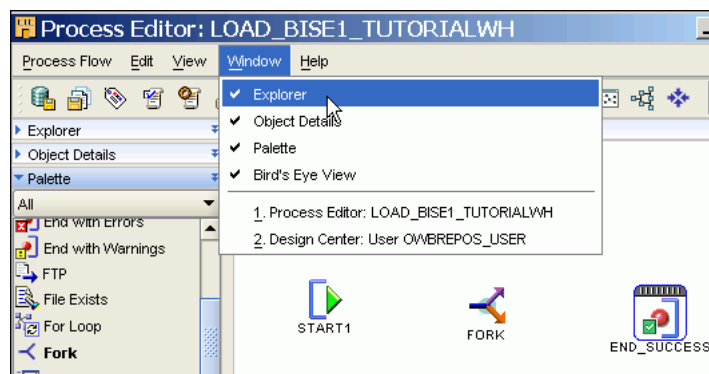
To add a Fork activity, in the Process Editor, from the Palette, drag a **Fork** activity to the canvas. Place the **FORK** activity after the **START** activity and before the **END** activity.

Figure 3–69 Adding a Fork Activity

3.6.3 Add the Mappings

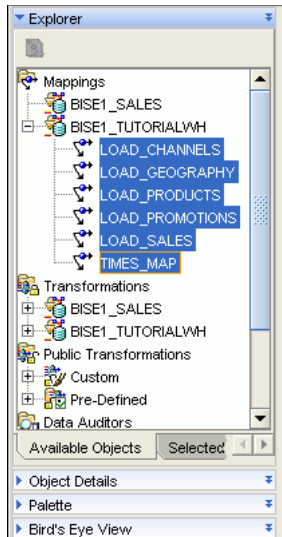
To add the mappings:

1. In the Process Editor, from the **Window** menu, select **Explorer** (if not already selected). The Explorer panel should be visible.

Figure 3–70 Selecting the Explorer Panel

2. In the Explorer panel, click the **Available Objects** tab. Expand **Mappings > BISE1_TUTORIALWH**. Drag the following mappings to the canvas:
 - LOAD_CHANNELS
 - LOAD_PROMOTIONS
 - LOAD_GEOGRAPHY
 - LOAD_PRODUCTS
 - TIMES_MAP
 - LOAD_SALES

Figure 3–71 *Selecting the Mappings*



3. Arrange the activities on the process flow canvas such that the following mapping activities appear after the Fork activity and before the END activity, and in the following order, from top to bottom:

LOAD_GEOGRAPHY, LOAD_CHANNELS, LOAD_PRODUCTS, LOAD_PROMOTIONS, TIMES_MAP

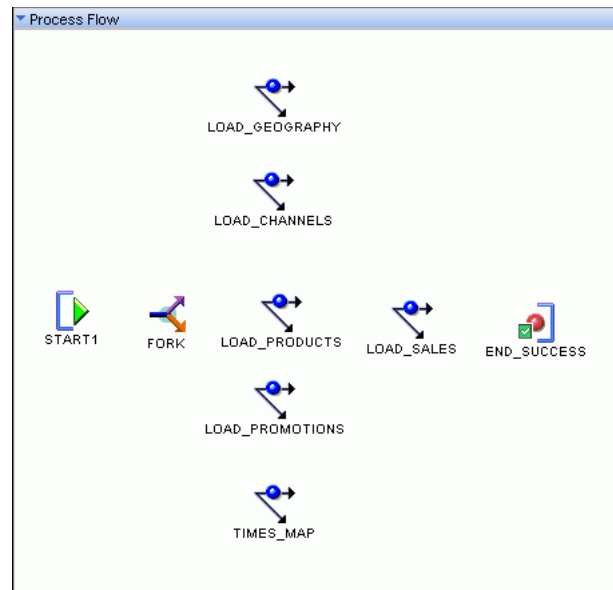
To move an activity within the canvas, select the activity. Place the cursor in the header area of the activity. The cursor changes to a single-headed arrow. Click in the header area and keep the left mouse button clicked and drag to the desired location.

Figure 3–72 *Moving an Activity*



Figure 3–73 shows the order of the activities.

Figure 3–73 Order of Activities

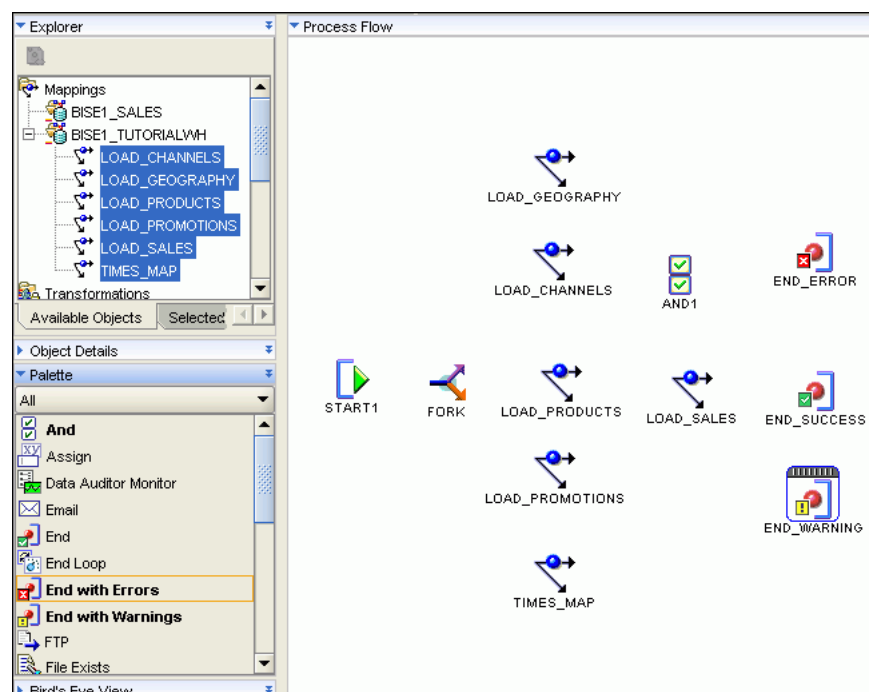


3.6.4 Add an AND and Control Activities

To add an AND and control activities:

1. From the Palette, drag the **AND** activity to the canvas. If the Palette is not visible, from the **Window** menu, select **Palette**.
2. On the Process Editor, from the Palette panel, drag the **End with Errors** and **End with Warnings** activities to the canvas, and position them as shown in Figure 3–74.

Figure 3–74 Dragging the End with Errors and End with Warnings Activities



3.6.5 Add Transitions

To add transitions between activities:

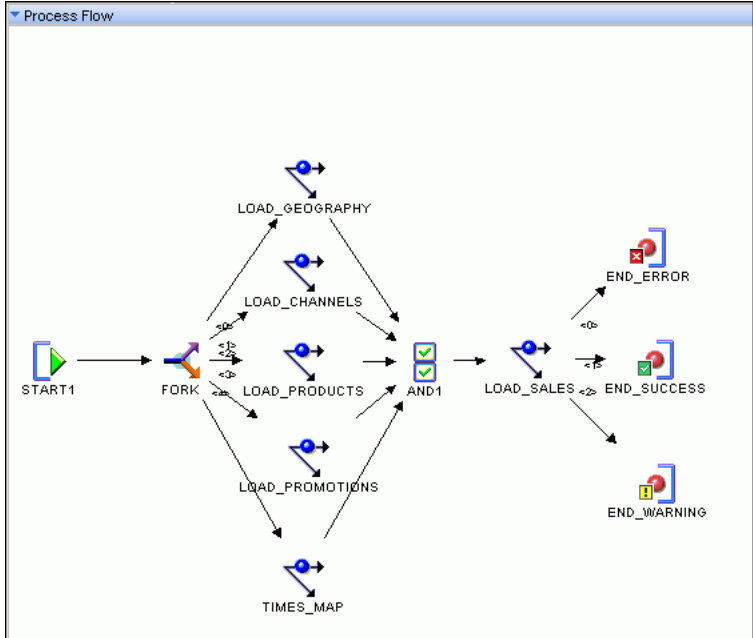
1. Place the cursor (it will change to a single-headed arrow) inside an activity that acts as the source (for example, START). Keeping the left mouse clicked, drag and release the left mouse pointer when you are in the center of the target activity. Create transitions as shown in [Table 3-7](#).

Table 3-7 Transitions for Activities

From	To
START	FORK
FORK	LOAD_CHANNELS
FORK	LOAD_PROMOTIONS
FORK	LOAD_GEOGRAPHY
FORK	LOAD_PRODUCTS
FORK	TIMES_MAP
LOAD_CHANNELS	AND
LOAD_PROMOTIONS	AND
LOAD_CUSTOMERS	AND
LOAD_PRODUCTS	AND
TIMES_MAP	AND
AND	LOAD_SALES
LOAD_SALES	END_SUCCESS
LOAD_SALES	END_ERROR
LOAD_SALES	END_WARNING

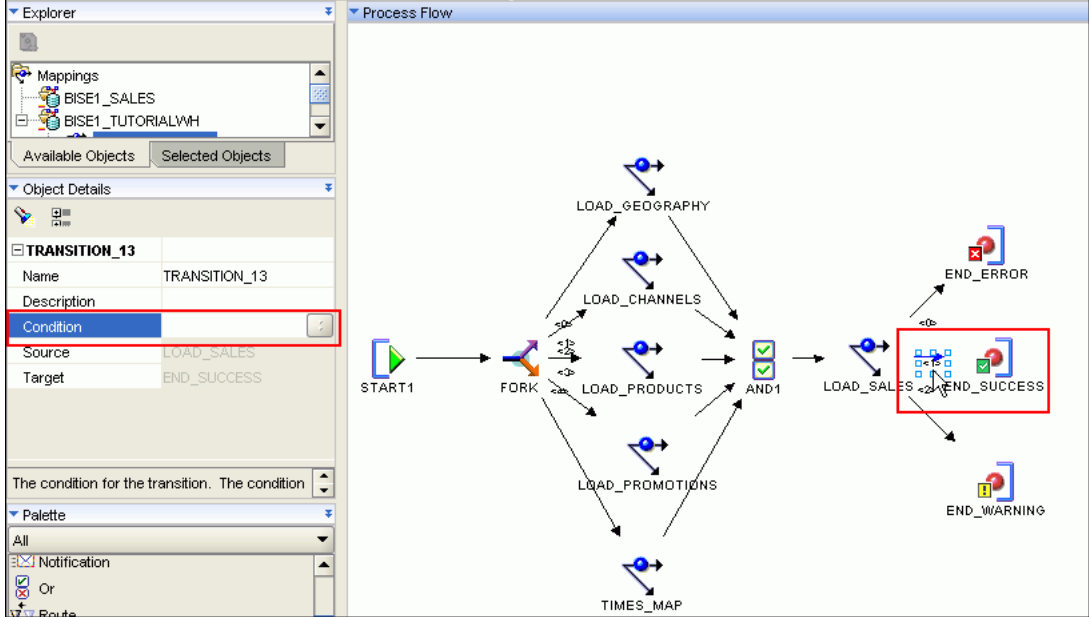
When you finish adding the transitions, your process flow should look like the one shown in [Figure 3-75](#).

Figure 3-75 Transitions for Activities



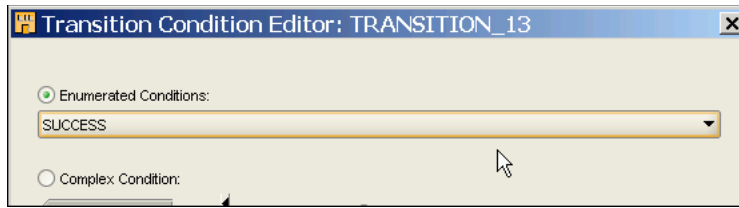
- 2. Click the transition line from the **LOAD_SALES** activity to the **END_SUCCESS** activity. From the Object Details panel, click **Condition**. Click the colon (:) button.

Figure 3-76 Creating a Transition from LOAD_SALES to END_SUCCESS



In the Transition Condition Editor, select **SUCCESS** from the Enumerated drop-down list. Click **OK**.

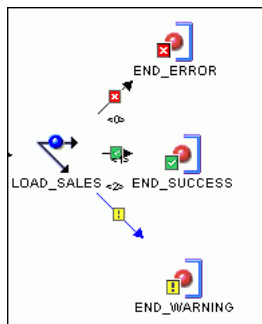
Figure 3–77 Transition Condition Editor



3. Click the transition line from the **LOAD_SALES** activity to the **END_ERROR** activity. From the Object Details panel, click **Condition**. Click the colon (:) button. In the Transition Condition Editor, select **ERROR** from the **Enumerated Conditions** drop-down list. Click **OK**.
4. Click the transition line from the **LOAD_SALES** activity to the **END_WARNING** activity. From the Object Details panel, click **Condition**. Click the colon (:) button. In the Transition Condition Editor, select **WARNING** from the **Enumerated Conditions** drop-down list. Click **OK**.

Your error handling process flow should appear as shown in [Figure 3–78](#).

Figure 3–78 Error Handling Process Flow



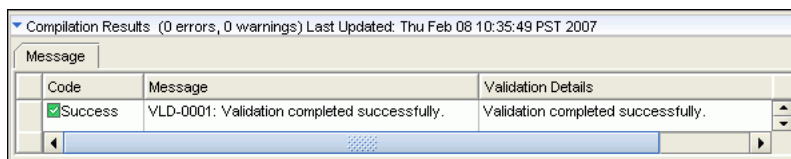
5. Save your work.

3.7 Validating and Generating the Process Flow

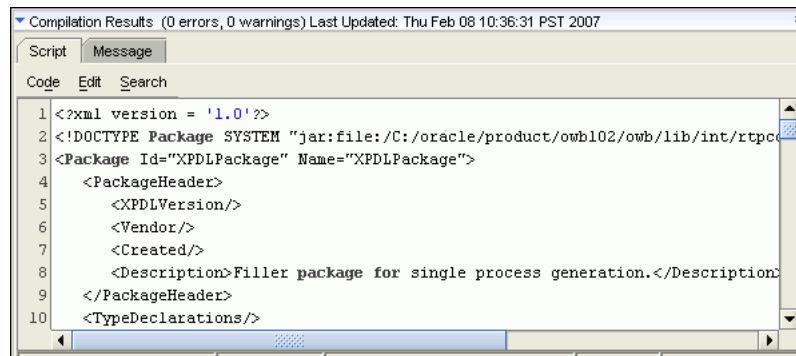
To validate and generate the process flow:

1. In the Process Editor, select **Validate** from the Process Flow menu. The validation results are displayed in the Compilation Results panel.

Figure 3–79 Compilation Results Panel



2. In the Process Editor, select **Generate** from the Process Flow menu. The script is generated and displayed in the Compilation Results window.

Figure 3–80 Generated Process Flow


```

Compilation Results (0 errors, 0 warnings) Last Updated: Thu Feb 08 10:36:31 PST 2007
Script Message
Code Edit Search
1 <?xml version = '1.0'?>
2 <!DOCTYPE Package SYSTEM "jar:file:/C:/oracle/product/owb102/owb/lib/int/rtpc
3 <Package Id="XPDLPackage" Name="XPDLPackage">
4   <PackageHeader>
5     <XPDLVersion/>
6     <Vendor/>
7     <Created/>
8     <Description>Filler package for single process generation.</Description>
9   </PackageHeader>
10  <TypeDeclarations/>

```

3. Save your work and close the Process Editor.

3.8 Deploying Mappings and Loading Data

Until now, you have designed and configured the logical definitions of your target system. Now, you learn how to deploy and create the physical instance of your target. You will deploy the relational and dimensional objects. You will then deploy the ETL mappings. Finally, you will deploy and execute the process flow.

Instead of separately deploying the objects, mappings, and process flow, you can also deploy them all in one step. For the purposes of this tutorial, we are deploying them individually so that you can pause after each deployment and view the results.

This section contains the following topics:

- [Section 3.8.1, "Register the Locations"](#)
- [Section 3.8.2, "Deploy Sequences"](#)
- [Section 3.8.3, "Deploy Tables"](#)
- [Section 3.8.4, "Deploy the Mappings"](#)
- [Section 3.8.5, "Deploy the Process Flow"](#)
- [Section 3.8.6, "Execute the Process Flow"](#)
- [Section 3.8.7, "View the SALES Cube Data"](#)

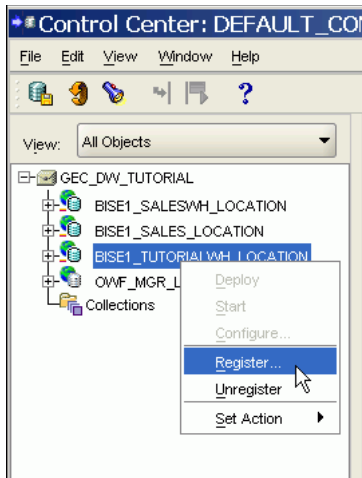
3.8.1 Register the Locations

Locations are specific to a type of module, such as the Oracle Database, SAP application, or flat file. In a flat file module, location is the path or the drive and directory in the file system where the flat files reside. In an Oracle Database module, location contains the database connection information.

To register the locations:

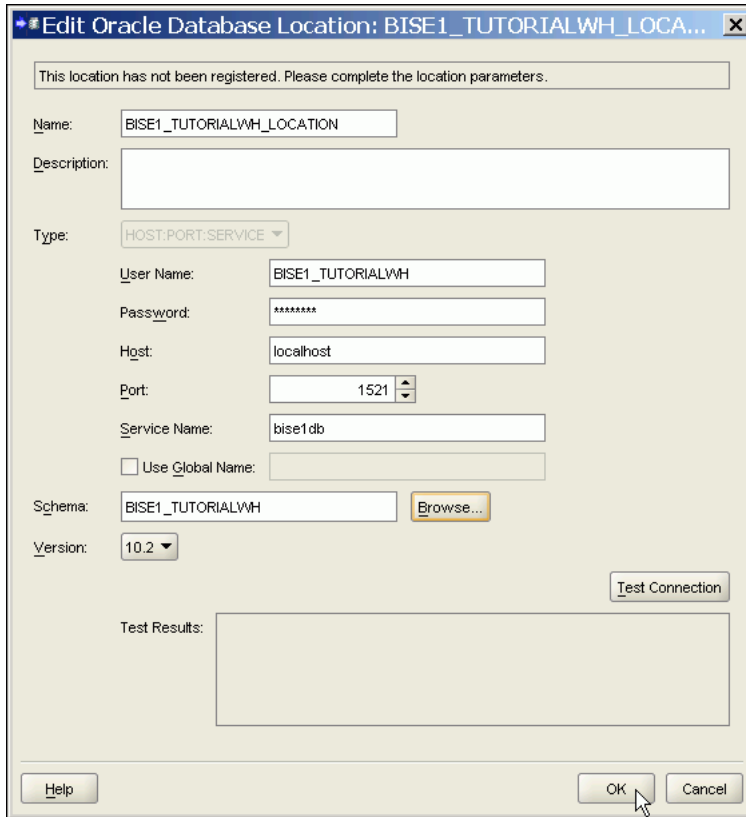
1. In the Design Center, select **Control Center Manager** from the **Tools** menu. The Control Center Manager is launched.
2. In the Control Center, from the navigation tree on the left, right-click **BISE1_TUTORIALWH_LOCATION** and select **Register**.

Figure 3–81 BISE1_TUTORIALWH_LOCATION: Register



3. The Edit Oracle Database Location dialog box appears. Review the details, enter the password, and click **Test Connection**. If the connection is successful, click **OK**.

Figure 3–82 Edit Oracle Database Location: BISE1_TUTORIALWH_LOCATION



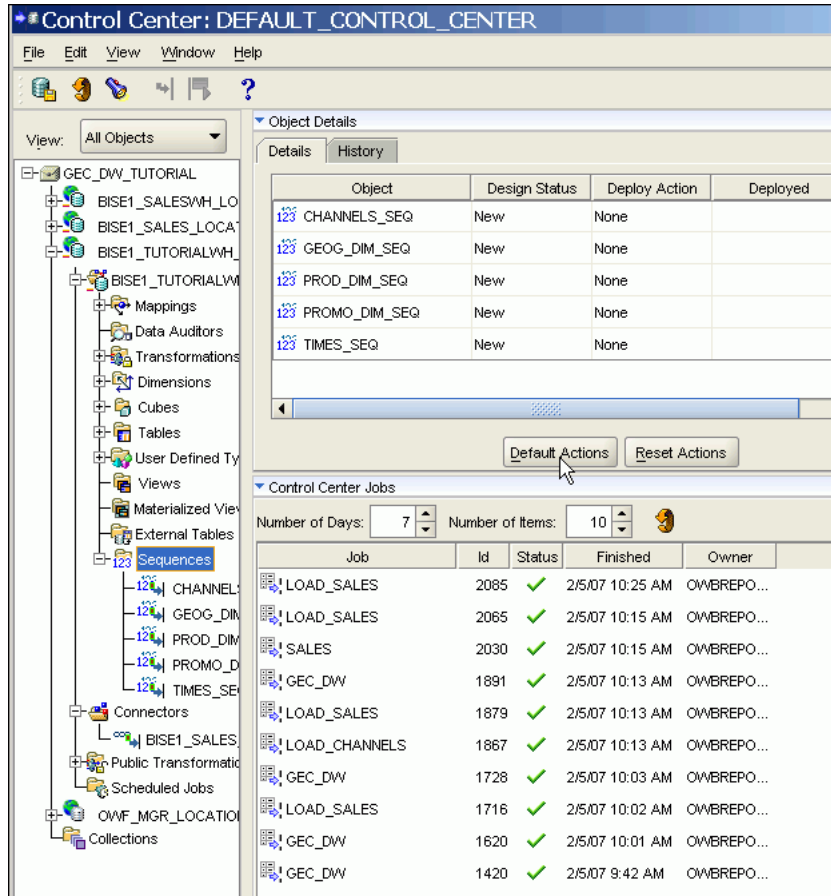
4. Repeat Steps 2 and 3 for **OWF_MGR_LOCATION** and **BISE1_SALES_LOCATION**.

3.8.2 Deploy Sequences

To deploy sequences:

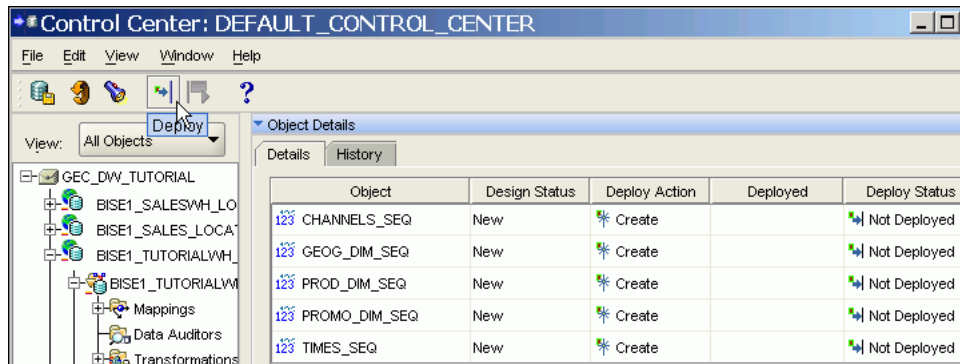
1. In the Control Center, in the left navigation tree, expand **BISE1_TUTORIALWH_LOCATION > BISE1_TUTORIALWH**. Select **Sequences**. On the right, in the Object Details panel, you see the sequences listed. Click the **Default Actions** button.

Figure 3–83 BISE1_TUTORIALWH Sequences



2. Click the **Deploy** button on the toolbar.

Figure 3–84 Deploy Button



3. Monitor the progress in the Control Center Jobs panel. The Deployment tab is selected. In the Control Center Jobs panel, monitor the deployment progress. The

deployment progress moves from **Generation** to **Running** to **Completed Successful**.

Figure 3–85 Control Center Jobs Panel

Object	Design Status	Deploy Action	Deployed	Deploy Status
123 CHANNELS_SEQ	Unchanged	None	2/8/07 10:51 AM	Success
123 GEOG_DIM_SEQ	Unchanged	None	2/8/07 10:51 AM	Success
123 PROD_DIM_SEQ	Unchanged	None	2/8/07 10:51 AM	Success
123 PROMO_DIM_SEQ	Unchanged	None	2/8/07 10:51 AM	Success
123 TIMES_SEQ	Unchanged	None	2/8/07 10:51 AM	Success

Job	Id	Status	Finished	Owner
GEC_DW_TUTORIAL	2121	✓	2/8/07 10:51 AM	OWBREPOS_USER

Note the value in the Deploy Status column. It displays Success against each object that has been deployed successfully.

3.8.3 Deploy Tables

Deploy the Tables in the BISE1_SALESWH target module. To do this:

1. In the Control Center, from the navigation tree, expand **BISE1_TUTORIALWH_LOCATION > BISE1_TUTORIALWH**. Select **Tables**. In the Object Details panel, the tables are listed. Note the values in the Design Status column, Deploy Action column, and Deploy Status column.
2. Click **Default Actions**.

Note: Clicking Default Action changes the value in the Deploy Action column from None to Create.

3. Click the **Deploy** button on the toolbar. In the Control Center Jobs panel, monitor the deployment progress. The deployment progress moves from **Generate** to **Run** to **Successful**. Wait until the Deploy Status column displays Success.

3.8.4 Deploy the Mappings

To deploy the mappings:

1. In the Control Center, from the navigation tree, expand **BISE1_TUTORIALWH_LOCATION > BISE1_TUTORIALWH**. Select **Mappings**. In the Object Details panel, the mappings are listed. Note the values in the Design Status column, Deploy Action column, and Deploy Status column.
2. Click **Default Actions**.

Note: Clicking Default Action changes the value in the Deploy Action column from None to Create.

3. Click the **Deploy** button on the toolbar. In the Control Center Jobs panel, monitor the deployment progress. The deployment progress moves from **Generate** to **Run** to **Successful**. Wait until the Deploy Status column displays Success.

3.8.5 Deploy the Process Flow

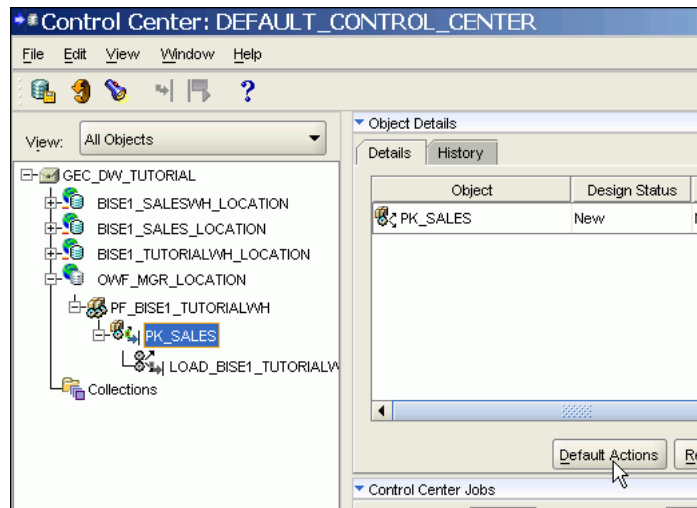
Deploy the LOAD_BISE1_TUTORIALWH process flow. Expand OWF_MGR_LOCATION to find the LOAD_BISE1_TUTORIALWH process flow, as follows:

1. In the Control Center, from the navigation tree, expand **OWF_MGR_LOCATION > PF_BISE1_TUTORIALWH**. Select **PK_SALES**. In the Object Details panel, the process flow package is listed.

Note: When you deploy a process flow, all process flows listed in the package are deployed.

2. Click **Default Actions**.
3. Click the **Deploy** button on the toolbar. In the Control Center Jobs panel, monitor the deployment progress. The deployment progress moves from **Generate** to **Run** to **Successful**. Wait until the Deploy Status column displays Success.

Figure 3–86 Deploying the LOAD_BISE1_TUTORIALWH Process Flow

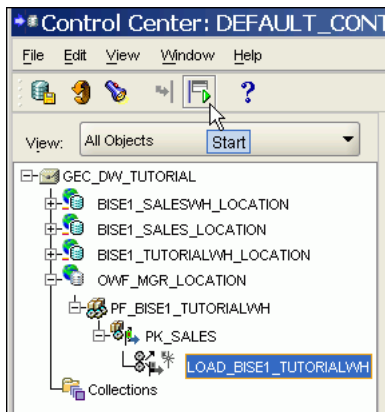


3.8.6 Execute the Process Flow

To execute the process flow:

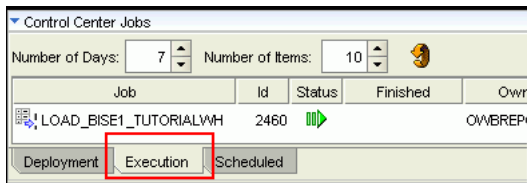
1. Select the **LOAD_BISE1_TUTORIALWH** process flow.
2. To execute a process flow, click the **Start** button on the toolbar.

Figure 3–87 Executing a Process Flow



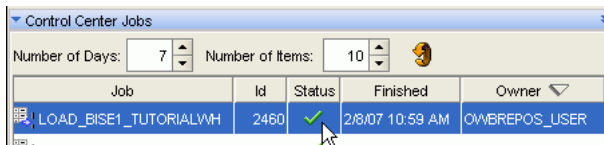
When you click the **Start** button, the Control Center Jobs panel in the lower section switches to the Execution tab.

Figure 3–88 Control Center Jobs Panel: Execution Tab



3. In the Control Center Jobs panel, the **Execution** tab is selected. Monitor the execution until it has completed successfully.

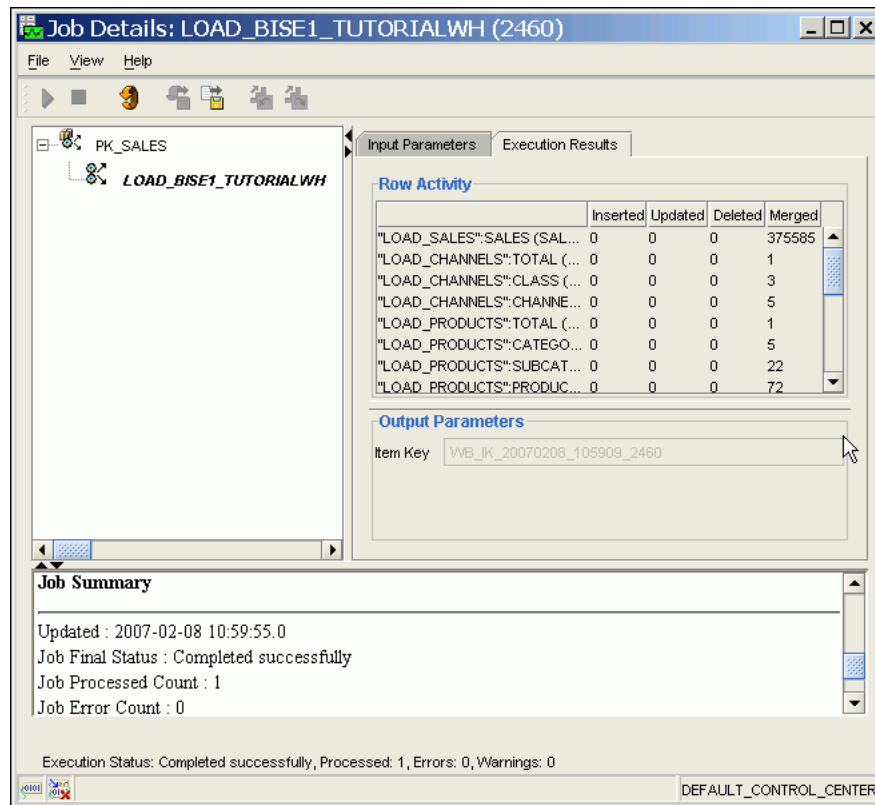
Figure 3–89 Completed Process Flow



Note: The executions are performed asynchronously, which means that you can close the Control Center and OWB client while the Process Flow or Mapping are being executed.

4. In the Control Center Jobs panel, ensure that the **Execution** tab is selected. Double-click **LOAD_BISE1_TUTORIALWH** from the Job column. The Job Details window is displayed.

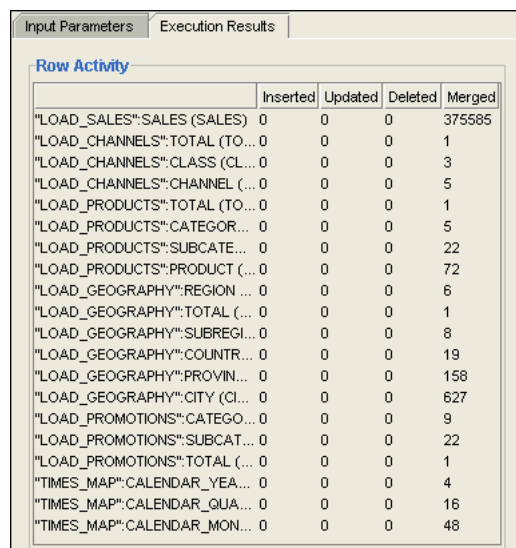
Figure 3–90 Job Details for LOAD_BISE1_TUTORIALWH



- Click the **Execution Results** tab and monitor the Row Activity.

Note: You get a count of how many rows were inserted in respective dimensions and the cube.

Figure 3–91 Execution Results Tab



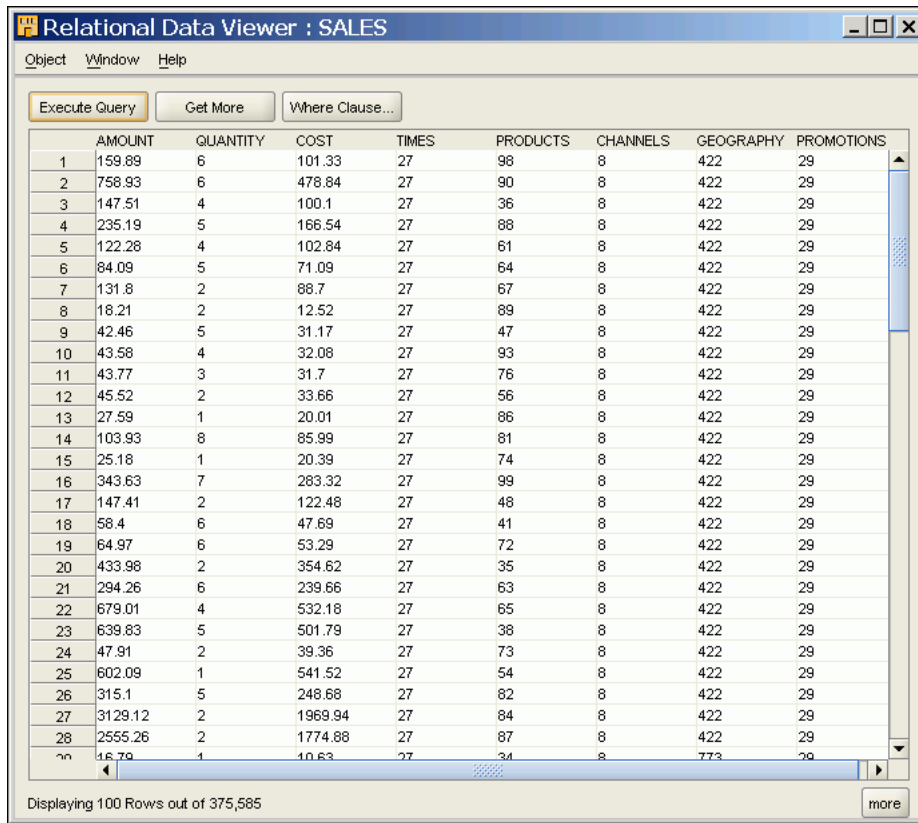
- Close the Control Center Manager.

3.8.7 View the SALES Cube Data

To check that the SALES cube has been populated, use the OWB Design Center Relational Data Viewer to view the data, as follows:

1. Select **View > Refresh** from the Design Center menu.
2. Expand **GEC_DW_TUTORIAL > Databases > Oracle > BISE1_TUTORIALWH > Tables**.
3. Right-click **SALES** and select **Data**. The Relational Data Viewer window appears, displaying the data in the SALES cube. When you are done viewing the data, close the Data Viewer window and exit the Design Center.

Figure 3–92 Relational Data Viewer Window



Congratulations! You have successfully set up your data mart.

Build the BI Repository

In this chapter, you build the Oracle BI metadata repository from the **BISE1_TUTORIALWH** data mart you just created.

Note that some of the later sections in this chapter have been flagged as "Advanced." You do not need to work through these topics during your initial pass through the tutorial, unless you wish to do so.

Prerequisites: If you have not completed the previous chapter, you must first populate the BISE1_TUTORIALWH schema before proceeding with this chapter. [Section 2.2.2.3, "BISE1_TUTORIALWH Schema"](#) explains how to do this.

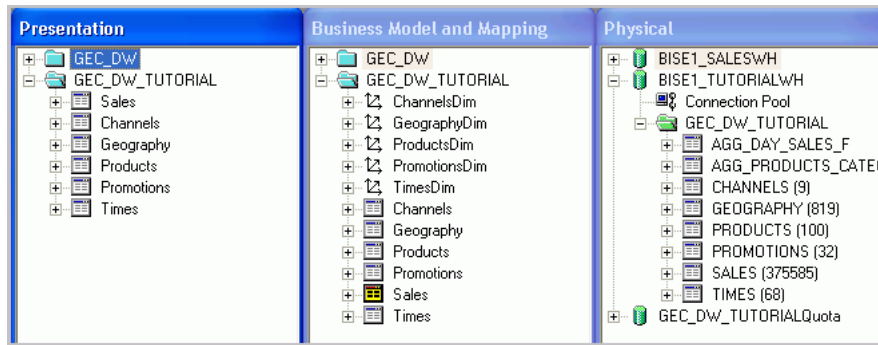
This chapter contains the following topics:

- [Section 4.1, "About the Oracle BI Administration Tool"](#)
- [Section 4.2, "Building the Physical Layer"](#)
- [Section 4.3, "Building the Business Model and Mapping Layer"](#)
- [Section 4.4, "Building the Presentation Layer"](#)
- [Section 4.5, "Testing and Validating a BI Repository"](#)
- [Section 4.6, "Creating Calculation Measures"](#)
- [Section 4.7, "Using Initialization Blocks and Variables - Advanced"](#)
- [Section 4.8, "Executing Direct Database Requests - Advanced"](#)
- [Section 4.9, "Using Aggregates - Advanced"](#)
- [Section 4.10, "Creating Time Series Measures - Advanced"](#)
- [Section 4.11, "Adding Multiple Sources - Advanced"](#)

4.1 About the Oracle BI Administration Tool

The Oracle BI Administration Tool is used to maintain BI metadata repositories. It provides a graphical representation of the three layers of the repository: a) Physical Layer, b) Business Model and Mapping Layer, and c) Presentation Layer. Each of these layers is explained later. You use the Oracle BI Server Administration tool to:

1. Import metadata from database and other data sources to create the physical layer of the BI repository.
2. Simplify and reorganize the imported metadata into a business model.
3. Create the presentation layer from the business model, for presentation to users who request business intelligence information through Oracle BI Answers.

Figure 4–1 Oracle BI Administration Tool

4.2 Building the Physical Layer

The Physical layer defines the data sources to which Oracle BI Server submits queries and the relationships between physical databases and other data sources that are used to process multiple data source queries. The recommended way to populate the Physical layer is by importing metadata from databases and other data sources. The data sources can be of the same or different varieties. You can import schemas or portions of schemas from existing data sources. Additionally, you can create objects in the Physical layer manually.

When you import metadata, many of the properties of the data sources are configured automatically based on the information gathered during the import process. After import, you can also define other attributes of the physical data sources, such as join relationships, that might not exist in the data source metadata. There can be one or more data sources in the Physical layer, including databases, spreadsheets, and XML documents. In this example, you import and configure tables from BISE1_TUTORIALWH schema, where you created your data mart using Warehouse Builder.

This section contains the following topics:

- [Section 4.2.1, "Verify an ODBC Data Source Exists"](#)
- [Section 4.2.2, "Start Oracle BI Services"](#)
- [Section 4.2.3, "Import Tables from the Database Schema"](#)

4.2.1 Verify an ODBC Data Source Exists

An ODBC data source is needed to import data source schema information into an Oracle BI Server repository. An ODBC data source has been pre-created during Oracle BI Standard Edition One installation.

Verify that the ODBC data source is configured correctly and can connect to the database, as follows:

1. Click **Start > Programs > Administrative Tools > Data Sources (ODBC)** to open the ODBC Data Source Administrator. Click the **System DSN** tab.
2. Select **bise1db** in the System Data Sources list, then click **Configure**.
3. Click **Test Connection** in the Oracle ODBC Driver Configuration dialog box. Change the **User Name** to BISE1_TUTORIALWH. Enter the password for the **BISE1_TUTORIALWH** database account. Click **OK**.

The BISE1_TUTORIALWH database account password was defined during installation.

4. You should see a **Connection successful** message. Click **OK**.
5. Click **OK** to exit the ODBC Data Source Administrator.

If you do *not* see **bise1db** in the System Data Sources list, then create it by following these steps:

1. Click **Add** in the ODBC Data Source Administrator dialog box.
2. In the Create New Data Source dialog box, select the **Oracle in BISE1Home1_1** database driver. Click **Finish** to open the Oracle ODBC Driver Configuration dialog box.
3. In the Oracle ODBC Driver Configuration dialog box, enter **BISE1DB** for **Data Source Name**, select the appropriate **TNS Service Name** from the drop-down list (**BISE1DB**), and enter **BISE1_TUTORIALWH** as the User ID for the schema.
4. Click **Test Connection** to open the Oracle ODBC Driver Connect dialog box. Enter the password for the **BISE1_TUTORIALWH** database schema and click **OK**.
5. You should see a **Connection successful** message. Click **OK**.
6. Click **OK** in the Oracle ODBC Driver Configuration dialog box. Verify that the **BISE1DB** system data source is added in the ODBC Data Source Administrator, then click **OK** to close the ODBC Data Source Administrator.

4.2.2 Start Oracle BI Services

In this step, you restart the Oracle BI Server service to load the **bise1** repository into memory, as follows:

1. Select **Start > Programs > Administrative Tools > Component Services**. Double-click **Services**.
2. Restart the **Oracle BI Server service**. Verify that the **Oracle BI Presentation Server** and **Oracle BI Java Host** services are started. If they are not started, start them. The services can be started in any order. Close the Component Services window.

4.2.3 Import Tables from the Database Schema

To import the dimensions and fact tables from the **BISE1_TUTORIALWH** schema into the repository, perform the following steps:

1. Click **Start > Programs > Oracle Business Intelligence > Administration** to open the Oracle BI Administration Tool. Click **File > Open > Online**.
2. In the Open Online AnalyticsWeb dialog box, type **Administrator** as userid and password. The password is case sensitive. Click **Open**.
3. The repository will open with **BISE1_SALESWH** in the Physical Layer and its corresponding **GEC_DW** Business Model and Presentation Layer in **online** mode. **GEC_DW** is a prebuilt subject area created from the prebuilt data mart, **BISE1_SALESWH**. Explore the repository. You will be creating a similar BI metadata repository based on the **BISE1_TUTORIALWH** data mart.

Note: In online mode, Oracle BI Server has opened the repository file and the operating system has put a write-lock on the file. In this mode, Oracle BI Server can act as an agent of the Administration Tool. When the Administration Tool tells it to, Oracle BI Server sends the Administration Tool a copy of its in-memory repository. Then it listens for messages from the Administration Tool about changes, makes those changes to its in-memory copy, and, when told by the Administration Tool, tells the operating system to save the changed file.

When you start the Administration Tool in online mode, you pick an Oracle BI ODBC DSN that points to the repository you want to edit (in other words, the repository that is the default repository in the DSN). The Administration Tool then communicates changes to Oracle BI Server, and Oracle BI Server makes the corresponding changes to its in-memory copy.

You can also work in offline mode. In offline mode, the relationship between the Administration Tool and the repository is like the relationship between any Windows application and a file. An application, the Administration Tool in this case, opens a file for editing, makes changes to its in-memory copy, and tells the operating system to save the changed file.

Typically, you develop a repository in offline mode and use online mode for minor updates and changes.

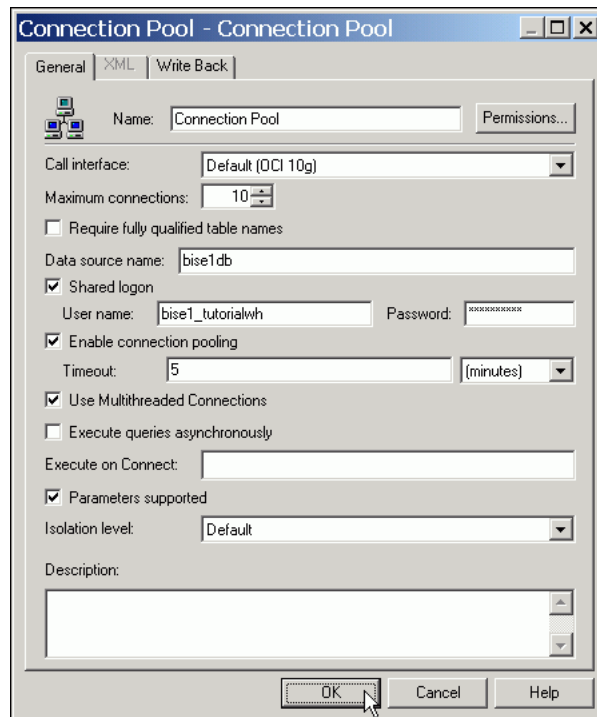
4. Import the tables from the **BISE1_TUTORIALWH** schema. Select **File > Import > from Database**, then select the **BISE1DB** ODBC connection. Enter **BISE1_TUTORIALWH** as the **User Name**. Enter the **Password** for the **BISE1_TUTORIALWH** database account. Click **OK**.
5. In the Import dialog box, expand the **BISE1_TUTORIALWH** schema folder and use **Ctrl + Click** to select the following tables: **CHANNELS**, **GEOGRAPHY**, **PRODUCTS**, **PROMOTIONS**, **SALES**, and **TIMES**.

Note: Do not select the **WB_RT_VERSION_FLAG** table for import. That is an internal table used by Oracle Warehouse Builder, and is not needed for the BI metadata repository.

6. **Tables** and **Keys** are the default options selected. Because you are familiar with the underlying schema structure and know that there are foreign keys already defined, go ahead and choose to import the foreign keys by selecting the **Foreign Keys** option. Click **Yes** when prompted about whether you wish to proceed with importing the foreign keys.

If you were not aware of the joins in the schema, or your queries need different joins, it is better to select only **Tables** and **Keys**, and create physical joins later.

7. Click **Import**. The Connection Pool dialog box opens.
8. In the Connection Pool dialog box, on the **General** tab, verify that the call interface is set correctly. Select **Default (OCI 10g)** from the drop-down list. Change the data source name to the appropriate `tnsnames.ora` entry (**BISE1DB** is used for the tutorial). Note that this is the TNS service name, not the ODBC DSN.

Figure 4–2 Connection Pool Dialog Box: General Tab

9. Leave the rest of the settings as they are and click **OK** to close the Connection Pool dialog box.
10. After the Import completes, click **Close** to exit the Import dialog box.
11. In the Physical layer of the repository, expand the **bise1db > BISE1_TUTORIALWH** schema folder, and verify that the correct tables are imported.
12. Right-click the **BISE1_TUTORIALWH** folder. Select **Rename** from the pop-up menu, and rename the folder to **GEC_DW_TUTORIAL**.
13. Check in the changes by selecting **File > Check In Changes**. To verify connectivity, select **Tools > Update All Row Counts**. When prompted about checking out the objects, click **Yes**.
14. When Update All Row Counts completes, verify that the row counts are displayed in the Physical layer of the Administration Tool. Select **Tools > Options**. Select **Show row count in physical view**, then click **OK**.

You should see the row counts for each of the tables, as shown in [Figure 4–3](#).

Figure 4–3 Tables with Row Counts

4.3 Building the Business Model and Mapping Layer

In this section, you use the Oracle BI Administration Tool to build the Business Model and Mapping layer of a repository.

The Business Model and Mapping layer of the Administration Tool defines the business, or logical, model of the data and specifies the mappings between the business model and the Physical layer schemas. This is where the physical schemas are simplified to form the basis for the users' view of the data. The Business Model and Mapping layer of the Administration Tool can contain one or more business model objects. A business model object contains the business model definitions and the mappings from logical to physical tables for the business model.

The main purpose of the business model is to capture how users think about their business using their own vocabulary. The business model simplifies the physical schema and maps the users' business vocabulary to physical sources. Most of the vocabulary translates into logical columns in the business model. Collections of logical columns form logical tables. Each logical column (and hence each logical table) can have one or more physical objects as sources.

There are two main categories of logical tables: fact and dimension. Logical fact tables contain the measures by which an organization gauges its business operations and performance. Logical dimension tables contain the data used to qualify the facts.

This section contains the following topics:

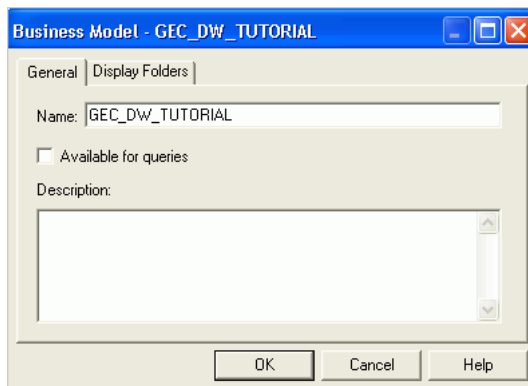
- [Section 4.3.1, "Create a Business Model"](#)
- [Section 4.3.2, "Populate the Business Model"](#)
- [Section 4.3.3, "Rename Business Model Objects"](#)
- [Section 4.3.4, "Delete Unnecessary Business Model Objects"](#)
- [Section 4.3.5, "Build Dimension Hierarchies"](#)

4.3.1 Create a Business Model

To create a business model:

1. In the Business Model and Mapping layer, right-click the white space and select **New Business Model**.
2. In the Business Model dialog box, name the business model **GEC_DW_TUTORIAL** and leave the **Available for queries** box unchecked. The **Description** field is used to add a comment for yourself or another developer. Leave it empty.

Figure 4–4 Business Model Dialog Box



3. Click **OK** to close the Business Model dialog. The new **GEC_DW_TUTORIAL** business model appears in the Business Model and Mapping layer. The red symbol on the business model indicates it is not yet enabled for querying. You

enable the business model for querying later, after the Presentation layer is defined and the repository passes a global consistency check.

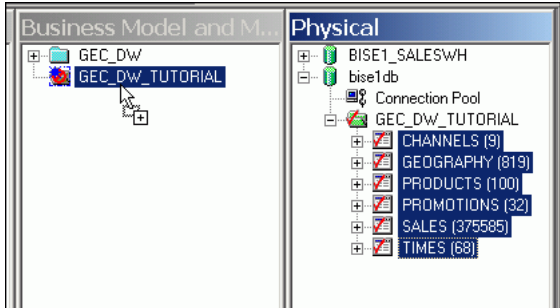
4.3.2 Populate the Business Model

You can manually define the logical tables, joins, and columns in the business model. Or, you can allow the BI Administration Tool to automatically create them by simply dragging Physical layer objects over to the Business Model and Mapping layer. In this tutorial, you will use the automatic definition. For details on how to manually create logical tables, joins, and columns, consult the Oracle BI Administration Tool product documentation.

To populate the business model:

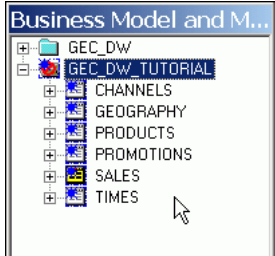
1. In the Physical layer, select all the tables in the **GEC_DW_TUTORIAL** folder, and drag them over to the **GEC_DW_TUTORIAL** business model folder.

Figure 4-5 Moving Tables to the Business Model



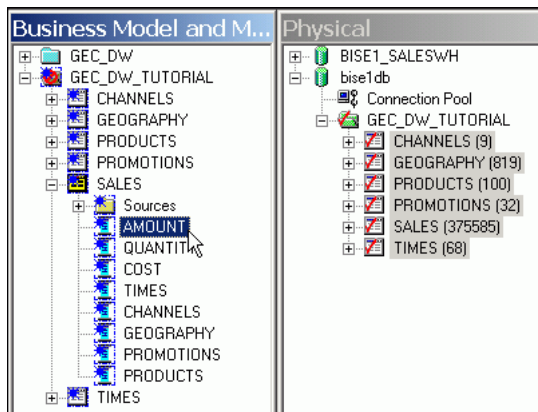
This action automatically creates logical tables in the Business Model and Mapping layer. Notice that the SALES logical table has a yellow table icon. In the Business Model and Mapping layer, this indicates a fact table. Dimension tables are denoted with white table icons.

Figure 4-6 Logical Tables in the Business Model and Mapping Layer



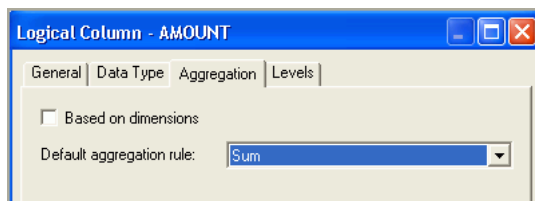
2. Expand the **SALES** logical table, then **Sources** to see the logical table source that was created automatically. Logical table sources define the mappings from a logical table to a physical table. A logical table's Sources folder contains the logical table sources. The logical table source name, **SALES**, is the same name as the physical table. However, it is possible to change names in the Business Model and Mapping layer without impacting the mapping.

Figure 4–7 Business Model and Mapping Layer: SALES Table



3. In the Business Model and Mapping layer, double-click the **AMOUNT** logical column to open the Logical Column dialog box. Click the **Aggregation** tab. In the **Default aggregation rule** drop-down list, select **SUM**. Click **OK**.

Figure 4–8 Logical Column Dialog Box



4. Save the repository. Click **OK** to check in changes, but answer **No** for consistency checking. Leave the Administration Tool and the repository open for the next topic.

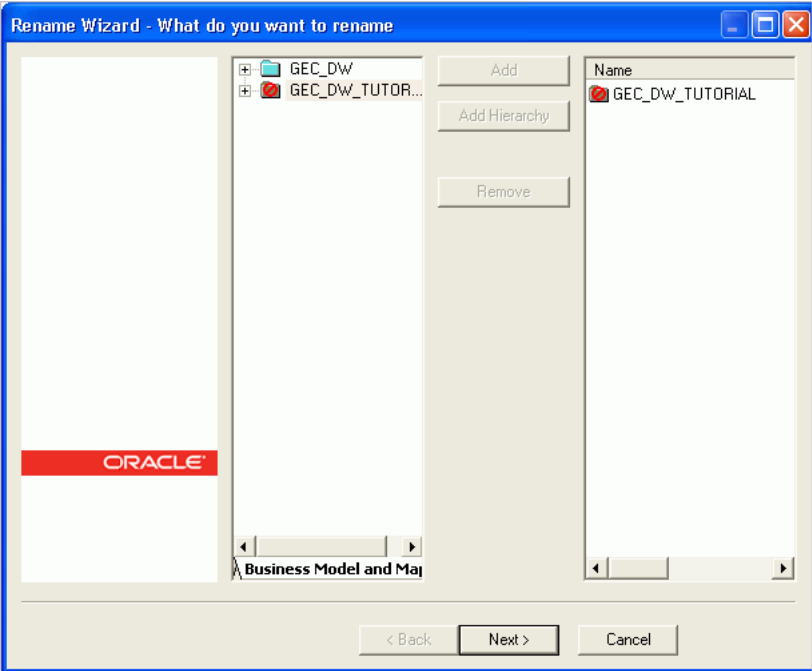
Note: Checking Global Consistency checks for errors in the entire repository. Some of the more common checks are done in the Business Model and Mapping layer and Presentation layer. Since these layers are not defined yet, bypass this check until the other layers in the repository are built. You learn more about consistency checking later in this tutorial.

4.3.3 Rename Business Model Objects

To rename business model objects:

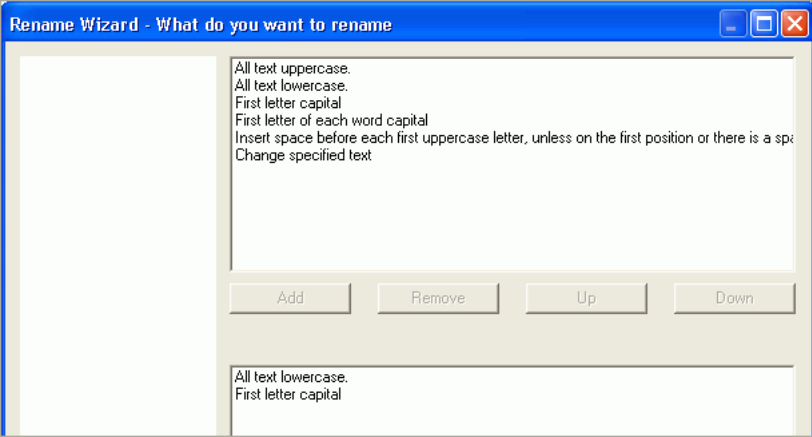
1. Click **Tools > Utilities**. In the Utilities dialog box, click **Rename Wizard** and then click **Execute**.
2. In the Rename Wizard, scroll the middle pane. Click the **Business Model and Mapping** tab and select the **GEC_DW_TUTORIAL** business model, then click the **Add Hierarchy** button. Click **Next**.

Figure 4–9 Rename Wizard: GEC_DW_TUTORIAL

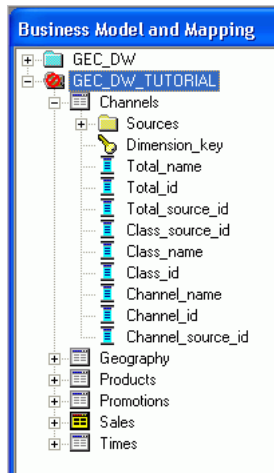


- 3. Select **Logical Table** and **Logical Column** only. Click **Next**.
- 4. Select **All text lowercase**. Click **Add**.
- 5. Select **First letter capital**. Click **Add**.

Figure 4–10 Rename Wizard: All Text Lowercase and First Letter Capital



- 6. Click **Next** and review changes. Click **Next** again.
- 7. Click **Finish** and verify that logical tables and logical columns in the Business Model and Mapping layer are changed as expected.

Figure 4–11 Business Model and Mapping Layer: GEC_DW_TUTORIAL Changes

8. Rename the following columns in the **Channels** logical table:

- **Total_name** to **Channel_total**
- **Class_name** to **Channel_class**

Note: To rename an object, right-click it, then select **Rename** from the pop-up menu. You can also rename by selecting the column and gently clicking on the name to open an edit box.

9. Rename the following columns in the **Times** logical table:

- **Calendar_year_name** to **Year**
- **Calendar_month_description** to **Month**
- **Calendar_quarter_name** to **Quarter**

10. Rename the following columns in the **Geography** logical table:

- **City_name** to **City**
- **Province_name** to **Province**
- **Country_name** to **Country**
- **Subregion_name** to **Subregion**
- **Region_name** to **Region**
- **Total_name** to **Geography_total**

11. Rename the following columns in **Products**:

- **Category_name** to **Product_Category**
- **Subcategory_name** to **Product_Subcategory**

12. Rename the following columns in **Promotion**:

- **Subcategory_name** to **Promotion_Subcategory**
- **Category_name** to **Promotion_Category**

13. In **Sales**, rename **Amount** to **Amount_Sold**.

14. Save the repository. Click **OK** to check in changes, but answer **No** for consistency checking.

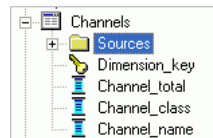
4.3.4 Delete Unnecessary Business Model Objects

To delete unnecessary business model objects:

1. For the **Channels** logical table in the Business Model and Mapping layer, use **Ctrl + Click** to select the **Channel_id**, **Channel_source_id**, **Class_id**, **Class_source_id**, **Total_id**, and **Total_source_id** logical columns. Right-click either of the highlighted columns and select **Delete** to delete the columns. Alternatively, you can use the Delete key on your keyboard. Click **Yes** to confirm the delete.

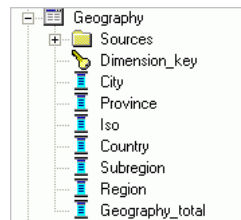
Verify that the **Channels** logical table now has only four logical columns, as shown in [Figure 4-12](#).

Figure 4-12 Deleting Logical Columns: Channels Table



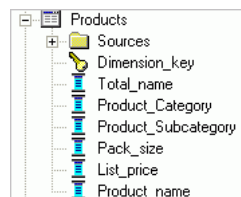
2. Repeat the steps to delete some of the logical columns in the **Geography** table. After you complete this step, you should only have the logical columns shown in [Figure 4-13](#).

Figure 4-13 Deleting Logical Columns: Geography Table



3. Repeat the steps to delete some of the logical columns in the **Products** table. After you complete this step, you should only have the logical columns shown in [Figure 4-14](#).

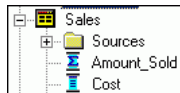
Figure 4-14 Deleting Logical Columns: Products Table



4. Repeat the steps to delete some of the logical columns in the **Promotion** table. After you complete this step, you should only have the logical columns shown in [Figure 4-15](#).

Figure 4–15 Deleting Logical Columns: Promotion Table

5. Repeat the steps to delete some of the logical columns in the **Sales** table. After you complete this step, you should only have the logical columns shown in [Figure 4–16](#).

Figure 4–16 Deleting Logical Columns: Sales Table

6. Do not delete any columns in the **Times** table.
7. Save the repository. Click **OK** to check in changes, but answer **No** for consistency checking.

4.3.5 Build Dimension Hierarchies

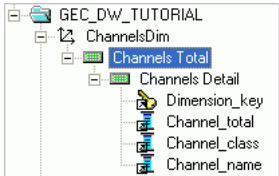
Dimension hierarchies introduce formal hierarchies into a business model, allowing Oracle BI Server to calculate useful measures and allowing users to drill down to more detail. In a business model, a dimension hierarchy represents a hierarchical organization of logical columns belonging to a single logical dimension table. Common dimension hierarchies used in a business model are time periods, products, customers, suppliers, and so forth.

Dimension hierarchies are created in the Business Model and Mapping layer, and end users do not see them in end user tools such as Oracle BI Answers or Interactive Dashboards. In each dimension hierarchy, you organize dimension attributes into hierarchical levels. These levels represent the organizational rules and reporting needs required by your business. They provide the structure that Oracle BI Server uses to drill into and across dimensions to get more detailed views of the data. Dimension hierarchy levels are used to perform aggregate navigation, configure level-based measure calculations, and determine what attributes appear when Oracle BI users drill down in their data requests.

To build dimension hierarchies:

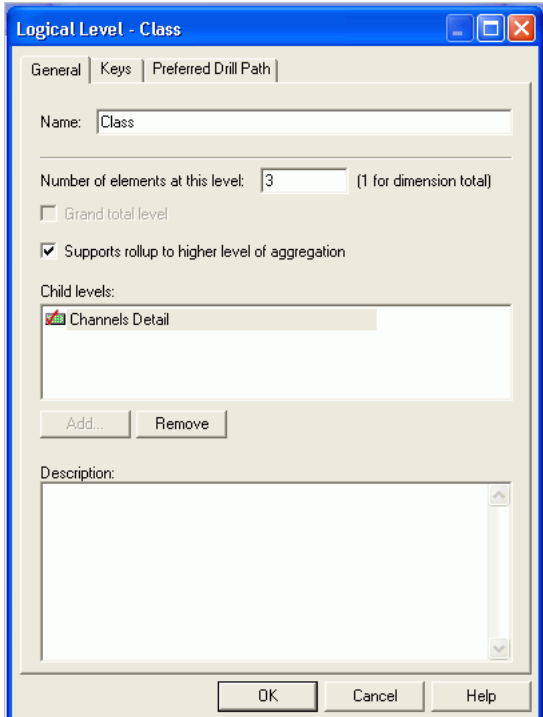
1. Right-click the **Channels** logical table and select **Create Dimension**. Click **Yes** if prompted on whether you want to check out the objects. Right-click the **ChannelsDim** object, which was created by the action in the previous step, and select **Expand All**.
2. Verify that the **ChannelsDim** dimension hierarchy matches the one shown in [Figure 4–17](#).

Figure 4-17 ChannelsDim Dimension Hierarchy



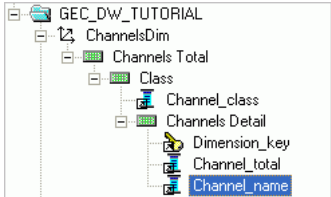
- 3. Right-click the **Channels Detail** level and select **New Object > Parent Level**. In the Logical Level dialog box, name the logical level **Class** and set the **Number of elements at this level** to **3**. This number does not have to be exact. The ratio from one level to the next is more important than the absolute number. These numbers only affect which aggregate source is used (optimization, not correctness of queries). Click **OK** to close the Logical Level dialog box. The new **Class** level is added to the hierarchy.

Figure 4-18 Logical Level: Class



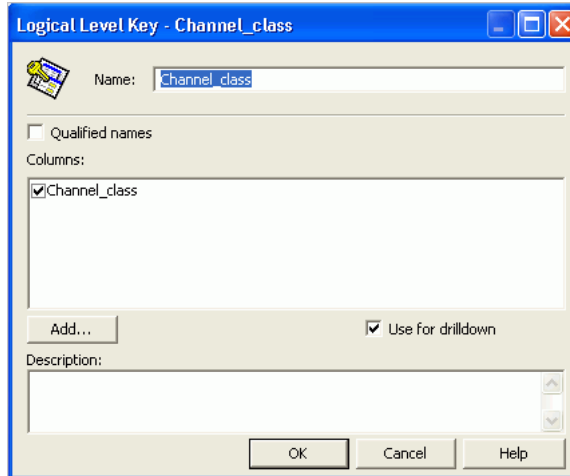
- 4. Right-click the **Class** level and select **Expand All**. Drag the **Channel_class** column from the **Channels Detail** level to the **Class** level to associate the logical column with this level of the hierarchy.

Figure 4-19 Moving the Channel_class Column to the Class Level



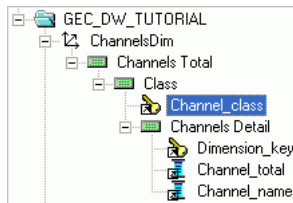
- Right-click **Channel_class** and select **New Logical Level Key**. In the Logical Level Key dialog box, verify that **Channel_class** and **Use for drilldown** are selected. The level key defines the unique elements in each logical level. Each level key can consist of one or more columns at this level.

Figure 4–20 Logical Level Key: Channel_class



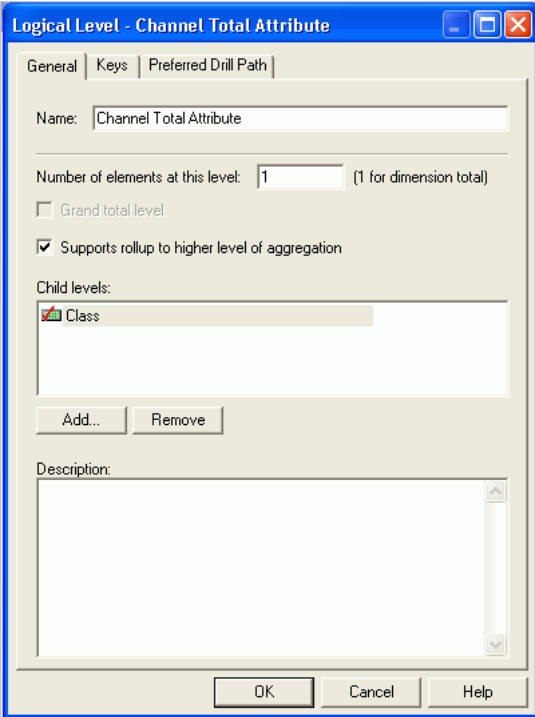
- Click **OK** to close the Logical Level Key dialog box. The **Channel_class** column now displays with a key icon.

Figure 4–21 Channel_class Column with Key Icon



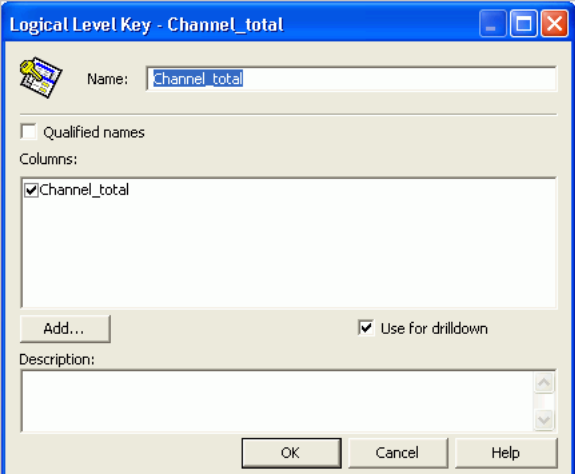
- Right-click the **Class** level and select **New Object > Parent Level**. In the Logical Level dialog box, name the logical level **Channel Total Attribute** and set the **Number of elements at this level** to 1. Close the dialog box.

Figure 4-22 Logical Level: Channel Total Attribute



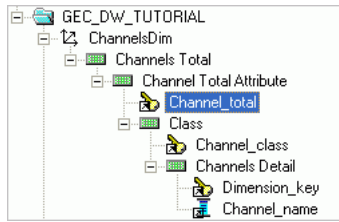
- 8. Right-click the **Channel Total Attribute** level and select **Expand All**. Drag the **Channel_total** column from the **Channels Detail** level to the **Channel Total Attribute** level. Then right-click **Channel_total** and select **New Logical Level Key**.

Figure 4-23 Logical Level Key: Channel_total



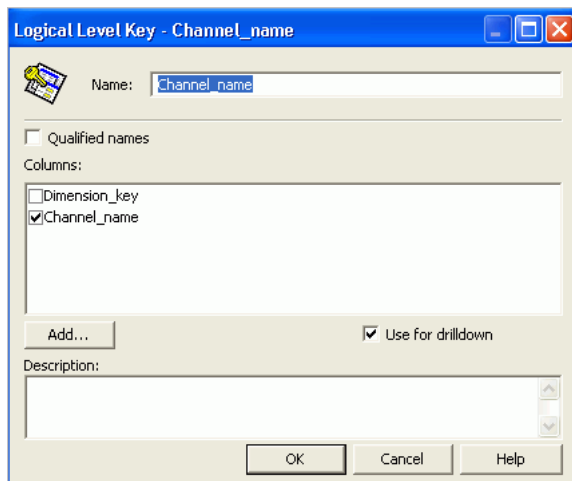
- 9. In the Logical Level Key dialog box, verify that **Channel_total** and **Use for drilldown** are selected. Click **OK** to close the dialog box. The **Channel_total** column now displays with a key icon.

Figure 4–24 Channel_total Column with Key Icon



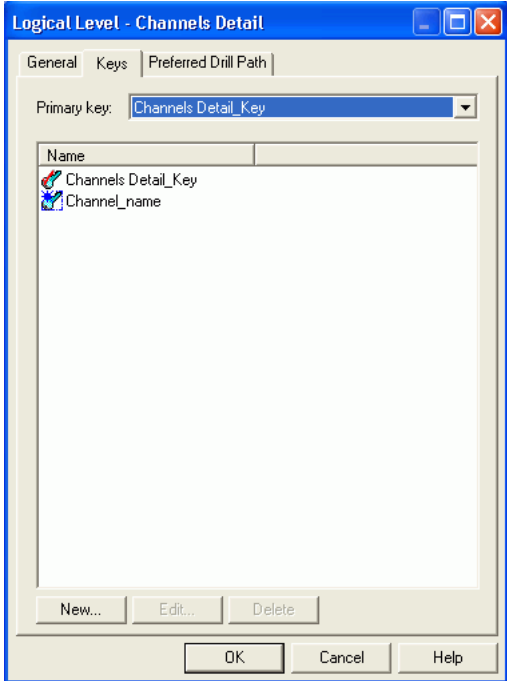
- Right-click the **Channel_name** column and select **New Logical Level Key**. In the Logical Level Key dialog box, notice that **Use for drilldown** is selected. Click OK to close the Logical Level Key dialog. Both **Channel_name** and **Dimension_key** display with key icons.

Figure 4–25 Logical Level Key: Channel_name



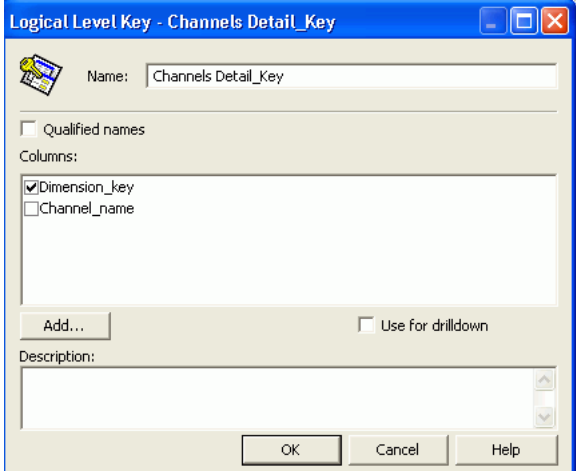
- Double-click the **Channels Detail** level to open the Logical Level dialog box. Click the **Keys** tab. Click **Channels Detail_Key** and then click **Edit**.

Figure 4-26 Logical Level: Channels Detail



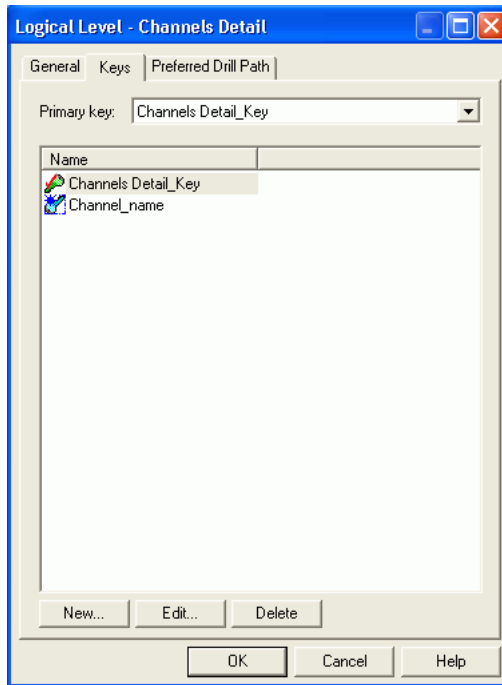
12. In the Logical Level Key dialog box, deselect **Use for drilldown**.

Figure 4-27 Logical Level Key: Channels Detail_Key



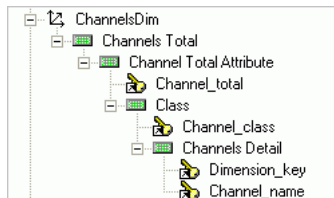
13. Click **OK** to close the Logical Level Key dialog box. Notice that the key icons are different. **Channel_name** is used for drill down, **Channels Detail_Key** is not. Later, when a user drills down in Answers or a dashboard, the default drill is to the level key that has **Use for drilldown** selected in the next lowest level. Based on this example, when a user drills down from the Channel Class column (the next highest level), the default is to drill down to the Channel_name column, not the Channels Detail_Key column.

Figure 4–28 Logical Level: Channels Detail



14. Click **OK** to close the Logical Level dialog box. The finished **ChannelsDim** hierarchy should look like the one shown in [Figure 4–29](#).

Figure 4–29 Channels Dim Hierarchy

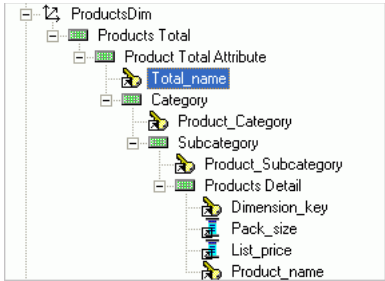


15. Save the repository. Click **OK** to check in changes, but answer **No** for consistency checking.

Next, you will build dimension hierarchies for the **Products**, **Geography**, **Times**, and **Promotions** logical tables. Use the preceding steps, the figures below, and the other characteristics below as a guide. If in doubt, you can check the dimensions in GEC_DW; the GEC_DW_TUTORIAL dimensions should be defined the same as the GEC_DW dimensions.

16. Build a dimension hierarchy for the **Products** logical table. Use the preceding steps, [Figure 4–30](#), and the characteristics below as a guide.

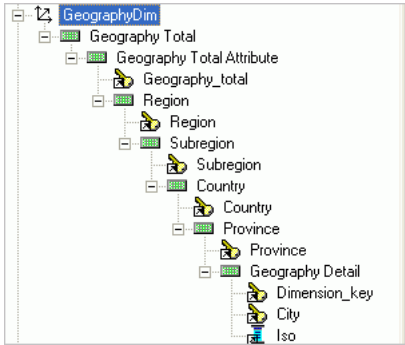
Figure 4-30 Dimension Hierarchy for Products



The ProductsDim dimension hierarchy should have the following characteristics:

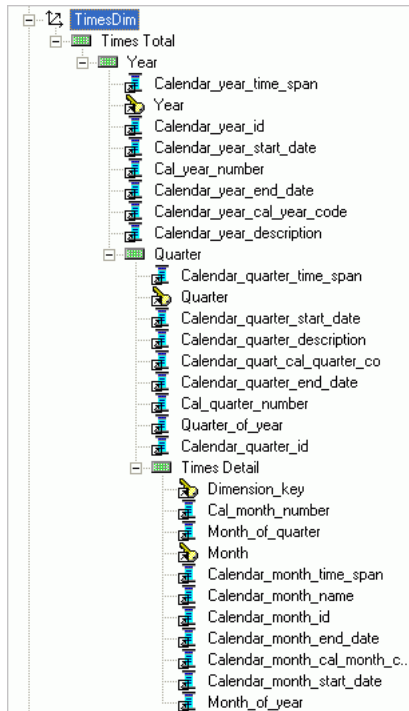
- a. For the **Subcategory** level, set **Number of elements at this level** to **21**.
 - b. For the **Category** level, set **Number of elements at this level** to **5**.
 - c. For the **Product Total Attribute** level, set **Number of elements at this level** to **1**.
 - d. For the **Products Detail** level, deselect **Use for drill down** for **Products Detail_key**.
 - e. Select **Use for drill down** for all other keys in the ProductsDim hierarchy.
 - f. If you complete this step, save the repository. Do not check global consistency.
17. Build a dimension hierarchy for the **Geography** logical table. Use the preceding steps and [Figure 4-31](#) as a guide. You can accept the default number of elements at each level.

Figure 4-31 Dimension Hierarchy for Geography



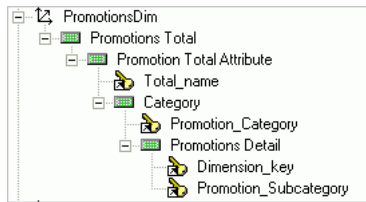
18. Build a dimension hierarchy for the **Times** logical table. Use the preceding steps and [Figure 4-32](#) as a guide. You can accept the default number of elements at each level.

Figure 4–32 Dimension Hierarchy for Times



19. Build a dimension hierarchy for the **Promotions** logical table. Use the preceding steps and [Figure 4–33](#) as a guide. You can accept the default number of elements at each level.

Figure 4–33 Dimension Hierarchy for Promotions



20. Save the repository. Click **OK** to check in changes, but answer **No** for consistency checking.

4.4 Building the Presentation Layer

In this section, you use the Oracle BI Administration Tool to build the Presentation layer of a repository.

The Presentation layer is built after the Physical layer and Business Model and Mapping layer, and adds a level of abstraction over the Business Model and Mapping layer. It is the view of the data seen by end users in client tools and applications, such as Oracle BI Answers. The Presentation layer provides a means to further simplify or customize the Business Model and Mapping layer for end users. For example, you can organize columns into catalogs and folders.

Simplifying the view of the data for users makes it easier to craft queries based on users' business needs, because you can expose only the data that is meaningful to the

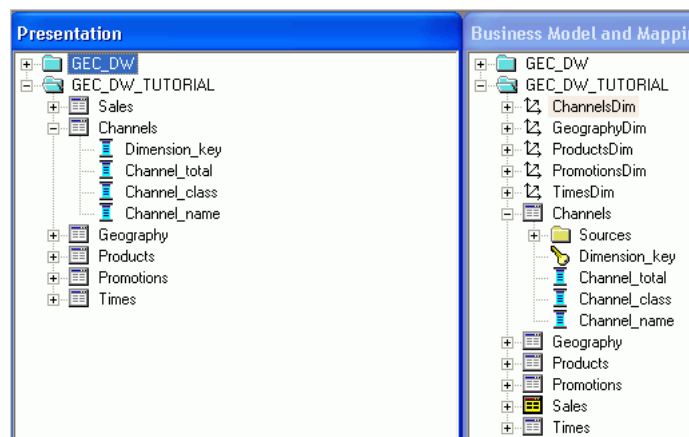
users, organize the data in a way that aligns with the manner in which users think about the data, and rename data as necessary for the set of users.

You typically create Presentation layer objects by dragging objects from the Business Model and Mapping layer. Corresponding objects are automatically created in the Presentation layer. Presentation layer objects can then be renamed and reorganized.

To build the Presentation layer:

1. Drag the **GEC_DW_TUTORIAL** business model from the Business Model and Mapping layer to the Presentation layer to create the **GEC_DW_TUTORIAL** catalog in the Presentation layer. Expand the **GEC_DW_TUTORIAL** catalog in the Presentation layer. Notice that the tables and columns in the Presentation layer exactly match the tables and columns in the Business Model and Mapping layer. Notice also that dimension hierarchies are not displayed.

Figure 4–34 Building the Presentation Layer



2. Save the repository. Do not check global consistency.

4.5 Testing and Validating a BI Repository

You have finished building the initial business model and now need to test the repository before continuing your development. You begin by checking the repository for errors using the check consistency option. You then test the repository by running queries using Oracle BI Answers. Finally, you examine the query log file to verify the SQL generated by Oracle BI Server.

This section contains the following topics:

- [Section 4.5.1, "Run a Consistency Check"](#)
- [Section 4.5.2, "Enable Query Logging"](#)
- [Section 4.5.3, "Use Oracle BI Answers to Execute Queries"](#)
- [Section 4.5.4, "Use the Query Log to Verify Queries"](#)

4.5.1 Run a Consistency Check

Consistency check is a utility in the Administration Tool that checks if a repository has met certain requirements. Repositories and the business models within them must pass the consistency check before you can make business models available for queries.

When a repository or business model is inconsistent, a detailed message alerts you to the nature of the inconsistency.

The Consistency Check Manager displays three types of messages:

- Error messages indicate errors that need to be fixed to make the repository consistent.
- Warning messages indicate conditions that may or may not be errors, depending upon the intent of the Oracle BI Server administrator. For example, if the Administrator user has an empty password, this should be addressed, but is not a requirement for a consistent repository.
- Best Practices messages provide information about conditions, but do not indicate an inconsistency. For example, if there are physical tables with no keys defined, a best practice message is displayed. Defining keys for physical tables is a best practice, but is not a requirement for a consistent repository.

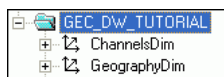
For each message, the Consistency Check Manager identifies the message type, the object type, the object, and provides a detailed description of the message. There are options to display only selected message types, display results using qualified names, check all objects in the repository, and copy the results to another file.

To run a consistency check:

1. Select **File > Check Global Consistency**.
2. You should receive a message indicating that the repository is consistent and asking if you want to make it available for queries. Click **Yes** to make the GEC_DW_TUTORIAL business model available for queries. Click **Yes** if prompted to check out the objects. The Consistency Check Manager appears.
3. If the Consistency Check Manager displays any **Error** messages, edit the repository to correct the inconsistencies and run the consistency check again.
4. If you see only **Warning** and **Best Practices** messages, you can ignore the messages for now and click **Close**.

In the Business Model and Mapping layer, notice that the GEC_DW_TUTORIAL business model icon has changed to indicate the business model is now available for queries (the red circle with a line is gone).

Figure 4–35 GEC_DW_TUTORIAL Business Model Icon

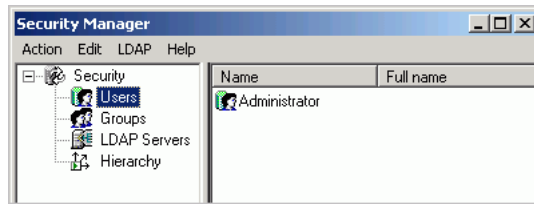


5. Save the repository. Click **No** when asked to check global consistency (you just checked it).
6. Double-click the **GEC_DW_TUTORIAL** business model object to open the Business Model properties dialog box. Click **Yes** if prompted on whether you want to check out the objects. Notice that **Available for queries** is selected. Click **OK** to close the dialog box.

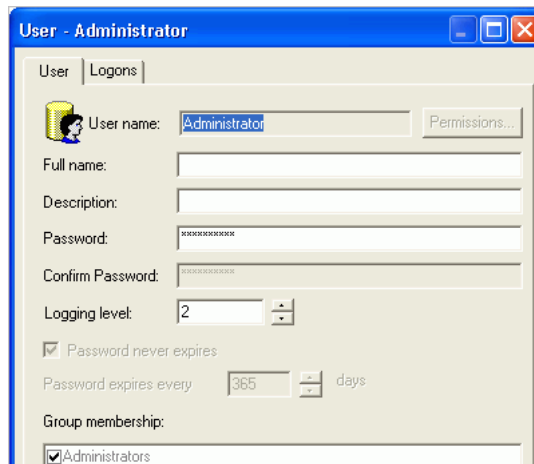
4.5.2 Enable Query Logging

To enable query logging:

1. Select **Manage > Security**. In the Security Manager, select **Users** in the left pane. The **Administrator** user appears in the right pane.

Figure 4–36 Security Manager: Users

2. In the right pane, double-click **Administrator**. Click **Yes** if prompted to check out the objects. The User dialog box opens. Verify that the **User** tab is selected. In the **Logging level** field, set the value to **2**. Click **OK**.

Figure 4–37 User Dialog Box: Administrator User

To test the repository, you will need to generate some queries, retrieve the results, and examine the query log. You log query activity at the individual user level. Logging is intended for testing, debugging, and technical support. In production mode, logging is normally disabled because query logging can impact performance by producing very large log files.

3. Close the Security Manager window.
4. Check global consistency. You can ignore the warning messages. Click **Close** in the Consistency Check Manager dialog box.
5. Save the repository.

4.5.3 Use Oracle BI Answers to Execute Queries

To use Oracle BI Answers to execute queries:

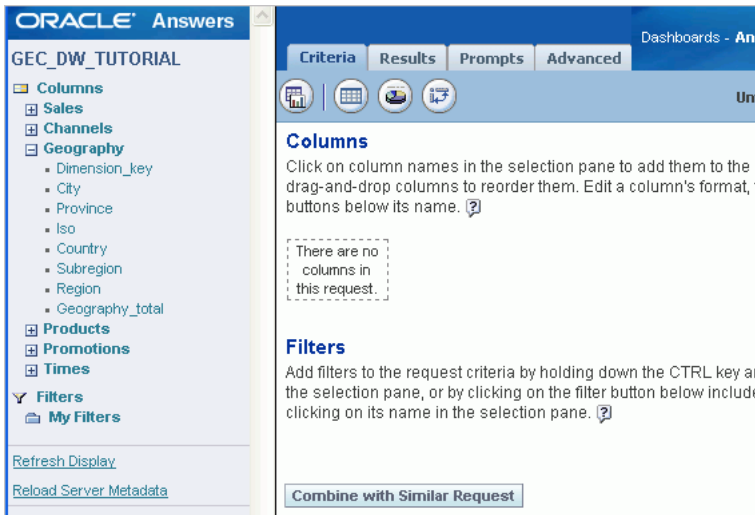
1. Select **Start > Programs > Oracle Business Intelligence > Presentation Services**. Log in to Oracle Business Intelligence as **Administrator** with password **Administrator**.
2. Click the **Answers** link, then click the **GEC_DW_TUTORIAL** subject area.

Figure 4–38 Oracle BI Answers: Choosing the GEC_DW_TUTORIAL Subject Area

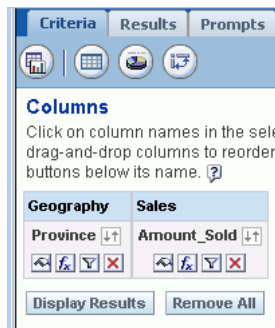


3. In the left pane, click the **Geography** folder to expand it. Notice that the folders and columns in Answers match the folders and column in the Presentation layer of the repository.

Figure 4–39 Oracle BI Answers: Geography Folder



4. Click the **Province** column in the left pane to add it to the request criteria on the right. Click the **Amount_Sold** column from the **Sales** folder to add it to the request criteria.

Figure 4–40 Oracle BI Answers: Province and Amount_Sold Columns

5. Click the **Results** tab. By default, results are displayed in a compound layout, consisting of a title and a table view.

Figure 4–41 Oracle BI Answers: Results Tab

The screenshot shows the 'Results' tab in Oracle BI Answers. It features a 'Compound Layout' dropdown menu and an 'Add View' button. Below this, there is a 'Title' field and a 'Table' view. The table displays the following data:

Province	Amount_Sold
Aichi (52543)	\$1,295,549
Alabama (52534)	\$1,165,702
Alaska (52533)	\$185,916
Alberta (52608)	\$165,042
Alberta (52637)	\$148,349
Alicante (52544)	\$499,041
Almeria (52545)	\$90,406
Alsace (52546)	\$104,161

4.5.4 Use the Query Log to Verify Queries

To use the query log to verify queries:

1. On the top right portion of the page, click **Settings > Administration** to open the Oracle BI Presentation Services Administration Window. Click the **Manage Sessions** link to open the Session Management Window.
2. In the Session Management Window, under Cursor Cache, click the **View Log** link for the last entry.

The log displays the last query executed by **Administrator**. The log file should look similar to [Figure 4–42](#).

Figure 4–42 Query Log File

```

+++Administrator:2b0000:2b0008:----2007/04/23 13:48:57
#####
----- SQL Request:
SET VARIABLE QUERY_SRC_CD='Report';SELECT Geography.Province saw_0, Sales.Amount_Sold

+++Administrator:2b0000:2b0008:----2007/04/23 13:48:57
----- General Query Info:
Repository: Star, Subject Area: GEC_DW_TUTORIAL, Presentation: GEC_DW_TUTORIAL

+++Administrator:2b0000:2b0008:----2007/04/23 13:48:57
----- Sending query to database named BISE1DB (id: <<2875>>):

select T13804.PROVINCE_NAME as c1,
       sum(T13862.AMOUNT) as c2
from
  GEOGRAPHY T13804,
  SALES T13862
where ( T13804.DIMENSION_KEY = T13862.GEOGRAPHY )

```

3. Locate the **SQL Request** section. This section contains the logical SQL issued from Answers.
4. Locate the **General Query Info** section, just below the SQL Request section. This section identifies the repository, subject area, and presentation catalog from which the query was run.
5. Locate the **Sending query to database named BISE1_TUTORIALWH** section, just below the General Query Info section. This section identifies the physical data source to which Oracle BI Server is connecting and the physical SQL that was generated.

The rest of the file contains information such as query status, number of rows returned, and so forth.

6. Close the query log. Close the Session Management window. Close the Oracle BI Presentation Services Administration window. Leave Answers open.

4.6 Creating Calculation Measures

Often, a business wants to compare values of a measure and needs a calculation to express the comparison. Oracle BI Server has a calculation engine to perform a multitude of calculations. Calculation measures allow end users to ask business questions like "Show me the accounts receivable balance as of Q3" or "Show me the difference between units ordered and units shipped." An Expression Builder enables you to create expressions that are similar to expressions created with SQL. In the examples in this lesson, you use the Expression Builder to create calculation measures that appear as columns to users in Answers. Users can then easily build queries using familiar terminology.

There are different methods for creating calculation measures in the Administration Tool. You can use existing logical columns as objects in a formula, use physical columns as objects in a formula, or use the Calculation Wizard to automate the process. All three methods are covered in this tutorial. You use physical columns for calculations that require an aggregation rule to be applied *after* the calculation. You use logical columns for calculation formulas that require an aggregation rule that is applied *before* the calculation. You can also build calculation measures in Answers. The advantages to building calculation measures in the repository is that the measures are built once and can be made available to all users. The advantage of defining a logical column formula based on existing logical columns is that you only have to define it

once. When you create formulas based on physical columns, you have to map for each physical source from which it could be derived.

This section contains the following topics:

- [Section 4.6.1, "Create a New Measure"](#)
- [Section 4.6.2, "Create a Calculation Measure Using Logical Columns"](#)
- [Section 4.6.3, "Create a Calculation Measure Using Physical Columns"](#)
- [Section 4.6.4, "Create a Calculation Measure Using the Calculation Wizard"](#)

4.6.1 Create a New Measure

To create a new measure:

1. Return to the Oracle BI Administration Tool. In the **Business Model and Mapping Layer**, navigate to the **Cost** logical column under the **Sales** logical column.
2. Set the aggregation rule for the **Cost** logical column to **Sum**. Double-click the column to open the Logical Column properties dialog box and click the **Aggregation** tab. If prompted to check out objects, click **Yes**. Click **OK**.
3. Click **File > Check In Changes** or click the **Check In Changes** icon on the toolbar to check in changes.
4. Click **Yes** when prompted to check global consistency. The Consistency Check Manager opens and displays Warnings and Best Practices messages. Review the messages.
5. Click **Close** to close the Consistency Check Manager. Save the repository.
6. Return to Answers to test the new column in Answers. If Answers is not open, select **Start > Programs > Oracle Business Intelligence > Presentation Server**, log in as **Administrator** with password **Administrator**, click the **Answers** link, and click the **GEC_DW_TUTORIAL** subject area.
7. If Answers was already open, and if the **Geography.Province** column is already selected (in the request created in the previous section), remove it by clicking the **X** under the column name.
8. Click **Reload Server Metadata**.
9. Build the following request: **Geography.Region, Sales.Amount_Sold, Sales.Cost**. Select these columns from the left navigation tree, and then click the **Results** tab to view the results.

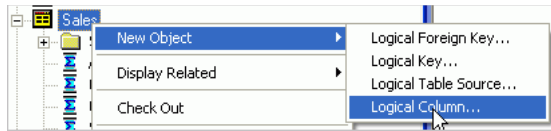
4.6.2 Create a Calculation Measure Using Logical Columns

In this topic, you define a new calculation measure named **Gross Profit** in the **Sales** logical table, using existing logical columns to define the calculation formula. This measure can be used by any requests in Answers. There is a similar calculated measure called **Profit** in **GEC_DW**, which is used in several requests on the **Sales Analysis** dashboard.

To create a calculation measure using logical columns:

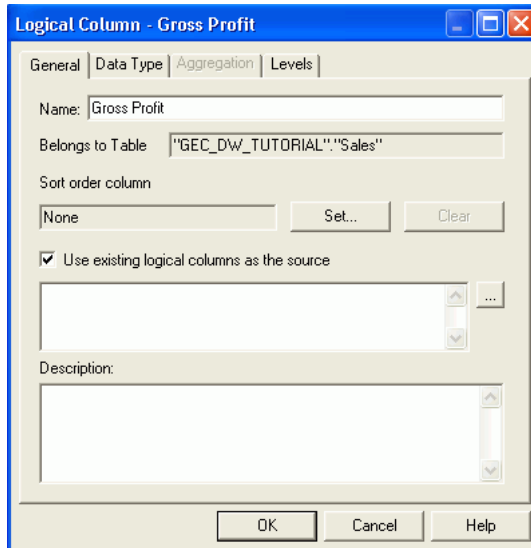
1. In the Oracle BI Administrator Tool, right-click the **Sales** logical table and select **New Object > Logical Column**.

Figure 4–43 Creating a Logical Column



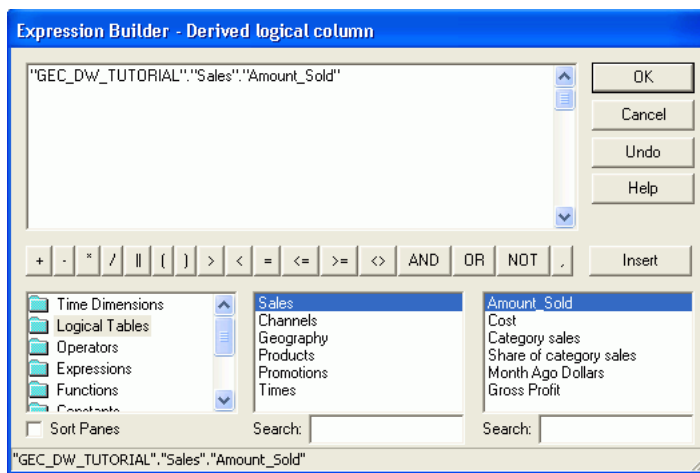
2. In the Logical Column dialog box, name the logical column **Gross Profit** and select **Use existing logical columns as the source**.

Figure 4–44 Logical Column: Gross Profit



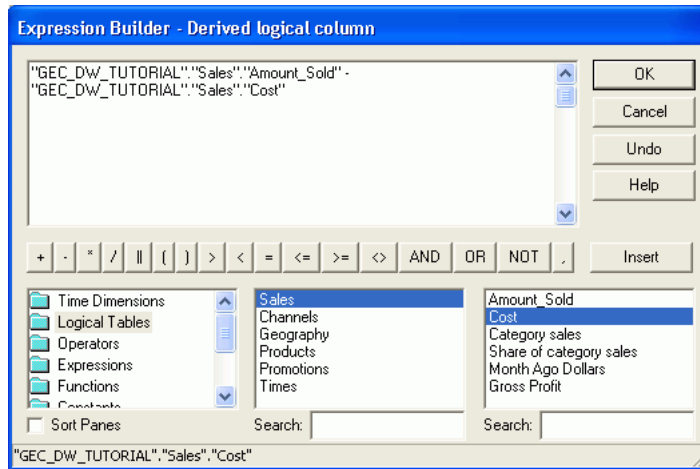
3. Click the ellipsis (...) to open the Expression builder. In the left pane, click **Logical Tables**. Select **Sales** in the middle pane, then double-click **Amount_Sold** in the right pane to add it to the formula.

Figure 4–45 Expression Builder: Amount_Sold



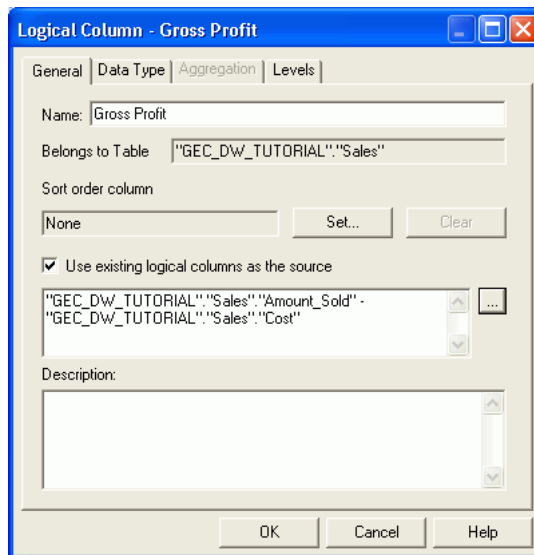
4. Click the **minus sign operator** to add it to the formula. Double-click **Cost** in the right pane to add it to the formula.

Figure 4–46 Expression Builder: Cost



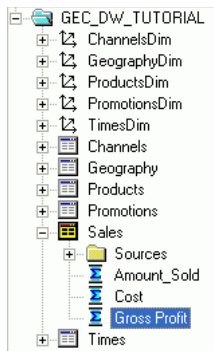
5. Click **OK** to close the Expression Builder. Notice that the formula appears in the Logical Column dialog box.

Figure 4–47 Logical Column: Formula



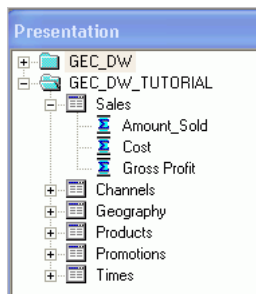
6. Click **OK** to close the Logical Column dialog box. The Gross Profit logical column appears in the business model. Check in the changes.

Figure 4–48 Business Model: Gross Profit Logical Column



7. Drag the **Gross Profit** logical column to the **Sales** table in the Presentation layer. Check in the changes. Save the repository and return to Answers.

Figure 4–49 Adding Gross Profit to the Presentation Layer



8. In Answers, click **Reload Server Metadata**. Expand **Sales** and verify that the **Gross Profit** column is now visible in Answers. Create the following request:
Geography.Region, Sales.Amount_Sold, Sales.Cost, Sales.Gross Profit
9. Click **Results**. Verify that the results are the same as the results shown in [Figure 4–50](#).

Figure 4–50 Oracle BI Answers: Results

Region	Amount_Sold	Cost	Gross Profit
Americas	\$55,318,792	43,903,858	\$11,414,934
Asia	\$10,351,539	8,199,199	\$2,152,340
Europe	\$28,664,416	22,689,256	\$5,975,159
Middle East	\$629	507	\$122
Oceania	\$3,960,308	3,128,068	\$832,241

10. Click **Settings > Administration > Manage Sessions > View Log** to view the query log. Verify that **Gross Profit** item was used.

Figure 4–51 Query Log: Gross Profit

```

+++Administrator:2a0000:2a0008:----2007/04/23 16:21:36
----- Sending query to database named biselddb (id: <<3187>>):
WITH
SAWITH0 AS (select sum(T12049.COST) as c1,
              sum(T12049.AMOUNT) as c2,
              T11994.REGION_NAME as c3
from
  GEOGRAPHY T11994,
  SALES T12049
where ( T11994.DIMENSION_KEY = T12049.GEOGRAPHY )
group by T11994.REGION_NAME)
select distinct SAWITH0.c3 as c1,
              SAWITH0.c2 as c2,
              SAWITH0.c1 as c3,
              SAWITH0.c2 - SAWITH0.c1 as c4
from
  SAWITH0

```

Note that the difference between AMOUNT and COST is being calculated in the outer query block (SAWITH0.c2 - SAWITH0.c1 as c4 in the example shown in [Figure 4–51](#)). Because you defined the Gross Profit calculation using logical columns, the columns are summed first and then the difference is calculated. You compare these results to the query results in the next practice.

11. Close the query log, Session Management, and BI Presentation Services Administration windows.

4.6.3 Create a Calculation Measure Using Physical Columns

In this section, you define a new calculation measure named **Gross Profit Physical** in the **Sales** logical table, using physical columns to define the calculation formula. To do this:

1. Return to the Oracle BI Administration Tool. In the **Business Model and Mapping Layer**, right-click the **Sales** logical table and select **New Object > Logical Column**.

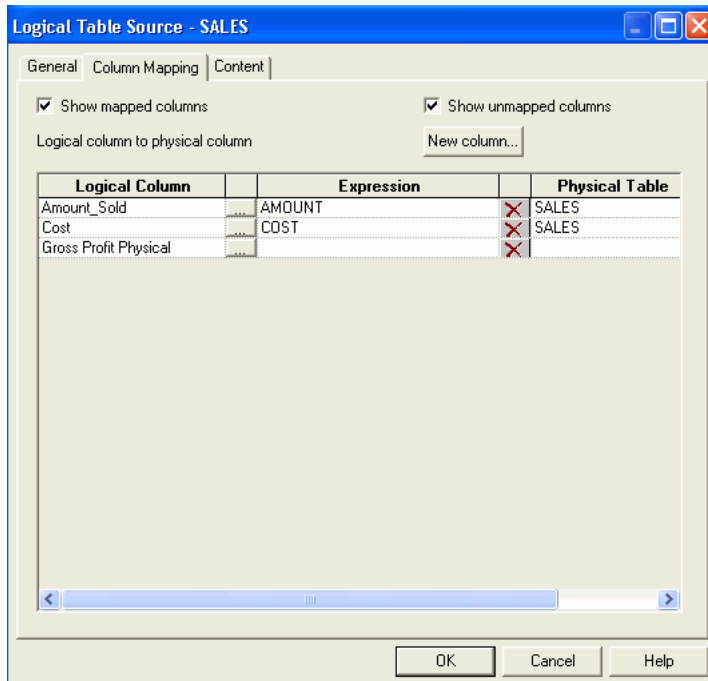
Note: Click **Yes** to check out objects whenever prompted to do so.

2. In the Logical Column dialog box, name the logical column **Gross Profit Physical**.
3. Click the **Aggregation** tab. Set the default aggregation rule to **Sum**.
4. Click **OK** to close the Logical Column dialog box. Gross Profit Physical is added to the business model.

Figure 4–52 Business Model: Gross Profit Physical

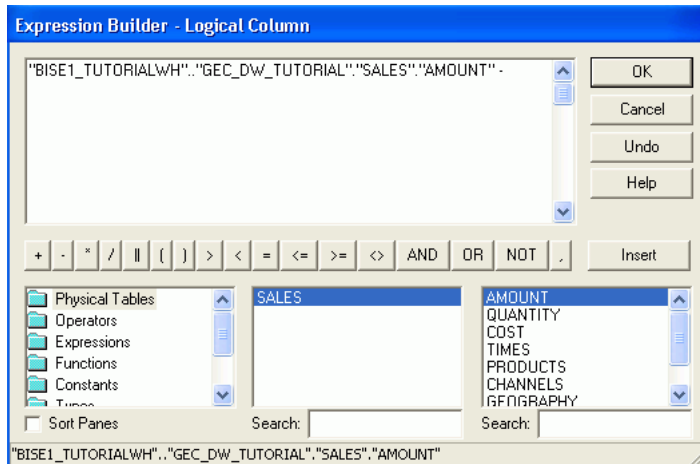
5. Expand **Sales > Sources** and double-click the **SALES** logical table source. The Logical Table Source dialog box opens. Click the **Column Mapping** tab.
6. Select the **Show unmapped columns** option.

Figure 4–53 Logical Table Source: Column Mapping Tab



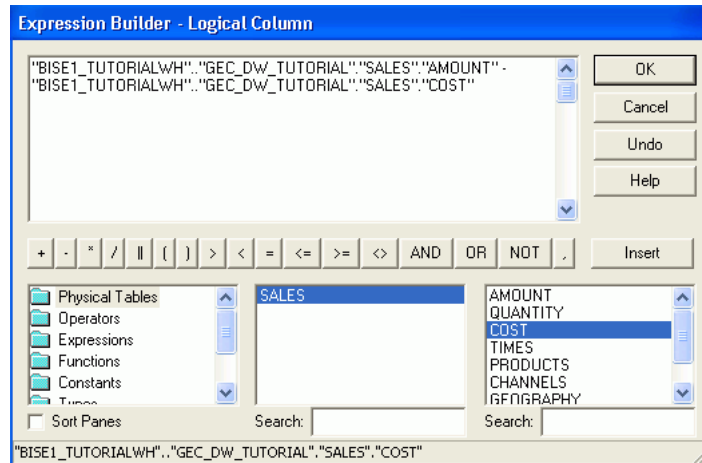
7. Click the ellipsis (...) for the **Gross Profit Physical** logical column.
8. In the Expression Builder, select **Physical Tables > SALES > AMOUNT**, then click **Insert** to add the column to the formula. Click the **minus sign operator** to add it to the formula.

Figure 4–54 Expression Builder: AMOUNT



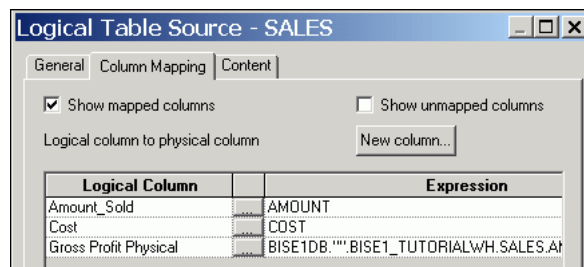
9. Select **Physical Tables > SALES > COST**, then click **Insert** to add the column to the formula.

Figure 4–55 Expression Builder: COST



10. Click **OK** to close the Expression Builder. Notice that the expression is added in the Logical Table Source dialog box.

Figure 4–56 Logical Table Source: SALES



11. Close the Logical Table Source dialog box. The icon for Gross Profit Physical changes to indicate an aggregation rule is applied.

Figure 4–57 Gross Profit Physical Icon



12. Drag **Gross Profit Physical** to **Sales** in the Presentation layer. Check in changes and save the repository. Return to Answers. Click **Reload Server Metadata**. Expand **Sales** and verify that the **Gross Profit Physical** column is now visible in Answers.
13. Create the following request:
Geography.Region, Sales.Amount_Sold, Sales.Cost, Sales.Gross Profit Physical
14. Click **Results**. Verify that the results are the same for the Gross Profit Physical (built using physical columns) as they were for the Gross Profit column (built using logical columns).

The calculation formula for the logical columns looks like this: `sum(Amount_Sold) - sum(Cost)`, whereas the calculation formula for the physical columns looks like this: `sum(AMOUNT_SOLD - COST)`. Because of arithmetic laws, you know that you can sum ColumnA and sum ColumnB and then take the differences of those sums, and have exactly the same results if you calculate the difference first (the value in ColumnA - the value in ColumnB for each row) and then sum the difference. So in this example, the results are the same for the logical column and the physical column calculations.

15. Click **Settings > Administration > Manage Sessions > View Log** to view the query log. Your results should look similar to [Figure 4-58](#).

Figure 4-58 Query Log: Gross Profit Physical

```

+++Administrator:2e0000:2e0004:----2007/02/12 14:04:58
----- Sending query to database named BISE1_TUTORIALWF
select T9220.REGION_NAME as c1,
       sum(T9278.AMOUNT) as c2,
       sum(T9278.COST) as c3,
       sum(T9278.AMOUNT - T9278.COST) as c4
from
  GEOGRAPHY T9220,
  SALES T9278
where ( T9220.DIMENSION_KEY = T9278.GEOGRAPHY )
group by T9220.REGION_NAME
order by c1
    
```

Note that the difference between Amount_Sold and Cost is calculated first and then summed: `sum(T9278.AMOUNT- T9278.COST)` in the example shown in [Figure 4-58](#).

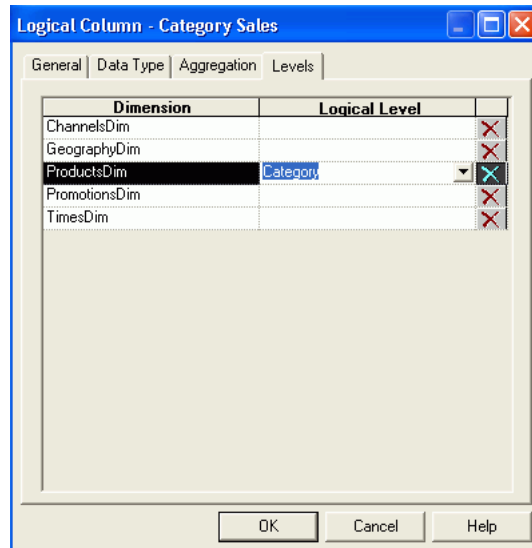
16. Close all the windows.

4.6.4 Create a Calculation Measure Using the Calculation Wizard

To create a calculation measure using the wizard:

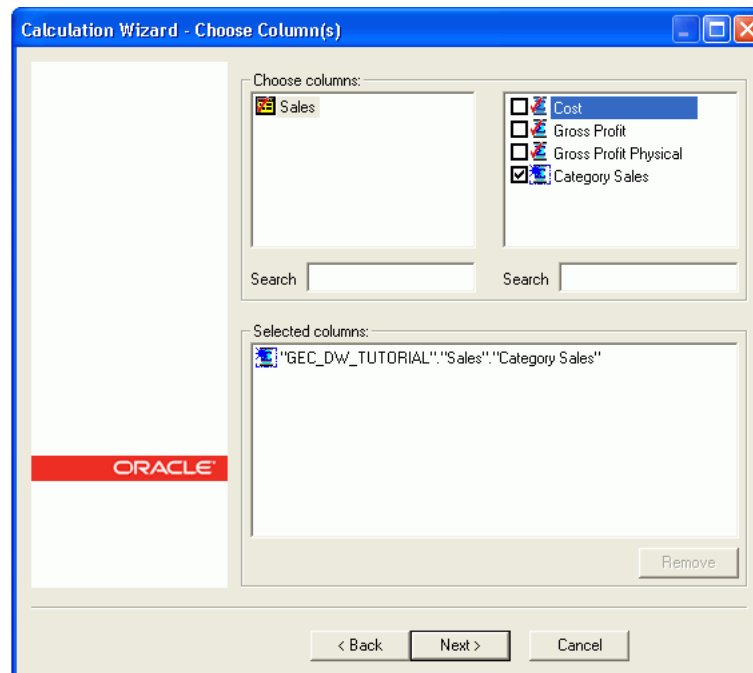
1. Return to the Oracle BI Administration Tool. In the **Business Model and Mapping Layer**, right-click the **Sales > Amount_Sold** logical column and select **Duplicate**. A new column named **Amount_Sold#1** is added to the business model. Rename **Amount_Sold#1** to **Category Sales**.
2. Double-click **Category Sales** to open the Logical Column dialog box. Click the **Levels** tab and select **Category** as the logical level for **ProductsDim**. Click **OK**.

Figure 4–59 Logical Column: Category Sales

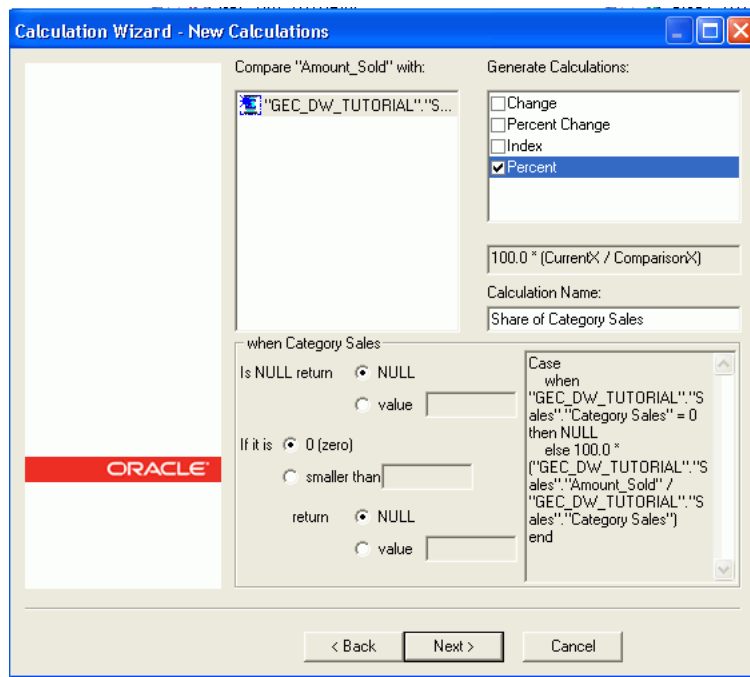


3. Category Sales is now a level-based measure that will calculate total sales at the category level when used in a query. Level-based measures are useful for creating share measures. You use the Calculation Wizard to create a share measure in the steps that follow. Close the dialog box.
4. Right-click **Amount_Sold** and select **Calculation Wizard**. The Calculation Wizard opens. Click **Next**. Select **Category Sales** as the column to compare with Amount_Sold. Click **Next**.

Figure 4–60 Calculation Wizard: Choose Columns



5. Deselect **Change** and **Percent Change**, but **select Percent**. Change **Calculation Name** to **Share of Category Sales**. Click **Next**, then click **Next** again if asked whether you want to check the objects out. Click **Finish**.

Figure 4–61 Calculation Wizard: New Calculations

6. **Share of Category Sales** is added to the business model. Drag **Category Sales** and **Share of Category Sales** to Sales in the Presentation layer. Check in the changes. Save the repository.
7. In Oracle BI Answers, click **Reload Server Metadata**. Expand **Sales** and verify that **Category Sales** and **Share of Category Sales** are now visible in Answers.

4.7 Using Initialization Blocks and Variables - Advanced

You can use variables in a repository to streamline administrative tasks and modify metadata content dynamically to adjust to a changing data environment. A variable has a single value at any point in time. Variables can be used instead of literals or constants in the Expression Builder in the Administration Tool. Oracle BI Server substitutes the value of a variable for the variable itself in the metadata.

You use the Variable Manager to define variables and initialization blocks. There are two classes of variables, repository variables and session variables:

- A repository variable has a single value at any point in time. There are two types of repository variables: static and dynamic. Static repository variables have values that are constant and do not change while Oracle BI Server is running. Dynamic repository variables have values that are refreshed by data returned from queries in initialization blocks. Repository variables are represented by a question mark icon in the Variable Manager.
- Session variables are created with and assigned a value when each user logs on. There are two types of session variables: system and nonsystem. System variables have reserved names and are used for specific purposes by Oracle BI Server, such as authenticating users. Non-system variables are application-specific variables created by an Administrator. System and nonsystem variables are represented by a question mark icon in the Variable Manager.

Initialization blocks are used to initialize dynamic repository variables, system session variables, and nonsystem session variables.

This section contains the following topics:

- [Section 4.7.1, "Explore an Initialization Block for Session Variables"](#)
- [Section 4.7.2, "Test the Initialization Block and Session Variables"](#)
- [Section 4.7.3, "Create a Dynamic Repository Variable"](#)

4.7.1 Explore an Initialization Block for Session Variables

Session variables are like dynamic repository variables in that they obtain their values from initialization blocks. Unlike dynamic repository variables, however, the initialization of session variables is not scheduled.

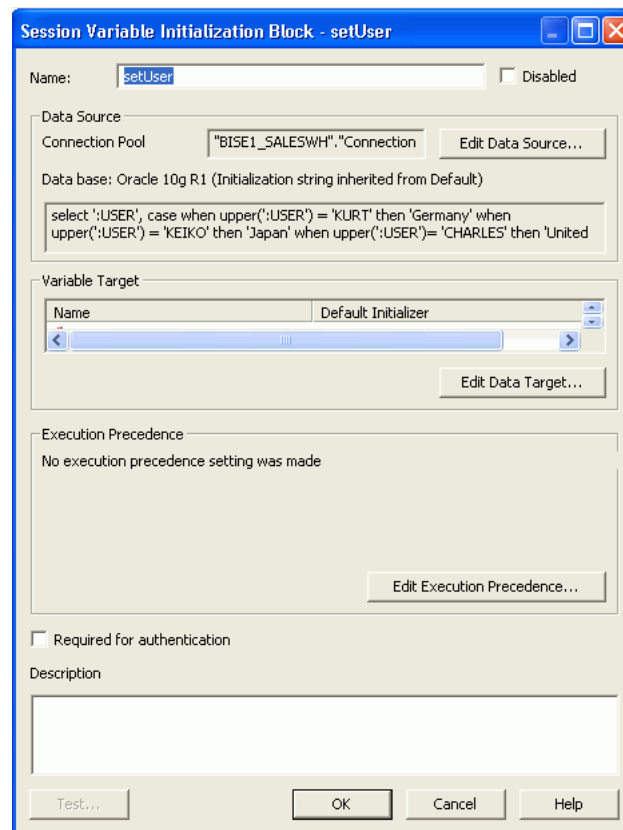
When a user begins a session, Oracle BI Server creates new instances of session variables and initializes them. Unlike a repository variable, there are as many instances of a session variable as there are active sessions on Oracle BI Server. Each instance of a session variable could be initialized to a different value.

A session is an instance of a user running the client application. The session starts when the application is started and ends when the application is exited.

To explore an initialization block for session variables:

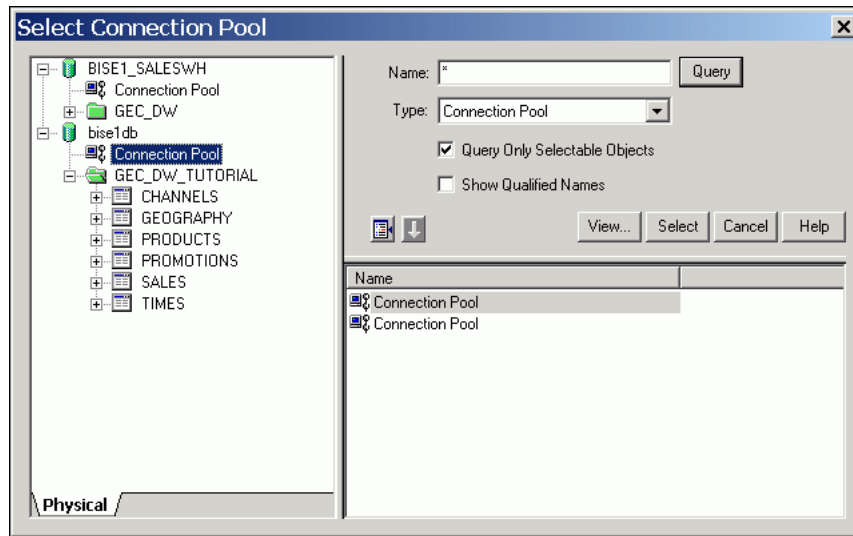
1. In the Oracle BI Administration Tool, click **Manage > Variables** to open the Variable Manager. Click **Session > Initialization Blocks**.
2. A session initialization block named `setUser` has already been created. Double-click `setUser` to open the Session Variable Initialization Block dialog box.

Figure 4–62 Session Variable Initialization Block: `setUser`



- Click **Edit Data Source** to open the Session Variable Initialization Block Data Source dialog box. Click **Browse**, then select **bise1db > Connection Pool** in the Select Connection Pool dialog box.

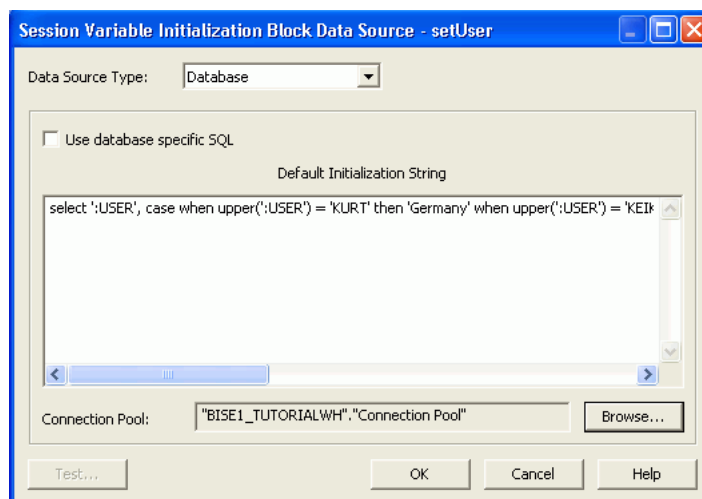
Figure 4–63 Select Connection Pool Dialog Box



- Click **Select** and notice that the connection pool is visible in the **Connection Pool** field in the Session Variable Initialization Block Data Source dialog box.
- In the **Default Initialization String** field, notice the initialization string. You may need to scroll to the right to see the entire string:

```
select ':USER', case when upper(':USER') = 'KURT' then 'Germany' when
upper(':USER') = 'KEIKO' then 'Japan' when upper(':USER')= 'CHARLES' then
'United Kingdom' when upper(':USER') = 'KAREN' then 'United States of America'
end, 'CountryManagers', 2 from Dual
```

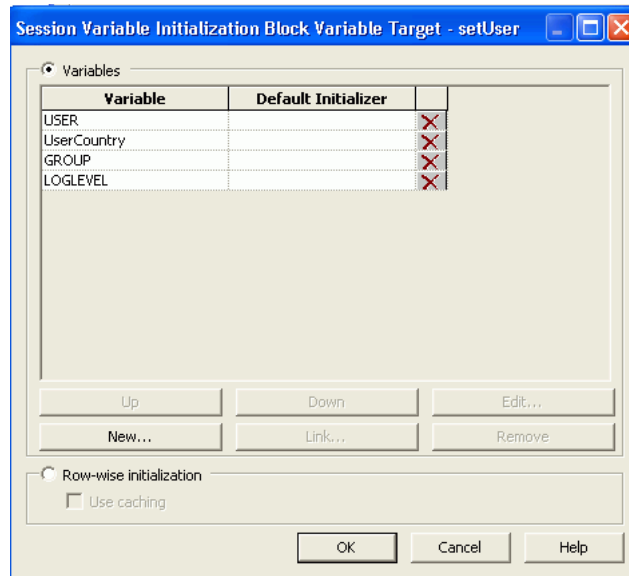
Figure 4–64 Session Variable Initialization Block Data Source



- Click **OK** to close the Session Variable Initialization Block Data Source dialog box. Notice that the initialization string and connection pool are visible in the Session Variable Initialization Block dialog box. Click **Edit Data Target** to open the Session

Variable Initialization Block Variable Target dialog box. Notice that the four variables have been created: USER, UserCountry, GROUP, and LOGLEVEL.

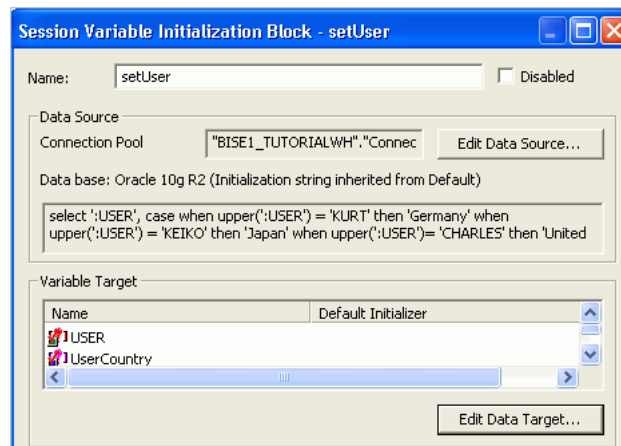
Figure 4–65 Session Variable Initialization Block Variable Target



Notice the order of the variables. The order is important. The order of the variables must match the order of the values for the variables in the initialization string in the initialization block.

7. Click **OK** to close the Session Variable Initialization Block Variable Target dialog box. The variables are displayed in the Variable Target section of the Session Variable Initialization Block dialog box. Close all the dialog windows.

Figure 4–66 Session Variable Initialization Dialog Box: Variable Targets



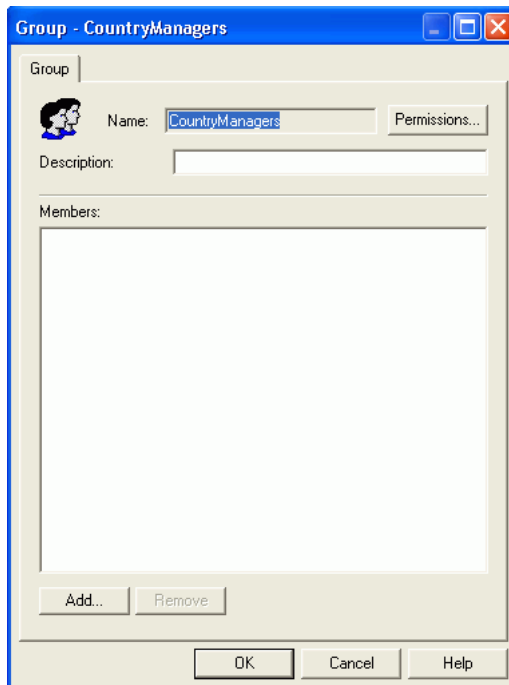
8. Check in the changes and save the repository.

4.7.2 Test the Initialization Block and Session Variables

To test the initialization block and session variables:

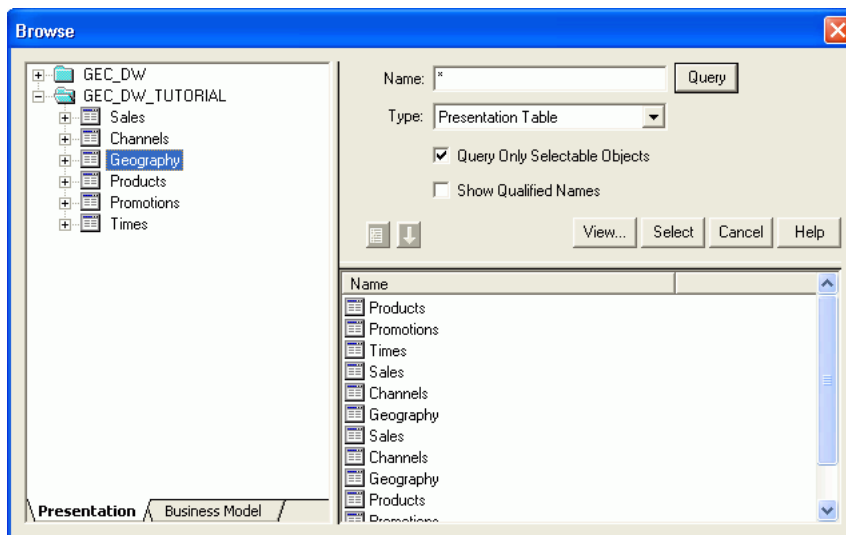
1. Click **Manage > Security** to open the Security Manager. Click **Groups** in the left pane. In the right pane, double-click the group **CountryManagers**.

Figure 4–67 Groups: CountryManagers

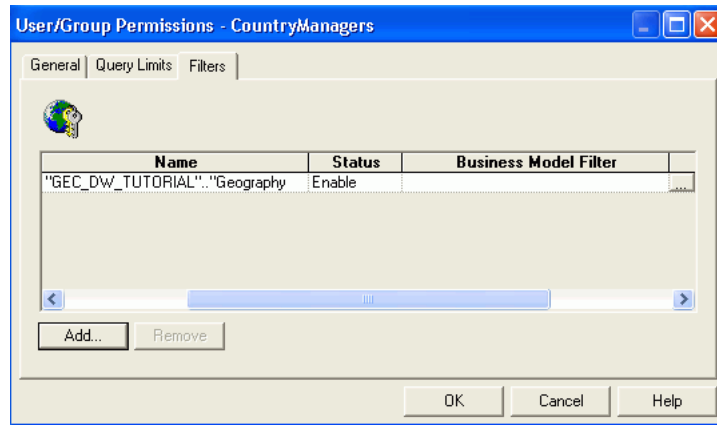


2. Click **Permissions** to open the User/Group Permissions dialog box. Click the **Filters** tab. Click the **Add** button.
3. Expand **GEC_DW_TUTORIAL** and click the **Geography** presentation table.

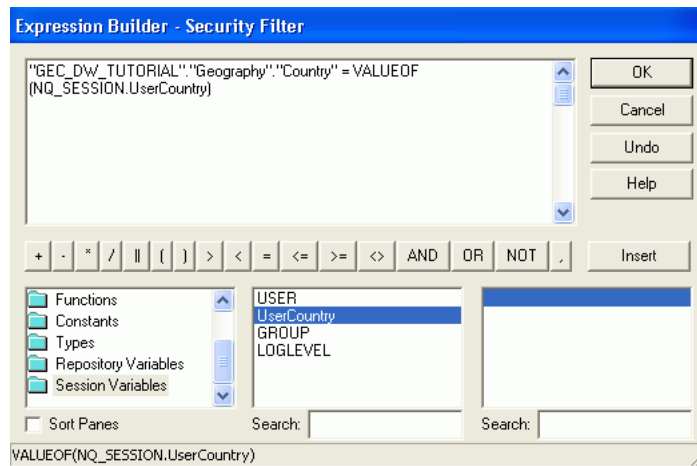
Figure 4–68 Geography Presentation Table



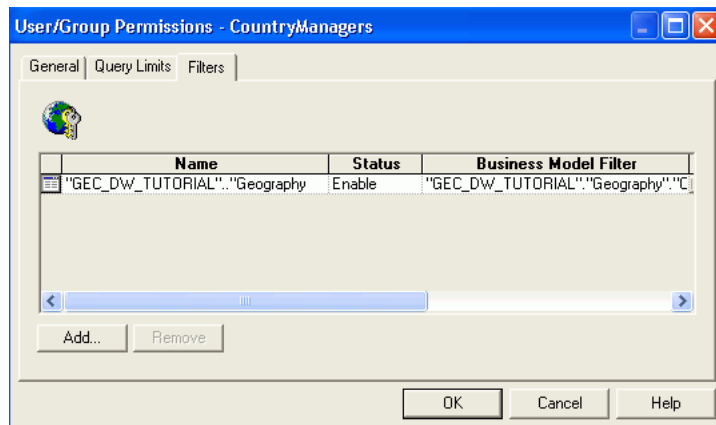
4. Click **Select** to add Geography to the **Name** field in the User/Group Permissions dialog box.

Figure 4–69 User/Group Permissions: Geography

5. Click the ellipsis (...) on the right to open the Expression Builder (you may need to scroll to see the button).
6. Select **Logical Tables > Geography > Country** and then click **Insert** to add **Country** to the formula. Click the equals (=) operator to add it to the formula.
7. Select **Session Variables > UserCountry** and click **Insert** to add **UserCountry** to the formula as an argument in the **VALUEOF()** function. Compare with the expression in [Figure 4–70](#).

Figure 4–70 Expression Builder: Security Filter

8. Click **OK** to close the Expression Builder. The filter is added in the User/Group Permissions dialog box. Close the User/Group Permissions dialog box.

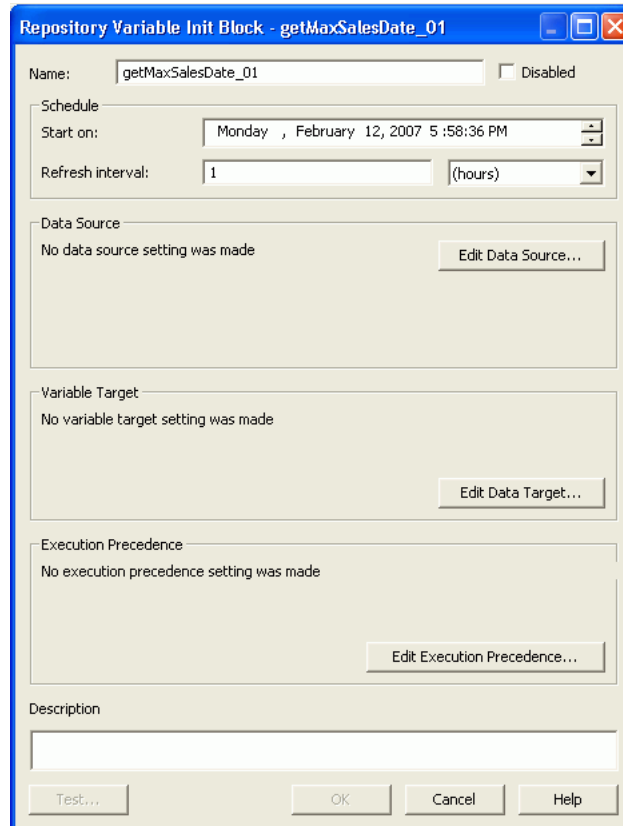
Figure 4–71 User/Group Permissions: Filters Tab

9. Close the Group dialog box. Close Security Manager. Check in the changes and save the repository.
10. Go to Oracle BI Answers. If you are already logged in as Administrator, log out, then log in using **Keiko** as the username, with no password.
11. Select the **GEC_DW_TUTORIAL** subject area. Create a new request with **Country** under **Geography**, and **Amount_Sold** under **Sales**.
12. Click the **Results** tab. Notice that the only country displayed is **Japan**.
13. Leave Answers open.

4.7.3 Create a Dynamic Repository Variable

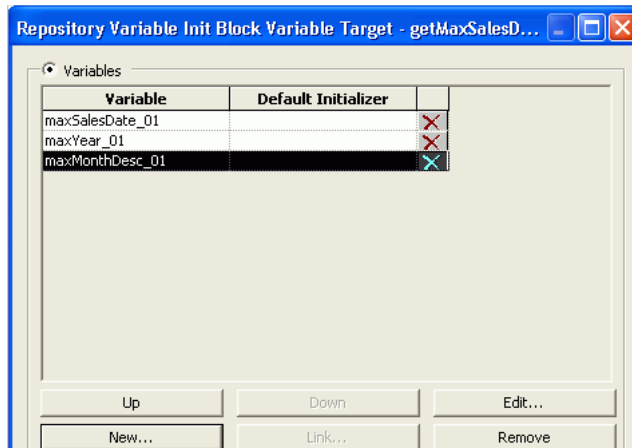
To create a dynamic repository variable:

1. In the Oracle BI Administration Tool, click **Manage > Variables** to open the Variable Manager. Click **Repository > Initialization Block**.
2. Right-click the white space and select **New Initialization Block** to open the Repository Variable Init Block dialog box. Name the initialization block **getMaxSalesDate_01**.

Figure 4–72 Repository Variable Init Block: getMaxSalesDate_01

3. Click **Edit Data Source** to open the Repository Variable Init Block Data Source dialog box. Click **Browse** to open the Select Connection Pool dialog box.
4. Double-click the **bise1db > Connection Pool** object to add it to the **Connection Pool** field in the Repository Variable Init Block Data Source dialog box. In the **Default Initialization String** field, type the following SQL:


```
select dimension_key, calendar_year_name, calendar_month_description from times
where dimension_key=(select max(times) from sales)
```
5. Click **OK** to close the Repository Variable Init Block Data Source dialog box. The connection pool and initialization string are added to the Repository Variable Init Block dialog box.
6. Click **Edit Data Target** to open the Repository Variable Init Block Variable Target dialog box.
7. Use the **New** button to create three variables: **maxSalesDate_01**, **maxYear_01**, and **maxMonthDesc_01**. The order is important. The order of the variables must match the column order in the initialization string.

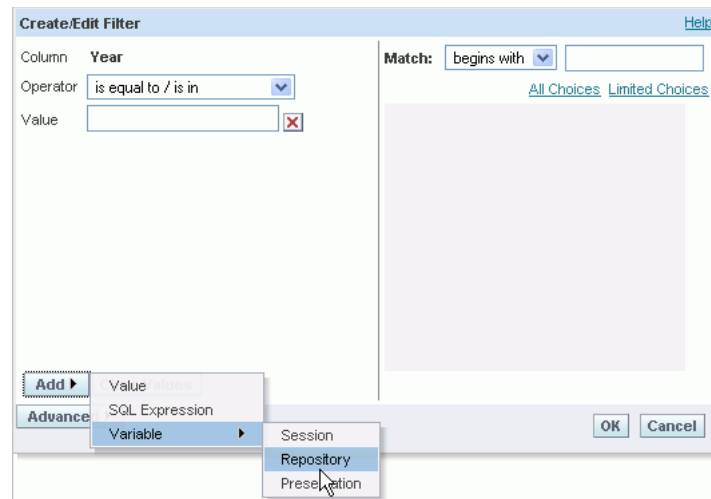
Figure 4–73 New Repository Variables

8. Click **OK** to close the Repository Variable Init Block Variable Target dialog box. The variables appear in the Variable Target field in the Repository Variable Init Block dialog box.
9. Click **Edit Data Source** to open the Repository Variable Init Block Data Source dialog box. Click **Test** and verify that you get the results shown in [Figure 4–74](#).

Figure 4–74 Results

Variable	Value
maxSalesDate_01	68.00
maxYear_01	2006
maxMonthDesc_01	December 2006

10. Close Results. Close the Repository Variable Init Block Data Source dialog box. The getMaxSalesDate_01 initialization block is displayed in the Variable Manager.
11. Select **Repository > Variables > Dynamic** to see the variables displayed in the Variable Manager. Close Variable Manager. Check in changes and save the repository.
12. In Oracle BI Answers, use the **GEC_DW_TUTORIAL** subject area to build a request with the following:
Times. Year, Sales.Amount_Sold
13. Click the **Add Filter** button for the **Year** column. In the Create/Edit Filter dialog box, click **Add > Variable > Repository**.

Figure 4–75 Create/Edit Filter: Add Repository

14. In the Server Variable field, type **maxYear_01**. Then, close the Create/Edit Filter dialog box and verify your results. Your result should look similar to [Figure 4–76](#).

Figure 4–76 Results

Year	Amount_Sold
2006	\$54,585,443

4.8 Executing Direct Database Requests - Advanced

Users with the appropriate permissions can create and issue a database request directly to a physical back-end database. The results of the request can be displayed and manipulated within Oracle BI Answers, and subsequently incorporated into Oracle BI Interactive Dashboards and Oracle BI Delivers.

The following privilege settings in Oracle BI Presentation Services Administration control whether you can create and issue physical requests:

- **Edit Direct Database Requests:** If this privilege is set for you, you can create direct database requests. By default, this privilege is set only for users defined as Presentation Server Administrators.
- **Execute Direct Database Requests:** If this privilege is set for you, you can issue physical requests. By default, this privilege is not enabled for anyone. A Presentation Server Administrator can change it.

To execute direct database requests:

1. If you are not already logged in as the **Administrator** account, log out, then log back in to Oracle BI Presentation Services as **Administrator**.
2. In Oracle BI Answers, click **Settings > Administration**. In the Oracle BI Presentation Service Administration screen, click **Manage Privileges** to open the Privilege Administration screen.
3. In the Privilege Administration screen, scroll down to **Answers**. Verify that the **Edit Direct Database Requests** privilege is granted to the Presentation Server

Administrators group. By default, the Administrator user is a member of this group. If this privilege is not granted to the Presentation Server Administrators group, then do so.

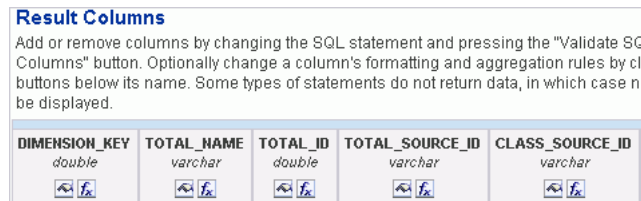
- In Oracle BI Answers, click the **Create Direct Request** link. In the Connection Pool field, type the connection pool name for the GEC_DW_TUTORIAL data source enclosed in double quotes ("**bise1db**".**Connection Pool**" in this example). In the SQL Statement field, type **SELECT * FROM channels**.

Figure 4–77 Create Direct Request Link



- Click **Validate SQL and Retrieve Columns** to display the columns from the Channels table.

Figure 4–78 Result Columns



- Click **Results** to see the contents of the Channels table.
- Click **Criteria**, to return to the direct database request specification page. Remain here for the next section.

4.9 Using Aggregates - Advanced

Aggregate tables store precomputed results, which are measures that have been aggregated (typically summed) over a set of dimensional attributes. Using aggregate tables is a very popular technique for speeding up query response times in decision support systems. This eliminates the need for run-time calculations and delivers faster results to users. The calculations are done ahead of time and the results are stored in the tables. Aggregate tables should have many fewer rows than the non-aggregate tables, and therefore, processing should be quicker.

The aggregate navigation capability of Oracle BI Server allows queries to use the information stored in aggregate tables automatically, without query authors or tools having to specify aggregate tables in the queries. Oracle BI Server allows users to concentrate on asking the right business questions, because the server decides which tables provide the fastest answers. For Oracle BI Server to have enough information to

navigate to aggregate tables, you need to configure certain metadata in the repository. You can also use the Aggregate Persistence Wizard utility in the Oracle BI Administration tool to create the aggregate tables.

This section contains the following topics:

- [Section 4.9.1, "Use a Direct Database Request to Create Aggregate Tables"](#)
- [Section 4.9.2, "Import the Aggregate Tables"](#)
- [Section 4.9.3, "Create Physical Joins to the Aggregate Tables"](#)
- [Section 4.9.4, "Map Logical Columns to Aggregate Columns"](#)
- [Section 4.9.5, "Test the Aggregation Using Oracle BI Answers"](#)

4.9.1 Use a Direct Database Request to Create Aggregate Tables

To use a direct database request to create aggregate tables:

1. Using Windows Explorer, navigate to the directory where you installed Oracle BI Standard Edition One (the default installation directory is `c:\oracle\bi\se1`). Go to the `tutorial\bi_ad` directory, and open the aggregate table SQL script file: `agg_table_sql.txt`.
2. In Oracle BI Answers direct database request Criteria page, verify that the Connection Pool field is still set to the connection pool name for the BISE1_TUTORIALWH data source enclosed in double quotes ("**bise1db**".**Connection Pool**" in this example).
3. Copy the first **CREATE TABLE** SQL for the first table, **AGG_DAY_SALES_F**, from the `agg_table_sql.txt` file and paste it into the SQL Statement field.
4. Click **Results**. You should see a **No Results** message.
5. Click the **Criteria** tab. Remove the **CREATE TABLE** SQL from the SQL Statement field and type **SELECT COUNT(*) FROM AGG_DAY_SALES_F** in the SQL Statement field.
6. Click **Validate SQL and Retrieve Columns**. Click **Results**. You should see **COUNT(*) = 229**.
7. Click the **Criteria** tab. Remove the **SELECT COUNT(*)** SQL statement from the SQL Statement field. Copy the second **Create Table** SQL from the `agg_table_sql.txt` file, for the second table **AGG_PRODUCTS_CATEGORY_D**, and paste it into the SQL Statement field.
8. Click **Results**. You should see a **No Results** message.
9. Click the **Criteria** tab. Type the following SQL in the SQL Statement field: **SELECT COUNT(*) FROM AGG_PRODUCTS_CATEGORY_D**.
10. Click **Results**. You should see **COUNT(*) = 6**.

Note: You may see a "Downloads" window when you click the **Results** tab. You can ignore the window.

4.9.2 Import the Aggregate Tables

In the Oracle BI Administration Tool, you need to import the two new aggregate tables from BISE1_TUTORIALWH schema. To do this:

1. Select **File > Import > from Database**, then select the **BISE1DB** ODBC connection. Enter the password for the BISE1_TUTORIALWH database account. Click **OK**.

Note: If the repository is not already opened, you can open it by selecting **File > Open > Online**.

2. In the Import dialog box, expand the **BISE1_TUTORIALWH** schema folder and use **Ctrl + Click** to select the following tables:
 - **AGG_DAY_SALES_F**
 - **AGG_PRODUCTS_CATEGORY_D**

Note that **Tables** and **Keys** are the default options selected.
3. Click **Import**. The Connection Pool dialog box opens. Click **Yes** if prompted to check out objects.
4. After the Import completes, click **Close** to exit the Import dialog box.
5. In the Physical layer, drag the two imported tables from the **BISE1_TUTORIALWH** folder into the **GEC_DW_TUTORIAL** folder.
6. Delete the **BISE1_TUTORIALWH** folder from the physical layer by right-clicking the **BISE1_TUTORIALWH** folder and selecting **Delete**.
7. Expand the **GEC_DW_TUTORIAL** folder in the Physical layer and verify that the aggregate tables are added. Check in changes.
8. Select the two aggregate tables in the Physical layer, then right-click and select **Update Row Count** to verify connectivity. You should see 229 rows for **AGG_DAY_SALES_F**, and 6 rows for **AGG_PRODUCTS_CATEGORY_D**.
9. Check in the changes and save the repository.

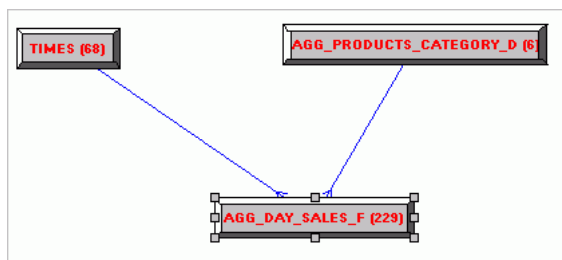
4.9.3 Create Physical Joins to the Aggregate Tables

To create physical joins to the aggregate tables:

1. In the Physical layer, use **Ctrl+Click** to select the two new aggregate tables and the **Times** table. Click the **Physical Diagram** icon on the toolbar.
2. In the Physical Diagram, use the **New Foreign Key** button to create the following joins. Click **Yes** when prompted to create matching table keys.

```
AGG_PRODUCTS_CATEGORY_D. CATEGORY_ID =
AGG_DAY_SALES_F. CATEGORY_ID
TIMES.DIMESION_KEY = AGG_SALES_F.TIMES
```

Figure 4–79 Joins



Note: The default join columns selected may not be correct. Ensure that the correct columns are selected for the joins.

3. If there is an error message, check the Physical column data type and make sure they are the same. You may have to manually change the datatype in the Physical Column dialog box. If a message says that the keys have to be created, click **Yes**.
4. Check in the changes and save the repository.

4.9.4 Map Logical Columns to Aggregate Columns

To map logical columns to aggregate columns:

1. Map existing logical columns to new sources by dragging columns from the Physical layer to corresponding columns in the Business Model and Mapping layer, as described in [Table 4-1](#).

Table 4-1 Mapping Logical Columns to Physical Columns

Logical Column Name	Physical Column Name
Sales.Amount_Sold	AGG_DAY_SALES_F.AMOUNT_SOLD
Products.Category	AGG_PRODUCTS_CATEGORY_D.CATEGORY_NAME
Products.Total_name	AGG_PRODUCTS_CATEGORY_D.TOTAL_NAME

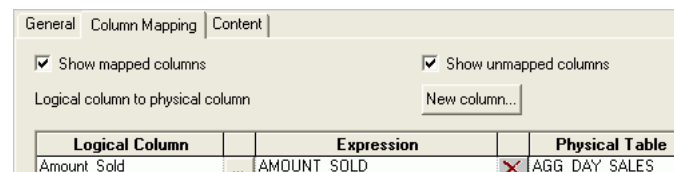
2. In the Business Model and Mapping layer, expand **Sales > Sources** and verify that a new logical table source, **AGG_DAY_SALES_F**, is created.

Figure 4-80 New Logical Table Source: AGG_DAY_SALES_F



3. Double-click the **AGG_DAY_SALES_F** logical table source to open the Logical Table Source dialog box, click the **Column Mapping** tab, and verify the mapping.

Figure 4-81 Column Mapping for AGG_DAY_SALES_F

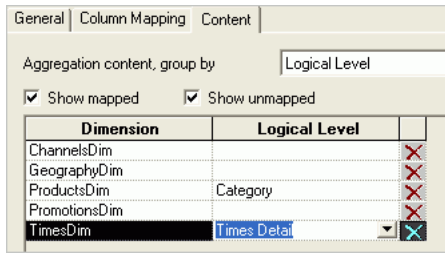


4. Click the **Content** tab. Use the drop-down menu in the **Logical Level** field to set the following logical levels:

ProductsDim = Category

TimesDim = Times Detail

Figure 4–82 Logical Levels for AGG_DAY_SALES_F



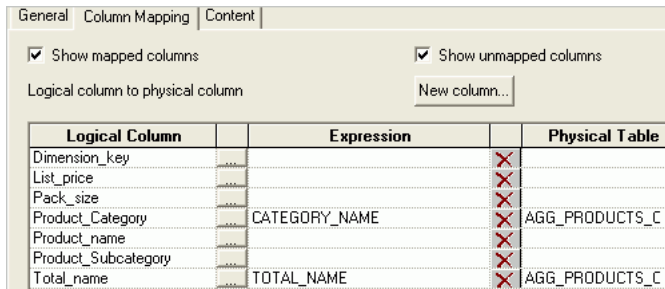
You are setting aggregation content for the fact table to the corresponding levels in the dimension hierarchies. In a subsequent step, you set similar levels for the dimension table aggregate sources. Later, when a user queries against a particular level, Oracle BI Server will access the aggregate tables instead of the detail tables.

For example, after setting aggregation content, if a user queries for Amount_Sold by Product Category, the server will access the AGG_DAY_SALES_F aggregate fact table and the corresponding aggregate dimension table, AGG_PRODUCTS_CATEGORY_D.

If a user queries for a level lower than the levels specified here, for example, Product Subcategory or Product Name, then the server will access the detail tables. If a user queries for a higher level, for example, Total_Name, the aggregate tables will be used as well, because whenever a query is run against a logical level or above, the aggregate tables are used.

5. Click **OK** to close the Logical Table Source dialog box.
6. Expand **Products > Sources** and verify that a new logical table source, **AGG_PRODUCTS_CATEGORY_D**, is created.
7. Double-click the **AGG_PRODUCTS_CATEGORY_D** logical table source, click the **Column Mapping** tab, and verify the mappings.

Figure 4–83 Column Mapping for AGG_PRODUCTS_CATEGORY_D



8. Click the **Content** tab. Use the drop-down menu in the Logical Level field to set the following logical level to specify what level of detail is stored in the aggregate table:

ProductsDim = Category

Figure 4–84 Logical Level for AGG_PRODUCTS_CATEGORY_D

Dimension	Logical Level
ProductsDim	Category

9. Click **OK** to close the Logical Table Source dialog box. Check in changes. Check consistency. Fix any errors before proceeding.
10. Save the repository. Note that you did not need to make any changes to the Presentation layer. You made changes in the business model that impact how the queries are processed and which sources are accessed. Based on how you specified the aggregation content, Oracle BI Server automatically uses the new sources when appropriate.

4.9.5 Test the Aggregation Using Oracle BI Answers

To test the aggregation:

1. In Oracle BI Answers, click **Reload Server Metadata**.
2. Click the **Answers** link. Select the **GEC_DW_TUTORIAL** subject area.
3. Create the following query:
Products.Products_Category, Sales.Amount_Sold
4. Click **Settings > Administration**, then click **Manage Sessions**. Locate the last query run under Cursor Cache and click **View Log**.
5. Examine the log and verify that the aggregate tables **AGG_CUSTOMER_STATE_D** and **AGG_DAY_SALES_F** were accessed.

Figure 4–85 Log View

```

+++Administrator:2d0000:2d0004:----2007/02/09 12:19:23
----- Sending query to database named BISE1_TUTORIALWH
select T10583.CATEGORY_NAME as c1,
       sum(T10579.AMOUNT_SOLD) as c2
from
  AGG_PRODUCTS_CATEGORY_D T10583,
  AGG_DAY_SALES_F T10579
where ( T10579.CATEGORY_ID = T10583.CATEGORY_ID )
group by T10583.CATEGORY_NAME
order by c1

```

6. Close all windows.

4.10 Creating Time Series Measures - Advanced

The ability to compare business performance with previous time periods is fundamental to understanding a business. Time comparisons allow businesses to analyze data that spans multiple time periods, providing a context for the data. Yet, as Ralph Kimball states, SQL was not designed to make comparisons over time straightforward:

"The most difficult area of data warehousing is the translation of simple business analyses into SQL. SQL was not designed with business reports in mind. SQL was really an interim language designed to allow relational table semantics to be expressed

in a convenient and accessible form, and to enable researchers and early developers to proceed with building the first relational systems in the mid-1970s. How else can you explain the fact that there is no direct way in SQL to compare this year to last year?"

- Ralph Kimball

The solution is to model time series data in the Oracle BI repository. This allows users to make one request for the desired result. Oracle BI Server runs multiple queries in parallel to get the results. The queries that run in the background to support the time measure are transparent to the user.

Oracle BI Server provides Ago and ToDate functions for time series comparisons. Both of these functions operate on measures. The Ago function calculates the aggregated value as of some time period shifted from the current time. For example, the Ago function can produce sales for every month of the current quarter, along with the corresponding quarter ago sales. The ToDate function is used to aggregate a measure attribute from the beginning of a specified time period to the currently displaying time. For example, the ToDate function can calculate Month to Date sales for a given year. You use the Expression Builder to apply the functions.

This section contains the following topics:

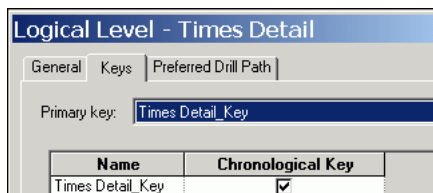
- [Section 4.10.1, "Identify a Dimension as a Time Dimension"](#)
- [Section 4.10.2, "Create a Month Ago Measure"](#)
- [Section 4.10.3, "Create a Change Month Ago Measure"](#)
- [Section 4.10.4, "Create a ToDate Measure"](#)
- [Section 4.10.5, "Test the Time Series Measures"](#)

4.10.1 Identify a Dimension as a Time Dimension

To identify a dimension as a time dimension:

1. In the Oracle BI Administration Tool, double-click the **TimesDim** dimension hierarchy in the Business Model and Mapping layer.
2. In the Dimension dialog box, select **Time dimension**. Click **OK** to close the Dimension dialog box.
3. Expand **TimesDim** to the **Times Detail** level, then double-click the **Times Detail** level to open the Logical Level dialog box. Click the **Keys** tab, then select the **Chronological Key** option for **Times Detail_Key**.

Figure 4–86 Logical Level: Times Detail_Key



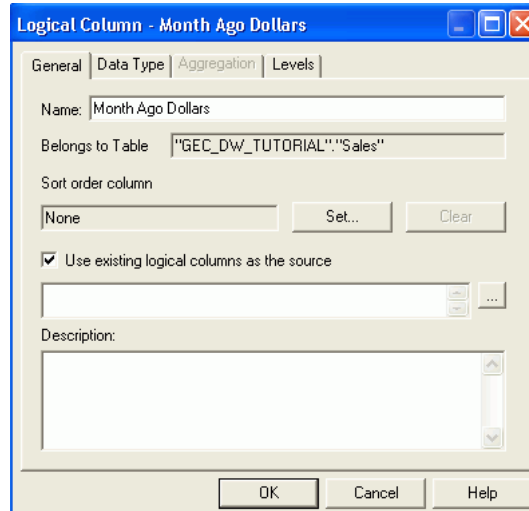
4. Click **OK** to close the Logical Level dialog box.

4.10.2 Create a Month Ago Measure

To create a Month Ago measure:

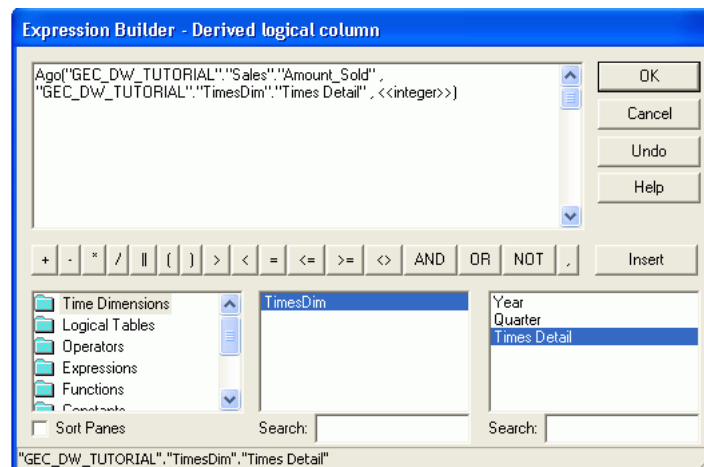
1. Right-click **Sales** and select **New Object > Logical Column** to open the Logical Column dialog box. Name the logical column **Month Ago Dollars**. Select **Use existing logical columns as the source**.

Figure 4–87 New Logical Column: Month Ago Dollars



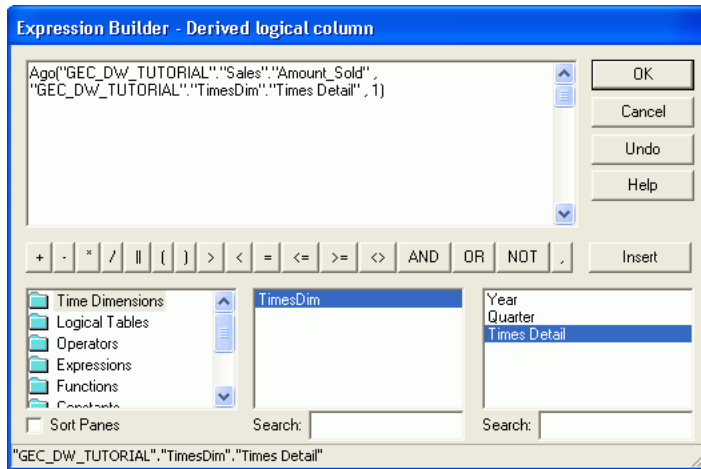
2. Click the ellipsis (...) to open the Expression Builder. Select **Functions > Time Series Functions > Ago**. Click **Insert** to add the Ago function to the Expression Builder.
3. Click the <<expr>> in the expression. Select **Logical Tables > Sales** and then double-click **Amount_Sold** to add it to the expression.
4. Click the <<level>> in the expression. Select **Time Dimensions > TimesDim** and then double-click **Times Detail** to add it to the expression.

Figure 4–88 Expression Builder: Times Detail



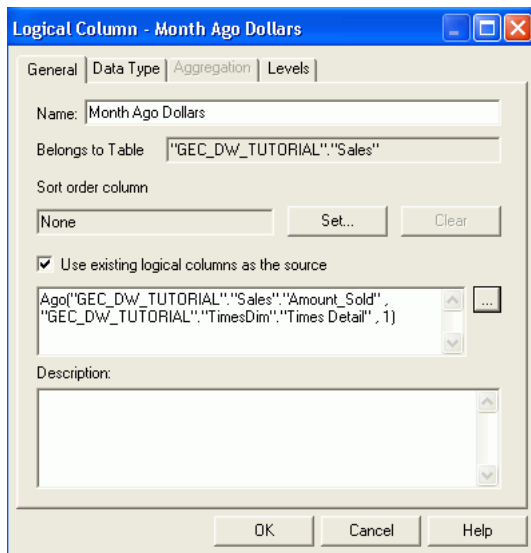
5. Click <<integer>> in the expression and type **1**. The Ago function calculates the Amount_Sold value one month prior to the current month.

Figure 4–89 Expression Builder: Replacing <<integer>> with 1



- Click **OK** to close the Expression Builder. The formula is displayed in the Logical Column dialog box.

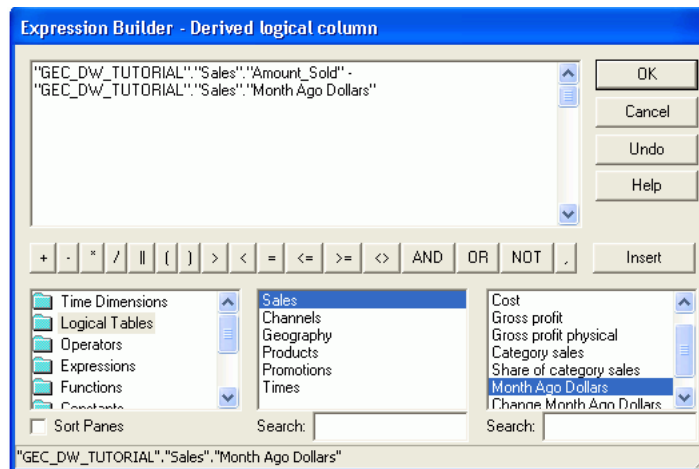
Figure 4–90 Logical Column Dialog Box with Ago Formula



- Click **OK** to close the Logical Column dialog box.

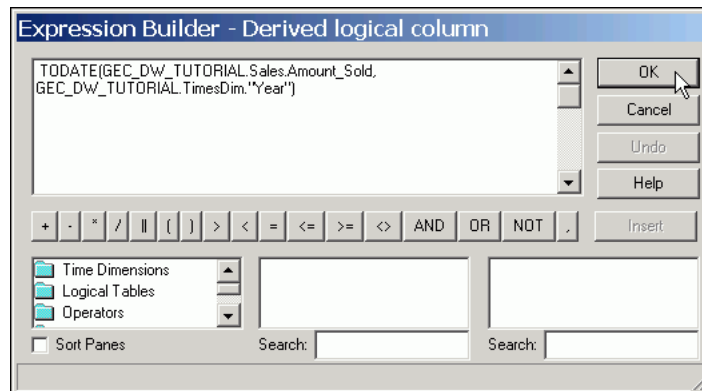
4.10.3 Create a Change Month Ago Measure

To create a Change Month Ago measure, create another logical column called **Change Month Ago Dollars** with the expression shown in [Figure 4–91](#). Be sure to select **Use existing logical columns as the source**.

Figure 4–91 Expression Builder: Change Month Ago Dollars

4.10.4 Create a ToDate Measure

To create a ToDate measure, create another logical column called **Year To Date Dollars** with the expression shown in [Figure 4–92](#). Be sure to select **Use existing logical columns as the source**.

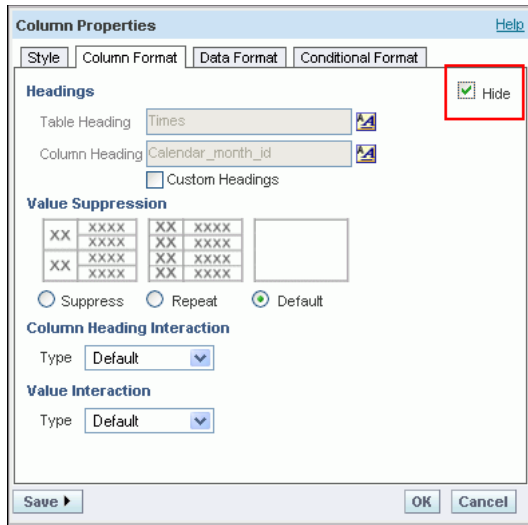
Figure 4–92 Expression Builder: Month To Date Dollars

4.10.5 Test the Time Series Measures

To test the time series measures:

1. Drag the three time series measures from the Business Model and Mapping layer to the Sales presentation table.
2. Check in the changes. Check for global consistency and save the repository.
3. In Oracle BI Answers, click **Reload Server Metadata**.
4. Create a new request using the `GEC_DW_TUTORIALWH` subject area.
5. Select `Times.Year`, `Times.Month`, and `Times.Calendar_month_id`.
6. Add a filter on `Year=2006`.
7. Sort in ascending order on `Calendar_month_id`. Then, hide the column. To do this, click **Column Properties** for `Calendar_month_id`, then, in the **Column Format** tab, select the **Hide** option.

Figure 4–93 Column Properties: Column Format Tab



8. Now add the following columns:

Sales.Amount_Sold

Sales.Month Ago Dollars

Sales.Change Month Ago Dollars

Your Answers result should look like the one shown in [Figure 4–94](#).

Figure 4–94 Answers Result with Sales.Month Ago Dollars

Year	Month	Amount_Sold	Month Ago Dollars	Change Month Ago Dollars
2006	January 2006	\$2,375,247	1,796,717	578,530
2006	February 2006	\$2,513,226	2,375,247	137,979
2006	March 2006	\$2,727,506	2,513,226	214,280
2006	April 2006	\$2,252,701	2,727,506	-474,805
2006	May 2006	\$3,086,104	2,252,701	833,403
2006	June 2006	\$3,246,797	3,086,104	160,693
2006	July 2006	\$3,406,984	3,246,797	160,187
2006	August 2006	\$3,044,269	3,406,984	-362,714
2006	September 2006	\$4,317,593	3,044,269	1,273,324
2006	October 2006	\$5,918,943	4,317,593	1,601,350
2006	November 2006	\$9,592,340	5,918,943	3,673,396
2006	December 2006	\$12,103,732	9,592,340	2,511,393

9. In Criteria, delete **Sales.Month Ago Dollars** and **Sales.Change Month Ago Dollars**. Add **Sales.Year To Date Dollars**.

Your Answers result should look like the one shown in [Figure 4–95](#).

Figure 4–95 Answers Result with Sales.Year To Date Dollars

Year	Month	Amount_Sold	Year to Date Dollars
2006	January 2006	\$2,375,247	2,375,247
2006	February 2006	\$2,513,226	4,888,473
2006	March 2006	\$2,727,506	7,615,979
2006	April 2006	\$2,252,701	9,868,680
2006	May 2006	\$3,086,104	12,954,784
2006	June 2006	\$3,246,797	16,201,581
2006	July 2006	\$3,406,984	19,608,565
2006	August 2006	\$3,044,269	22,652,834
2006	September 2006	\$4,317,593	26,970,427
2006	October 2006	\$5,918,943	32,889,371
2006	November 2006	\$9,592,340	42,481,710
2006	December 2006	\$12,103,732	54,585,443

4.11 Adding Multiple Sources - Advanced

Data is often partitioned into multiple physical sources for a single logical table in a business model. When a logical table source does not contain the entire set of data at a given level, you need to specify the portion of the set that it does contain. When individual sources at a given level contain information for a portion or fragment of the domain, Oracle BI Server needs to know the content of the sources in order to pick the appropriate source for the query. The goal is to provide seamless and efficient access from the users' perspective. When there are multiple sources, the metadata is built so that Oracle BI Server handles the navigation to the appropriate source. Oracle BI Server can seamlessly access and process data from multiple sources in an efficient manner to satisfy user requests.

In this example, sales quota numbers are stored in an Excel workbook. The workbook, `Quota.xls`, is stored on your computer. You incorporate the quota numbers into the business model and create business measures to report variance from quota and percent of quota.

This section contains the following topics:

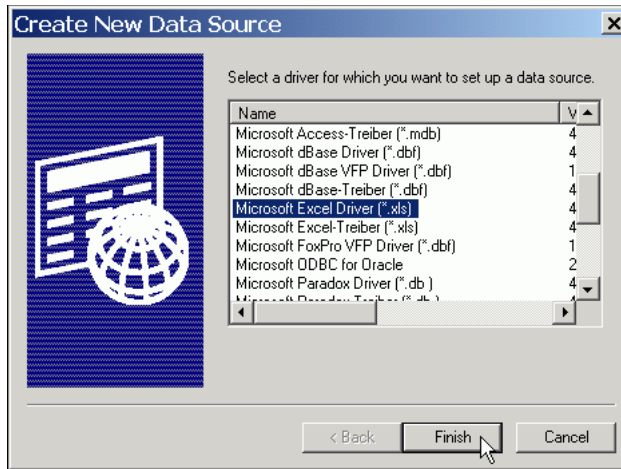
- [Section 4.11.1, "Create an ODBC DSN"](#)
- [Section 4.11.2, "Import the Excel Data Source Into the BI Repository"](#)
- [Section 4.11.3, "Create a Physical Key"](#)
- [Section 4.11.4, "Create the Physical Joins"](#)
- [Section 4.11.5, "Set the Dimension Key"](#)
- [Section 4.11.6, "Map the Logical Dimension Columns"](#)
- [Section 4.11.7, "Create the Quota Measures"](#)
- [Section 4.11.8, "Test the Quota Measures"](#)

4.11.1 Create an ODBC DSN

To create an ODBC DSN:

1. Click **Start > Programs > Administrative Tools > Data Sources (ODBC)** to open the ODBC Data Source Administrator. Click the **System DSN** tab and click **Add**.
2. The Create New Data Source dialog box opens. In the Create New Data Source dialog box, select the **Microsoft Excel Driver**.

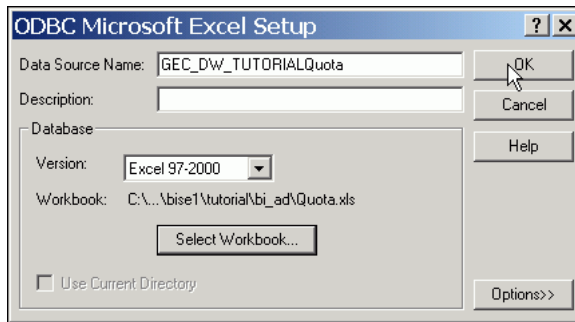
Figure 4–96 Create New Data Source Dialog Box



3. Click **Finish** to open the ODBC Microsoft Excel Setup dialog box. In the ODBC Microsoft Excel Setup dialog box, enter **GEC_DW_TUTORIALQuota** as the **Data Source Name**.
4. Click **Select Workbook** to open the Select Workbook dialog box and go to the location where you installed Oracle BI Standard Edition One. Then, go to the tutorial\bi_ad directory, select the Quota.xls file, and click **OK**.

The path to the workbook is displayed in the ODBC Microsoft Excel Setup dialog box. Click **OK** to close the ODBC Microsoft Excel Setup dialog box.

Figure 4–97 ODBC Microsoft Excel Setup Dialog Box



5. Verify that the Excel system data source is added in the ODBC Data Source Administrator, then click **OK** to close the ODBC Data Source Administrator.

4.11.2 Import the Excel Data Source Into the BI Repository

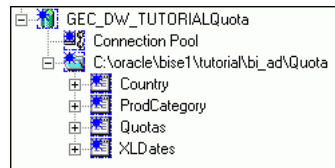
To import the Excel data source:

1. In BI Administration Tool, select **File > Import > from Database**. In the Select Data Source dialog box, select the ODBC DSN you created in the previous steps. Leave **User Name** and **Password** blank.
2. Click **OK**. The Import dialog box opens. In the Import dialog box, select the **GEC_DW_TUTORIALQuota** object. Verify that only **Tables** and **Keys** are selected, and click **Import**.

Note: Each named range in the Excel worksheets are treated as a table.

- When the import completes, click **Close** to close the Import dialog box and verify that all four range tables have been imported into the Physical layer.

Figure 4–98 Physical Layer with Imported Range Tables



The Country table will not be used.

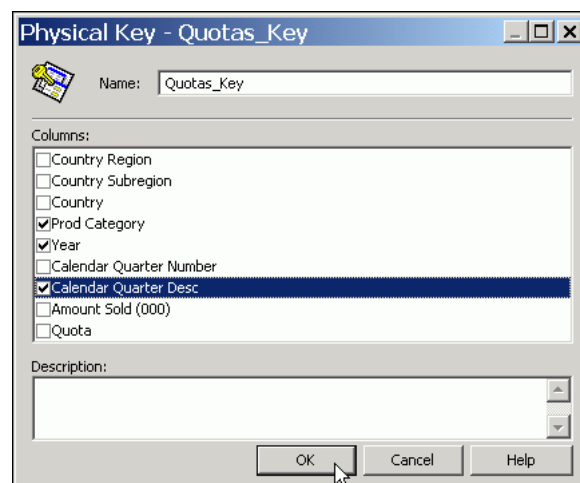
- Check in changes and save the repository.

4.11.3 Create a Physical Key

To create a physical key:

- In the Physical layer, double-click the **Quotas** physical table.
- Click the **Keys** tab, then click **New**.
- Enter **Quotas_Key** as the **Name** of the key.
- Select **Prod Category**, **Year**, and **Calendar Quarter Desc**. Click **OK** to close the Physical Key dialog box.

Figure 4–99 Physical Key Dialog Box



- Click **OK** to close the Physical Table dialog box.

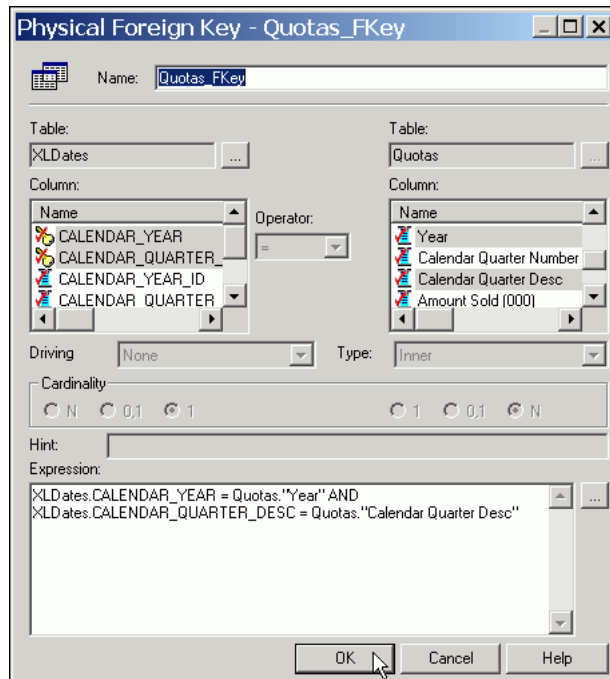
4.11.4 Create the Physical Joins

To create the physical joins:

1. Use **Ctrl+Click** to select table ProdCategory, XLDates, Quotas in the Physical layer. Right-click any one of the three tables and select **Physical Diagram > Selected Object(s) Only** to open the Physical Diagram.
2. In the Physical Diagram, click the **New foreign key** button to create the joins.
3. Click **XLDates**, then click and drag the join to **Quotas**. In the **Expression** field, enter the following expression, then click **OK**:

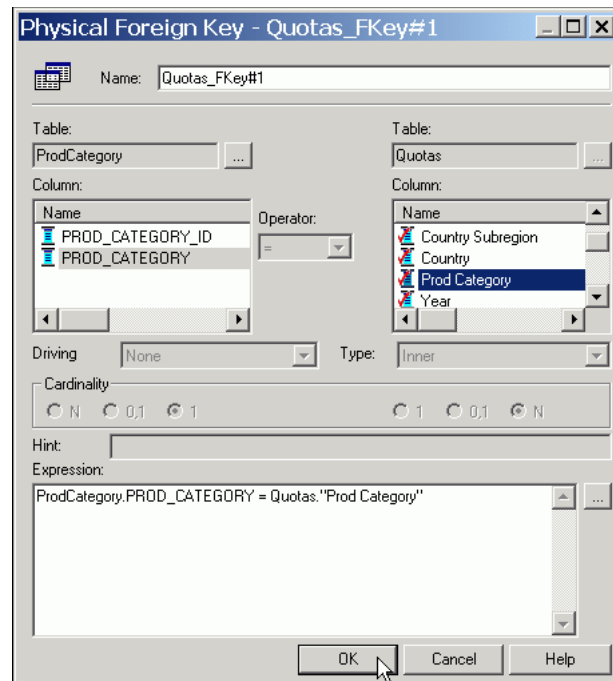
```
XLDates.CALENDAR_YEAR = Quotas."Year" AND
XLDates.CALENDAR_QUARTER_DESC = Quotas."Calendar Quarter Desc"
```

Figure 4–100 New Foreign Key for XLDates



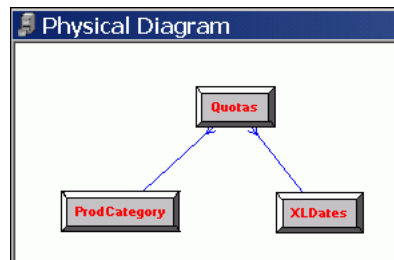
4. When prompted for new key creation, click **Yes**.
5. Click **ProdCategory**, then click and drag the join to **Quotas**. In the **Expression** field, enter the following expression, then click **OK**:

```
ProdCategory.PROD_CATEGORY = Quotas."Prod Category"
```


Figure 4–101 New Foreign Key for ProdCategory

6. When prompted for new key creation, click **Yes**.

The Physical Diagram should appear like the one shown in [Figure 4–102](#).

Figure 4–102 Physical Diagram

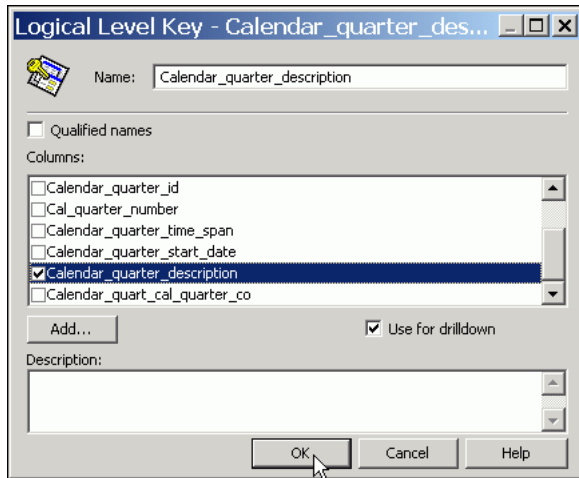
7. Close the Physical Diagram.
8. Check in changes and save the repository.

4.11.5 Set the Dimension Key

To set the dimension key:

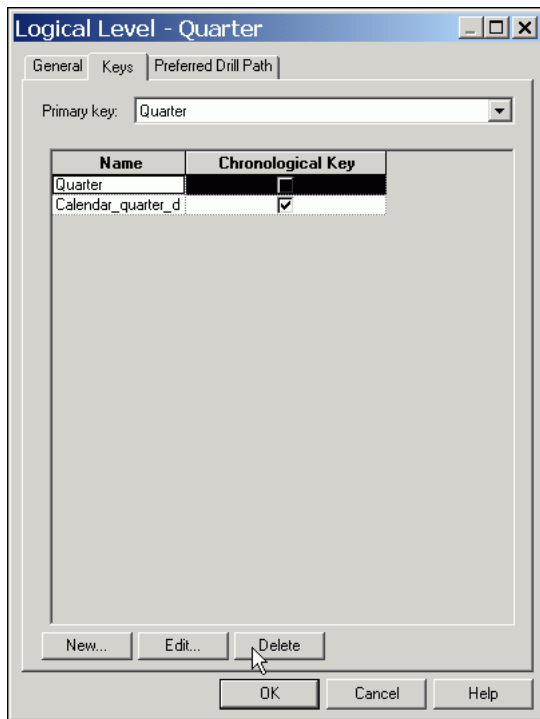
1. In the **Business Model and Mapping layer**, expand the **TimesDim** dimension.
2. Double-click the **Quarter** level.
3. In the Logical Level dialog box, click **New**.
4. Select **Calendar_quarter_description**. Click **OK** to close the Logical Level Key dialog box.

Figure 4–103 Logical Level Key Dialog Box: Calendar_quarter_description



5. In the Logical Level dialog box, select the **Chronological** key option for **Calendar_quarter_description**.
6. Select **Quarter** and click **Delete**, and then click **Yes** to confirm the deletion.

Figure 4–104 Logical Level Dialog Box: Deleting Quarter



7. Click **OK** to close the Logical Level dialog box.

4.11.6 Map the Logical Dimension Columns

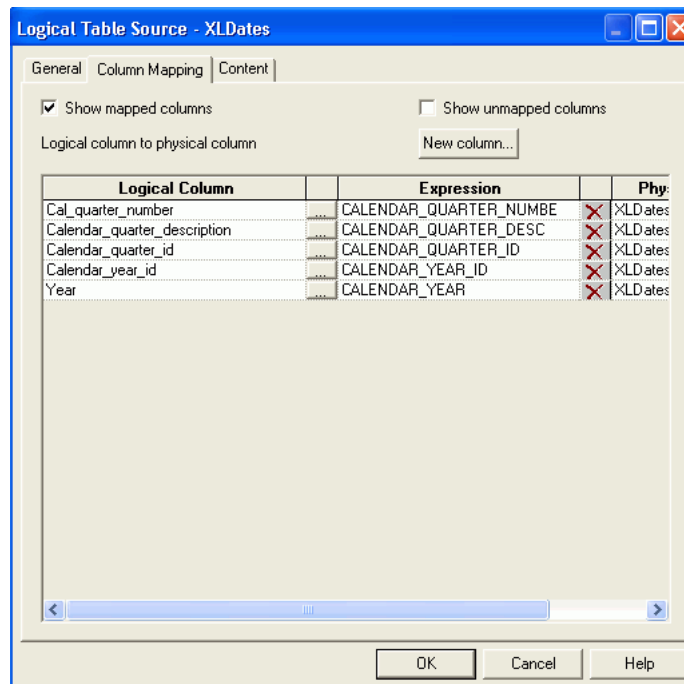
To map the logical dimension columns:

1. Drag the physical column **CALENDAR_YEAR** from **XLDates** in the Physical layer to the logical column **Times.Year** in the Business Model and Mapping layer.

Notice that a new logical table source, **XLDates**, is created automatically. Drag the remaining physical columns from **XLDates** in the Physical layer to the corresponding logical columns in the **Times** table in the Business Model and Mapping layer.

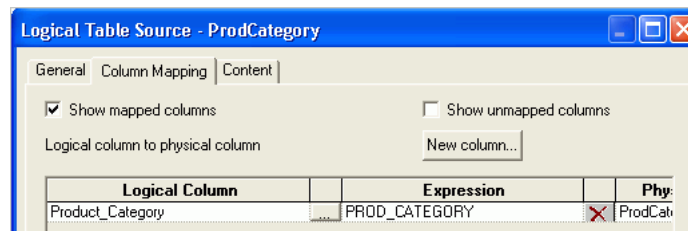
2. Double-click the **XLDates** logical table source in the **Times > Sources** folder to open the Logical Table Source dialog box, click the **Column Mapping** tab, and verify the mappings. Deselect **Show unmapped columns**.

Figure 4–105 Logical Table Source: XLDates



3. Click **OK** to close the Logical Table Source dialog box.
4. Repeat the steps and map **Products.Product_Category** to **ProdCategory.PROD_CATEGORY** and verify the logical table source mapping.

Figure 4–106 Logical Table Source: ProdCategory

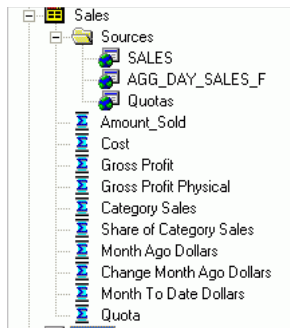


4.11.7 Create the Quota Measures

To create the quota measures:

1. Drag the column **Quotas.Quota** from the Physical layer onto the **Sales** logical table. Notice that a new logical table source, **Quotas**, and a new logical column, **Quota**, are created.

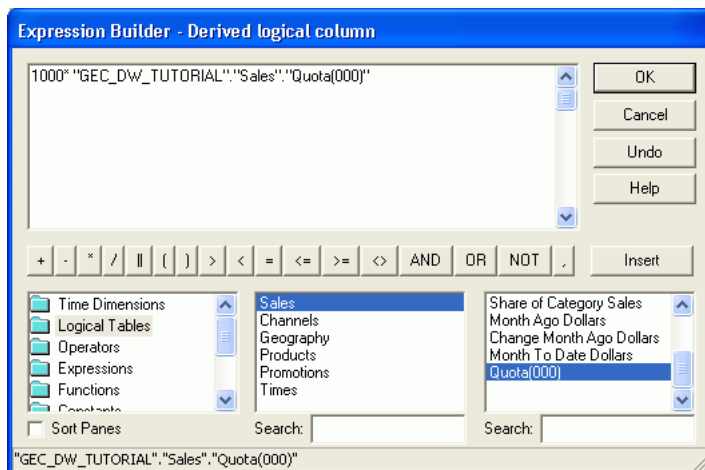
Figure 4–107 Quotas Logical Table Source and Quota Logical Column



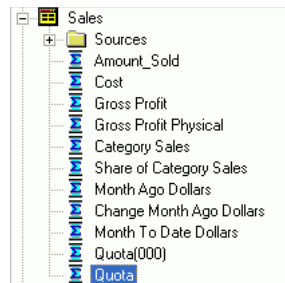
2. Double-click **Sales.Quota** to open the Logical Table dialog box. Click the **Aggregation** tab and set the aggregation rule to **Sum**. Close the Logical Table dialog box.
3. The Quota logical column states quota in thousands, so rename **Quota** to **Quota(000)**.
4. Create a measure for actual Quota based on Quota(000). To do this, right-click **Sales** and select **New Object > Logical Column** to open the Logical Column dialog box. Click the **General** tab and select **Use existing logical columns as the source**. Enter **Quota** in the **Name** field.
5. Click the ellipsis (...) to open the Expression Builder.
6. Specify the following formula:


```
1000*"GEC_DW_TUTORIAL"."Sales"."Quota(000)"
```

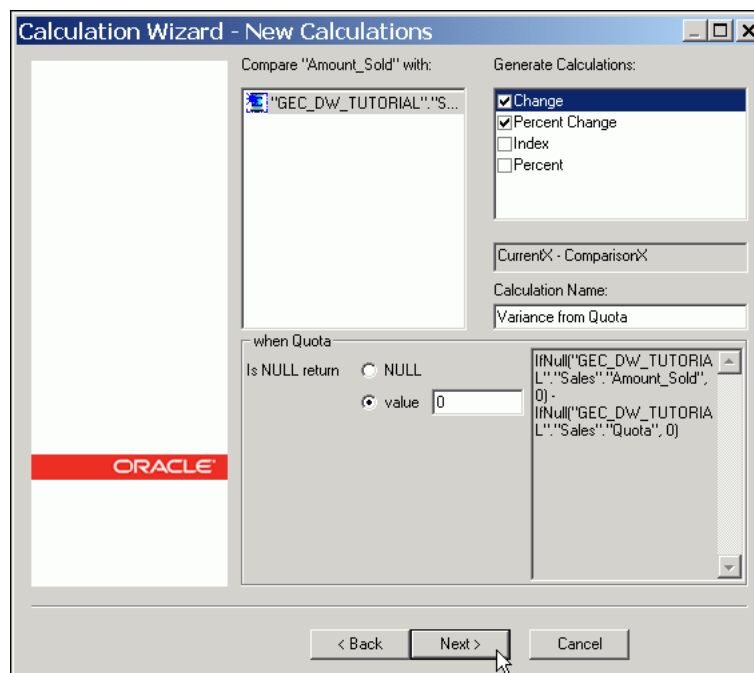
Figure 4–108 Expression Builder: Quota(000)



7. Click **OK** to close the Expression Builder.
8. Click **OK** to close the Logical Column dialog box. The **Quota** column is added to the business model.

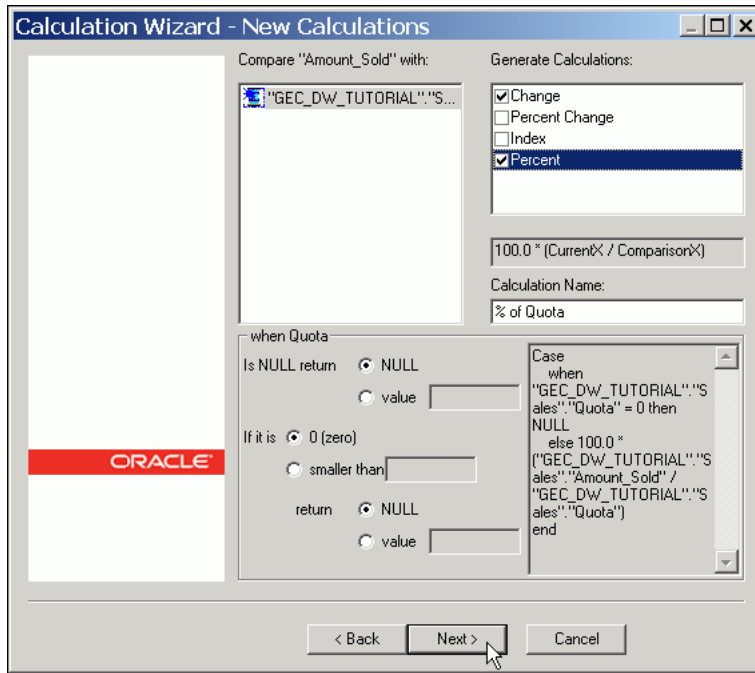
Figure 4–109 Business Model with Quota Column

9. Create two more measures based on **Amount_Sold** and **Quota**. Right-click **Sales.Amount_Sold** and select **Calculation Wizard**. Click **Next**. Select the **Quota** column.
10. Click **Next**. Make sure **Change** is selected. In the Calculation Name field, name the calculation **Variance from Quota**.

Figure 4–110 Calculation Wizard: Naming the Calculation "Variance from Quota"

11. Deselect **Percent Change**. Select **Percent** and make sure it is selected. In the **Calculation Name** field, leave the name as is: **% of Quota**.

Figure 4–111 Calculation Wizard: Selecting "Percent"



12. Click **Next**. In the Finish window, verify the calculations that will be created by the Calculation Wizard.
13. Click **Finish**. The calculation measures are added to the business model.
14. Add the **Quota (000)**, **Quota**, **Variance from Quota**, and **% of Quota** measures to the Sales folder in the Presentation Layer.
15. Check in the changes. Check global consistency. If you receive any Error messages, edit the repository to correct the errors before proceeding. Save the repository.

4.11.8 Test the Quota Measures

To test the quota measures:

1. Return to Oracle BI Answers. Log out and then log back in as **Administrator** with password **Administrator**. Click **Reload Server Metadata**.
2. Create the following request:

Times, **Year**, **Sales.Amount_Sold**, **Sales.Quota**, **Sales.Variance from Quota**, **Sales.% of Quota**

Figure 4–112 Oracle BI Answers: Request

Times	Sales			
Year	Amount_Sold	Quota	Variance from Quota	% of Quota

3. Because Quota is only for 2006, create a filter on the **Year** column as equal to 2006, as shown in [Figure 4–113](#).

Figure 4–113 Oracle BI Answers: Filter on Year Column

Year is equal to / is in 2006

4. Click **Results**.

Figure 4–114 Oracle BI Answers: Results

Year	Amount_Sold	Quota	Variance from Quota	% of Quota
2006	\$54,585,443	72,102,000	-17,516,558	76

5. Click **2006** to drill down to the quarters. Your Answers result should look like the one shown in [Figure 4–115](#).

Figure 4–115 Oracle BI Answers: 2006

Year	Quarter	Amount_Sold	Quota	Variance from Quota	% of Quota
2006	Quarter 1 2006	\$7,615,979	18,319,000	-10,703,021	42
	Quarter 2 2006	\$8,585,602	18,019,000	-9,433,398	48
	Quarter 3 2006	\$10,768,846	18,673,000	-7,904,154	58
	Quarter 4 2006	\$27,615,015	17,091,000	10,524,015	162

6. Examine the query log and verify that Quotas and XLDates spreadsheets are accessed. Refer to [Section 4.5.4, "Use the Query Log to Verify Queries"](#) for detailed steps on viewing query logs.

Figure 4–116 Query Log

```
+++Administrator:480000:480004:----2007/04/24 11:19:47
----- Sending query to database named bise1db (id: <<40581>
WITH
SAWITH0 AS (select sum(T12857.AMOUNT_SOLD) as c1,
              T12058.CALENDAR_YEAR_NAME as c2
from
  TIMES T12058,
  AGG_DAY_SALES_F T12857
where ( T12058.DIMENSION_KEY = T12857.TIMES and T12058.CALENDAR_YEAR_NAME
group by T12058.CALENDAR_YEAR_NAME)
select SAWITH0.c2 as c1,
       SAWITH0.c1 as c2
from
  SAWITH0

+++Administrator:480000:480004:----2007/04/24 11:19:47
----- Sending query to database named GEC_DW_TUTORIALQuota
select sum(T12988."Quota") as c1,
       T12998."CALENDAR_YEAR" as c2
from
  "XLDates" T12998,
  "Quotas" T12988
where ( T12988."Year" = T12998."CALENDAR_YEAR" and T12988."Year" = '2006'
group by T12998."CALENDAR_YEAR"
```

Analyze the Data

In this chapter, you analyze the data in the data mart. To extract business intelligence from the data mart, you use Oracle BI Answers to:

- Create queries, format views, charts, and add user interactivity and dynamic content to enhance the user experience.
- Work with views, including pivot tables, and narratives.
- Create selectors to drive interactivity in your BI Answers requests.

This chapter contains the following topics:

- [Section 5.1, "Prerequisites"](#)
- [Section 5.2, "Creating a Query and a Chart"](#)
- [Section 5.3, "Working with Pivot Tables"](#)
- [Section 5.4, "Creating Column Selectors"](#)
- [Section 5.5, "Creating a Narrative View"](#)
- [Section 5.6, "Creating View Selectors"](#)
- [Section 5.7, "Creating Conditional Formats, Gauges, and Navigation"](#)

5.1 Prerequisites

If you have not completed the previous chapters, you must first populate the BISE1_TUTORIALWH schema before proceeding with this chapter. [Section 2.2.2.3, "BISE1_TUTORIALWH Schema"](#) explains how to do this.

If you have not completed the previous chapters, you must also replace the BI metadata repository file, as follows:

1. Shut down the BI Server service.
2. Copy `tutorial\bi_ad\bise1_soln.rpd` to the `bi\oraclebi\server\repository` directory.
3. Rename `bi\oraclebi\server\repository\bise1_soln.rpd` to `bise1.rpd`.
4. Start the BI Server service.

If you specified a password other than `welcome1` during installation, perform the following steps as well:

1. Go to **Start > Programs > Oracle Business Intelligence > Administration**.
2. Select **File > Open > Online**.

3. Enter **Administrator** in the Password field. Click **OK**.
4. In the **Physical Layer** pane, expand **BISE1_SALESWH**. Double-click **Connection Pool**.
5. In the **Password** field, enter the password you specified during installation. Click **OK**.
6. Enter the password again in the **Confirm Password** dialog box. Click **OK**.
7. Repeat steps 4 through 6 for the **BISE1DB** node in the **Physical Layer**.
8. Select **File > Save**. Click **No** when asked whether you wish to check global consistency. Exit the Oracle BI Administration tool.

If you have not completed the previous chapters, you must also create the BI aggregate tables in the database, as follows:

1. Open a command prompt window.
2. Go to the directory where Oracle BI Standard Edition One is installed, then go to the `tutorial\bi_ad` directory.
3. Run the `cr_tabs.bat` batch script. Enter the password for the **BISE1_TUTORIALWH** database account when prompted. Close the command prompt window when done.

5.2 Creating a Query and a Chart

You build and format a business intelligence request using Oracle BI Answers, and create and format a chart.

This section contains the following topics:

- [Section 5.2.1, "Create an Answers Query"](#)
- [Section 5.2.2, "Add Filters on Columns"](#)
- [Section 5.2.3, "Create Totals and Format Results"](#)
- [Section 5.2.4, "Create a Chart"](#)

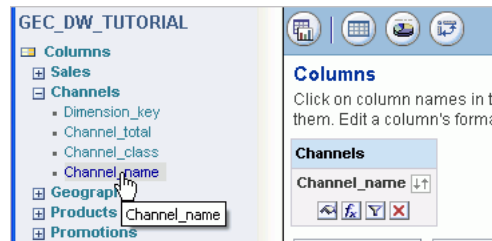
5.2.1 Create an Answers Query

To create an Answers query:

1. Select **Start > Programs > Oracle Business Intelligence > Presentation Services**. Log in to Oracle Business Intelligence as **Administrator** with password **Administrator**.
2. Explore the Sales Analysis Dashboard. Click the **Answers** link to navigate to the Answers start page, and explore the requests created from subject area **GEC_DW**. Expand **GEC_DW** under **My Folders** from the left pane. Under **GEC_DW**, there are several folders. Open each of the folders and double-click the requests to view them.
3. To ensure the latest changes in the BI metadata repository are reflected in the Oracle BI Presentation Server, click **Settings > Administration**. Click **Reload Files and Metadata**. Click **Close Window**.
4. You will be creating requests using the **GEC_DW_TUTORIAL** subject area you just created in the Oracle BI Administration Tool. Select the **GEC_DW_TUTORIAL** subject area by clicking the **GEC_DW_TUTORIAL** link in the Subject Areas list.

5. In this example, there are two subject areas, but there could be more depending on the metadata that is defined in the Oracle Business Intelligence repository, which can contain multiple subject areas. Subject areas are sets of related information with a common business purpose.
6. In the left-hand selection pane of the Answers interface, click the **Plus** icon next to **Channels** to expand it. Click the **Channel_name** column to add it to your query criteria, which appears in the right pane. The query you are building has two measures and three attributes.

Figure 5-1 Adding Channel_name to the Query Criteria



7. From **Geography** click **Region**, from **Times** click **Year**, and from **Sales** click **Amount_Sold** and **Gross_Profit**. Your query should look like the one shown in [Figure 5-2](#).

Figure 5-2 Query with Channel_name, Region, Year, Amount_Sold, and Gross_Profit

Channels	Geography	Times	Sales	
Channel_name ↓↑	Region ↓↑	Year ↓↑	Amount_Sold ↓↑	Gross Profit ↓↑
⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖

8. You can reorder the columns in your query by clicking and dragging them. Drag the **Year** column in front of the columns from the **Times** table in your query. Your query criteria should now look like the one shown in [Figure 5-3](#).

Figure 5-3 Reordered Query

Times	Channels	Geography	Sales	
Year ↓↑	Channel_name ↓↑	Region ↓↑	Amount_Sold ↓↑	Gross Profit ↓↑
⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖	⏪ ⏩ ⏴ ⏵ ✖

9. Now drag the column back to its original location.
10. Click the **Advanced** tab. The Advanced tab can be made available only to specific users. The XML fully defines the query (including chart formats when charts are used). The SQL defines the content of the query. Note that any query or reporting tool that can issue SQL over an ODBC connection can issue a query to the BI server, just like Answers. Examine the request XML that defines the view and the logical SQL that will be issued for the query.

The Request XML defines the whole analysis, including logical SQL and views for the query. The **SQL Issued** field contains only the logical SQL that will be issued to the Oracle BI Server for processing. Note the saw_X column aliases, which are added automatically. Editing the logical SQL will change your view definitions.

11. Click the **Criteria** tab.

5.2.2 Add Filters on Columns

To add filters on columns:

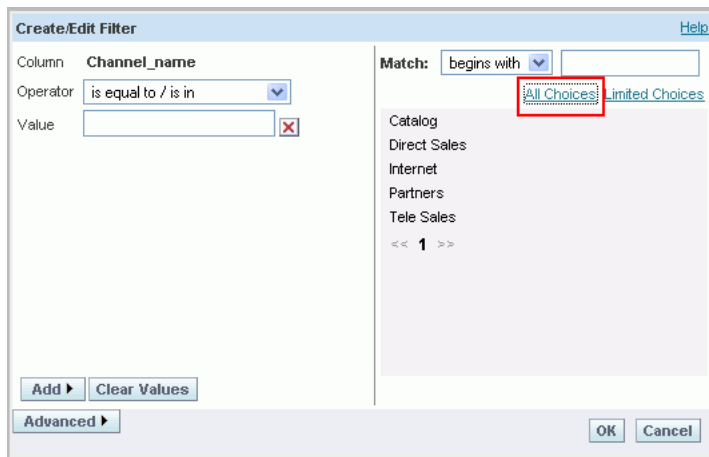
1. Create a filter for the Channel_name column. Click the **Filter** button below the Channel_name column to add a filter on that column.

Figure 5–4 Filter Button on Channel_name Column



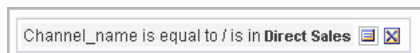
2. In the Create/Edit Filter dialog box, click the **All Choices** link to display all values for the column.

Figure 5–5 Create/Edit Filter: All Choices Link



3. You can also limit the list of choices by setting a match criteria using the **Match** drop-down menu and entering a string, then clicking the **All Choices** link. The **Limited Choices** link will limit choices to those that are consistent with any preexisting filters.
4. Verify that the **Operator** for the filter is set to **is equal to / is in** and then click the **Direct Sales** value in the list of choices. Direct Sales is added as a value in the filter. Click **OK** to create the filter.

Figure 5–6 Filter for Direct Sales



5. Click the **Results** tab to view your results.

Figure 5–7 Results for Direct Sales

Channel_name	Region	Year	Amount_Sold	Gross Profit	Gross Profit
Direct Sales	Americas	2003	2,663,224	686,573	686,573
		2004	6,089,259	1,462,790	1,462,790
		2005	7,778,032	1,406,391	1,406,391
		2006	15,080,493	3,166,638	3,166,638
	Asia	2003	465,766	124,635	124,635
		2004	1,166,796	282,667	282,667
		2005	1,204,618	224,163	224,163
		2006	2,527,941	530,876	530,876
	Europe	2003	1,891,989	496,932	496,932
		2004	3,699,488	893,983	893,983
		2005	4,652,341	863,704	863,704

- Click the **Criteria** tab. Add another filter on **Year = 2006**.

Figure 5–8 Create/Edit Filter for Year=2006

- You should have two filters now joined with an **AND** operator.

Figure 5–9 Two Filters Joined with an AND Operator

- Click the **Results** tab to view the results of your query.

Figure 5–10 Results for Direct Sales and Year 2006

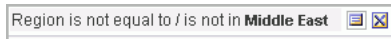
Channel_name	Region	Year	Amount_Sold	Gross Profit
Direct Sales	Americas	2006	\$15,080,493	\$3,166,638
	Asia	2006	\$2,527,941	\$530,876
	Europe	2006	\$8,524,757	\$1,771,096
	Middle East	2006	\$629	\$122
	Oceania	2006	\$1,181,632	\$247,640

- Other ways to view results are by clicking the view buttons below the tabs in the **Criteria** pane, or by clicking the **Display Results** button below the columns in your query criteria. By default, the results are displayed in the **Compound Layout** view, which contains two other views, a **Title** view and a **Table** view. As you will see,

you can delete these default views and add other views to the Compound Layout. Later, you will add these different types of views to a dashboard.

10. Notice **Amount_Sold** for the **Middle East** Region is significantly lower than the other Regions. Return to the **Criteria** tab and create a filter for Regions that does not include Middle East, as shown in [Figure 5–11](#).

Figure 5–11 Filter for Regions



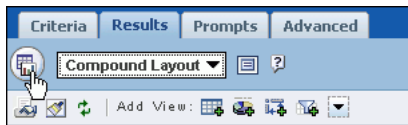
11. Remove the other two filters by clicking the **Delete** icon next to the filters.
12. Save the filter by clicking the **Save filter** button. In the Save Filter dialog box, select **My Filters**. Enter **Filter on Region** in the **Name** field and click **OK**.

5.2.3 Create Totals and Format Results

To create totals and format results:

1. Click the **Combine individual views for display on dashboards** button.

Figure 5–12 Combine Individual Views for Display on Dashboards



2. To open the Edit Table view for your results, click the **Edit view** icon for the **Table** view in the Compound Layout.

Figure 5–13 Edit View Icon



The column controls for each column are displayed with the results. Using the view-level controls, you can also set table-wide formatting properties, import formatting from other queries, and set grand totals for the entire table.

3. Click the **Total By** icon above the **Channel_name** column to add subtotals by channel to your results.

Figure 5–14 Total By Icon

Channel_name	Region	Amount_Sold
Direct Sales	Americas	\$31,611,009
	Asia	\$5,365,120
	Europe	\$18,768,575
	Oceania	\$2,429,295
Internet	Americas	\$8,042,944
	Asia	\$1,631,094
	Europe	\$3,408,698
	Oceania	\$578,433
Partners	Americas	\$15,664,839
	Asia	\$3,355,325
	Europe	\$6,487,143

- The measure is totaled each time the value in the Channel_name column changes. In this case, the default aggregation rule (SUM) is applied. The default aggregation rule is set in the Oracle BI repository metadata, but can be overridden using controls in the Edit Formula dialog box accessed through the Edit Formula icon in the Edit Table view or the Criteria tab.

Figure 5–15 Subtotals by Channel

Channel_name	Region	Amount_Sold
Direct Sales	Americas	\$31,611,009
	Asia	\$5,365,120
	Europe	\$18,768,575
	Oceania	\$2,429,295
Direct Sales Total		\$58,173,998
Internet	Americas	\$8,042,944
	Asia	\$1,631,094
	Europe	\$3,408,698
	Oceania	\$578,433
Internet Total		\$13,661,169
Partners	Americas	\$15,664,839
	Asia	\$3,355,325
	Europe	\$6,487,143
	Oceania	\$952,581
Partners Total		\$26,459,888

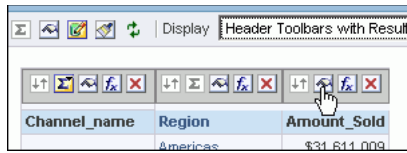
- Select **Table** in the **View** menu. Click the **Grand Total** button at the table view level to add a grand total to your results.

Figure 5–16 Grand Total Button

Channel_name	Region	Amount_Sold
	Americas	\$31,611,009

- Scroll down to the bottom of the Results pane and verify that the grand total that you set for the results is present.
- Click the **Column Properties** icon above the **Amount_Sold** column.

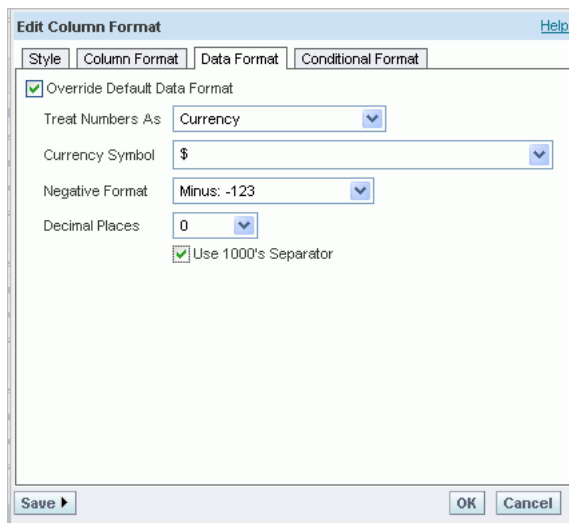
Figure 5–17 Column Properties Icon



- In the Column Properties dialog box, click the **Data Format** tab, select the **Override Default Data Format** option, then select the **Use 1000's Separator** option and click **OK**. This displays a comma separator in the number results for the column.

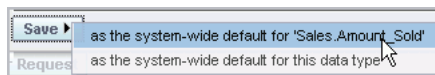
If the column is already formatted, ignore these steps.

Figure 5–18 Edit Column Format: Data Format Tab



- In the Edit Column Format dialog box, click **Save**. If you have permissions as a Web administrator, you can save the data format as the systemwide default for the column you are working with, or for all columns with the same data type.

Figure 5–19 Saving as the System-Wide Default



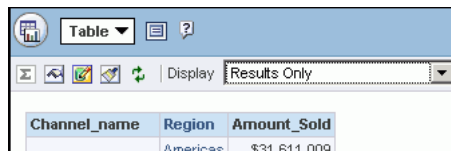
- Click **OK** and verify that a 1000's separator is displayed in the results for the **Amount_Sold** column.

Figure 5–20 Amount_Sold with 1000's Separator

Channel_name	Region	Amount_Sold
Direct Sales	Americas	\$31,611,009
	Asia	\$5,365,120
	Europe	\$18,768,575
	Oceania	\$2,429,295
Direct Sales Total		\$58,173,998
Internet	Americas	\$8,042,944
	Asia	\$1,631,094
	Europe	\$3,408,698
	Oceania	\$578,433
Internet Total		\$13,661,169
Partners	Americas	\$15,664,839
	Asia	\$3,355,325
	Europe	\$6,487,143
	Oceania	\$952,581
Partners Total		\$26,459,888
Grand Total		\$98,295,055

11. In the **Display** drop-down menu, select **Results Only** to eliminate the Header Toolbars.

Figure 5–21 Display: Results Only

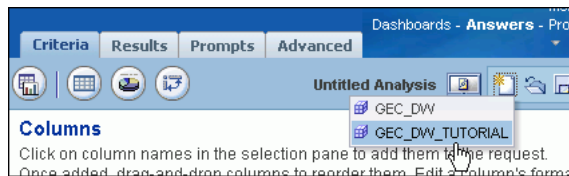


5.2.4 Create a Chart

To create a chart:

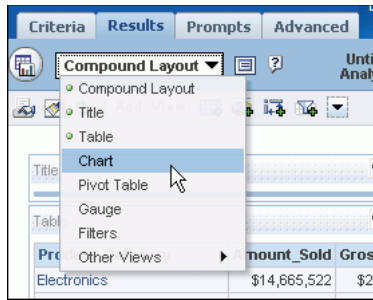
1. Click the **Create a new request** icon and select the **GEC_DW_TUTORIAL** subject area.

Figure 5–22 Creating a New Request for the GEC_DW_TUTORIAL Subject Area



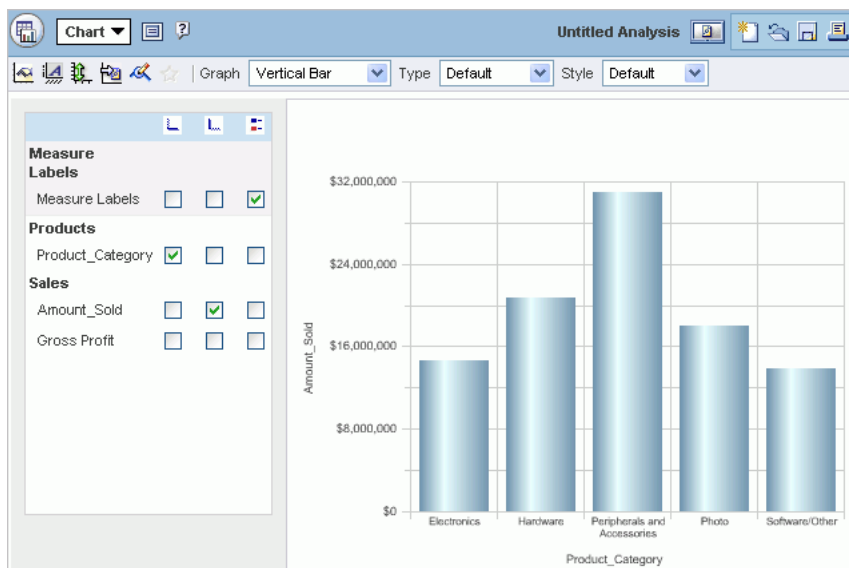
2. Create a request with **Product_Category** under **Products**, and **Amount_Sold** and **Gross Profit** under **Sales**.
3. In the **Results** tab, verify that **Amount_Sold** and **Gross Profit** have the **\$** currency symbol. If necessary, format them so that they have the currency symbol in all requests.
4. Pick **Chart** in the View menu.

Figure 5–23 Selecting Chart



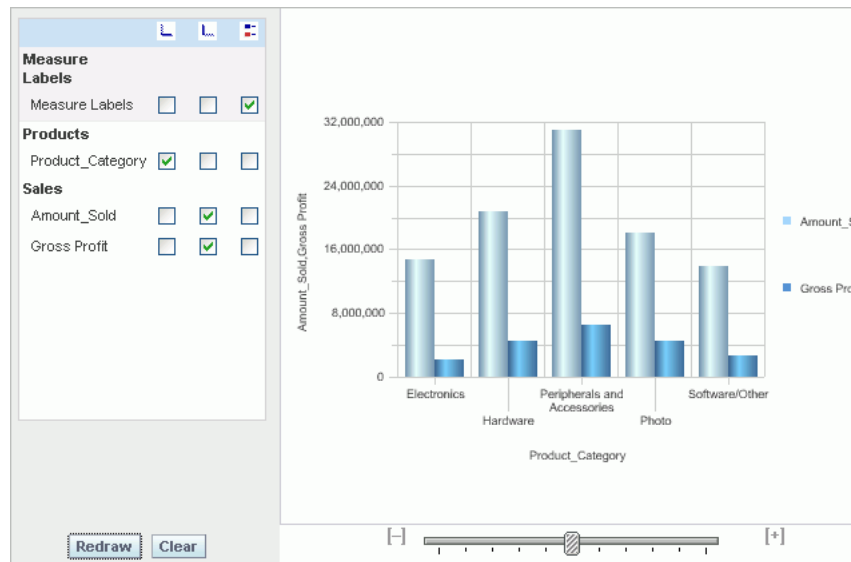
By default, charts display as a Vertical Bar graph.

Figure 5–24 Chart View



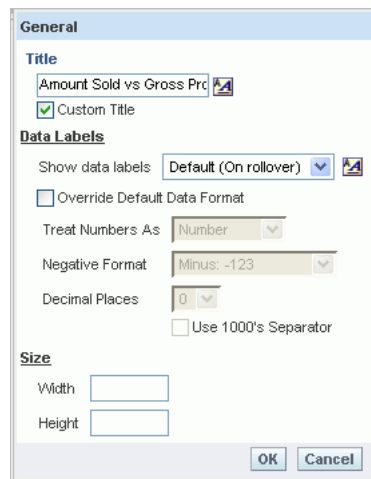
5. Select the y-axis box for the **Gross Profit** measure. Click the **Redraw** button at the bottom. Notice that the chart redraws with a legend for measures.

Figure 5–25 Chart with Legend for Measures



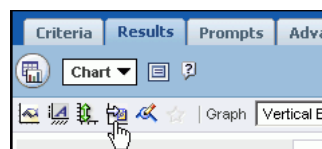
6. Click the **General** chart properties icon (first icon from the left). Click the **Custom Title** box. Enter **Amount Sold vs Gross Profit**, then click **OK**.

Figure 5–26 General Chart Properties



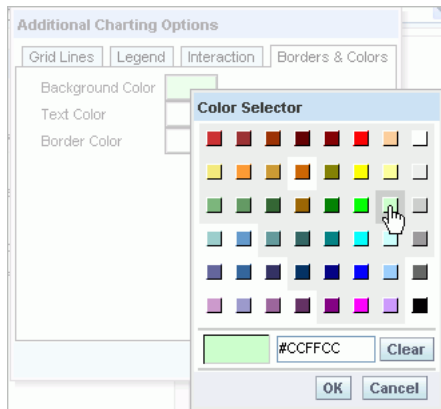
7. Explore some of the other options in the Axis Titles & Labels dialog box (second icon from the left). For example, you can toggle whether scale labels displays on the axes and set orientation guidelines for your labels. In addition, you can set overrides for the default data format on measures.
8. Click the **Additional Charting Options** icon.

Figure 5–27 Additional Charting Options icon



- In the Additional Charting Options dialog box, click the **Borders & Colors** tab. Click the color box for **Background Color** and, in the Color Selector dialog box, select **light green** from the palette and click **OK**.

Figure 5–28 Additional Charting Options: Borders & Colors



- In the Additional Charting Options dialog box, click the **Grid Lines** tab and set the major grid line color to **white** and the minor grid line color to **light gray**, using the same method as described in the previous step. After you have set both colors, click **OK** to apply your changes.
- Select the graph **Type** as **3D**. Your chart should look like the one shown in [Figure 5–29](#).

Figure 5–29 Chart After Selecting Additional Charting Options



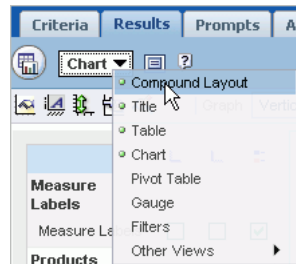
- Sort the request by **Gross Profit**. Click the **Criteria** tab and click the **Order By** icon (the icon with two arrows) in the **Gross Profit** column. The arrow points up to indicate an ascending sort. Click again, and the arrow points downward to indicate descending order.
- Click the **Results** tab to verify that your sort has been applied to your chart.

Figure 5–30 Chart with Sorting By Gross Profit



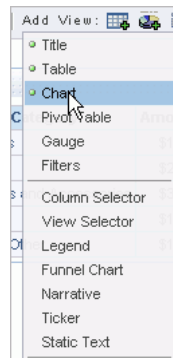
14. Select **Compound Layout** from the **View** drop-down menu.

Figure 5–31 Selecting Compound Layout



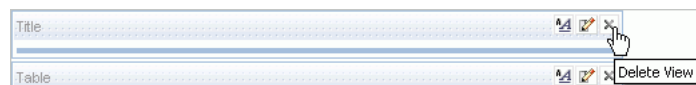
15. Click the **Add View** link and select **Chart** to add the Chart view to the Compound Layout view.

Figure 5–32 Adding the Chart View



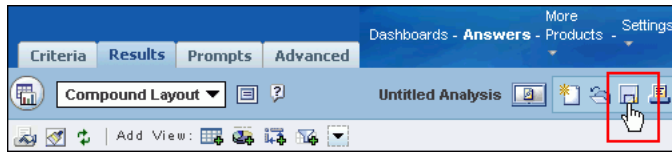
16. Click the **Delete View** icon in the Title view to delete it from the Compound Layout.

Figure 5–33 Deleting a View



17. Click the **Save Request** icon.

Figure 5–34 Save Request icon



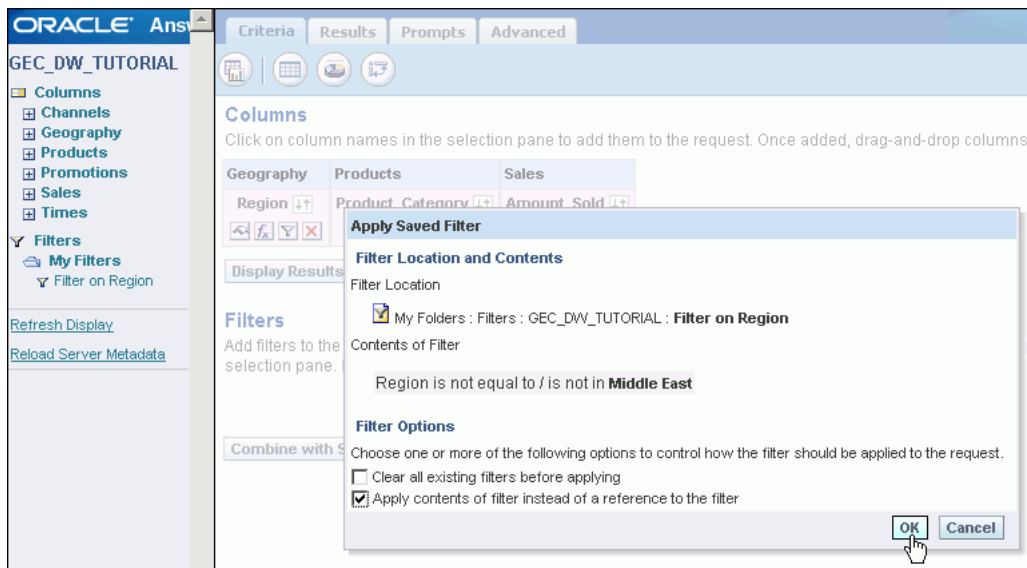
18. In the Save Request dialog box, click **My Folders**, then click the **Create Folder** button. In the Create Folder In My Folders dialog box, enter **GEC_DW_TUTORIAL** and click **OK**.
19. Click **Create Folder** again. In the Create Folder In My Folders dialog box, enter **Learn** and click **OK**.
20. Enter **Category Gross Profits** as the **Name** and click **OK**.

5.3 Working with Pivot Tables

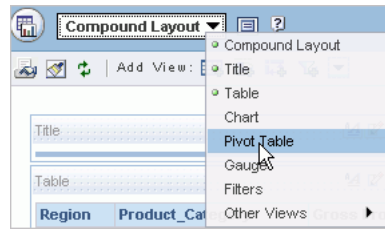
To work with pivot tables:

1. Create a new request with **Region** under **Geography**, **Product_Category** under **Products**, and **Gross Profit** under **Sales**.
2. Apply the previously saved filter called **Filter on Region**. Click **Filter on Region** in the left navigation tree. In the Apply Saved filter dialog box, select **Apply contents of filter instead of a reference to the filter** and click **OK**.

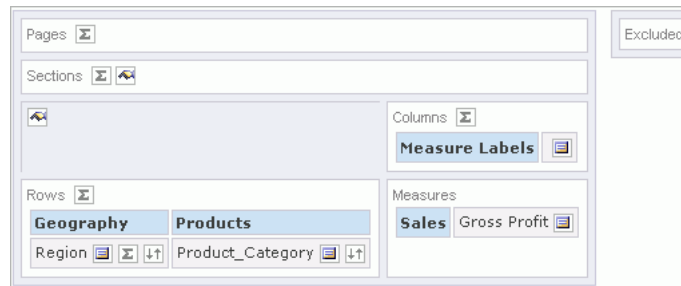
Figure 5–35 Applying a Previously Saved Filter



3. In the **Results** tab, pick **Pivot Table** in the **Compound Layout View** menu. This opens the Pivot Table Layout page.

Figure 5–36 Opening the Pivot Table Layout Page

4. Examine the default pivot table layout and the pivot table that is created. Measure labels for the measures in your query appear in blue as columns in the pivot table. Row headings for the dimensional attributes in your query are displayed in gray as rows by default. The measures are displayed at the intersection of the rows and columns.

Figure 5–37 Default Pivot Table Layout**Figure 5–38** Pivot Table

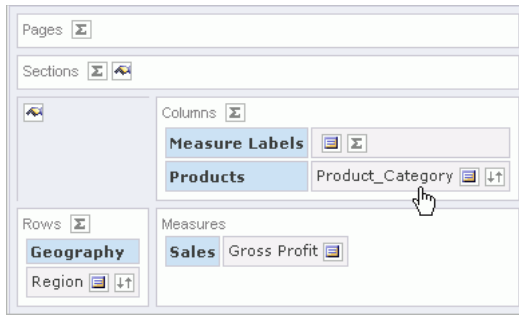
Region	Product_Category	Gross Profit
Americas	Electronics	1,267,454
	Hardware	2,373,452
	Peripherals and Accessories	3,688,919
	Photo	2,497,317
	Software/Other	1,587,793
	Electronics	212,503

5. Deselect the **Display Results** option. This option displays the results of any layout modifications you make as you work in the Pivot Table layout. To speed performance, you will only view your results periodically.

Figure 5–39 Deselecting Display Results

6. Drag the **Product_Category** column below the Measure Labels in the Columns area in your layout controls. When you see a blue line appear, you have a valid insertion point and can drop the column.

Figure 5–40 Product_Category



7. Click the **Display Results** link to verify your changes in the pivot table.
8. Drag the **Product_Category** column above the Measure Labels in the column section.

Your pivot table should look like the one shown in [Figure 5–41](#).

Figure 5–41 Pivot Table with Product_Category Column

	Electronics	Hardware	Peripherals and Accessories	Photo	Software/Other
Region	Gross Profit	Gross Profit	Gross Profit	Gross Profit	Gross Profit
Americas	\$1,267,454	\$2,373,452	\$3,688,919	\$2,497,317	\$1,587,793
Asia	\$212,503	\$513,427	\$679,203	\$519,079	\$228,127
Europe	\$647,881	\$1,370,214	\$1,860,473	\$1,380,126	\$716,466
Oceania	\$92,845	\$190,690	\$254,658	\$201,451	\$92,597

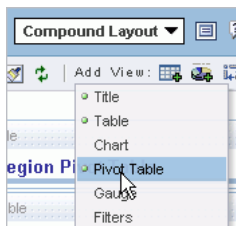
9. Interchange **Region** with **Product_Category**. Your pivot table should like the one shown in [Figure 5–42](#).

Figure 5–42 Pivot Table with Interchanged Columns

	Americas	Asia	Europe	Oceania
Product_Category	Gross Profit	Gross Profit	Gross Profit	Gross Profit
Electronics	\$1,267,454	\$212,503	\$647,881	\$92,845
Hardware	\$2,373,452	\$513,427	\$1,370,214	\$190,690
Peripherals and Accessories	\$3,688,919	\$679,203	\$1,860,473	\$254,658
Photo	\$2,497,317	\$519,079	\$1,380,126	\$201,451
Software/Other	\$1,587,793	\$228,127	\$716,466	\$92,597

10. Select **Compound Layout** from the **View** drop-down menu. Click the **Add View** link and select **Pivot Table** to add the Pivot table view to the Compound Layout view.

Figure 5–43 Selecting Compound Layout > Add View > Pivot Table



- The Pivot table is added at the bottom. Drag the Pivot Table and place it above the Table.

Figure 5–44 Moving the Pivot Table Up

	Americas	Asia	Europe	Oceania
Product_Category	Gross Profit	Gross Profit	Gross Profit	Gross Profit
Electronics	\$1,267,454	\$212,503	\$647,881	\$92,845
Hardware	\$2,373,452	\$513,427	\$1,370,214	\$190,690
Peripherals and Accessories	\$3,688,919	\$679,203	\$1,860,473	\$254,658
Photo	\$2,497,317	\$519,079	\$1,380,126	\$201,451
Software/Other	\$1,587,793	\$228,127	\$716,466	\$92,597

Region	Product_Category	Gross Profit
Americas	Electronics	\$1,267,454
	Hardware	\$2,373,452
	Peripherals and Accessories	\$3,688,919
	Photo	\$2,497,317
	Software/Other	\$1,587,793

- You will be adding Column Selector to this request, so save the request as **Region Pivot Table with Column Selector** under the folder **GEC_DW_TUTORIAL > Learn**.

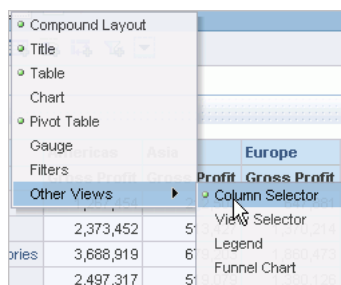
5.4 Creating Column Selectors

In this section, you build a Column Selector and see the effects in a request. Column Selectors allow users to select from a group of columns, substituting columns in their queries for comparative analysis.

To create column selectors:

- In the **Results** tab, select **Column Selector** from the **View > Other Views** drop-down menu.

Figure 5–45 Choosing the Column Selector View



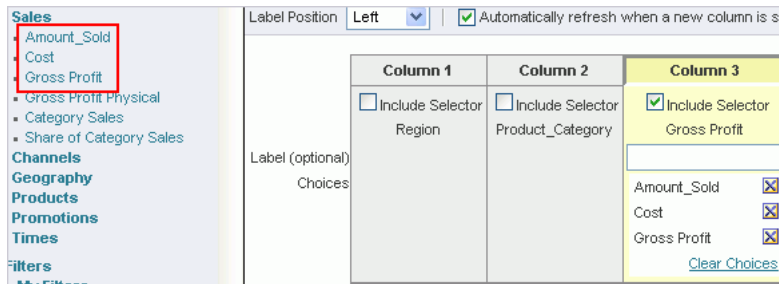
- Select the **Include Selector** option in Column 3, currently Gross Profit.

Figure 5–46 Selecting the Include Selector Option

Column 1	Column 2	Column 3
<input type="checkbox"/> Include Selector Region	<input type="checkbox"/> Include Selector Product_Category	<input checked="" type="checkbox"/> Include Selector Gross Profit
		<input type="text"/> Click on a column in the selection pane to add choices.

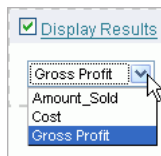
- In the selection pane on the left, click the following columns to make them available in the Column Selector: **Amount_Sold**, **Cost**, and **Gross Profit**.

Figure 5–47 Choosing Amount_Sold, Cost, and Gross Profit



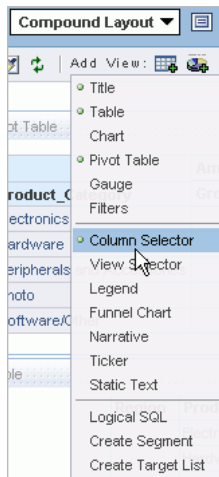
Your Column Selector in the Results pane should look like the one shown in Figure 5–48.

Figure 5–48 Column Selector



- Select **Compound Layout** from the **View** drop-down menu. Click the **Add View** link and select **Column Selector** to add the Column Selector view to the Compound Layout view.

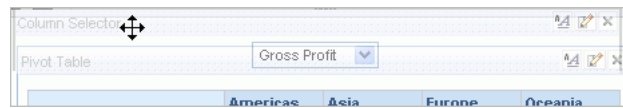
Figure 5–49 Adding the Column Selector View to the Compound Layout View



Notice that views that you have built for the active request appear in the **View** and **Add View** drop-down menus with green circle icons next to them.

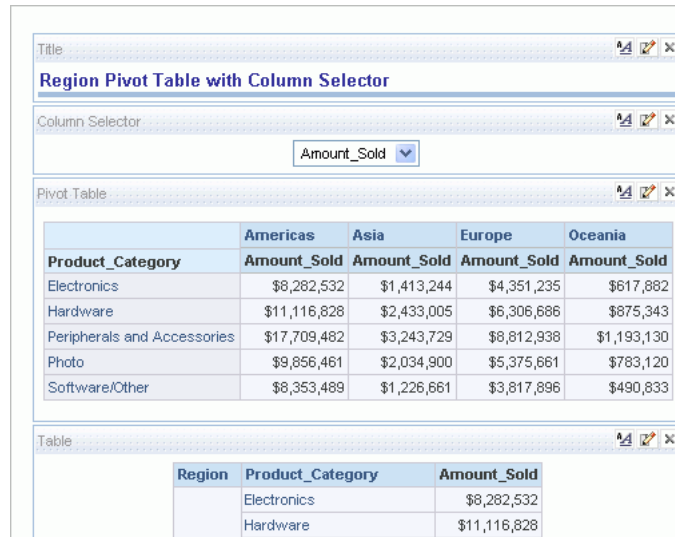
- Scroll to the bottom of the Compound Layout and drag the Column Selector view above both Pivot Table and Table views.

Figure 5–50 Positioning the Column Selector View



6. Scroll down, locate the **Title** box, and drag it above the Column Selector view.
7. In the Column Selector, select **Amount_Sold**. Your Compound Layout view should look like the one shown in Figure 5–51.

Figure 5–51 Compound Layout View with Column Selector



8. Note that **Amount_Sold** is selected in the Column Selector, and both the Pivot Table and the Table displays values for Amount_Sold. Select **Cost** and see the change.
9. In the Compound Layout, you now have a Column Selector, a Pivot Table, a Table, and a Title.
10. Save the request with the same name: **Region Pivot Table with Column Selector**.

5.5 Creating a Narrative View

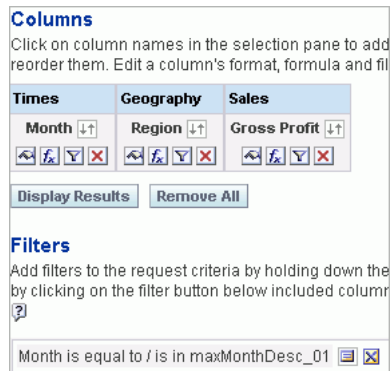
In this section, you build and format a business intelligence request using Oracle BI Answers, and create and format a Narrative View. The Narrative view lets you add text that appears with the results to provide information such as context, explanatory text, or extended descriptions.

To create a Narrative View:

1. Create a new request with the following columns: **Month** under **Times**, **Region** under **Geography**, and **Gross Profit** under **Sales**. Add a filter on **Month** to get the last transaction month in the data source. The filter will use the repository variable **maxMonthDesc_01** you created earlier.
2. Click the filter icon on the Month column. In the Create Filter dialog box, set the Operator to be **is equal to/is in**. Click **Add>Variable>Repository**.
3. In the **Value** field, enter **maxMonthDesc_01**. Click **OK**.

Your request should look like the one shown in [Figure 5–52](#).

Figure 5–52 Request with maxMonthDesc_01 Filter



In the **Results** tab, the request should display like the one shown in [Figure 5–53](#).

Figure 5–53 Results Tab with maxMonthDesc_01 Filter

The screenshot shows the 'Results' tab of the Oracle BI request editor. It displays a table with the following data:

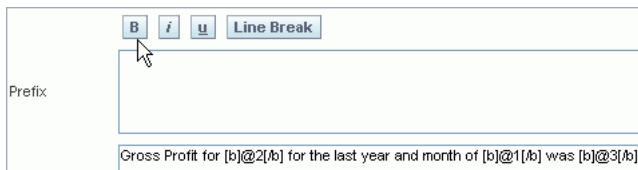
Month	Region	Gross Profit
December 2006	Americas	\$1,425,648
	Asia	\$308,502
	Europe	\$680,652
	Oceania	\$79,716

4. Select **Narrative** from the **View > Other Views** drop-down menu. In the Narrative view workspace, in the **Narrative** field, enter the following: **Gross Profit for @2 for the last year and month of@1 was @3.**

The narrative is a combination of text and query column values. In this example, @1 refers to the first column in the query, Month, @2 refers to the second column, Region, and @3 refers to Gross Profit. Note that you can control the number of row values returned in the Narrative view by setting the Rows to display value. By default, all queried rows are displayed.

5. To highlight the column values in the narrative, select @1 in the narrative and click the **Bold** button. Also add bold tags to @2 and @3.

Figure 5–54 Choosing Bold for Portions of the Narrative Field



6. Position the cursor at the end of the narrative text, and then click the **Line Break** button to insert a line feed between each sentence. Enter 2 in the **Rows to display** field.
7. In the **Compound Layout**, add the **Narrative View** and delete the **Table** and **Title** views.

- Save the request as **Narrative View for Region Gross Profit** under **GEC_DW_TUTORIAL > Learn**.

5.6 Creating View Selectors

In this section, you build a View Selector and experiment with its use in requests. View Selectors allow users to quickly navigate between different views of their queries (for example, viewing different charts of the same data, or quickly navigating to a pivot table to do trend analysis).

To create view selectors:

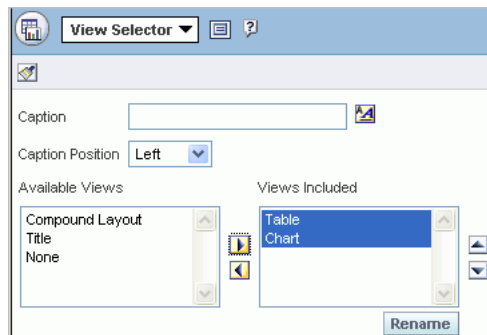
- In Oracle BI Answers, click the **Answers** link. Go to the **My Folders > GEC_DW_TUTORIAL > Learn** folder and open the first request, **Category Gross Profit**, that you saved.
- Click **Modify**, then click the **Results** tab.

Figure 5–55 Category Gross Profit Request



- From the **View** drop-down menu, select **Other Views > View Selector**. In the View Selector design workspace, select **Table** and **Chart** in the Available Views field, using **Ctrl+Click** to select multiple views.
- Click the **Move Right** icon to add them to the **Views Included** field.

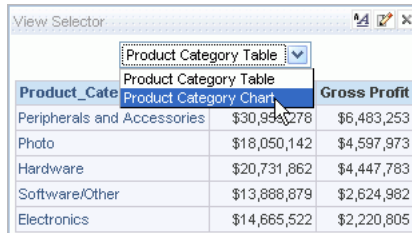
Figure 5–56 Selecting the Table and Chart Views



- Select **Table** in the **Views Included** field, click the **Rename** button, and, in the Rename dialog box, rename it to **Product Category Table**. Click **OK**.
- Rename **Chart** to **Product Category Chart**. Click **OK**.

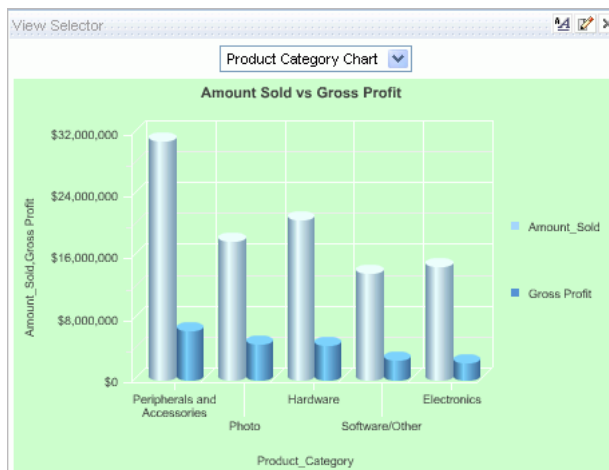
7. Use the **View** drop-down menu to go to the **Compound Layout** view. Delete the **Table** and **Chart** view from the Compound Layout.
8. Add the **View Selector** view using the **Add View** button. Also add the **Title** view.
9. Select **Product Category Chart** from the View Selector.

Figure 5–57 Choosing Product Category Chart from the View Selector



The chart should change to the Chart view, as shown in [Figure 5–58](#).

Figure 5–58 Product Category Chart View



10. Save the request as **Product Category View Selector** under **GEC_DW_TUTORIAL > Learn**.

5.7 Creating Conditional Formats, Gauges, and Navigation

In this section, you create a report with conditional formats, add gauges, and from one measure field, navigate to a different report to get further details.

This section contains the following topics:

- [Section 5.7.1, "Create a Report with Conditional Formats"](#)
- [Section 5.7.2, "Add Gauges to the Report"](#)
- [Section 5.7.3, "Add Navigation to the Report"](#)

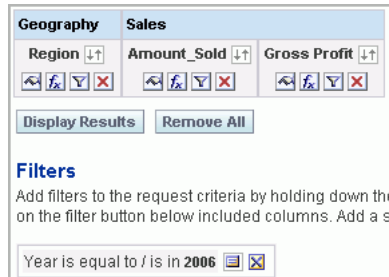
5.7.1 Create a Report with Conditional Formats

To create a report with conditional formats:

1. Create a request with **Year**, **Region**, **Amount_Sold**, and **Gross Profit**. Create a filter condition on the **Year** column, where **Year=2006**.
2. Delete the **Year** column.

The request criteria should appear as shown in [Figure 5-59](#).

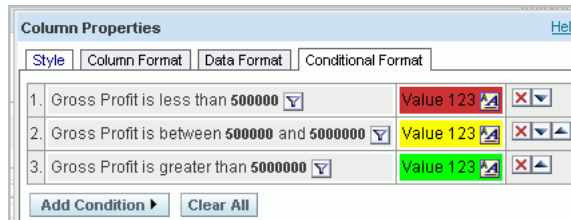
Figure 5-59 *Creating a Request with Year=2006 Filter*



3. Click the **Column properties** icon for the **Gross Profit** column. Click the **Conditional Format** tab. Add the following conditions by clicking **Add Condition** and selecting **Gross Profit** each time. Click the **Value** Property icon, and set the **Cell Background Color** for the conditions as follows:

- Gross Profit < 500,000 = red
- Gross Profit between 500,000 and 5,000,000 = yellow
- Gross Profit > 5,000,000 = green

Figure 5-60 *Conditions for Gross Profit Column*



4. Click the **Results** tab. The report should appear like the one shown in [Figure 5-61](#).

Figure 5-61 *Results with Conditions for Gross Profit*

Region	Amount_Sold	Gross Profit
Americas	\$31,120,994	\$5,000,000
Asia	\$5,820,345	\$1,192,527
Europe	\$15,395,828	\$3,138,300
Middle East	\$629	\$100
Oceania	\$2,247,646	\$403,085

5. Save the request as **Advanced Report** under the **GEC_DW_TUTORIAL > Learn** folder.

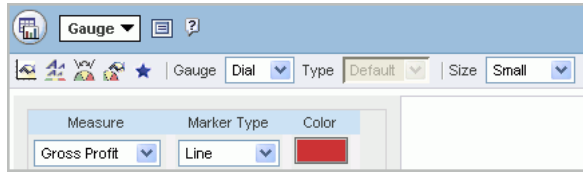
5.7.2 Add Gauges to the Report

To add gauges to the report:

1. In the **Results** tab for the **Advanced Report**, select **Gauge** from the **View** drop-down list. In the **Gauge** drop-down list, select **Dial**. In the **Size** drop-down

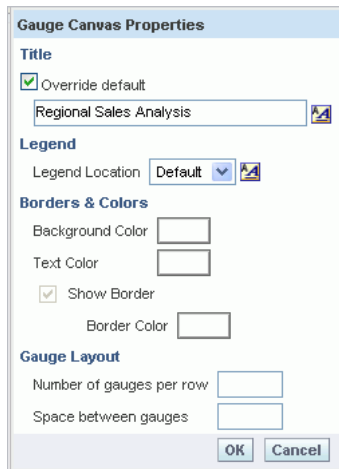
list, select **Small**. In the **Measure** drop-down list, select **Gross Profit**. In the **Marker Type** drop-down list, select **Line**. Click the color button to change the color for the indicator needle. Select a bright red color. Click **OK**.

Figure 5–62 Formatting Results for Advanced Report



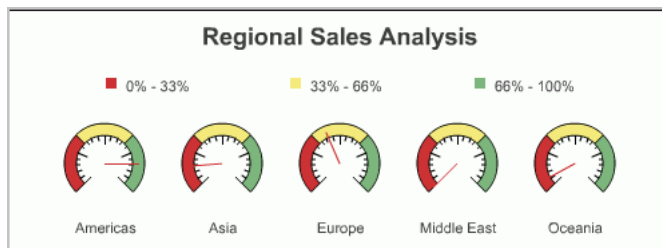
2. Click the **Gauge Canvas Properties** button at the top of the Gauge view page. In the **Title** section, select **Override default**. In the **Title** field, type **Regional Sales Analysis**. Click **OK**.

Figure 5–63 Gauge Canvas Properties



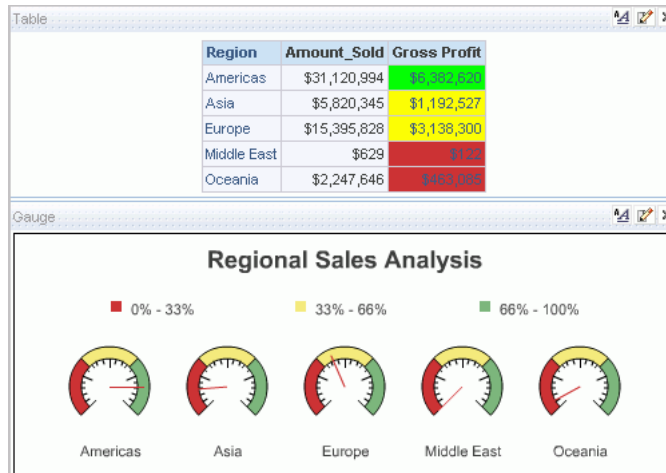
The gauges should look like the ones shown in [Figure 5–64](#).

Figure 5–64 Gauges for Advanced Report



3. Add the gauge to the compound layout, and remove the title. The report should look like the one shown in [Figure 5–65](#)

Figure 5–65 Results and Gauges for Advanced Report



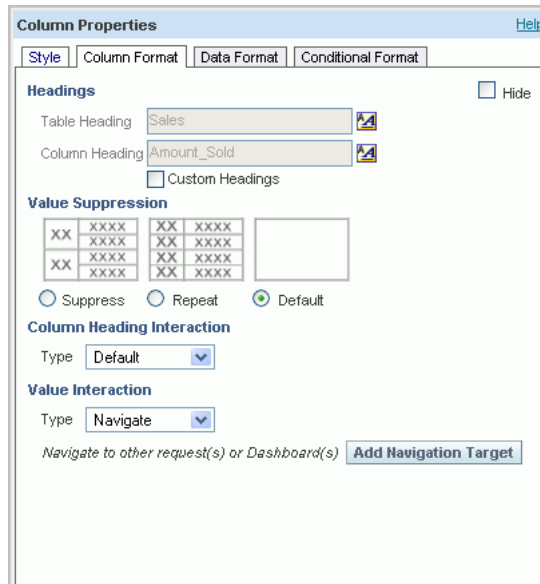
4. Save the request with the same name.

5.7.3 Add Navigation to the Report

To add navigation to the report:

1. In the **Criteria** tab for the **Advanced Report**, select **Column Properties** for the **Amount_Sold** column. Click the **Column Format** tab. In the **Value Interaction** section, select **Navigate** from the **Type** drop-down list.

Figure 5–66 Column Format Tab of Column Properties



2. Click the **Add Navigation Target** button. Click the **Browse** button and select the **Product Category View Selector** request under the **GEC_DW_TUTORIAL > Learn** folder. Click **OK**, then click **OK** again.

Figure 5–67 Selecting the Product Category View Selector Request

Value Interaction

Type: Navigate

Navigate to other request(s) or Dashboard(s) Add Navigation Target

Target: /Users/administrator/GEC_DW_TUTORIAL Browse...

Caption: Remove

- In the **Results** tab, click the **Amount_Sold** column for any region.

Figure 5–68 Choosing the Amount_Sold Column

Region	Amount_Sold	Gross Profit
Americas	\$31,120,994	\$6,382,620
Asia	\$5,820,345	\$1,192,527
Europe	\$15,582,828	\$3,138,300
Middle East	\$629	\$122
Oceania	\$2,247,646	\$463,085

A new windows appears with the **Product Category View Selector** request that shows the product categories for the region.

Figure 5–69 Product Category View Selector Request

Product Category View Selector

Product Category Table

Product_Category	Amount_Sold	Gross Profit
Peripherals and Accessories	\$30,959,278	\$6,483,253
Photo	\$18,050,142	\$4,597,973
Hardware	\$20,731,862	\$4,447,783
Software/Other	\$13,888,879	\$2,624,982
Electronics	\$14,665,522	\$2,220,805

[Download](#)

- Save the request with the same name.

Publish Data on Dashboards

In this chapter, you publish data on dashboards for user access. You use Oracle BI Interactive Dashboards to:

- Build a custom Dashboard to contain the BI Answers requests and views you created in the last chapter.
- Work with dashboard prompts to filter your dashboard and populate variables.

This chapter contains the following topics:

- [Section 6.1, "Prerequisites"](#)
- [Section 6.2, "Creating an Interactive Dashboard"](#)
- [Section 6.3, "Using Dashboard Prompts and Presentation Variables"](#)

6.1 Prerequisites

If you have not completed the previous chapters, you must first populate the BISE1_TUTORIALWH schema before proceeding with this chapter. [Section 2.2.2.3, "BISE1_TUTORIALWH Schema"](#) explains how to do this.

If you have not completed the previous chapters, you must also replace the BI metadata repository file, as follows:

1. Shut down the BI Server service.
2. Copy `tutorial\bi_ad\bise1_soln.rpd` to the `bi\oraclebi\server\repository` directory.
3. Rename `bi\oraclebi\server\repository\bise1_soln.rpd` to `bise1.rpd`.
4. Start the BI Server service.

If you specified a password other than **welcome1** during installation, perform the following steps as well:

1. Go to **Start > Programs > Oracle Business Intelligence > Administration**.
2. Select **File > Open > Online**.
3. Enter **Administrator** in the Password field. Click **OK**.
4. In the **Physical Layer** pane, expand **BISE1_SALESWH**. Double-click **Connection Pool**.
5. In the **Password** field, enter the password you specified during installation. Click **OK**.

6. Enter the password again in the **Confirm Password** dialog box. Click **OK**.
7. Repeat steps 4 through 6 for the **BISE1DB** node in the **Physical Layer**.
8. Select **File > Save**. Click **No** when asked whether you wish to check global consistency. Exit the Oracle BI Administration tool.

If you have not completed the previous chapters, you must also replace the BI Web catalog, as follows:

1. Shut down the BI Presentation Server service.
2. Copy tutorial\bi_ad\bise1_webcat_soln.zip to the bidata\web\catalog directory.
3. Remove the bidata\web\catalog\bise1 directory and its contents.
4. Unzip bidata\web\catalog\bise1_webcat_soln.zip.
5. Start the BI Presentation Server service.

6.2 Creating an Interactive Dashboard

In this section, you build a new shared Interactive Dashboard and add content you have saved in previous steps.

To create an Interactive Dashboard:

1. Click the **Settings** link and select **Administration**.

Figure 6–1 *Choosing Settings > Administration*



2. In the Oracle BI Presentation Services Administration window, click the **Manage Interactive Dashboards** link.

The Oracle BI Presentation Services Administration window offers access to many administrative features, including session monitoring and management of user and group privileges across the Oracle BI Presentation Catalog.

3. In the Manage Dashboards window, click **Create Dashboard**. In the Create Dashboard window, set the Group Folder to **Sales Managers**, name the dashboard **Sales Analysis1**, and click **Finished**.
4. The dashboard inherits the security of the group folder. Click **Finished** again to close the Manage Dashboards window. Finally, click **Close Window** to close the Oracle BI Presentation Services Administration screen.
5. Click the **Dashboards** link, then click the **Sales Analysis1** Interactive Dashboard link to navigate to the new dashboard.

Figure 6–2 *Navigating to the New Dashboard*



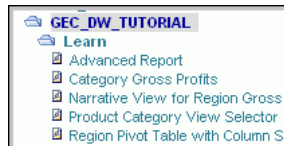
Your new dashboard is empty, and should appear like the one shown in [Figure 6–3](#).

Figure 6–3 Sales Analysis1 Interactive Dashboard



6. Click **Page Options** in the upper right corner and select **Edit Dashboard** to open the Dashboard Editor. In the Dashboard Editor, you can drag saved content from your Presentation Catalog directly onto the dashboard. Sections are automatically created in the layout to contain the requests and other objects you add to the dashboard. In the left-hand selection panel, expand the **GEC_DW_TUTORIAL > Learn** folder where you have saved your work.

Figure 6–4 Expanding the GEC_DW_TUTORIAL > Learn Folder



7. Drag the **Category Gross Profits** request onto the layout workspace. The layout area is highlighted in blue to indicate that you have a valid insertion point for the object.

Your dashboard layout should look like the one shown in [Figure 6–5](#).

Figure 6–5 Dashboard with Category Gross Profits Request



Notice that a section was automatically added to contain the request. You could also have dragged a Section object from the Dashboard Objects palette to create the section before dragging content into the section. Sections and columns are containers you can use to control the layout of your dashboard.

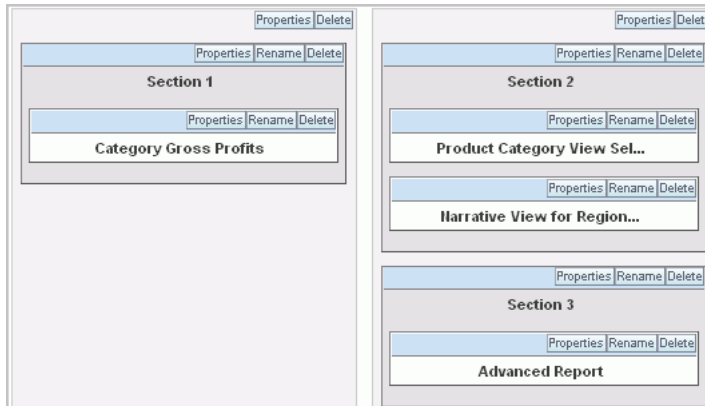
8. Click the insert column **plus (+)** icon to add another column to the page.

Figure 6–6 Plus Icon: Insert Column



9. Drag the **Product Category View Selector** request into **Section 2** of the second column. Drag the **Narrative View for Region Gross Profit** below the Product Category View Selector.
10. Drag **Section** from the **Dashboard Objects** list and add a section below **Section 2**.
11. Drag **Advanced Report** onto **Section 3**. The Dashboard layout should look like the one shown in [Figure 6–7](#).

Figure 6–7 Updated Dashboard Layout



12. Click **Save** at the top to save your layout for Page 1. The Dashboard should look like the one shown in [Figure 6–8](#).

Figure 6–8 Updated Dashboard



Note that the Column Selector only applies to the request that contains it. If you select a column in the control, it will only apply to the table and chart, not the narrative request in the same dashboard.

- Click **Page Options** and then **Edit Dashboard**. In the Dashboard Editor page, click the add page plus (+) icon on the top to add a new page to the Dashboard.

Figure 6–9 Plus Icon: Add Page

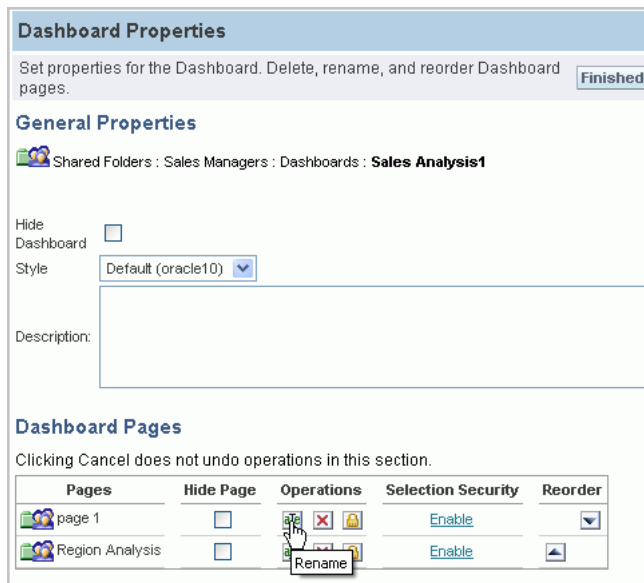


- Name the page **Region Analysis** and click **OK**.
- In the new page, drag the **Region Pivot Table with Column Selector** request. Click **Save**. Your Dashboard page should like the one shown in [Figure 6–10](#).

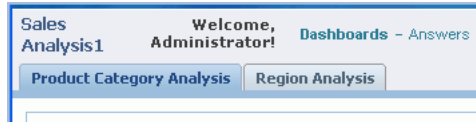
Figure 6–10 Region Analysis Dashboard Page



- Rename Page 1 of the Dashboard to **Product Category Analysis**. To do this, click **Page Options** and **Edit Dashboard**. In the Dashboard Editor, click the **Dashboard Properties** icon at the top, next to the name of the dashboard (**Sales Analysis 1**). In the Dashboard Properties page, click **Rename** under Operations for page 1.

Figure 6–11 Dashboard Properties Page: Choosing Rename

17. Rename the page to **Product Category Analysis**. Click the **Update** button. Click **Finished** in the Dashboard Properties page. Click **Save** in the Dashboard Editor. Your Dashboards pages should have two tabs, like the ones shown in [Figure 6–12](#).

Figure 6–12 Sales Analysis1 Dashboard with Product Category Analysis and Region Analysis Pages

6.3 Using Dashboard Prompts and Presentation Variables

In this section, you build a dashboard prompt to filter your dashboard for a specific country region. Then, you use a Presentation variable to dynamically update a request title in your dashboard that includes the filtered name of the country region. Dashboard prompts are used both to allow users to filter the results of embedded requests in a dashboard as well as to populate and update Presentation Variables, which are variables that can be defined in Oracle BI Answers.

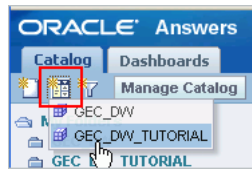
This section contains the following topics:

- [Section 6.3.1, "Create a Dashboard Prompt"](#)
- [Section 6.3.2, "Use a Presentation Variable to Populate a Title"](#)

6.3.1 Create a Dashboard Prompt

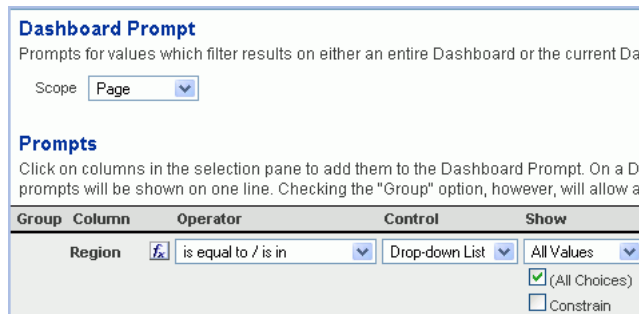
To create a Dashboard prompt:

1. Click the **Answers** link. In the selection pane, click the **New Dashboard Prompt** icon and select **GEC_DW_TUTORIAL** in the drop-down **Subject Area** menu.

Figure 6–13 Choosing GEC_DW_TUTORIAL

- In the selection pane, click **Geography.Region** to add it to the prompt. The default selection for the prompt is **Dashboard**. This means the filter will be applied to all pages in any dashboard with which it is associated. Select **Page** from the **Scope** drop-down list, because you will apply the prompt to the **Region Analysis** page only.

The prompt should look like the one shown in [Figure 6–14](#).

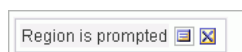
Figure 6–14 Dashboard Prompt

This prompt will filter the Region column in any requests that contain this filter in a dashboard with which the prompt is associated. Note that you can control several aspects of the prompt. You can select the type of control users will employ to enter their selections in the prompt, and you can determine what values will be presented in a drop-down list or multi-select control, limiting the available values either through SQL or by constraining the values based on the results of another dashboard prompt.

- Click the **Save** icon at the top and save the prompt under **GEC_DW_TUTORIAL > Learn** as **Region Prompt**.

You have created the prompt, but for it to take effect on any embedded requests in a dashboard, the requests must contain filters on the column being prompted. This task is described in the following steps.

- Click the **Answers** link. Open the **Region Pivot Table with Column Selector** request from **GEC_DW_TUTORIAL > Learn**. Click **Modify**.
- Remove the previously saved filter from the request.
- Click the filter icon for **Geography.Region**, and in the Create/Edit Filter dialog box, select **is prompted** as the operator for the filter and click **OK**. Your filter should look like the one shown in [Figure 6–15](#).

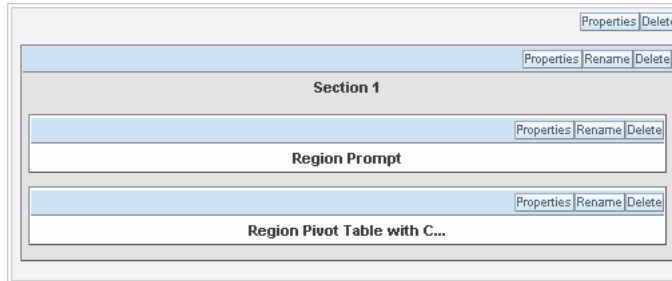
Figure 6–15 Region Is Prompted Filter

- Save the request.

8. Click the **Dashboards** link and go to the **Region Analysis** page of the Sales Analysis1 dashboard, then click **Page Options > Edit Dashboard** to open the Dashboard Editor.
9. From the left-hand selection panel, expand the **GEC_DW_TUTORIAL > Learn** folder and drag **Region Prompt** above the **Region Pivot Table with Column Selector** request in Section 1 of the dashboard. The area will be highlighted in blue when you have found a valid insertion point.

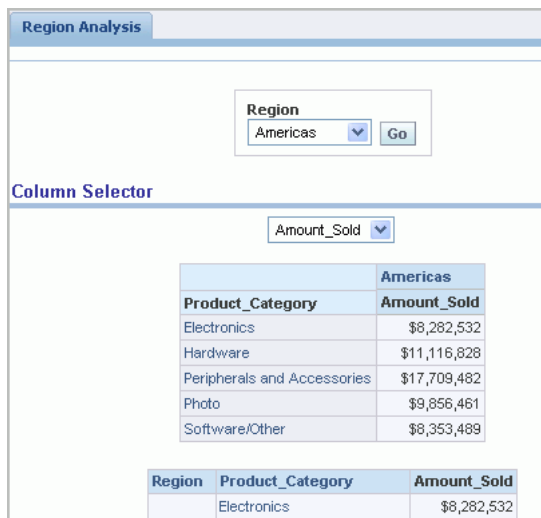
Your dashboard layout should look like the one shown in [Figure 6–16](#).

Figure 6–16 Dashboard with Region Prompt



10. Click the **Save** button. Your dashboard should look like the one shown in [Figure 6–17](#).

Figure 6–17 Updated Dashboard



11. Explore the report by selecting other regions.

6.3.2 Use a Presentation Variable to Populate a Title

To use a presentation variable to populate a title:

1. Click the **Answers** link. Open the **Region Pivot Table with Column Selector** request, then click **Modify**.
2. Click the **Results** tab. Select **Compound Layout View**, and edit the Title view using the **Edit View** icon. If the Title is not there, add a Title view to the

Compound Layout view using the **Add View** link, and drag the new Title view to the top of the layout.

Your Compound Layout should look like the one shown in [Figure 6–18](#).

Figure 6–18 Compound Layout

The screenshot shows a dashboard layout with three main components:

- Title:** A text field at the top.
- Column Selector:** A dropdown menu currently set to "Amount_Sold".
- Pivot Table:** A table displaying sales data by region and product category.

	Americas	Asia	Europe	Middle East	Oceania
Product_Category	Amount_Sold	Amount_Sold	Amount_Sold	Amount_Sold	Amount_Sold
Electronics	\$6,282,532	\$1,413,244	\$4,351,235	\$629	\$617,882
Hardware	\$11,116,828	\$2,433,005	\$6,306,686		\$875,343

- In the Title field, enter **Sales Analysis for @region**. This is a reference to the Presentation variable you are going to create. Clear the **Display Saved Name** option. Click OK.

Figure 6–19 Entering a Value in the Title Field

The screenshot shows the configuration for a Title view:

- Title:** Sales Analysis for @region
- Display Saved Name:** (unchecked)
- Logo:** (empty field)
- Subtitle:** (empty field)
- Started Time:** Do not display
- Help URL:** (empty field)

Optional - URL of a title image. Note: When running located on the Oracle BI Presentation Server machine, the relative path prefixed with "FMAP".

Note: The syntax for calling a Presentation variable, whether in a column or a Title view, is `@{VariableName}`.

- Save the request.
- Create the variable by designating that it be populated by the **Region Prompt**. Click the **Open** icon and open the **Region Prompt** from the **GEC_DW_TUTORIAL > Learn** folder.
- In the **Set Variable** field of the prompt, select **Presentation Variable**.

Figure 6–20 Setting the Variable to Be a Presentation Variable

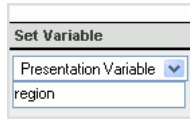
The screenshot shows a dropdown menu titled "Set Variable" with the following options:

- Presentation Variable (selected)
- None
- Presentation Variable
- Request Variable

Note that you could also reference a Request Variable. Request Variables are defined as Session Variables in the Oracle BI metadata and are instantiated when the user's session begins. Their values for any request can be updated by dashboard prompts.

- In the Set Variable text field, enter **region**.

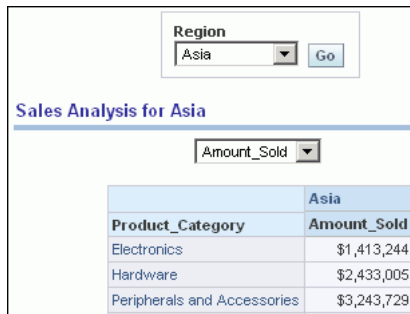
Figure 6–21 Entering "region" in the Text Field



By entering the name of the variable here, you are creating the variable as well as setting it to be populated by the user selection in the Prompt for Country Region.

- Save the dashboard prompt.
- Click the **Dashboards** link to navigate to the Region Analysis page of the Sales Analysis1 dashboard. Select **Asia** in the Region prompt and click **Go**.
- Verify that the Prompt for Region dashboard prompt is filtering the dashboard and updating the **region** Presentation variable.

Figure 6–22 Prompt for Region Dashboard Prompt



Create and Publish Reports

In this chapter, you create reports using Oracle BI Publisher to:

- Create an Invoices report from the Oracle Database
- Add parameters and lists of values to the report
- Build the template, and publish the report
- Schedule automated delivery of the report
- Create a report from an Oracle BI Answers request and publish the report in Oracle BI Interactive Dashboards
- Create a report from Oracle BI metadata

Prerequisites: There are no prerequisite steps for this chapter.

This section contains the following topics:

- [Section 7.1, "Logging in to Oracle BI Publisher and Setting Preferences"](#)
- [Section 7.2, "Creating a Report Based on the Operational Data"](#)
- [Section 7.3, "Automatically Delivering BI Publisher Reports"](#)
- [Section 7.4, "Publishing a BI Publisher Report in BI Dashboards"](#)
- [Section 7.5, "Creating an Oracle BI Publisher Report Using BI Server Metadata"](#)

7.1 Logging in to Oracle BI Publisher and Setting Preferences

To log in to Oracle BI Publisher and set preferences:

1. In Windows, click **Start > Programs > Oracle Business Intelligence > BI Publisher**. This opens the Log In screen for Oracle BI Publisher. Enter **Administrator** for both the **User ID** and **Password**. Click **Log In**.

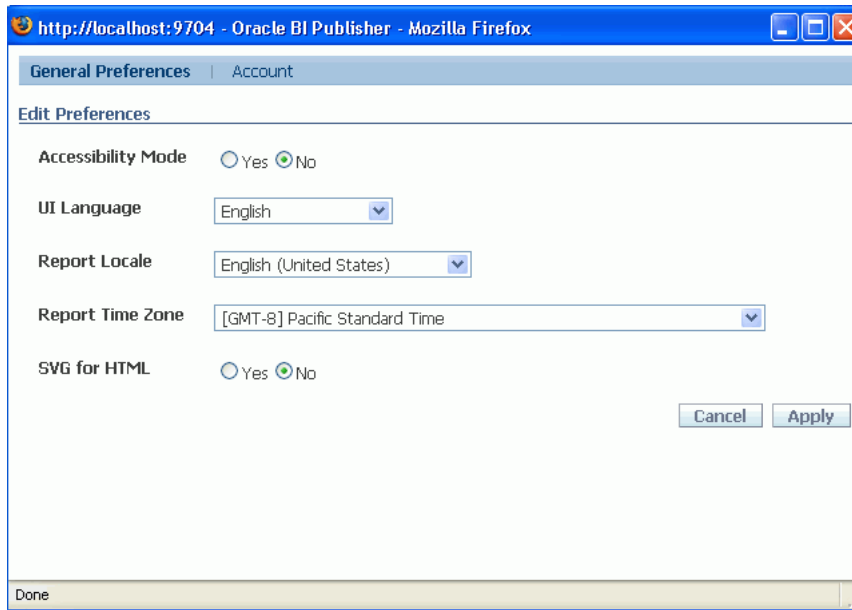
Figure 7–1 Oracle BI Publisher



Note: Because you have logged in as the Administrator, the welcome page displays the **Reports**, **Schedules**, and **Admin** tabs.

2. Click the **Preferences** link. The Preferences screen appears, with tabs for **General Preferences** and **Account**. Observe the options in the General Preferences tabbed page, and select appropriate options for **UI Language**, **Reports Locale**, and **Reports Time Zone** according to your requirements. You can also enable the accessibility option.

Figure 7–2 Setting Preferences



3. Click **Apply**, then close the window.

7.2 Creating a Report Based on the Operational Data

In this topic, you will create invoices from the **BISE1_SALES** operational data source.

This section contains the following topics:

- [Section 7.2.1, "Create a JDBC Connection to the Data Source"](#)

- [Section 7.2.2, "Define a New Report"](#)
- [Section 7.2.3, "Create an RTF Template for the Report"](#)
- [Section 7.2.4, "View the Report"](#)

7.2.1 Create a JDBC Connection to the Data Source

You need to configure a JDBC connection from Oracle BI Publisher to the Oracle Database. To do this:

1. While logged into BI Publisher as Administrator, click the **Admin** tab.
2. Click the **JDBC Connection** link under **Data Sources**.
3. Click **Add Data Source**.
4. Enter **bise1_sales** for the **Data Source Name**.
5. Enter **jdbc:oracle:thin:@localhost:1521:bise1db** in the **Connection String** field.
6. Enter the **Username** and **Password**, as follows:
 - **Username:** bise1_sales
 - **Password:** welcome1 (or the password specified during installation)
7. Enter **oracle.jdbc.driver.OracleDriver** for the **Database Driver Class**.
8. Click **Test Connection**. If the test is successful, click **Apply**.

7.2.2 Define a New Report

This section contains the following topics:

- [Section 7.2.2.1, "Create the New Report"](#)
- [Section 7.2.2.2, "Create a Data Model for the Report"](#)
- [Section 7.2.2.3, "Create Lists of Values and Parameters"](#)
- [Section 7.2.2.3, "Create Lists of Values and Parameters"](#)

7.2.2.1 Create the New Report

To define a new report:

1. While logged into Oracle BI Publisher as Administrator, click the **Reports** tab.
2. Click **Home**, then click the **My Folders** link.
3. Click **Create a new report** in the **Folders and Report Tasks** menu.
4. Enter **Invoices** as the report name, then click **Create**.
5. Click **Edit** under the newly created report.
6. Click **Report**.
7. Select **bise1_sales** from the **Default Data Source** drop-down list.
8. Deselect the **Auto Run** option.
9. Click **Save**.

7.2.2.2 Create a Data Model for the Report

To create a data model:

1. Click **Data Model**.
2. Click **New**.
3. Enter **Customer Invoices** in the **Name** field.
4. Select **bise1_sales** as the **Data Source**.
5. Click **Query Builder**.
6. Select the following tables and the specified columns for each table:
 - **Customers: Name**
 - **Addresses: Cust_Street_Address, Cust_Postal_Code**
 - **Cities: Name, State_Province**
 - **Countries: Name**
 - **Orders: ID**
 - **Order_Items: Product_ID, Order_Date, Amount, Quantity**
 - **Products: Name**
7. Join the tables by clicking the box to the right of the columns to be joined:
 - **Customers.ID = Addresses.Customer_ID**
 - **Addresses.City_ID = Cities.ID**
 - **Cities.Country_ISO_Code = Countries.ISO_CODE**
 - **Customers.ID = Orders.Customer_ID**
 - **Orders.ID = Order_Items.Order_ID**
 - **Order_Items.Product_ID = Products.Identifier**

These boxes, when marked for joins, turn to light gray. Also note that a fine line joining the tables appears in the Model canvas.

8. Click **Conditions**.
9. Specify **Aliases** for the following columns:
 - **Customers.Name = CUST_NAME**
 - **Countries.Name = COUNTRY**
 - **Cities.Name = CITY**
 - **Products.Name = PRODUCT_NAME**
 - **Orders.ID = ORDER_ID**
10. Accept the default Aliases for the other columns. Click **Save**.
11. Edit the **SQL Query**, and add the following conditions:

```
and ORDERS.PROMOTION_ID =999
and ORDERS.CHANNEL ='Direct Sales'
and ORDER_ITEMS.ORDER_DATE between :start_date and :end_date
```

The last condition references parameters which will be created next.

12. Click **Save**.

7.2.2.3 Create Lists of Values and Parameters

Instead of generating all invoices at once, the invoices will be generated based on a date range. You will create two lists of values and two parameters to capture the start and end dates at report run time. To do this:

1. Create the first list of values. Click **List of Values**.
2. Click **New**.
3. Enter **start_date_lov** in the **Name** field.
4. Select **bise1_sales** as the **Connection**.
5. Enter the following statement in the **SQL Query** box:

```
select DISTINCT TO_CHAR (ORDERS.ORDER_FINISHED, 'DD-MON-YYYY') AS START_DATES
from BISE1_SALES.ORDERS
order by TO_DATE(START_DATES)
```

6. Click **Save**.
7. Create the second list of values. Click **List of Values**.
8. Click **New**.
9. Enter **end_date_lov** in the **Name** field.
10. Select **bise1_sales** as the **Connection**.
11. Enter the following statement in the **SQL Query** box:

```
select DISTINCT TO_CHAR (ORDERS.ORDER_FINISHED, 'DD-MON-YYYY') AS END_DATES
from BISE1_SALES.ORDERS
order by TO_DATE(END_DATES)
```

12. Click **Save**.
13. Create the first parameter. Click **Parameters**.
14. Click **New**.
15. Complete the Parameter Settings page as follows:
 - **Identifier:** start_date
 - **Data Type:** String
 - **Default Value:** 12-DEC-2006
 - **Parameter Type:** Menu
 - **Display Label:** Start Date
 - **List of Values:** start_date_lov
 - **Options:** deselect **Can select all**, then select **Refresh other parameters on change**
16. Click **Save**.
17. Create the second parameter. Click **Parameters**.
18. Click **New**.
19. Complete the Parameter Settings page as follows:
 - **Identifier:** end_date
 - **Data Type:** String

- **Default Value:** 12-DEC-2006
- **Parameter Type:** Menu
- **Display Label:** End Date
- **List of Values:** end_date_lov
- **Options:** deselect **Can select all**, then select **Refresh other parameters on change**

20. Click **Save**.

7.2.2.4 Create a Layout

You will upload an existing RTF template as the starting point for the report. After uploading the template and creating a layout based on the template, you will modify the template using Microsoft Word in the next section.

To create the layout:

1. Click **Layouts**.
2. Click **Browse** under the **Manage Template Files** section.
3. Locate the tutorial\bi_pub\invoices.rtf file, then click **Upload**.
4. Click **New**.
5. Complete the Layout Settings page as follows:
 - **Name:** Invoices
 - **Template:** invoices.rtf
 - **Template Type:** RTF Templates
 - **Output Format:** Limit Output to
Deselect all the format types, except for **PDF**.
6. Click **Save**.

7.2.3 Create an RTF Template for the Report

Before you can create or modify any templates in Microsoft Word, you need to make sure that it has the BI Template Builder plug-in, formally known as Oracle BI Publisher Desktop.

This section contains the following topics:

- [Section 7.2.3.1, "Install Oracle BI Publisher Desktop"](#)
- [Section 7.2.3.2, "Open the Report Layout Template"](#)
- [Section 7.2.3.3, "Insert Report Fields Into the Template"](#)
- [Section 7.2.3.4, "Apply Format Masks to the Fields"](#)

7.2.3.1 Install Oracle BI Publisher Desktop

To install Oracle BI Publisher Desktop:

1. Click the **Reports** tab in BI Publisher. Click the **Home** link, then click the **Shared Folders** link. Click **Template Builder** in the **Folders and Reports Tasks** section in the left menu box.

2. Click **Save File** to save the `BIPublisherDesktop.exe` file to a directory on the Windows client computer.
3. Locate the `BIPublisherDesktop.exe` file using Windows Explorer. Double-click the file to begin the installation. Choose the language, then click **Next**.
4. Click **Next** on the following two screens, then click **Finish** in the InstallShield Wizard Complete screen.

7.2.3.2 Open the Report Layout Template

To open the report layout template:

1. Start the Microsoft Word application from the program menu. You will notice an additional menu item called **Oracle BI Publisher** at the top.

Note: The Oracle BI Publisher menu is displayed in Word only when you have successfully installed the Oracle BI Publisher Desktop.

Also, there is a new Toolbar with the following menu items: Data, Insert, Preview, Tools, and Help. Oracle BI Publisher has similar items, and a few extra.

2. From the **Oracle BI Publisher** menu, select **Log On**.
3. In the Login screen that appears, enter **Administrator** as the **Username** and **Password**, then click **Login**. The first time you connect, you will need to specify the report server to which you want to connect. Enter the report server URL in this format: `http://localhost:9704/xmlpserver`.

Note: If you are not running Microsoft Word on the computer where Oracle BI Standard Edition One is installed, you must replace `localhost` with the actual hostname of the computer on which Oracle BI Standard Edition One is installed.

4. In the Open Template dialog box, click **My Folders**.
5. Click the **Invoices** report. In the **Template Layouts** pane on the bottom, select the **Invoices** template. Click **Open Layout Template**.
6. In the template, notice the words and letters with the gray background. These are called form fields. Form fields are Word objects that allow you to reference other data (for example, a mail merge letter). BI Publisher uses form fields in two ways. The first use is to reference data fields from the report definition (like YEAR and MONTH). The second use is to embed instructions that control how the data fields will be laid out (like G, F, and E). If you want to know what these instructions are, double-click the form field and view the Help text. It is important to treat these form fields carefully and not accidentally delete or move them. Doing so will change the layout of your report. You may want to modify or add your own form fields with XSL commands to do more sophisticated things with your layout.

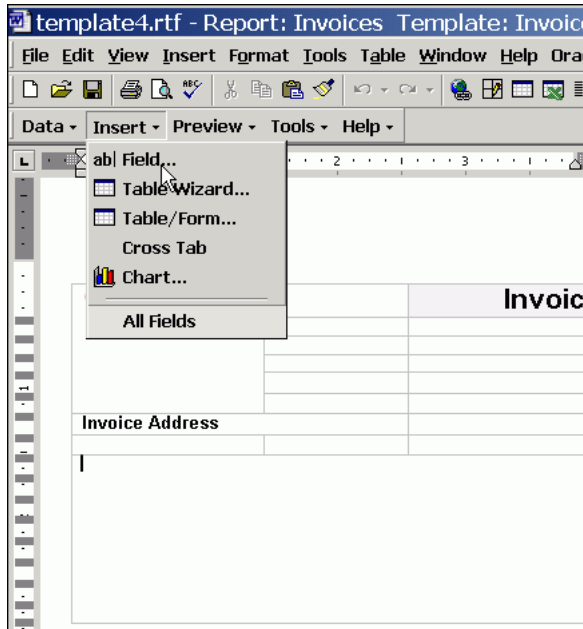
7.2.3.3 Insert Report Fields Into the Template

To insert report fields into the template:

1. Scroll down to the second page of the template.

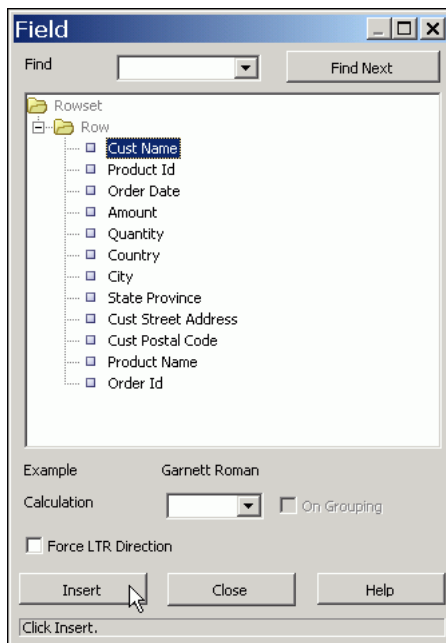
- Position the cursor in the large box beneath **Invoice Address**. From the Oracle BI Publisher toolbar, select **Insert > Field**.

Figure 7-3 Inserting a Field



- Select **Cust Name** from the **Field** dialog box. Click **Insert**.

Figure 7-4 Selecting Cust Name



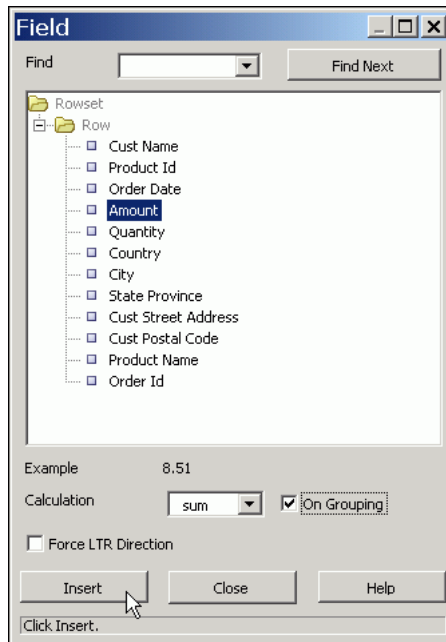
- Continue to populate the invoice address box. For each field, position the cursor on the appropriate location in the template before you select the field and insert it. Keep the **Field** dialog box open.

When you are done, the invoice address box of the template should look like the one shown in [Figure 7-5](#).

Figure 7-5 Invoice Address Box

- Next, you need to populate the **Invoice Information** part of the template. Insert the **ORDER_ID** field next as the **Invoice Number**. Insert the **sum** of the **Amount** field for **Payment Due**. Be sure to select the **On Grouping** option.

Figure 7-6 Selecting the Amount Field



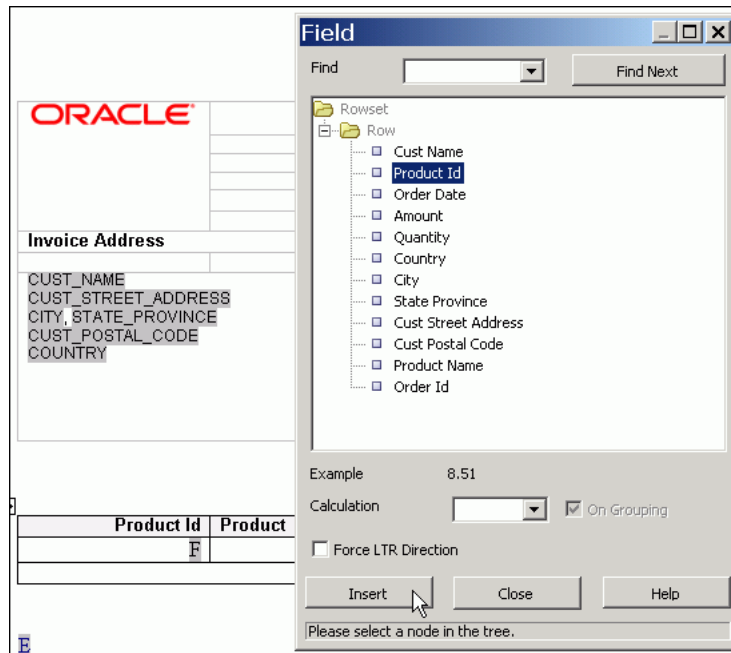
When you are done, the **Invoice Information** section of the template should look like the one shown in [Figure 7-7](#).

Figure 7-7 Invoice Information Section

Invoice Information	
Invoice Number:	ORDER_ID
Invoice Date:	16 March 2007
Payment Terms:	Immediate
Payment Due:	sum AMOUNT

- Next, you need to populate the invoice line items table. Position the cursor under the **Product ID** column in the table, in front of the gray letter F. Select the **Product ID** field, making sure that the **Calculation** field is set to blank. Click **Insert**.

Figure 7–8 *Selecting the Product ID Field*



- Continue to insert fields into the invoice line items table. When you are done, the invoice line items table should look like the one shown in [Figure 7–9](#).

Figure 7–9 *Invoice Line Items Table*

Product Id	Product	Quantity	Amount
PRODUCT_ID F	PRODUCT_NAME	QUANTITY	AMOUNT E
Total Amount Due			sum AMOUNT

Note: The gray letter **E** at the end of the line item **Amount** field should remain. It is an Oracle BI Publisher formatting code. Also, note that the **Total Amount Due** is the **sum** of the **Amount** field, just like **Payment Due**.

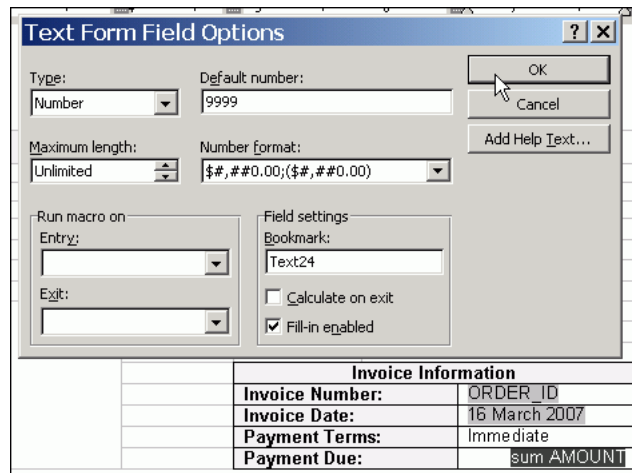
- Close the **Field** dialog box.

7.2.3.4 Apply Format Masks to the Fields

To apply format masks:

- Double-click the **sum Amount** field of **Payment Due**. In the Text Form Field Options dialog box, specify the following, then click **OK**:
 - Type:** Number
 - Default number:** 9999
 - Number format:** \$\$\$,##0.00

Figure 7–10 Text Form Field Options



2. Apply the same format mask as above for the two **Amount** fields in the invoice line item table.
3. Double-click the **Quantity** field. In the Text Form Field Options dialog box, specify the following, then click **OK**:
 - **Type:** Number
 - **Default number:** 1
 - **Number format:** 0
4. The **Quantity** and **Amount** columns of the invoice line item table should now look like the ones shown in [Figure 7–11](#).

Figure 7–11 Quantity and Amount Columns

	Quantity	Amount
	1	\$9,999.00E
Total Amount Due		\$9,999.00

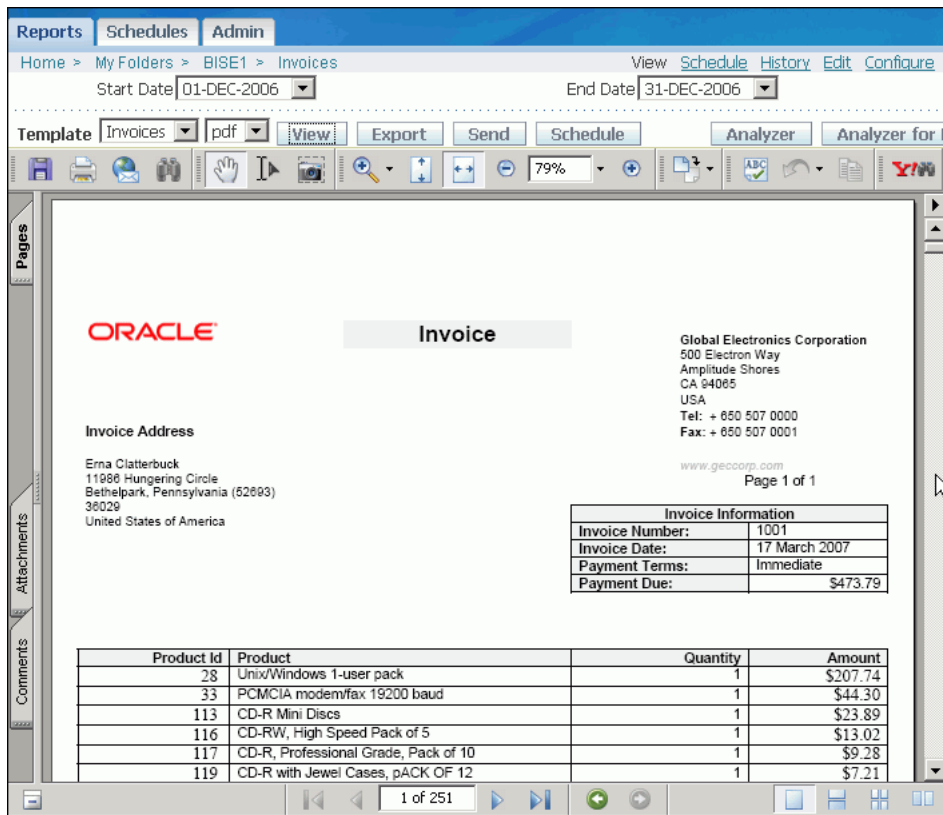
5. Upload the modified template to the Oracle BI Publisher server. In Microsoft Word, select **Oracle BI Publisher > Update Layout Template**.
6. Click **OK** in the confirmation dialog box.

7.2.4 View the Report

To view the report:

1. Log in to Oracle BI Publisher as **Administrator**.
2. Click the **Invoices** report link.
3. Set the **Start Date** to **01-DEC-2006**, and the **End Date** to **31-DEC-2006**. Click **View**.
4. Scroll through the generated PDF file:

Figure 7–12 Generated PDF File



Note: Though the invoices are all in one single PDF file, the page count (**Page x of y**) is automatically reset for each invoice. This is achieved through the form field format codes mentioned earlier.

7.3 Automatically Delivering BI Publisher Reports

This section contains the following topics:

- [Section 7.3.1, "Configure Oracle BI Publisher Scheduler Report Delivery Options"](#)
- [Section 7.3.2, "Schedule FTP Delivery of a Report"](#)

7.3.1 Configure Oracle BI Publisher Scheduler Report Delivery Options

To set up the Oracle BI Publisher Scheduler report delivery options, perform the following steps:

1. Log in to Oracle BI Publisher as **Administrator**, specifying **Administrator** as the password.
2. Click the **Admin** tab.
3. Click the **Delivery Configuration** link under the Delivery section.
4. Configure an Email report delivery server. Click the **Email** tab.
5. Click **Add Server**.
6. Fill in the fields as follows, then click **Apply**:

- **Server Name:** The name for this Email report delivery server (can be the same as the Host value)
 - **Host:** The host name of your SMTP server
 - **Port:** The port number for your SMTP server (typically 25)
7. Next, configure an FTP report delivery server. Click the **FTP** tab.
 8. Click **Add Server**.
 9. Fill in the fields as follows, then click **Apply**:
 - **Server Name:** The name for this FTP report delivery server (can be the same as the Host value)
 - **Host:** The host name of your FTP server
 - **Port:** The port number for your FTP server (typically 21)

Note: The Host you specify must have an FTP service running. Typically, computers running Windows do not run FTP services, but computers running Linux or UNIX do.

 10. Click **Reports** to return to the home page.

7.3.2 Schedule FTP Delivery of a Report

To schedule FTP delivery:

1. Schedule the **Invoices** report for automatic delivery. Click **Invoices**.
2. Click the **Schedule** button.
3. Keep all the default field values.
4. Scroll down to the Delivery section. Select the **FTP** destination.
5. Notice that additional fields appear after the FTP destination is selected. Fill in the fields as follows, then click **Submit**:
 - **FTP Server:** The name of the FTP report delivery server
 - **Username:** Operating system user account name
 - **Password:** Operating system user account password
 - **Remote Filename:** The full path and file name for the report
6. You see a message that the report delivery job has been scheduled.
7. Click the **Schedules** tab to view the status of the report delivery job.
8. Click the **Invoices** job link.
9. Notice the delivery status is **Success**. Click the **Reports** tab to return to the home page.
10. You can also verify the existence of the report file by logging into the FTP server and viewing the directory where the report has been stored.

7.4 Publishing a BI Publisher Report in BI Dashboards

In this topic, you will create a new report with a template in BI Publisher from a BI Answers request in the **GEC_DW** subject area. You will publish this report in BI Interactive Dashboards.

This section contains the following topics:

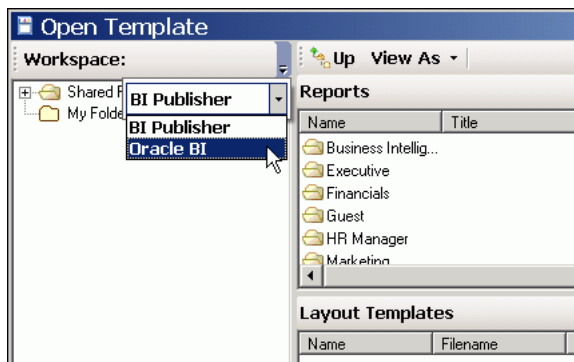
- [Section 7.4.1, "Create an Oracle BI Publisher Report from an Oracle BI Answers Request"](#)
- [Section 7.4.2, "Publish the Oracle BI Publisher Report on Oracle BI Interactive Dashboards"](#)

7.4.1 Create an Oracle BI Publisher Report from an Oracle BI Answers Request

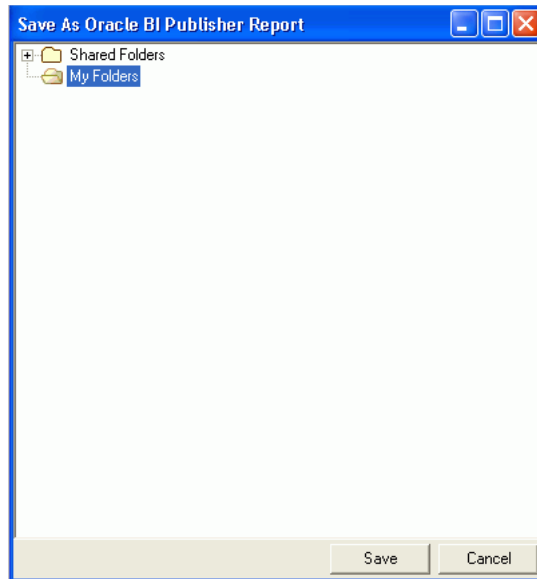
To create an Oracle BI Publisher report from an Oracle BI Answers request:

1. Open the `regional_sales.rtf` file in Microsoft Word. This is a pre-created template file in Word, which has a header and footer, and an image inserted.
2. Log in to Oracle BI Publisher from the Word **Oracle BI Publisher** menu. This opens the Log In screen for Oracle BI Publisher. Enter **Administrator** as the **Username** and the **Password**, then click **Log In**.
3. After you log in, the Open Template Window opens. Select **Oracle BI** from the **Workspace** drop-down list.

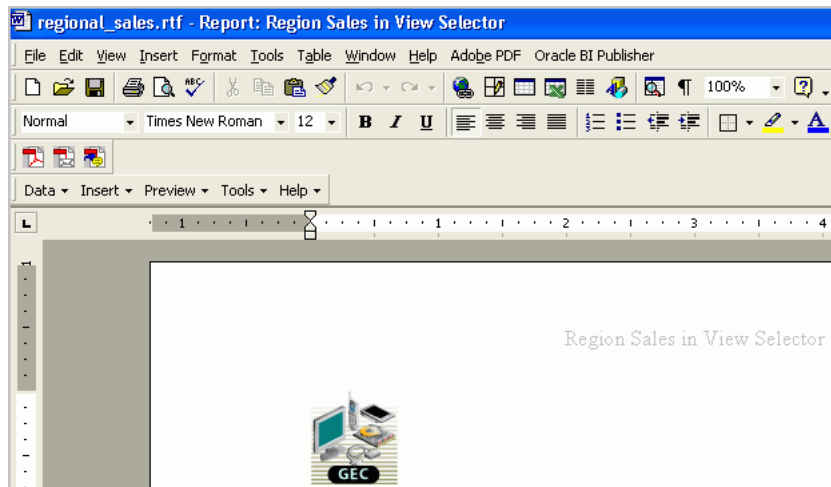
Figure 7–13 Open Template Window



4. Go to the **users > Administrator > GEC_DW > Sales Trends** folder. Double-click the **Region Sales in View Selector** report. This report has a dashboard page prompt associated with it.
5. The Save As Oracle BI Publisher Report Window opens. Select the **My Folder** folder to save the same report in Oracle BI Publisher.
6. Click **Save**. You will be able to see this report in Oracle BI Publisher now.

Figure 7–14 Save As Oracle BI Publisher Report Window

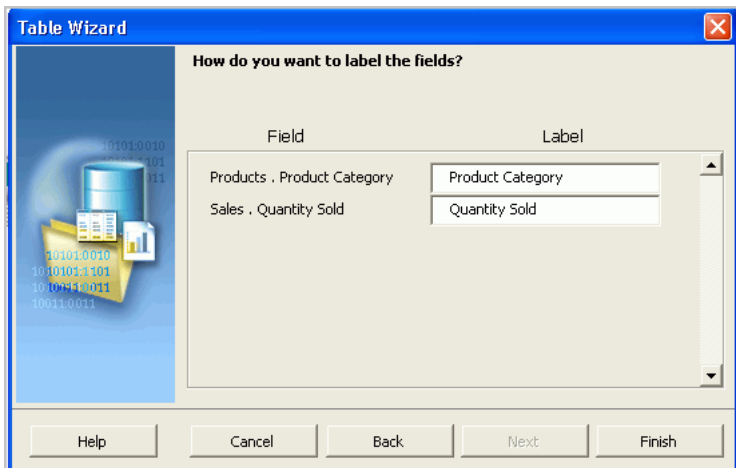
Notice the name of the document has changed in Word and includes the name of the Oracle BI Answers request.

Figure 7–15 Word Document with Answers Request Name

7. In the Word document that loads data from this Answers request, place the cursor a few lines below the image. From the BI Toolbar, select **Insert > Table Wizard** to define a format for the table data in the report. This displays the **Table Wizard**. Do the following:
 - a. Select **Table** and click **Next**.
 - b. Ensure that **ROWSET/ROW** is selected as the **Grouping Field**, and click **Next**.
 - c. Select all the fields and click **Next**.
 - d. Select **Region** from the **Group By** drop-down list, accept the defaults for other options, and click **Next**.
 - e. Select the sort orders for the measures **Quantity Sold**. Accept the defaults in this step and click **Next**.

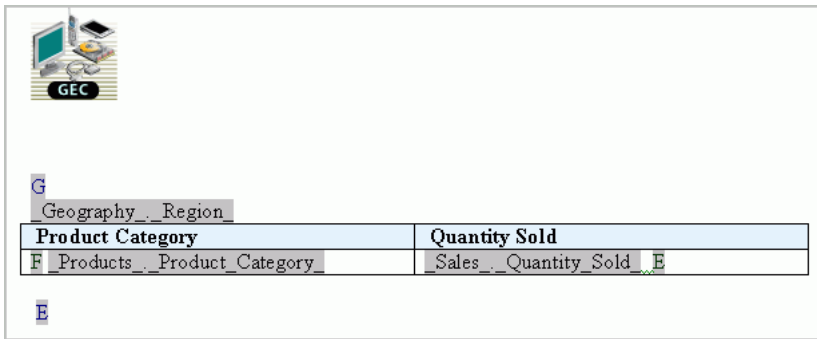
- f. Edit the labels of the fields to **Product Category** and **Quantity Sold**, and click **Finish** to complete the creation of the table template.

Figure 7-16 Table Wizard



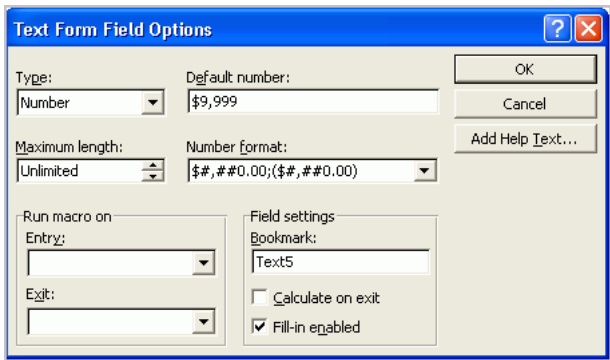
The template you created looks like the one shown in [Figure 7-17](#).

Figure 7-17 Template



- 8. Double-click `_Sales_.Quantity_Sold_` and format the field as shown in [Figure 7-18](#). Click **OK** when you are done.

Figure 7-18 Text Form Field Options



- 9. Use Word formatting features and format the other fields as shown. Create a title for the report called **Regional Sales Report**. Change the color to red and the font

size to Header 1. Change the column headers color to blue and increase the font size to 14. Change the **Region** field font size to Header 2 and the color to green. Right-align the **Amount Sold** field and column header. The final format should look like [Figure 7-19](#).

Figure 7-19 Final Format of Regional Sales Report

Region Sales in View Selector



Regional Sales Report

G

Geography . Region


Product Category	Quantity Sold
F Products . Product_Category_	\$9,999.00 E

E

- From the BI Toolbar menu, select **Preview > RTF** to preview the template. Scroll down to see the footer.

Figure 7-20 Template in Preview Mode

Region Sales in View Selector



Regional Sales Report

Americas

Product Category	Quantity Sold
Peripherals and Accessories	\$167,793.00
Software/Other	\$246,423.00
Photo	\$56,148.00
Electronics	\$68,388.00
Hardware	\$8,311.00

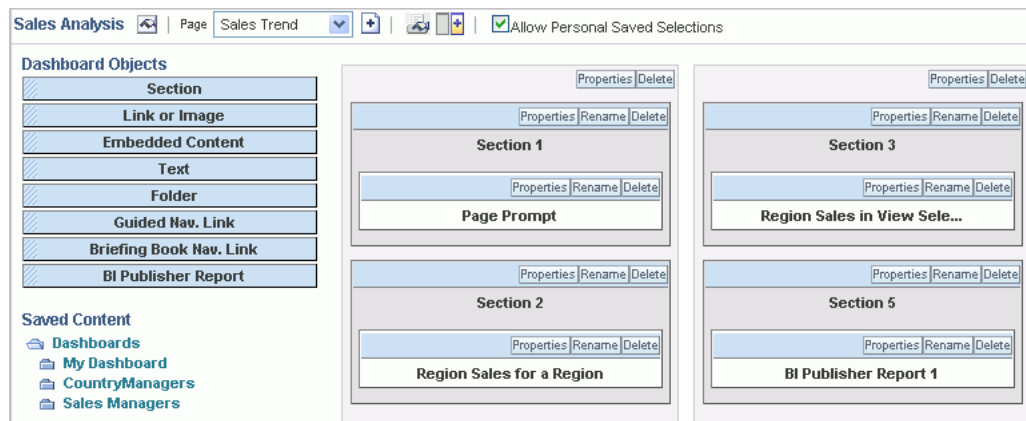
- Switch back to **Normal** view. Publish the template by selecting **Publish Template As** from the Oracle BI Publisher menu. Enter **Regional_Template** as the name of the template.
- Save the template as an RTF file for future use.

7.4.2 Publish the Oracle BI Publisher Report on Oracle BI Interactive Dashboards

In this topic, you will publish the Oracle BI Publisher report that you created from the Oracle BI Answers request onto an Oracle BI Dashboard. To do this:

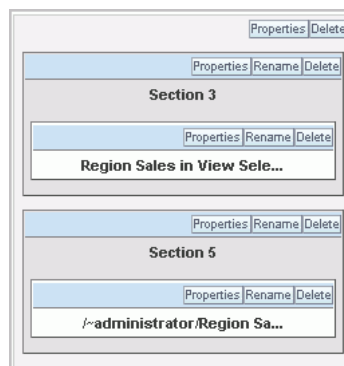
1. Return to Oracle BI Interactive Dashboards. Select the **Sales Analysis** dashboard and the **Sales Trend** page. This dashboard page has a prompt for the reports on the page. Alternately, if you have created the **Sales Analysis1** dashboard, you can select the **Regional Analysis** page. The figures in this procedure are based on the Sales Trend page, in the Sales Analysis dashboard, but the steps are the same for any dashboard.
2. In the Sales Trend page, from the **Page Options** drop-down list (found on the right top corner of the page), select **Edit Dashboard**. The Dashboard Editor screen appears.
3. Drag and drop the **BI Publisher Report** from the **Dashboard Objects** list to the section as shown in [Figure 7-21](#).

Figure 7-21 BI Publisher Report on Sales Analysis Dashboard



4. You can see that the **BI Publisher Report** object is added to the dashboard. Click the **Properties** link on this object.

Figure 7-22 Properties Link



5. The BI Publisher Report Properties screen appears. Click **Browse** to browse and specify the path for BI Publisher Report. Select the **Region Sales in View Selector** report from **My Folders** in BI Publisher, and click **OK**. Click **OK** again to close the BI Publisher Report Properties screen.

Figure 7–23 BI Publisher Report Properties

BI Publisher Report Properties
Specify the path to a report or a document from BI Publisher Enterprise.

Path

Display Mode

Embedded Content
Width
Height

Link

View Latest Version

6. This will take you back to the Dashboard Editor page. Click **Save** to save the changes you made to the dashboard.
7. This dashboard page has a prompt on **Region**. Select **Americas** from the **Region** drop-down list. Click the **Go** button for the prompt. All reports on this page will display sales for **Americas** only. The dashboard is shown; scroll down and observe that the BI Publisher Report is displayed on the dashboard in the section that you have added.

Figure 7–24 Regional Sales Report

Template:

Region Sales in View Selector

 **Regional Sales Report**

Americas

Product Category	Quantity Sold
Peripherals and Accessories	\$167,793.00
Software/Other	\$246,423.00
Photo	\$56,148.00
Electronics	\$68,388.00
Hardware	\$8,311.00

Page 1 3/10/2007

7.5 Creating an Oracle BI Publisher Report Using BI Server Metadata

In this topic, you will create an Oracle BI Publisher report from BI Server Metadata. You will also associate a pre-created template with the report to view the data.

This section contains the following topics:

- [Section 7.5.1, "Create an Oracle BI Publisher Report from BI Server Metadata"](#)

- [Section 7.5.2, "Publish the Template to View the Report Data in Oracle BI Publisher"](#)

7.5.1 Create an Oracle BI Publisher Report from BI Server Metadata

To create an Oracle BI Publisher report from BI Server metadata:

1. Log in to Oracle BI Publisher as **Administrator** (password **Administrator**).

Note: You can log in to Oracle BI Publisher by logging in to Presentation Services and selecting the **More Products > BI Publisher** option, or you can log in to Oracle BI Publisher directly by selecting **Start > Programs > Oracle Business Intelligence > BI Publisher**.

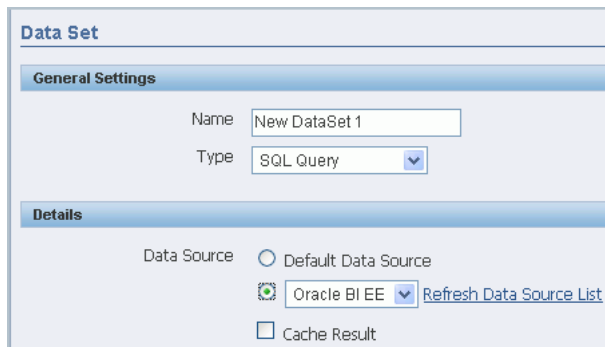
2. Navigate to **My Folder** and click **Create a New Report**. Enter **From BI Server** as the name of the report and click **Create**. The newly created report is displayed on the page.

Figure 7–25 From BI Server Report



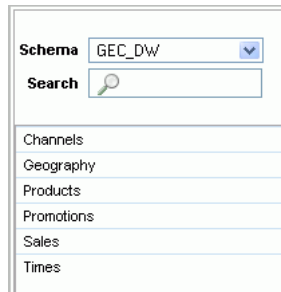
3. Click the **Edit** link displayed below the report name to edit the properties. The report is displayed in the edit mode. Click **Data Model** and click **New** to define the data source for this report.
4. In the Data Set screen, select **SQL Query** from the **Type** drop-down list.
5. Select **Oracle BI EE** as the data source.

Figure 7–26 Data Set Screen



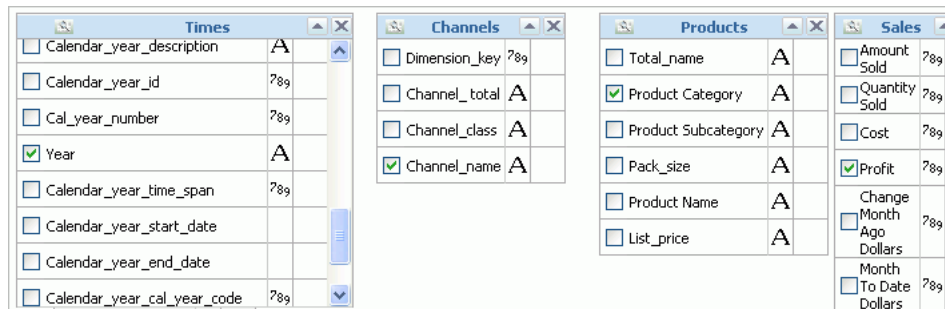
6. Click **Query Builder** to create a SQL Query. The Query Builder screen opens, displaying the **GEC_DW** subject area repository objects on the left.

Figure 7–27 Query Builder Screen



7. Now, drag and drop the **Time**, **Channels**, **Products**, and **Sales** from the **GEC_DW** subject area one by one, to the **Model** canvas on the right. Select the following columns to be displayed in the query by selecting the checkboxes next to the column names:
 - **Year** from **Times**
 - **Channel_name** from **Channels**
 - **Product Category** from **Products**
 - **Profit** from **Sales**

Figure 7–28 Selecting the Columns to Be Displayed In the Query



8. Click **Save**. This will take you back to the BI Publisher Data Set screen. Observe that the query is displayed in the **SQL Query** field. Click the **Save** icon to save the report.

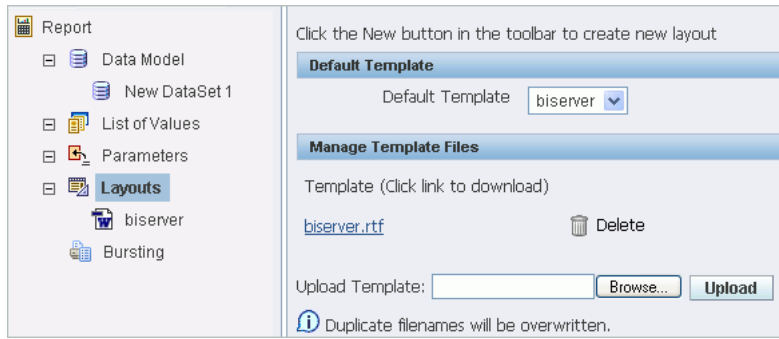
7.5.2 Publish the Template to View the Report Data in Oracle BI Publisher

In this section, you will associate a pre-created template to the report you just created and then view the report in Oracle BI Publisher with this template. To do this:

1. In Microsoft Word, open the `biserver.rtf` file.
2. Through Word, log in to **Oracle BI Publisher** as **Administrator** (password **Administrator**). The Open Template screen appears.
3. Select the **From BI Server** report. Click **Open Report** at the bottom. This will bring you back to Word.
4. From the **Oracle BI Publisher** menu, select the **Publish Template As** option. This opens the Upload as New dialog box. Enter **biserver** as the name of the template, then click **OK**.

5. After the template is uploaded, it displays the message **biserver was added to the report From BI Server**. Click **OK** again
6. Connect to Oracle BI Publisher as **Administrator**. Open the **From BI Server** report from **My Folders**.
7. Click **Layouts**. Notice the `biserver.rtf` template is associated with the report.

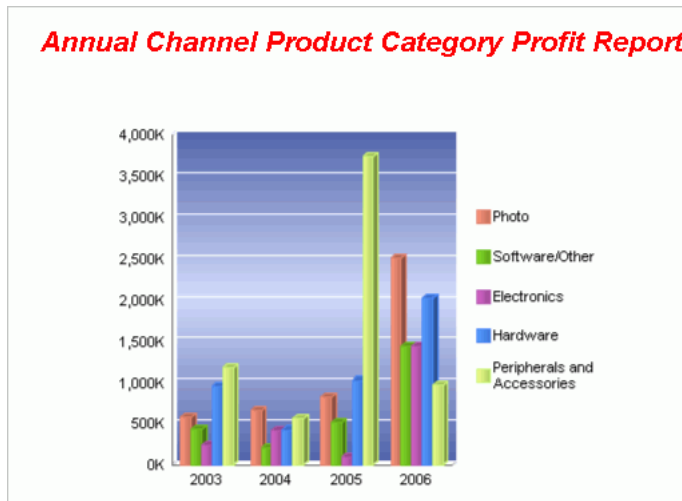
Figure 7–29 *biserver.rtf* Template Under Manage Template Files



8. Click **View** to display the report.

Note: You can also view the data in other formats, such as PDF, RTF, MS Excel, and so on.

Figure 7–30 *Report in HTML Format*



Scroll down and see the tabular report, as shown in [Figure 7–31](#).

Figure 7-31 Tabular Report

2003	
Direct Sales	
Product Category	Profit
Electronics	\$86,106.92
Hardware	\$248,540.41
Peripherals and Accessories	\$430,099.79
Photo	\$433,339.95
Software/Other	\$171,951.37

Reset the Tutorial Exercise Environment

If you or your colleagues would like to work through the tutorial exercises again, you can reset your tutorial exercise environment by following the instructions appropriate to the component environment you would like to reset.

This chapter contains the following topics:

- [Section 8.1, "Reset the Oracle Warehouse Builder Environment"](#)
- [Section 8.2, "Reset the BI Server Metadata Repository"](#)
- [Section 8.3, "Reset the BI Presentation Services Web Catalog"](#)

8.1 Reset the Oracle Warehouse Builder Environment

If you want to work through the exercises in [Chapter 3, "Set Up the Data Mart"](#) again, perform these steps first:

1. Open a command prompt window.
2. Navigate to the directory where you installed Oracle BI Standard Edition One.
3. Navigate to the `tutorial\owb` subdirectory.
4. Run the `reset_bise1_tutorialwh.bat` script. When prompted, enter the password for the `BISE1_TUTORIALWH` database account.
5. Go to the Windows Start menu, then select **Programs > Oracle - BISE1_ WarehouseBuilder > Warehouse Builder > Design Center**.
6. Log into Oracle Warehouse Builder Design Center using the `owbrepos_owner` account.
7. In the Design Center, right-click the `GEC_DW_TUTORIAL` project and select **Delete**.
8. Select **Design > Save All**.
9. Select **Design > Import > Warehouse Builder Metadata**.
10. Click **Browse** next to the **File Name** field. Go to the Oracle BI Standard Edition One installation directory, and then go to the `tutorial\owb\mdl` directory.
11. Select `gec_dw_tutorial_start.mdl`. Click **Open**.
12. In the Metadata Import dialog box, click **Advanced**.
13. Select the options **Import user-defined definition** and **Import security information**. Click **OK**.
14. Click **Import**.

15. When the import completes successfully, click **Close** to close the Metadata Import Progress dialog box.
16. Exit the Design Center.

8.2 Reset the BI Server Metadata Repository

If you want to work through all of the exercises in [Chapter 4, "Build the BI Repository"](#) again, perform these steps first:

1. Shut down the BI Server service, and the BI Presentation Services service.
2. Copy `tutorial\bi_ad\bise1.rpd` to `bi\server\repository`, replacing the existing `bise1.rpd`.
3. Copy `tutorial\bi_ad\bise1.zip` to `bidata\web\catalog`.
4. Unzip `bise1.zip`, replacing the contents of `bidata\web\catalog\bise1`.
5. Start the BI Server service and the BI Presentation Services service.

If you want to work through just the Advanced exercises in Chapter 4 ([Section 4.8](#) through [Section 4.11](#)) again, or you want to work through [Chapter 5, "Analyze the Data"](#) again, perform these steps first:

1. Shut down the BI Server service.
2. Copy `tutorial\bi_ad\bise1_soln_noadv.rpd` to `bi\server\repository`.
3. Rename `bi\server\repository\bise1_soln_noadv.rpd` to `bi\server\repository\bise1.rpd`, replacing the existing `bise1.rpd` file.
4. Copy `tutorial\bi_ad\bise1_soln_noadv.zip` to `bidata\web\catalog`.
5. Unzip `bise1_soln_noadv.zip`, replacing the contents of `bidata\web\catalog\bise1`.
6. Start the BI Server service and the BI Presentation Services service.

8.3 Reset the BI Presentation Services Web Catalog

To reset the BI Presentation Services Web catalog:

1. Shutdown the BI Presentation Services service.
2. Copy `tutorial\bi_ad\bise1.zip` to `bidata\web\catalog`.
3. Unzip `bise1.zip`, replacing contents of `bidata\web\catalog\bise1`.
4. Start the BI Presentation Services service.

Next Steps

Congratulations on reaching the end of the Oracle Business Intelligence Standard Edition One Tutorial!

One of the key benefits of Oracle BI Standard Edition One is that you can begin by implementing only those components that are needed to satisfy your immediate requirements, and then evolve the solution architecture as your business requirements grow. If your business success carries you beyond the capabilities of Oracle BI Standard Edition One, the built-in upgrade path enables you to preserve the investment you have made today, and does not hinder your growth tomorrow.

Now that you have worked through the tutorial and have learned to use the basic features of the Oracle BI Standard Edition One components, where do you go from here?

This chapter lists some additional tasks and features you should become familiar with, if you serve as the administrator or provide technical support, and points you to the resources that will help you.

This chapter contains the following topics:

- [Section 9.1, "Enhancing the Security of Your Oracle BI Environment"](#)
- [Section 9.2, "Administering the Database"](#)
- [Section 9.3, "Using Oracle Warehouse Builder"](#)
- [Section 9.4, "Administering Oracle Warehouse Builder"](#)
- [Section 9.5, "Administering Oracle BI Server"](#)
- [Section 9.6, "Using Oracle BI Answers and Dashboards"](#)
- [Section 9.7, "Administering Oracle BI Answers and Dashboards"](#)
- [Section 9.8, "Using Oracle BI Publisher"](#)
- [Section 9.9, "Administering Oracle BI Publisher"](#)

9.1 Enhancing the Security of Your Oracle BI Environment

You may want to enhance security of your Oracle BI environment by creating additional accounts to limit access to administrative privileges, as well as protect sensitive data in your data sources.

- For information about configuring Oracle BI Server security, refer to Chapter 15, "Security in Oracle BI" in *Oracle Business Intelligence Server Administration Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b31770.pdf).

- To configure access to Oracle BI Answers requests, and dashboards, refer to Chapter 8, "Managing Oracle BI Presentation Services Security" in *Oracle Business Intelligence Presentation Services Administration Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b31766.pdf).
- To configure access to Oracle BI Publisher reports, refer to the "Defining a Security Model" section (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T475591.htm) in *Oracle Business Intelligence Publisher User's Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/toc.htm).

9.2 Administering the Database

The *Oracle Database 2 Day DBA* book (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/toc.htm) provides a quick start to understanding and administering the Oracle Database. In particular, the following chapters in the book are useful to review:

- Getting Started with Oracle Enterprise Manager (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/em_manage.htm#sthref126)
- Managing the Oracle Instance (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/instance.htm#sthref212)
- Managing Database Storage Structures (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/storage.htm#sthref276)
- Administering Users and Security (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/users_secure.htm#sthref380)
- Managing Schema Objects (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/schema.htm#sthref452)
- Performing Backup and Recovery (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/backrest.htm#i1004902)
- Monitoring and Tuning the Database (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/montune.htm#sthref714)
- Managing Oracle Software (http://download.oracle.com/docs/cd/B19306_01/server.102/b14196/software.htm#sthref800)

9.3 Using Oracle Warehouse Builder

Refer to the following chapters in *Oracle Warehouse Builder User's Guide* (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/toc.htm) to learn more about how to build a data mart with Oracle Warehouse Builder:

- Overview (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_overview.htm#i1086382)
- Creating an Oracle Data Warehouse (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_basics.htm#CHDEJAI)
- Setting Up Warehouse Builder (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_preferences.htm#BABIGHFA)
- Designing Target Schemas (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_data_modeling.htm#CHDICECI)
- Identifying Data Sources and Importing Metadata (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_integrate_import.htm#CHDHEGEJ)
- Creating Mappings (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_mappings.htm#CIHGDHFE)
- Deploying to Target Schemas and Executing ETL Logic (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/concept_deploy.htm#BABBDACA)
- Reference for Using Oracle Data Objects (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_orcl_data_objx.htm#i85798)
- Defining Dimensional Objects (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_dim_objects.htm#BABEJGDC)
- Defining Flat Files and External Tables (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_def_flatfiles.htm#BABICCDG)
- Validating Data Objects (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_validation.htm#CHDDAAC)
- Source and Target Operators (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_src_tgt_ops.htm#i1123494)
- Scheduling ETL Objects (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_schedule.htm#CHDGEGJF)
- Deploying Target Systems (http://download.oracle.com/docs/cd/B31080_01/doc/owb.102/b28223/ref_deployment.htm#i1064950)

9.4 Administering Oracle Warehouse Builder

To learn how to administer Oracle Warehouse Builder (OWB), refer to *Oracle Warehouse Builder Installation and Administration Guide*. The chapter entitled "Implementing Security in Warehouse Builder" (http://download.oracle.com/docs/cd/B31080_01/doc/install.102/b28224/security.htm#CDDDIEFD) is important if you want to set up additional OWB users. You can also attend the Oracle University course

(<http://education.oracle.com>) called *Oracle Warehouse Builder 10g: Administration*.

To explore additional capabilities of Oracle Warehouse Builder, see the Oracle Warehouse Builder - Extending Your Knowledge OBE series (http://www.oracle.com/technology/obe/10gr2_owb/10gr2_owb_extend/index.html).

9.5 Administering Oracle BI Server

Review these chapters from *Oracle Business Intelligence Server Administration Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b31770.pdf) to better understand how you can administer and control your Oracle BI Server, such as creating subject areas, users, and setting up query caching:

- Chapter 2: Oracle BI Administration Tool Basics
- Chapter 3: Planning and Creating an Oracle BI Repository
- Chapter 4: Creating and Administering the Physical Layer in an Oracle BI Repository
- Chapter 5: Creating and Administering the Business Model
- Chapter 6: Creating and Maintaining the Presentation Layer
- Chapter 7: Completing Setup and Managing Oracle BI Repository Files
- Chapter 8: Oracle BI Administration Tool Utilities and Expression Builder
- Chapter 10: Administering the Oracle BI Server Query Environment
- Chapter 11: Query Caching in the Oracle BI Server
- Chapter 15: Security in Oracle BI

9.6 Using Oracle BI Answers and Dashboards

Refer to the following chapters in *Oracle Business Intelligence Answers, Delivers, and Interactive Dashboards User Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b31767.pdf) to learn more about how to analyze, explore, and present your data:

- Chapter 2: Getting Started with Oracle Business Intelligence
- Chapter 3: Basics of Working with Requests in Oracle BI Answers
- Chapter 4: Filtering Requests in Oracle BI Answers
- Chapter 5: Formatting Results in Oracle BI Answers
- Chapter 6: Working with Oracle BI Views in Oracle BI Answers
- Chapter 8: Using Oracle BI Interactive Dashboards
- Chapter 9: Managing Content in the Oracle BI Presentation Catalog

9.7 Administering Oracle BI Answers and Dashboards

Review these chapters from *Oracle Business Intelligence Presentation Services Administration Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b31766.pdf) to better understand how you can administer and control your Oracle BI Answers and Dashboard environment:

- Chapter 2: Administering Oracle BI Presentation Services
- Chapter 3: Administering Oracle BI Answers
- Chapter 5: Administering Oracle BI Dashboards
- Chapter 6: Administering the Oracle BI Presentation Catalog
- Chapter 7: Managing Presentation Catalogs Using Oracle BI Catalog Manager
- Chapter 8: Managing Oracle BI Presentation Services Security

9.8 Using Oracle BI Publisher

Refer to the following chapters in *Oracle Business Intelligence Publisher User's Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/toc.htm) to learn more about how to create highly formatted reports:

- Getting Started (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T430846.htm)
- Viewing and Scheduling Reports (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T421742.htm)
- Creating a New Report (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T423025.htm)
- Creating an RTF Template (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T481157.htm)
- Translating Reports (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T432386.htm)

9.9 Administering Oracle BI Publisher

Refer to the following chapters in *Oracle Business Intelligence Publisher User's Guide* (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/toc.htm) to learn more about how to set up security, users, report destinations (for example, printers), and so on for Oracle BI Publisher:

- Defining a Security Model (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T475591.htm)
- Using the Admin Functions (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T421745.htm)
- Setting Up Print Servers (http://download.oracle.com/docs/cd/B40078_02/doc/bi.1013/b40017/T421739T479665.htm)

Data Mart Concepts

This chapter reviews some basic concepts relating to data marts and establishes some working definitions for use in the rest of this book. Although there is a lot of agreement among users and vendors on the definitions and terminology, they have not yet reached complete consensus. If you talk to a dozen people, you are likely to hear about half a dozen similar but slightly differing answers for even something as basic as "What is a data mart?" This chapter takes a quick look at some definitions and explains what a data mart is (and is not).

This chapter contains the following topics:

- [Section A.1, "What Is a Data Mart?"](#)
- [Section A.2, "How Is It Different from a Data Warehouse?"](#)
- [Section A.3, "Dependent and Independent Data Marts"](#)
- [Section A.4, "What Are the Steps in Implementing a Data Mart?"](#)

A.1 What Is a Data Mart?

A data mart is a simple form of a data warehouse that is focused on a single subject (or functional area), such as Sales, Finance, or Marketing. Data marts are often built and controlled by a single department within an organization. Given their single-subject focus, data marts usually draw data from only a few sources. The sources could be internal operational systems, a central data warehouse, or external data.

A.2 How Is It Different from a Data Warehouse?

A data warehouse, unlike a data mart, deals with multiple subject areas and is typically implemented and controlled by a central organizational unit such as the corporate Information Technology (IT) group. Often, it is called a central or enterprise data warehouse. Typically, a data warehouse assembles data from multiple source systems.

Nothing in these basic definitions limits the size of a data mart or the complexity of the decision-support data that it contains. Nevertheless, data marts are typically smaller and less complex than data warehouses; hence, they are typically easier to build and maintain. [Table A-1](#) summarizes the basic differences between a data warehouse and a data mart.

Table A-1 *Differences Between a Data Warehouse and a Data Mart*

Category	Data Warehouse	Data Mart
Scope	Corporate	Line of Business (LOB)

Table A-1 (Cont.) Differences Between a Data Warehouse and a Data Mart

Category	Data Warehouse	Data Mart
Subject	Multiple	Single subject
Data Sources	Many	Few
Size (typical)	100 GB-TB+	< 100 GB
Implementation Time	Months to years	Months

A.3 Dependent and Independent Data Marts

There are two basic types of data marts: dependent and independent. The categorization is based primarily on the data source that feeds the data mart. Dependent data marts draw data from a central data warehouse that has already been created. Independent data marts, in contrast, are standalone systems built by drawing data directly from operational or external sources of data, or both.

The main difference between independent and dependent data marts is how you populate the data mart; that is, how you get data out of the sources and into the data mart. This step, called the Extraction-Transformation-and Loading (ETL) process, involves moving data from operational systems, filtering it, and loading it into the data mart.

With dependent data marts, this process is somewhat simplified because formatted and summarized (clean) data has already been loaded into the central data warehouse. The ETL process for dependent data marts is mostly a process of identifying the right subset of data relevant to the chosen data mart subject and moving a copy of it, perhaps in a summarized form.

With independent data marts, however, you must deal with all aspects of the ETL process, much as you do with a central data warehouse. The number of sources is likely to be fewer and the amount of data associated with the data mart is less than the warehouse, given your focus on a single subject.

The motivations behind the creation of these two types of data marts are also typically different. Dependent data marts are usually built to achieve improved performance and availability, better control, and lower telecommunication costs resulting from local access of data relevant to a specific department. The creation of independent data marts is often driven by the need to have a solution within a shorter time.

A.4 What Are the Steps in Implementing a Data Mart?

Simply stated, the major steps in implementing a data mart are to design the schema, construct the physical storage, populate the data mart with data from source systems, access it to make informed decisions, and manage it over time.

This section contains the following topics:

- [Section A.4.1, "Designing"](#)
- [Section A.4.2, "Constructing"](#)
- [Section A.4.3, "Populating"](#)
- [Section A.4.4, "Accessing"](#)
- [Section A.4.5, "Managing"](#)

A.4.1 Designing

The design step is first in the data mart process. This step covers all of the tasks from initiating the request for a data mart through gathering information about the requirements, and developing the logical and physical design of the data mart. The design step involves the following tasks:

- Gathering the business and technical requirements
- Identifying data sources
- Selecting the appropriate subset of data
- Designing the logical and physical structure of the data mart

A.4.2 Constructing

This step includes creating the physical database and the logical structures associated with the data mart to provide fast and efficient access to the data. This step involves the following tasks:

- Creating the physical database and storage structures, such as tablespaces, associated with the data mart
- Creating the schema objects, such as tables and indexes defined in the design step
- Determining how best to set up the tables and the access structures

A.4.3 Populating

The populating step covers all of the tasks related to getting the data from the source, cleaning it up, modifying it to the right format and level of detail, and moving it into the data mart. More formally stated, the populating step involves the following tasks:

- Mapping data sources to target data structures
- Extracting data
- Cleansing and transforming the data
- Loading data into the data mart
- Creating and storing metadata

A.4.4 Accessing

The accessing step involves putting the data to use: querying the data, analyzing it, creating reports, charts, and graphs, and publishing these. Typically, the end user uses a graphical front-end tool to submit queries to the database and display the results of the queries. The accessing step requires that you perform the following tasks:

- Set up an intermediate layer for the front-end tool to use. This layer, the metalayer, translates database structures and object names into business terms, so that the end user can interact with the data mart using terms that relate to the business function.
- Maintain and manage these business interfaces.
- Set up and manage database structures, like summarized tables, that help queries submitted through the front-end tool execute quickly and efficiently.

A.4.5 Managing

This step involves managing the data mart over its lifetime. In this step, you perform management tasks such as the following:

- Providing secure access to the data
- Managing the growth of the data
- Optimizing the system for better performance
- Ensuring the availability of data even with system failures

Design the Data Mart

This chapter looks at the issues involved in the design of a data mart. Think of this chapter as a collection of tips on how to run your data mart implementation project. Just as important as learning what you should do is learning what to watch out for—the things that can trip you up on a project like this (and these may not always be technical issues).

The driving business factor for the data mart is the need for information, and the best way to start the design process is by identifying the business needs. You should involve those who have an investment in the data mart, such as the business sponsor and the end user, as early as possible in the design process. Together, you should agree on the information requirements that the data mart must fulfill, the data sources, the technical requirements (such as how often the data needs to be refreshed from the source), and the success criteria for the project.

The steps in designing a data mart are:

1. Conducting a study to define the scope of the project
2. Defining the business and technical requirements for the data mart
3. Developing the logical and physical design of the data mart

This chapter contains the following topics:

- [Section B.1, "Defining the Scope of the Data Mart Project"](#)
- [Section B.2, "Defining the Requirements for the Data Mart"](#)
- [Section B.3, "Data Mart Design"](#)

B.1 Defining the Scope of the Data Mart Project

Before you begin to implement the data mart, you need to develop a plan for its delivery. Critical inputs to this plan are the information requirements and priorities of your users. After this information has been defined and approved by your business sponsor, you can develop your list of key deliverables and assign responsibilities to your team.

Your first task is to define the scope of the project. The scope of the data mart defines the boundaries of the project and is typically expressed in some combination of geography, organization and application, or business functions. Defining scope usually requires making compromises as you try to balance resources (such as people, systems, and budget) with the scheduled completion date and the capabilities you promised to deliver. Defining your scope and making it clear to everyone involved is important because it:

- Sets the right expectations

- Prioritizes incremental development
- Highlights risks and issues
- Allows you to estimate costs

B.2 Defining the Requirements for the Data Mart

To start the implementation of the data mart, you need to define the business and technical requirements. However, you should expect the requirements to change as users use the initial implementation and are better able to communicate their requirements. The development of the data mart is an iterative process, because the data mart evolves in response to feedback from users.

This section contains the following topics:

- [Section B.2.1, "Define Business Requirements"](#)
- [Section B.2.2, "Identify Technical Requirements"](#)
- [Section B.2.3, "How Do You Know If You Have Done It Right?"](#)

B.2.1 Define Business Requirements

The purpose of the data mart is to provide access to data that is specific to a particular department or functional area. The data should be at a meaningful level of detail for the kind of analysis that the end users want to perform, and should be presented in the business terms that they understand. The expectation is that the analysis of the data in a data mart will lead to more informed business decisions. Therefore, you need to understand how the business person makes decisions—what questions the users ask in the decision-making process, and what data is necessary to answer those questions. The best way to understand the business processes is through interviews with the business people. The requirements identified as a result of these interviews comprise the business requirements for your data mart.

As you gather requirements for the data mart, you should focus your efforts on the information needs of a single subject area. Remember that no requirements document will be complete enough to get all information at the outset, and you need to design the data mart to accommodate changing needs. However, if you introduce new subjects or deviate from your primary theme, you will lose your focus and schedule.

Even though the data mart addresses one subject area, it usually has many business users, each with different requirements and expectations. Try to identify at least one representative from each area of the business for your interviews. For example, if you are building a marketing data mart, interview several people involved in various aspects of the marketing function (such as a marketing analyst, channel specialist, direct marketing manager, and promotion manager).

Use a consistent set of questions or an interview template for each interview. The questions should focus on the users' information requirements, such as content, frequency of update, priorities, and level of detail. Finally, set a definite time limit on the interview and requirements gathering phase; otherwise, it could continue indefinitely as you try to refine each requirement. You cannot collect all requirements in this time frame, but you will get enough to create a road map. Needs will change during the implementation period, and you will need a way to evaluate and accommodate requests for changes or to reject and consider them for a future phase.

B.2.2 Identify Technical Requirements

You must identify the technical requirements. These specify where you get the data that feeds the data mart. The primary sources of data for data marts are the operational systems that handle the day-to-day transactional activities. Usually, these operational systems are online transaction processing (OLTP) systems. Your data mart may be fed from more than one such operational source. However, you cannot usually transfer the data from the operational system into the data mart without intermediate processing. You need to understand how clean the operational data is and how much formatting or translation is needed to integrate it with other sources. Also, you need to determine how often you must refresh or update the data. For example, if you use the data for relatively long-term planning and analytical horizons, you may need a weekly or monthly feed rather than a daily feed. Note that the frequency of update does not necessarily determine the level of detail in the data mart. You can have a monthly feed of data summarized by week. In this initial phase of data mart design, you need to identify data sources, the kind of data cleansing needed, and the frequency with which data should be refreshed.

B.2.3 How Do You Know If You Have Done It Right?

When you finish your interviews, you have a set of information and performance requirements that your data mart application must meet. You should be realistic and prioritize the needs and develop a list of success criteria. To prioritize your list, ask yourself these questions:

- Is performance the primary concern?
- Are you constrained by your systems configuration?
- How often do you want to update or append to the data?
- Do the users expect the data mart to be a comprehensive source for departmental data, or is the data mart limited in scope to a particular topic within that department?
- Is your scope consistent with IT architecture, or can you develop autonomously?

Consider the answers to these questions as you develop your priorities and critical success factors.

In summary, here are some guidelines to facilitate your requirements definition process:

- Involve the end users throughout the process.
- Classify the requirements analysis framework: define the requirements for the business sponsor, the IT architect, the data mart developer, and the end users.
- Manage the expectations of the end users.

B.3 Data Mart Design

At the beginning of the design stage, business requirements are already defined, the scope of your data mart application has been agreed upon, and you have a conceptual design. Now, you need to translate your requirements into a system deliverable. In this step, you create the logical and physical design for the data mart and, in the process, define the specific data content, relationships within and between groups of data, the system environment supporting your data mart, the data transformations required, and the frequency with which data is refreshed.

The logical design is more conceptual and abstract than the physical design. In the logical design, you look at the logical relationships among the objects. In the physical design, you look at the most effective way of storing and retrieving the objects.

Your data mart design should be oriented toward the needs of your end users. End users typically want to perform analysis and look at aggregated data, rather than at individual transactions. Your design is driven primarily by end-user utility, but the end users may not know what they need until they see it. A well-planned design allows for growth and changes as the needs of users change and evolve.

The quality of your design determines your success in meeting the initial requirements. Because you do not have the luxury of unlimited system and network resources, optimal utilization of resources is determined primarily by your design. By beginning with the logical design, you focus on the information requirements without getting bogged down immediately with implementation detail.

Note that you are not forced to work in a top-down fashion. You can reverse-engineer an existing data schema and use this as a starting point for your design. If your data requirements are very clear and you are familiar with the source data, you might be able to begin at the physical design level and then proceed directly to the physical implementation. In practice, it takes several iterations before you achieve the right design.

This section contains the following topics:

- [Section B.3.1, "Creating a Logical Design"](#)
- [Section B.3.2, "Creating a Wish List of Data"](#)
- [Section B.3.3, "Identifying Sources"](#)
- [Section B.3.4, "Classifying Data for the Data Mart Schema"](#)
- [Section B.3.5, "Designing the Star Schema"](#)
- [Section B.3.6, "Moving from Logical to Physical Design"](#)

B.3.1 Creating a Logical Design

A logical design is a conceptual, abstract design. You do not deal with the physical implementation details yet; you deal only with defining the types of information that you need. The process of logical design involves arranging data into a series of logical relationships called entities and attributes. An entity represents a chunk of information. In relational databases, an entity often maps to a table. An attribute is a component of an entity and helps define the uniqueness of the entity. In relational databases, an attribute maps to a column.

While entity-relationship diagramming has traditionally been associated with highly normalized models, such as OLTP applications, the technique is still useful in dimensional modeling. You just approach it differently. In dimensional modeling, instead of seeking to discover atomic units of information and all of the relationships between them, you try to identify which information belongs to a central fact table and which information belongs to its associated dimension tables.

Attention to design is critical. Keep your business requirements on hand throughout the design process. Nothing else is more important!

As part of the design process, you map operational data from your source into subject-oriented information in your target data mart schema. You identify business subjects or fields of data, define relationships between business subjects, and name the attributes for each subject.

The elements that help you to determine the data mart schema are the model of your source data and your user requirements. Sometimes, you can get the source model from your company's enterprise data model and reverse-engineer the logical data model for the data mart from this. The physical implementation of the logical data mart model may require some changes due to your system parameters—size of computer, number of users, storage capacity, type of network, and software. You will need to make decisions as you develop the logical design:

- Facts and dimensions
- Granularity of the facts
- Relationship between the entities
- Historical duration of the data

B.3.2 Creating a Wish List of Data

You generate the wish list of your data elements from the business user requirements. This tutorial assumes that the scope of the data mart is fully specified by the users. Often, you must look beyond the specific requests of the users and anticipate future needs.

Start with the business parameters that matter to your subject area. For a Sales and Marketing data mart, parameters might be Customer, Geography, Product, Sales, and Promotions. Remember Time—do you want to look at monthly, daily, or weekly figures?

Then, create a list of desired data elements, either from the requirements provided by the users, or by brainstorming with them. At the end of this exercise, you should have the following:

- A list of data elements, both raw and calculated
- Attributes of the data, such as character or numeric data types
- Reasonable groupings of the data, such as geographical regions for the elements country, county, city
- An idea of the relationship between the data, such as "a city is within a county"

Typical data fields of interest in the Sales and Marketing example might be dollar sales, unit sales, product names, packages, promotion characteristics, regions, and countries. Identify the critical fields—those that drove the creation of the data mart. Data such as dollar sales or unit sales are critical for a sales data mart.

Users may provide you with reports to give you an idea of their data requirements. These reports may be existing reports, or the kind of reports they would like to see. Reports are a good vehicle to get the users to articulate their needs.

At this point, you can separate the data into numeric data (the facts) and textual or descriptive data (the dimensions). During the iterative process of interaction with the end user, ask why certain data is important—what decisions are driven by this data? Some insight into the business processes will help you anticipate future data needs.

B.3.3 Identifying Sources

Now, you have a list of dimensions and facts that you want for your data mart. The question is, can you get the data? And if yes, at what price? Data sources can range from operational systems, such as order processing systems, to spreadsheets. You need to map the individual elements from your wish list to the sources. You should start with the largest, most comprehensive source and seek other sources as needed.

Typically, a large percentage of the data comes from one or two sources. The dimensions can usually be mapped to lookup tables in your operational system. In their raw form, the facts can be mapped to the transaction tables. For use in the data mart, the transaction data usually needs to be aggregated, based on the specified level of granularity. Granularity is the lowest level of information that the user might want. You may find that some of the requested data cannot be mapped. This usually happens when groupings in the source system are not consistent with the desired groups within the data mart. For example, in a telecommunications company, calls can be aggregated easily by area code. However, your data mart needs data by postal code. Because an area code contains multiple postal codes and one postal code may span multiple area codes, it is difficult to map these dimensions.

You may find that some data is too costly to acquire. For example, the promotion data that the users requested may not be obtained easily because the information is not consistent across time or promotion. To translate to a common system format would be very costly.

B.3.4 Classifying Data for the Data Mart Schema

At this point, you have started thinking about the classification of your data as facts and dimensions. A common representation of facts, dimensions, and the relationships between them in data mart applications is the star schema. Typically, it contains a dimension of time and is optimized for access and analysis. It is called a star schema because the graphical representation looks like a star with a large fact table in the center and the smaller dimension tables arranged around it.

Advanced design modeling may involve schemas, called snowflake or constellation schemas, which are more complex than the simple star schema shown. The following sections provide more information about dimensions, facts, and level of granularity.

This section contains the following topics:

- [Section B.3.4.1, "Dimensions"](#)
- [Section B.3.4.2, "Facts"](#)
- [Section B.3.4.3, "Granularity"](#)

B.3.4.1 Dimensions

In your classification exercise, many of the fields from the OLTP source will end up as dimensions. The big design issue is to decide when a field is just another item in an existing dimension, or when it should have its own dimension. The time dimension is generated independently using the discrete dates in the OLTP source. This offers flexibility in doing any time series analysis. For a true star schema, the creation order of the dimension tables does not matter as long as they are created before the fact table. Generally, a table must be created before other tables can reference it. Therefore, be sure to create all dimension tables first.

B.3.4.2 Facts

Facts are the numeric metrics of the business. They support mathematical calculations used to report on and analyze the business. Some numeric data are dimensions in disguise, even if they seem to be facts. If you are not interested in a summarization of a particular item, the item may actually be a dimension. Database size and overall performance will improve if you categorize borderline fields as dimensions.

For example, assume that you have a membership database for a health club and want to find out how much of the club brand vitamins the members buy. In your wish list,

you have several queries like "Give me the usage by age by..." and "Give me the average age of members by..." Is age a fact or a dimension? Make it a dimension.

B.3.4.3 Granularity

After you define the facts and dimensions, you determine the appropriate granularity for the data in the data mart. At this point, you know why your users have requested a particular level of information within a dimension. You need to estimate the resource requirements to provide the requested level of granularity and, based on the costs, decide whether or not you can support this level of granularity.

B.3.5 Designing the Star Schema

After you have a list of all facts, dimensions, and the desired level of granularity, you are ready to create the star schema. The next step is to define the relationships between the fact and dimension tables using keys.

A primary key is one or more columns that make the row within a table unique. The primary key of the fact table can consist of several columns. Such a key is called a composite or concatenated key.

It is a good idea to use system-generated keys (synthetic keys), in place of natural keys, to link the facts and the dimensions. This provides the data mart administrator with control of the keys within the data mart environment, even if the keys change in the operational system.

A synthetic key is a generated sequence of integers. You include the synthetic keys in the dimension table, in addition to the natural key. Then, you use the synthetic key in the fact table as the column that joins the fact table to the dimension table.

Although creating synthetic keys requires additional planning and work, the keys can provide benefits over natural keys:

- Natural keys are often long character strings, such as in a product code. Because synthetic keys are integers, response time to queries is improved.
- The data mart administrator has control over the synthetic key. If a manufacturing group changes the product code naming conventions, the changes do not affect the structure of the data mart. Consider using synthetic keys for most dimension tables. (In the rest of this tutorial, we refer to synthetic keys as warehouse keys.)

The process of translating the data from the OLTP database and loading the target star schema requires mapping between the schemas. The mapping may require aggregations or other transforms.

B.3.6 Moving from Logical to Physical Design

During the physical design process, you convert the data gathered during the logical design phase into a description of the physical database, including tables and constraints. This description optimizes the placement of the physical database structures to attain the best performance. Because data mart users execute certain types of queries, you want to optimize the data mart database to perform well for those types of queries. Physical design decisions, such as the type of index or partitioning, have a huge impact on query performance.

As the data mart becomes successful and more widely used, more and more users will access it. Over time, the volume of data will also grow. Scalability, the ability to increase the volume of data and number of users, is an important consideration when you move from your logical design to a physical representation. To accommodate the

need for scalability, you should minimize the limitations of factors such as hardware capacity, software, and network bandwidths.

This section contains the following topics:

- [Section B.3.6.1, "Estimating the Size of the Data Mart"](#)
- [Section B.3.6.2, "What Is Metadata?"](#)

B.3.6.1 Estimating the Size of the Data Mart

In estimating the size of your data mart, you need to develop a method that will accommodate its future growth. There are several methods for estimating the size of the database. Here is one approach:

1. Use a representative sample of the source data to determine the number of rows in the fact table.
2. Estimate the size of one row in the fact table.
3. Estimate the size of the fact table by multiplying the number of rows by the size of one row.
4. Estimate the size of the data mart. Generally, the total size of the data mart is three to five times the size of the fact table.

This process is usually iterative. Each time the design changes, you should estimate the size again. Even if you think that your star schema is small, you should do this calculation once.

After you calculate the size, you can validate your assumptions by doing the following:

1. Extract sample files.
2. Load data into the database.
3. Compute exact expected row lengths.
4. Add overhead for indexing, rollback, and temporary tablespaces, and a file system staging area for flat files.

To plan for future growth, you can use the ratio of the estimated size to the largest possible size of the fact table to calculate the future size of the data mart:

1. For each dimension, check the granularity that you want and estimate the number of entries in the finest level.
2. Multiply the number of entries of all dimensions to get the maximum possible rows.
3. Calculate the ratio of actual rows from representative data to possible rows.
4. Estimate the growth for each dimension table over a period of time.
5. Multiply the number of rows of all dimension tables.
6. Adjust the number, using the ratio calculated in Step 3.
7. Multiply the result by the fact table row size.

You may need to schedule a regular batch job to refresh your data mart from your sources. Depending on the data volumes and system load, this job may take several hours. Plan your data mart refresh so that under normal circumstances it can be accomplished within the time allowed for batch processing, usually at night. In your planning process, you should also estimate the data volume that will be refreshed. Develop a strategy for purging the data beyond the specified retention period.

B.3.6.2 What Is Metadata?

Metadata is information about the data. For a data mart, metadata includes:

- A description of the data in business terms
- Format and definition of the data in system terms
- Data sources and frequency of refreshing data

The primary objective for the metadata management process is to provide a directory of technical and business views of the data mart metadata. Metadata can be categorized as technical metadata and business metadata.

Technical metadata consists of metadata created during the creation of the data mart, as well as metadata to support the management of the data mart. This includes data acquisition rules, the transformation of source data into the format required by the target data mart, and schedules for backing up and refreshing data.

Business metadata allows end users to understand what information is available in the data mart and how it can be accessed.

You use the technical metadata to determine data extraction rules and refresh schedules for the Oracle Warehouse Builder component. Similarly, you use the business metadata to define the business layer used by the Oracle BI Answers tool.

