

Oracle BI 11g R1: Build Repository Practice Guide

Table of Contents

Practices for Lesson 1: Course Introduction	1-1
Practices for Lesson 1.....	1-3
Practices for Lesson 2: Repository Basics.....	2-1
Overview of Practices for Lesson 2.....	2-3
Practice 2-1: Exploring an Oracle BI Repository.....	2-4
Practices for Lesson 3: Building the Physical Layer of a Repository.....	3-1
Overview of Practices for Lesson 3.....	3-3
Practice 3-1: ABC Business Scenario.....	3-4
Practice 3-2: Gathering Information to Build an Initial Business Model	3-9
Solutions 3-2: Gathering Information to Build an Initial Business Model.....	3-10
Practice 3-3: Creating a Repository and Importing a Data Source.....	3-11
Practice 3-4: Creating Alias Tables.....	3-17
Practice 3-5: Defining Keys and Joins.....	3-20
Practices for Lesson 4: Building the Business Model and Mapping Layer of a Repository	4-1
Overview of Practices for Lesson 4.....	4-3
Practice 4-1: Creating the Business Model.....	4-4
Practice 4-2: Creating Simple Measures	4-11
Practices for Lesson 5: Building the Presentation Layer of a Repository.....	5-1
Overview of Practices for Lesson 5.....	5-3
Practice 5-1: Creating the Presentation Layer	5-4
Solutions 5-1: Creating the Presentation Layer.....	5-10
Practices for Lesson 6: Testing and Validating a Repository.....	6-1
Overview of Practices for Lesson 6.....	6-3
Practice 6-1: Testing the Repository	6-4
Practices for Lesson 7: Managing Logical Table Sources.....	7-1
Overview of Practices for Lesson 7.....	7-3
Practice 7-1: Enhancing the Product Dimension.....	7-4
Practice 7-2: Creating Multiple Sources for a Logical Table Source (Manual)	7-9
Practice 7-3: Creating Multiple Sources for a Logical Table Source (Automated).....	7-13
Practice 7-4: Adding a New Logical Table Source.....	7-23
Practices for Lesson 8: Adding Calculations to a Fact.....	8-1
Overview of Practices for Lesson 8.....	8-3
Practice 8-1: Creating Calculation Measures.....	8-4
Practice 8-2: Creating Calculation Measures by Using the Calculation Wizard.....	8-15
Practice 8-3: Creating a Rank Measure.....	8-22
Practices for Lesson 9: Working with Logical Dimensions	9-1
Overview of Practices for Lesson 9.....	9-3
Practice 9-1: Creating Logical Dimension Hierarchies.....	9-4
Practice 9-2: Creating Level-Based Measures.....	9-19
Practice 9-3: Creating Share Measures.....	9-21
Practice 9-4: Creating Dimension-Specific Aggregation Rules	9-22
Practice 9-5: Creating Presentation Hierarchies	9-24
Practice 9-6: Creating Parent-Child Hierarchies	9-34

Practice 15-1: Enabling Query Caching..... 15-4

Practice 15-2: Modifying Cache Parameters..... 15-11

Practice 15-3: Seeding the Cache 15-16

Practices for Lesson 1: Course Introduction

Chapter 1

Practices for Lesson 1

Practices Overview

There are no practices for this lesson.

Practices for Lesson 2: Repository Basics

Chapter 2

Overview of Practices for Lesson 2

Lesson Overview

In these practices, you will explore an existing Oracle Business Intelligence repository.

Practice 2-1: Exploring an Oracle BI Repository

Goal

To explore the three layers of an Oracle BI repository

Scenario

Before beginning the development of a repository, you use the Administration Tool to explore an existing repository to get a better understanding of its three layers and how the layers relate to one another, and to understand the link between physical data sources and the information presented in the Oracle BI user interface.

Tasks -

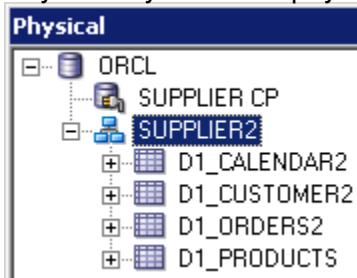
Import **Supplier2** Database dump file present in PracticeFiles folder . Use the syntax below,

impdp Supplier2 DUMPFILE=DATA_PUMP_DIR:SUPPLIER2.DMP FULL=YES LOGFILE=DATA_PUMP_DIR:full_imp.log.

Create and ODBC Data Source for Supplier2 Database. Go to Control Panel -> Administrative Tools – ODBC Data Sources -> System DSN tab

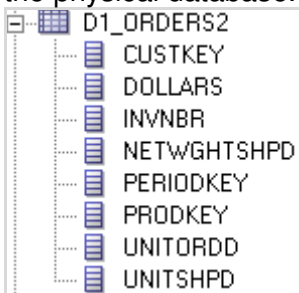
1. Copy the repository for this practice to the appropriate directory.
 - a. Navigate to **D:\PracticeFiles**.
 - b. Copy the **ClassStart.rpd** file.
 - c. Paste the file in
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
2. Start the Oracle BI Administration Tool and open the ClassStart repository in offline mode.
 - a. Select **Start > Programs > Oracle Business Intelligence > BI Administration**.
 - b. Select **File > Open > Offline**.
 - c. In the **Open** dialog box, double-click **ClassStart.rpd**.
 - d. Enter **welcome1** as the repository password and click **OK**. The repository opens in offline mode.
3. Examine the properties of the ORCL database object.
 - a. In the Physical layer, double-click the **ORCL** database object to view its properties.
 - b. Click the **General** tab.
 - c. Notice that the database platform type is **Oracle 11g/Exadata**.
 - d. Click the **Features** tab. Each database comes with a set of features that determine the SQL that the Oracle BI Server will issue for this database. Features can have a Boolean value (on or off), integer value, or a string value. Scroll to the right to view the Value and Default columns. A check mark in the Default column indicates that the feature is supported by this database type and a check mark in the Value column indicates that the feature is enabled.
 - e. Click the **Connection Pools** tab. This tab identifies all the connection pools associated with this database. In this example, there is only one connection pool, **SUPPLIER CP**.
 - f. Click the **Display Folders** tab. Physical layer objects can be organized into display folders. When there are display folders in the Physical layer, they are listed here.
 - g. Click **Cancel** to close the Properties dialog box.
4. Explore the properties of a connection pool object.
 - a. In the Physical layer, expand the **ORCL** database object.
 - b. Double-click the **SUPPLIER CP** connection pool object.

- c. Notice that the call interface type for this connection pool is OCI 10g/11g and the data source name is ORCL. The call interface is the application programming interface (API) used to access the data source. Some databases may be accessed using native APIs, some using ODBC. In this example, the ORCL data source is accessed by the Oracle Call Interface (OCI) native API. The data source name, ORCL, is a tnsnames.ora entry.
 - d. Click **Cancel**.
5. Examine the properties of a physical schema and its physical table objects.
 - a. Expand the **SUPPLIER2** schema folder to display the physical table objects in the Physical layer. These physical table objects map to tables in the physical database.



There are more tables in the physical database. The tables displayed here are the tables that have been imported into the Physical layer. You learn more about importing tables in the lesson titled “Building the Physical Layer of a Repository.”

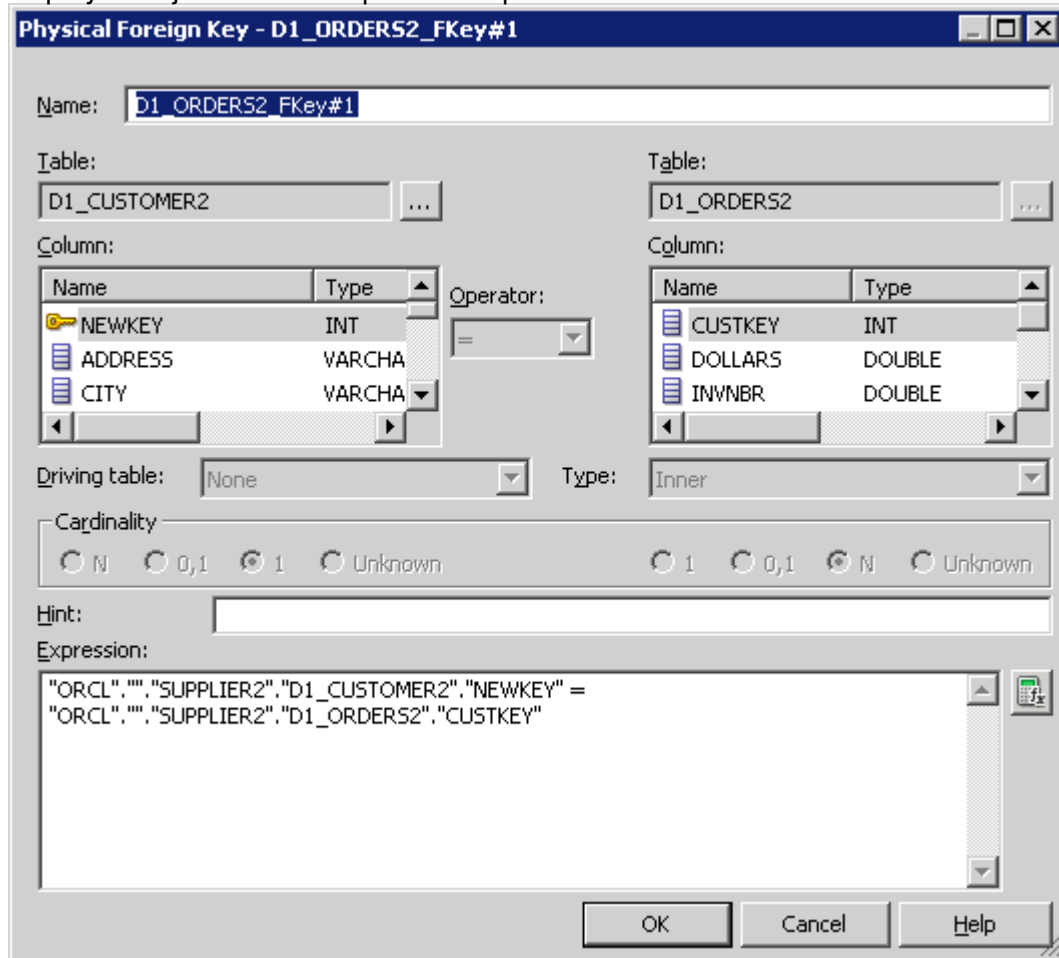
- b. Expand **D1_ORDERS2** to view the physical columns for this table. D1_ORDERS2 is the “fact” table in this business model. These columns correspond to the columns in the physical database.



- c. Right-click **D1_ORDERS2** and select **View Data**. The first 100 rows of data for this table are displayed.

View Data from Table "orcl"."SUPPLIER2"."D1_ORDERS2"					
351636 rows		<input type="checkbox"/> Distinct		Query	
Show	100	rows starting from	0	Close	
	CUSTKEY	DOLLARS	INVNBR	NETWGHTSHPD	PERIODKEY
0	1052	349.14	1694415.0	378.0	20080105.0
1	1052	462.5	1694416.0	233.0	20080105.0
2	1052	465.81	1694417.0	211.27	20080105.0
3	1052	60.2	1694417.0	150.0	20080105.0
4	1052	24.36	1694417.0	14.25	20080105.0
5	1052	872.92	1694417.0	502.48	20080105.0
6	1052	395.26	1694417.0	530.55	20080105.0
7	1052	1607.0	1694417.0	2945.55	20080105.0
8	1052	549.87	1694417.0	1077.0	20080105.0
9	1052	209.87	1694417.0	401.0	20080105.0

- d. Click **Close**.
- e. Double-click the **D1_ORDERS2** table object to view its properties.
- f. Click the **Columns** tab to view the columns in this table. This is another way to create, view, and modify physical columns.
- g. Click the **Foreign Keys** tab.
- h. Notice that all three tables have join relationships with D1_ORDERS2: D1_CUSTOMER2, D1_CALENDAR2, and D1_PRODUCTS.
- i. Double-click one of the foreign keys. The Physical Foreign Key dialog box opens and displays the join relationship in the Expression field.

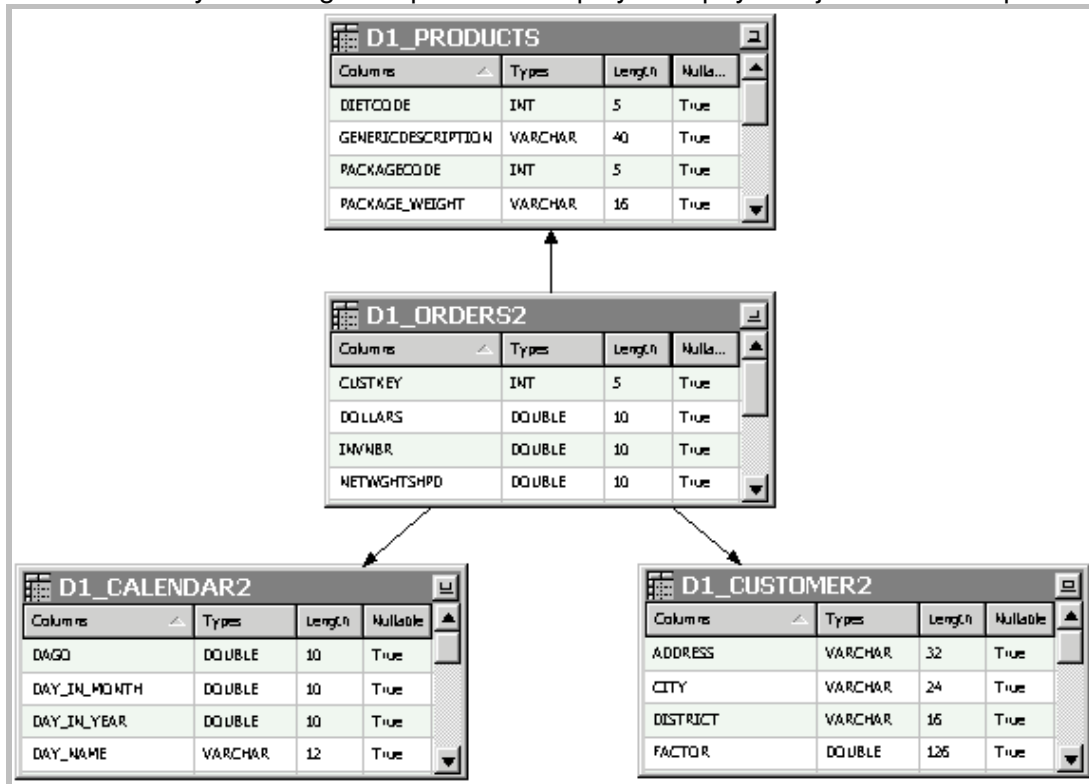


The dialog box is titled "Physical Foreign Key - D1_ORDERS2_FKey#1". It contains the following fields and controls:

- Name:** D1_ORDERS2_FKey#1
- Table:** D1_CUSTOMER2 (with a selection button "...")
- Column:** A list box showing columns: NEWKEY (INT), ADDRESS (VARCHAR), and CITY (VARCHAR). The "NEWKEY" column is selected and marked with a key icon.
- Operator:** =
- Table:** D1_ORDERS2 (with a selection button "...")
- Column:** A list box showing columns: CUSTKEY (INT), DOLLARS (DOUBLE), and INVNBR (DOUBLE). The "CUSTKEY" column is selected.
- Driving table:** None
- Type:** Inner
- Cardinality:** Two sets of radio buttons. The first set has options N, 0,1, 1 (selected), and Unknown. The second set has options 1, 0,1, N (selected), and Unknown.
- Hint:** (empty text field)
- Expression:** A text area containing the expression: "ORCL", "", "SUPPLIER2", "D1_CUSTOMER2", "NEWKEY" = "ORCL", "", "SUPPLIER2", "D1_ORDERS2", "CUSTKEY"
- Buttons:** OK, Cancel, and Help.

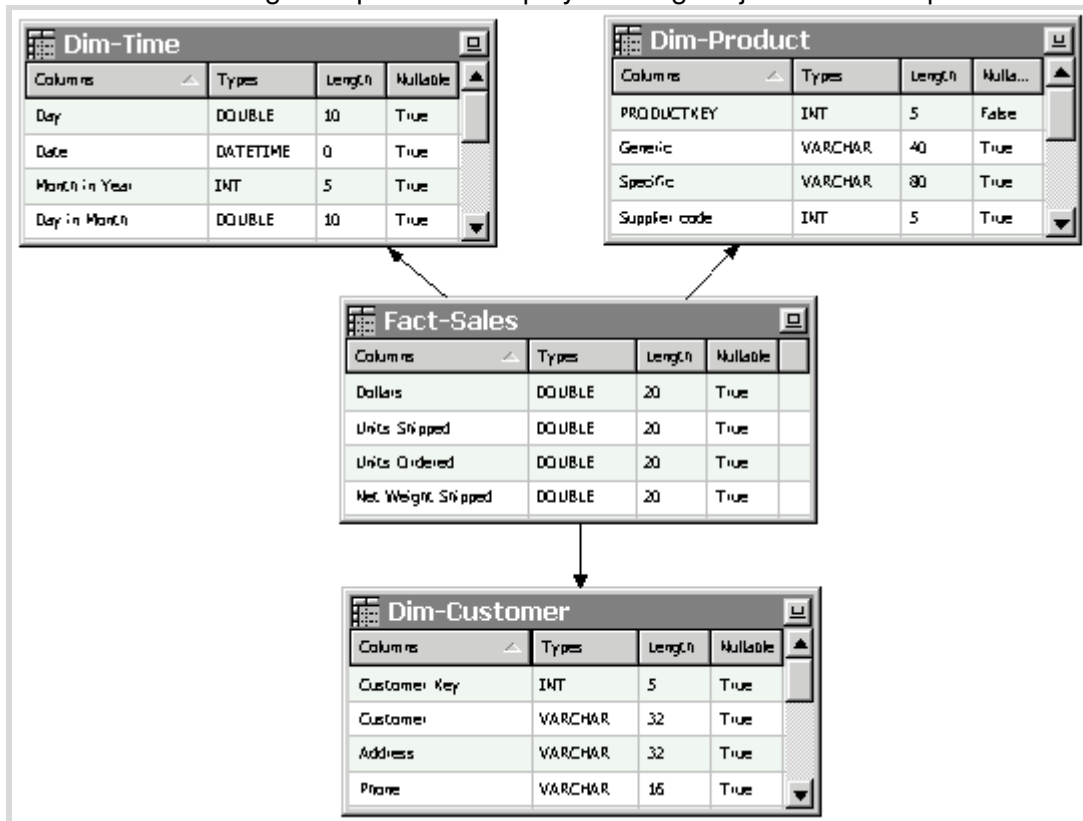
- j. Click **Cancel** to close the Physical Foreign Key dialog box.
- k. Click **Cancel** to close the Physical Table dialog box.

- I. Right-click **D1_ORDERS2** and select **Physical Diagram > Object(s) and Direct Joins**. The Physical Diagram opens and displays the physical join relationships.



- m. Double-click the **connector** between D1_CUSTOMER2 and D1_ORDERS2. The Physical Foreign Key dialog box opens and displays the join relationship in the Expression field. This is another way to view, build, and modify joins between tables in the Physical layer.
- n. Click **Cancel**.
- o. Close the Physical Diagram.
6. Examine the properties of a physical column object.
 - a. Expand the **D1_CUSTOMER2** table object.
 - b. Double-click the **Address** physical column to view the properties.
 - c. Click **Cancel**.
7. Examine the properties of a logical table in the SupplierSales business model.
 - a. If necessary, expand the **SupplierSales** business model in the Business Model and Mapping layer.
 - b. Notice that there are four logical table objects in the SupplierSales business model: Fact-Sales, Dim-Time, Dim-Customer, and Dim-Product.
 - c. Expand the **Fact-Sales** logical table to view the logical columns for this table. Fact-Sales is the logical “fact” table in this business model. These logical columns map to columns in the Physical layer.
 - d. Double-click the **Fact-Sales** logical table object.
 - e. Click the **General** tab. Notice that the logical columns and their corresponding properties are listed. On this tab you can change the name of the logical table, reorder the columns, and add, edit, or remove a column.

- f. Click the **Sources** tab. The source for this logical table is the D1_ORDERS2 table that you explored in the Physical layer. In a more complex business model, there may be many physical sources for a logical table.
- g. Click the **Keys** tab. Typically no keys are defined for a logical fact table.
- h. Click the **Foreign Keys** tab. Foreign key joins are typically not used in the Business Model and Mapping layer. All joins in the Business Model and Mapping layer are logical joins that do not require primary key – foreign key relationships.
- i. Click **Cancel** to close the Logical Table properties dialog box.
- j. Right-click **Fact-Sales** and select **Business Model Diagram > Whole Diagram**. The Business Model Diagram opens and displays the logical join relationships.



- k. Double-click the **connector** between Dim-Customer and Fact-Sales. The Logical Join dialog box opens. Notice that there is no join expression in the Expression field and that there is a one-to-many relationship between the Dim-Customer logical dimension table and the Fact-Sales logical fact table. You learn more about logical joins in the

lesson titled “Building the Business Model and Mapping Layer of a Repository.”

Logical Join - Relationship_2004:4711150758995

Name: Relationship_2004:4711150758995

Business model: SupplierSales

Table: Dim-Customer

Column: Customer Key (INT)

Table: Fact-Sales

Column: Units Shipped (DOUBLE)

Operator: Inner

Driving table: None

Type: Inner

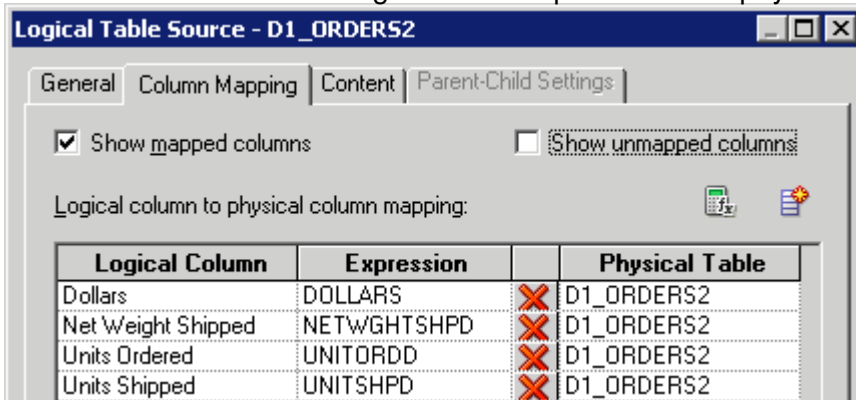
Cardinality: 1

Expression:

OK Cancel Help

- l. Click **Cancel** to close the Logical Join dialog box.
- m. Close the Business Model Diagram.
8. Examine the logical table source for the Fact-Sales logical table.
 - a. Expand **Fact-Sales > Sources** to display the D1_ORDERS2 logical table source for this logical table. In this example, there is only one logical table source. However, it is possible to have many logical table sources for a single logical table.
 - b. Double-click the **D1_ORDERS2** logical table source to view the properties.
 - c. Click the **General** tab.
 - d. Notice that the D1_ORDERS2 logical table source maps to the D1_ORDERS2 physical table.
 - e. Click the **Column Mapping** tab. This tab shows the mappings between the logical columns and physical columns.

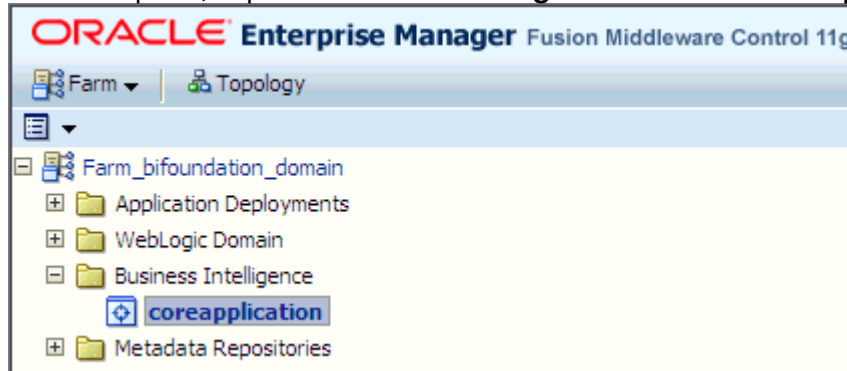
- f. If necessary, scroll to the right to view the Physical Table column. Notice that all columns in the Fact-Sales logical table map to the same physical table.



- g. Notice that some logical columns have names that are different from the physical columns to which they map. This is because the column names were changed in the Business Model and Mapping layer. You learn how to rename logical columns in the lesson titled “Building the Business Model and Mapping Layer of a Repository.”
 - h. Click the **Content** tab. Currently there is no information on the tab. You learn how to use this tab to identify aggregation content and fragmentation content in the lesson titled “Using Aggregates.”
 - i. Click **Cancel** to close the Logical Table Source dialog box.
9. Examine the properties of a logical column in the Fact-Sales logical table.
 - a. Double-click the **Dollars** logical column to open the properties window. Dollars is a measure in this business model.
 - b. Click the **General** tab. This tab provides general information about the column, such as the column name, the table it belongs to, and a description of the column.
 - c. Click the **Column Source** tab. This tab provides information about the physical table and physical column that the logical column maps to, or whether the logical column is derived from other existing logical columns.
 - d. Double-click the **D1_ORDERS2** logical table source in this dialog box. Notice that this is another way to access the Logical Table Source properties dialog box.
 - e. Click **Cancel** to close the Logical Table Source properties dialog box.
 - f. Click the **Aggregation** tab. Notice that the default aggregation rule is set to Sum. It is common to apply aggregation rules to measures in business models. Open the Default aggregation rule drop-down list to see the other available aggregation rules.
 - g. Make sure that the Sum aggregation rule is still selected and click **Cancel** to close the Logical Column dialog box.
 10. Examine the properties of the SupplierSales presentation catalog object.
 - a. In the Presentation layer, double-click the **SupplierSales** subject area to open the Subject Area properties dialog box.
 - b. Click the **General** tab. Notice that the subject area name is the same as the business model name. This is because the subject area was created by dragging the business model from the Business Model and Mapping layer to the Presentation layer. If desired, you could use this tab to change the name of the subject area. The subject area and its description appear in the Analysis Editor user interface. Many subject areas can map to a single business model, but each subject area can map to only one business model.

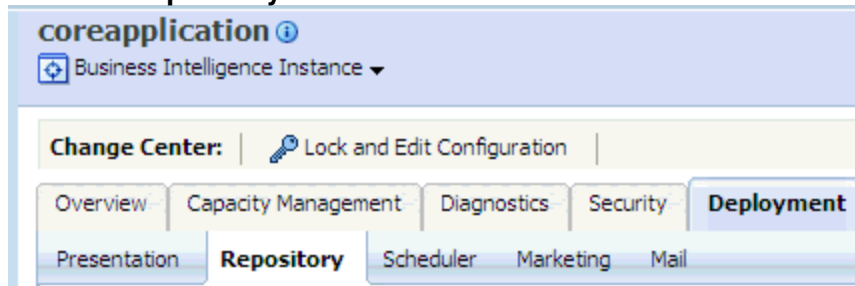
- c. Click **Permissions** to open the Permissions dialog box. This shows the permissions for all users and application roles in the repository. Currently, no permissions are defined for users or application groups, so the default is to give everyone read permission for this subject area. You learn more about setting object permissions in the lesson titled "Security."
 - d. Click **Cancel** to close the Permissions dialog box.
 - e. Click the **Presentation Tables** tab to display a list of presentation tables. You can use this tab to add, remove, edit, or change the display order of the presentation tables in the subject area.
 - f. Click the **Aliases** tab. If you change the name of a subject area, the tool automatically creates an alias based on the previous name. You can use this tab to specify or delete an alias.
 - g. Click **Cancel** to close the Subject Area dialog box.
11. Examine the properties of a presentation table in the SupplierSales subject area.
- a. In the Presentation layer, expand **SupplierSales** to view the presentation tables.
 - b. Double-click the **Fact-Sales** table to view the properties.
 - c. Click the **General** tab. You can use this tab to change the name of the presentation table. A description would appear as a "tool tip" in the Oracle BI user interface.
 - d. Click the **Columns** tab to see a list of columns and their mappings in the Fact-Sales presentation table. You can use this tab to add, remove, edit, or change the display order of the presentation columns.
 - e. Double-click **Dollars** to open the Presentation Column properties dialog box. This is one method for viewing and modifying presentation column properties.
 - f. Click **Cancel** to close the Presentation Column properties dialog box.
 - g. Click the **Hierarchies** tab. Currently no hierarchies are defined for this presentation table. You learn about presentation hierarchies in the lesson titled "Working with Logical Dimensions."
 - h. Click **Cancel** to close the Presentation Table properties dialog box.
12. Examine the properties of a presentation column in the Fact-Sales table.
- a. Expand the **Fact-Sales** presentation table.
 - b. Double-click **Dollars** to open the Presentation Column properties dialog box. Notice that this is the same dialog box that you saw earlier. This is another method for viewing and modifying presentation column properties.
 - c. Click **Cancel** to close the Presentation Column properties dialog box.
 - d. Click **File > Close** to close the repository without saving any changes.
 - e. Leave the Administration Tool open.
13. Use Fusion Middleware Control Enterprise Manager to upload the repository.
- a. Open the Internet browser and enter the following URL:
http://localhost:7001/em
 - b. Enter **weblogic** as the username and **welcome1** as the password.

- c. In the left pane, expand **Business Intelligence** and select **coreapplication**.

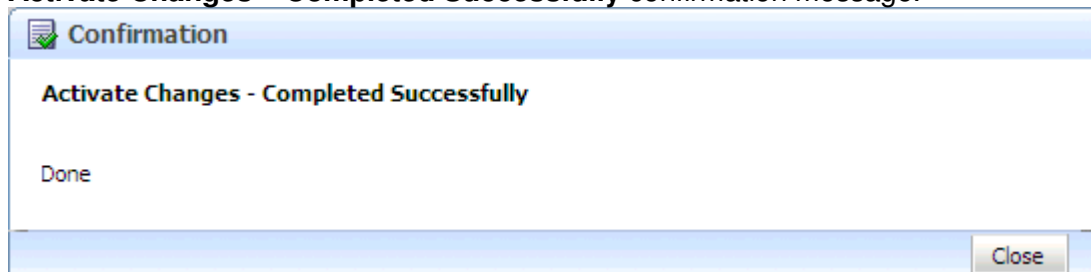


- d. In the right pane, click the **Deployment** tab.

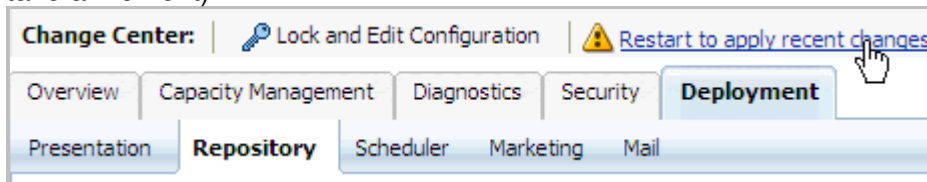
- e. Click the **Repository** subtab.



- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the confirmation message "Lock and Edit configuration – Completed Successfully."
- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository**.
- j. Select **ClassStart.rpd** and click **Open**.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays ClassStart with an extension (for example, ClassStart_BI0001).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the **Activate Changes – Completed Successfully** confirmation message.



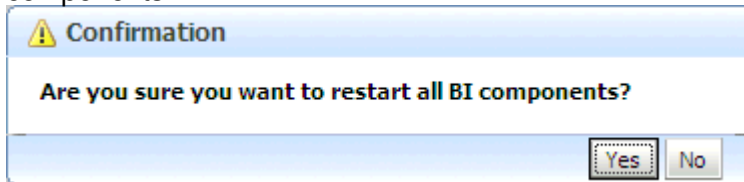
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



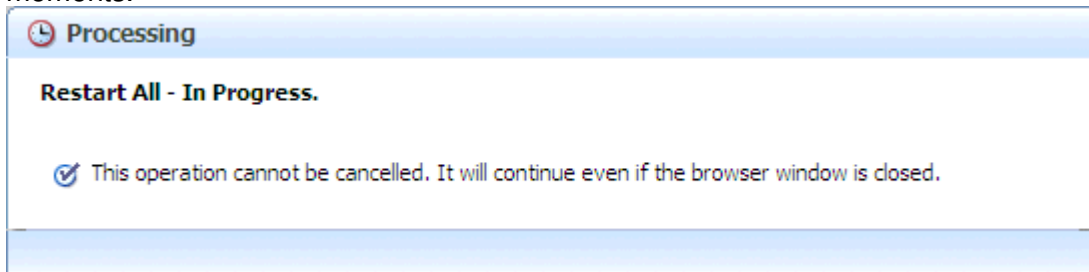
- p. On the Overview page, click **Restart**.



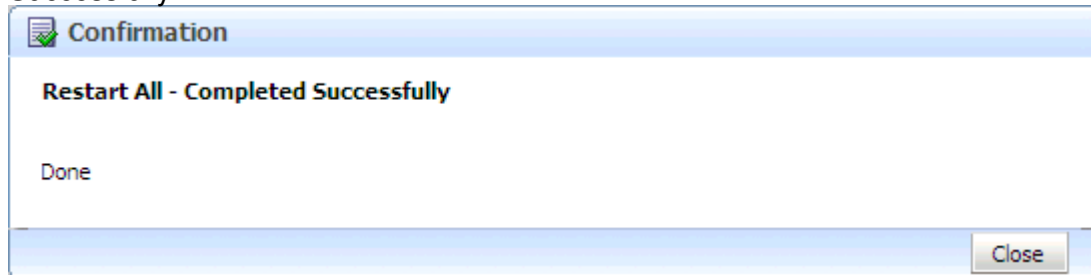
- q. Click **Yes** when you receive the message “Are you sure you want to restart all BI components?”



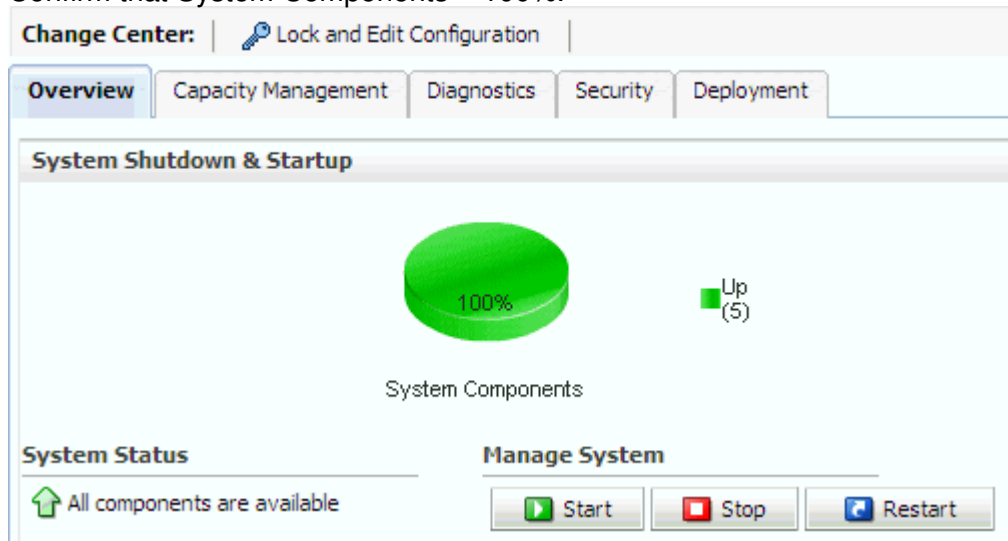
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the confirmation message “Restart All – Completed Successfully.”



- t. Confirm that System Components = 100%.



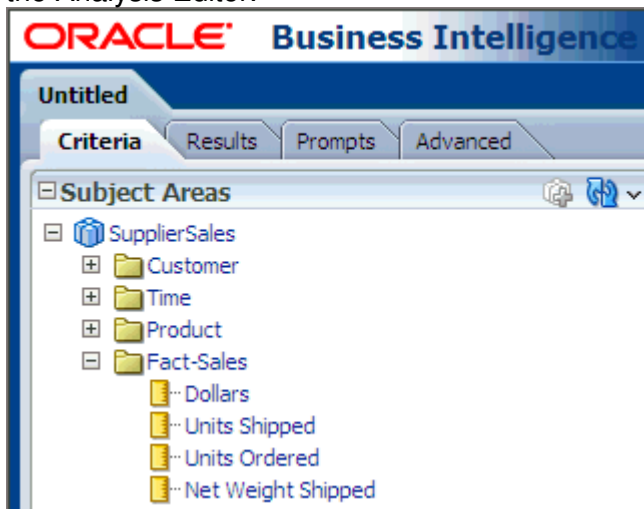
- u. Leave Enterprise Manager open.
14. Examine the relationship between the columns in the Presentation layer and the columns displayed in Oracle BI Analysis Editor.
- Return to the Administration Tool, which should still be open.
 - Select **File > Open > Offline**.
 - Double-click **ClassStart.rpd** to open it in offline mode.
 - Enter **welcome1** as the repository password and click OK. The ClassStart repository opens in offline mode.
 - Return to the browser, which should still be open with FMW Enterprise Manager.
 - On a new tab, enter the following URL:
http://localhost:7001/analytics

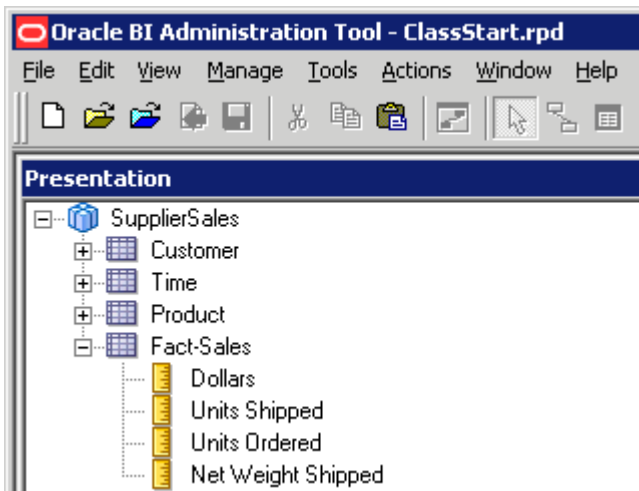
- g. Sign in as **weblogic** with password **welcome1**.



The image shows the Oracle Business Intelligence Sign In page. At the top, the Oracle Business Intelligence logo is displayed in red and blue, with a 'Help' link to the right. Below the logo is a 'Sign In' section with the instruction 'Enter your user id and password.' There are two input fields: 'User ID' containing the text 'weblogic' and 'Password' containing a series of dots. Below these fields is a 'Sign In' button. At the bottom of the sign-in section is a language selection dropdown menu showing 'English'.

- h. In the Create section on the left, click **Analysis**.
- i. Click the **SupplierSales** subject area to open the Analysis Editor.
- j. Size the windows of the browser and the Administration Tool so that you can see the two applications side by side. Notice that the SupplierSales subject area in the Presentation layer of the repository corresponds to the SupplierSales subject area in the Analysis Editor.

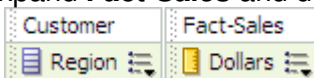




The key point to understand is this: What you see in the Analysis Editor is driven by what is defined in the Presentation layer of the repository in the Administration Tool.

15. Create a simple request.

- a. In the Analysis Editor, expand the **Customer** table and double-click the **Region** column to add it to the analysis criteria in the right pane.
- b. Expand **Fact-Sales** and double-click the **Dollars** column to add it to the request.



- c. Click the **Results** tab. The table displays the total dollars for each region.

Region	Dollars
Central	13,423,387
East	25,460,351
West	25,728,722

- d. Sign out of Oracle Business Intelligence.



- e. Click **OK** when you receive the message "Are you sure you want to navigate away from this page..."
- f. Leave the browser open.
- g. In the Administration Tool, close the **ClassStart** repository without saving.
- h. Leave the Administration Tool open.

Practices for Lesson 3: Building the Physical Layer of a Repository

Chapter 3

Overview of Practices for Lesson 3

Lesson Overview

In these practices, you will build the physical layer of an Oracle Business Intelligence repository.

Practice 3-1: ABC Business Scenario

Goal

To read the business scenario for the fictitious company used throughout the course

Time

10 minutes

Background

ABC sells food and nonfood items to restaurant chain customers throughout the United States. ABC's product line includes food, condiments, cookware, clothing, and other miscellaneous restaurant supplies.

ABC employees currently rely on reports generated by their IT department to analyze sales and shipment data. These reports are generated monthly and, therefore, contain static information. ABC is looking for a way to generate dynamic, interactive reports to analyze this data in order to effectively manage its orders, monitor sales performance, and increase overall customer satisfaction. ABC recognizes the importance of a business intelligence solution that will provide its employees with the data and tools that they need to query large data sets, generate reports, analyze data, identify trends, and monitor business performance.

To achieve these goals, ABC has decided to implement an Oracle Business Intelligence (BI) solution. The solution will enable its sales executives and operations managers to answer business questions that are important to running the part of the business for which they are responsible. The desired system will provide timely, up-to-date sales order data and provide sales and shipment performance monitoring, which will allow ABC to improve its customer service efforts. ABC also has the vision of making this data directly available to customers, so that they can analyze their purchase history better and compare it with what is being purchased nationally or regionally. This initiative is consistent with ABC's commitment to deliver a high-quality business intelligence solution that will set it apart from its competitors, and result in additional sales and revenue.

Business Requirements

You are a consultant hired by ABC to implement a business intelligence solution. From your initial interviews with the managers at ABC, you obtain the following information about ABC:

Managers want to use their own information to ask and answer questions about the sales history of their products, the buying history of their customers, the order fulfillment performance of their operations group, and the selling performance of the sales force.

The company has little experience in data analysis and does not expect to hire any data analysts in the future. Therefore, the managers want to be able to ask relevant questions and analyze the results themselves using an intuitive user interface.

The database has approximately 350,000 invoice-level records that span the period from January 2, 2008 to April 21, 2009.

To make this kind of information widely available within the enterprise, the business intelligence solution must have a structure that is consistent with the way employees think about the business. From further interviews and an examination of the existing sample reports presenting invoice-based data, you determine the following:

ABC employees think about their business in terms of sales, products and product hierarchies, time periods, and relationships between customers and the sales force.

ABC employees measure product data at five levels. The product levels from the top level to the bottom (most detailed) level are:

- ☐ Total
- ☐ Type
- ☐ Subtype
- ☐ Generic Product
- ☐ Specific Product

ABC employees measure sales organization data at five levels. These levels mirror organizational management responsibilities. They are (in descending order):

- ☐ Total
- ☐ Region
- ☐ District
- ☐ Sales Representative
- ☐ Customer

ABC employees measure time data at five levels:

- ☐ Total time
- ☐ Year
- ☐ Quarter
- ☐ Month
- ☐ Day

Many existing reports contain data presented at these levels. An example of a report containing product data levels is:

- ☐ Total Product sales
- ☐ Product Type sales: Cheese
- ☐ Product Subtype sales: American Cheese
- ☐ Generic Product sales: American Cheese Slices
- ☐ Specific Product sales: 2 Pack American Cheese Slices 16 Slices

In addition to these hierarchies, ABC wants to group and analyze customers using geographical attributes such as region, city, and state. They also want to analyze products by characteristics such as diet type and suppliers.

The most common measures used in reports are dollar sales, units ordered, units shipped, and net weight shipped. ABC would like to analyze these measures at all levels of the hierarchies, by the important product and customer attributes, by the various time periods, and be able to compare performance with previous years.

Source Data

By interviewing ABC's database administrators and examining the documents that they provide, you learn that the source data resides in an invoice system on an Oracle database. The core data containing the business measures is stored in an invoice (orders) table.

In addition to the invoice information, there are various tables providing information about customers, the product lines, the time periods, and the relationship between customers and the company's sales organization.

The following screenshots show examples of data in some of the tables:

D1_PRODUCTS

PRODUCTKEY	PGCODE	GENERICDESCRIPTION	SPECIFICDESCRIPTIN	SUPPLIERCODE
1130	130 - Pepper; Pc	Black Pepper	100 Ct Single-serving Black Pepper 1/8 tsp	2300
1131	131 - Boxes	Take-out Containers	"100 Ct Foam Containers 6""x6""x3""	1400
1132	132 - Checks	Guest Checks	10 Pak Guest Check Tablet w/Tearoff Rec...	1400
1133	133 - Wraps	Shrink Wrap	1000' Deli-style Shrink Wrap	2100
1134	134 - Containers	Refrigerator Jars	Refrigerator Jars	2200
1135	135 - Disposable Papers	Placemats	500 Ct Disposable Placemats	1400
1136	136 - Sanitary	Liquid Disinfectant	Liquid Disinfectant	1400
1137	137 - Bags	Take-out bags	Take-out Bags 200-pk	1400
1138	138 - Can Liners	Can Liners	Can Liners 2 qt 100-pak	1400
1139	139 - Dishes	Plain Dinnerware	20 Piece Setting for 4	2100
1140	140 - Cutlery/Flatware	Flatware Fancy	20 Piece Service for 5	1400

D1_CUSTOMER2

NEWKEY	NAME	ADDRESS	PHONE	CITY	STATE
1111	Midway Cafe	400 Crescent Ct	(214) 871-3240	Dallas	TX
1112	Penn Brewery	1717 N Akard St	(214) 720-5291	Dallas	TX
1113	Soulfood Seafood	1706 Commerce St	(214) 653-1900	Dallas	TX
1114	Caesar's Frozen Custard	6930 N Mesa St	(915) 833-4100	El Paso	TX
1115	Cristina's	6951 N Mesa St	(915) 585-6994	El Paso	TX
1116	Johnnie's Charcoal Broiler	5151 Fairbanks Dr # B	(915) 757-9000	El Paso	TX
1117	Aibonitos Restaurant	5480 Fm 1960 Rd W	(281) 893-2281	Houston	TX
1118	Mayflower Cuisinier	14025 Memorial Dr	(281) 293-7676	Houston	TX
1119	Peter's Pub	14715 Hempstead Rd	(713) 896-4000	Houston	TX
1120	El Burrito	4109 Fredericksburg Rd	(210) 732-3571	San Antonio	TX

D1_CALENDAR2

YYYYMMDD	FULL_GREGORIAN_DTE	MONTH_IN_YEAR	DAY_IN_MONTH	DAY_NAME	DAY_IN_YEAR
20080406	06-APR-08	4	6	Monday	96
20080407	07-APR-08	4	7	Tuesday	97
20080408	08-APR-08	4	8	Wednesday	98
20080409	09-APR-08	4	9	Thursday	99
20080410	10-APR-08	4	10	Friday	100
20080411	11-APR-08	4	11	Saturday	101
20080412	12-APR-08	4	12	Sunday	102
20080413	13-APR-08	4	13	Monday	103
20080414	14-APR-08	4	14	Tuesday	104
20080415	15-APR-08	4	15	Wednesday	105

D1_ORDERS2

PRODKEY	PERIODKEY	INVNBR	CUSTKEY	UNITSHPD	UNITORDD	DOLLARS	NETWGHTSHPD
1009	20080105	1694415	1052	14	14	349.14	378
1143	20080105	1694416	1052	21	21	462.5	233
1126	20080105	1694417	1052	33	33	465.81	211.27
1097	20080105	1694417	1052	6	6	60.2	150
1107	20080105	1694417	1052	1	1	24.36	14.25
1108	20080105	1694417	1052	24	24	872.92	502.48
1118	20080105	1694417	1052	31	31	395.26	530.55
1040	20080105	1694417	1052	100	100	1607	2945.55
1042	20080105	1694417	1052	44	50	549.87	1077
1043	20080105	1694417	1052	14	14	209.87	401

Training Objective

The primary objective of this training is to build the metadata and administer Oracle BI Server to support the business requirements of ABC. The metadata will allow ABC employees to build interactive reports and dashboards that they can use to better analyze, monitor, and manage their business, and improve overall customer satisfaction.

The recommended strategy for building metadata is to use an iterative approach. You begin by building a relatively simple repository:

- Minimize the number of source tables.
- Expose only stored measures with simple aggregation rules.
- Use the query log to check query results.
- Create presentation objects and test with Oracle BI Analysis Editor.

After the initial repository is built and tested, you expand the business model:

- Import additional physical tables needed to support the business model.
- Add calculated measures that involve operations on existing columns.
- Add more complex calculated measures (for example, level-based measures and share measures).
- Add time series calculations (for example, percentage change in a measure compared to that in the same period in the previous year).
- Add security information.
- Add aggregate table data sources to improve performance.
- Localize Oracle BI metadata and data.
- Configure many-to-many relationships.
- Configure implicit fact columns.

In the lessons that follow, you follow this strategy to build the business model to support the business requirements of ABC. At first, the repository metadata includes only the basic set of physical tables needed to support the initial business model. You add more tables and complexity in future iterations.

Practice 3-2: Gathering Information to Build an Initial Business Model

Goal

To analyze the business requirements to begin building the business model

Scenario

Before you begin building the business model, you need to gather and analyze the business requirements of the ABC company. In this practice, you use the information provided in the ABC document that you read in the previous practice to determine the structure of the initial business model.

Time

10 minutes

Tasks:

Use the information in the ABC document to help you determine the following information, which you will need to implement the initial business model.

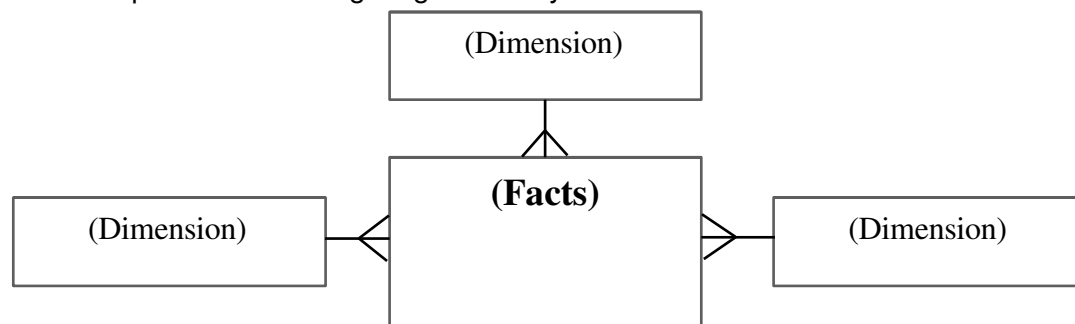
1. What measures (facts) does ABC want to report on?

2. What hierarchies can you identify?

3. Against which attributes (dimensions) does ABC want to analyze its facts?

4. By which geographical attributes does ABC want to analyze data?

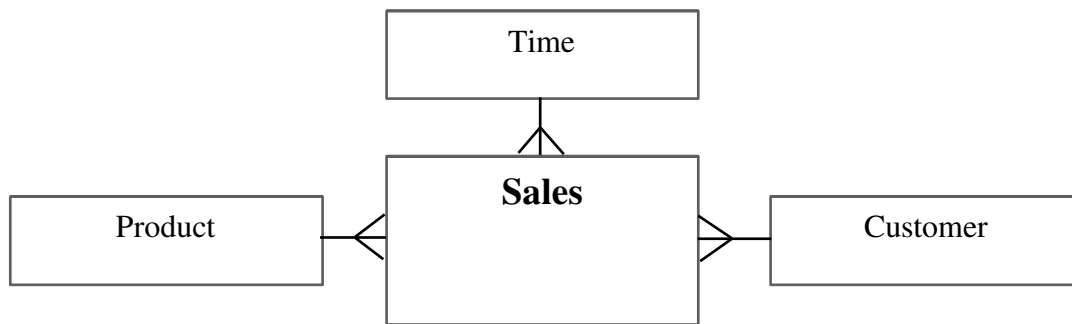
5. Complete the following diagram with your identified facts and dimensions:



Solutions 3-2: Gathering Information to Build an Initial Business Model

Answers

1. What measures (facts) does ABC want to report on?
Dollar sales, units ordered, units shipped, net weight shipped
2. What hierarchies can you identify?
Product, Time, Customer
3. Against which attributes (dimensions) does ABC want to analyze its facts?
Product, Time, Customer
4. By which geographical attributes does ABC want to analyze data?
Region, city, state
5. Complete the following diagram with your identified facts and dimensions:



Practice 3-3: Creating a Repository and Importing a Data Source

Goal

To create a new repository and import the table schema from an external data source

Scenario

You use the Import Wizard of the Administration Tool to create a new repository and import tables from the SUPPLIER2 schema into the Physical layer of the repository.

Outcome


You have a new repository file, ABC.rpd, which contains the D1_CALENDAR2, D1_CUSTOMER2, D1_ORDERS2, and D1_PRODUCTS tables in the Physical layer.

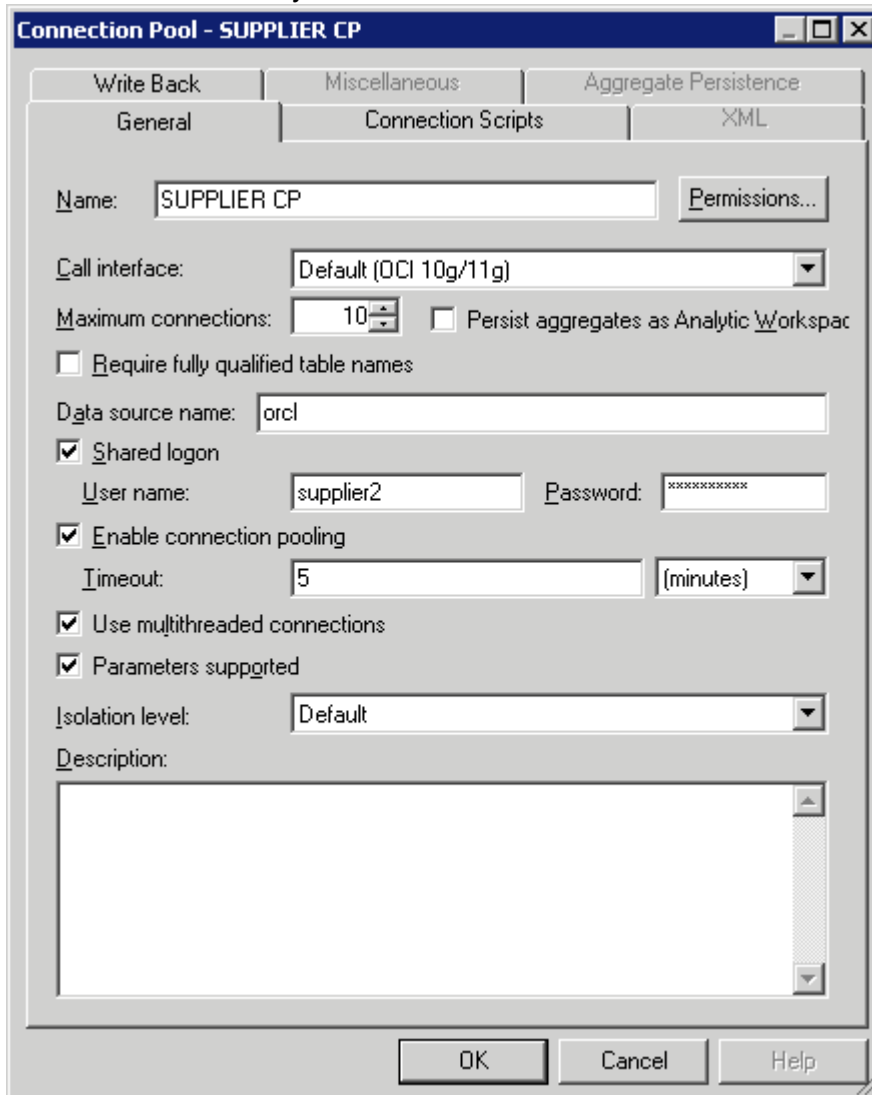
Time

15 minutes

Tasks

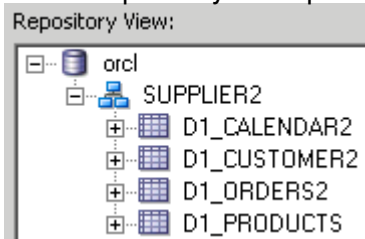
1. Use the Import Wizard to create a new ABC repository file.
 - a. Return to the Administration Tool, which should still be open. If not, select **Start > Programs > Oracle Business Intelligence > Administration**.
 - b. Select **File > New Repository** to open the Import Wizard. The Import Wizard guides you through the steps to create a new repository and import metadata.
 - c. Enter **ABC** in the Name field.
 - d. In the **Location** field, accept the default location to store the repository.
 - e. For **Import Metadata**, accept the default selection: **Yes**. When Yes is selected, the Import Wizard continues with windows for importing metadata. When No is selected, an empty repository is saved to the selected location.
 - f. Enter **welcome1** as the repository password.
 - g. Retype the password.
 - h. Click **Next** to open the new ABC repository and the Select Data Source window.
2. Use the Import Wizard to import metadata.
 - a. Select **OCI 10g/11g OR ODBC 3.5** from the Connection Type drop-down list. The window displays connection fields based on the connection type that you selected.
 - b. Enter the following connection information:
Data Source Name: **orcl**
User Name: **supplier2**
Password: **supplier2**
 - c. Click **Next** to continue to the Select Metadata Types window.
 - d. In the Select Metadata Types window, accept the defaults: **Tables, Keys, and Foreign Keys**. The check boxes allow you to select the information to import. You can import tables, keys, foreign keys, system tables, aliases, synonyms, and views. As a general rule, import only those objects that are needed to support your business model. If you do import extra objects at this point, you can delete them later if you determine that they do not support your business model.
 - e. Click **Next** to open the Select Metadata Objects window.
 - f. In the Data source view pane, expand the **SUPPLIER2** schema folder.

- g. Scroll to view all the tables in the SUPPLIER2 schema.
- h. Select the **D1_CALENDAR2** table. This automatically deselects any higher-level container objects in the tree.
- i. Press and hold **Ctrl** and select the remaining three tables to import to the Physical layer to build ABC's initial business model: **D1_CUSTOMER2**, **D1_ORDERS2**, and **D1_PRODUCTS**.
- j. Scroll to ensure that no higher-level container objects are selected, and that only the four tables for import are selected.
- k. Click the **Import Selected** button  to move the metadata objects to the Repository View pane. The Connection Pool dialog box opens automatically.
- l. Change the Connection Pool name to **SUPPLIER CP**.
- m. Ensure that the call interface is set to **Default (OCI 10g/11g)** or **ODBC 3.5**.
- n. Ensure that the data source name is **orcl**. The username and password fields are automatically populated with **supplier2**. The data source name, orcl, is the same as the tnsnames.ora entry for this Oracle database instance.

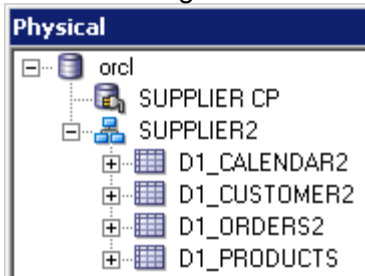


The image shows the 'Connection Pool - SUPPLIER CP' dialog box. It has tabs for 'Write Back', 'Miscellaneous', and 'Aggregate Persistence'. The 'General' tab is selected. The 'Name' field is 'SUPPLIER CP'. The 'Call interface' is 'Default (OCI 10g/11g)'. The 'Maximum connections' is '10'. The 'Persist aggregates as Analytic Workspac' checkbox is unchecked. The 'Require fully qualified table names' checkbox is unchecked. The 'Data source name' is 'orcl'. The 'Shared logon' checkbox is checked. The 'User name' is 'supplier2' and the 'Password' is '*****'. The 'Enable connection pooling' checkbox is checked. The 'Timeout' is '5' minutes. The 'Use multithreaded connections' checkbox is checked. The 'Parameters supported' checkbox is checked. The 'Isolation level' is 'Default'. The 'Description' field is empty. The 'OK', 'Cancel', and 'Help' buttons are at the bottom.

- o. Click **OK**. The Connection Pool dialog box closes and the metadata objects are visible in the Repository View pane. Expand SUPPLIER2 to see the objects.



- p. Click **Finish** to complete the import process. The ABC repository opens.
- q. Notice that the metadata is imported into the Physical layer of the ABC repository and the orcl database object appears in the Physical layer. Recall that every repository contains three layers. The Physical layer is where information about the physical data sources is stored. The Business Model and Mapping layer is where measurements and terms used in business are mapped to the physical data sources. The Presentation layer is where the business model is customized for presentation to the user. You can work on each layer at any stage in creating a repository, but the typical order is to create the Physical layer first, then the Business Model and Mapping layer, and then the Presentation layer.
- r. To display the tables, expand the **orcl** database object and then expand the **SUPPLIER2** schema folder.
- s. Ensure that the four tables are imported successfully and that the connection pool name is changed to SUPPLIER CP.

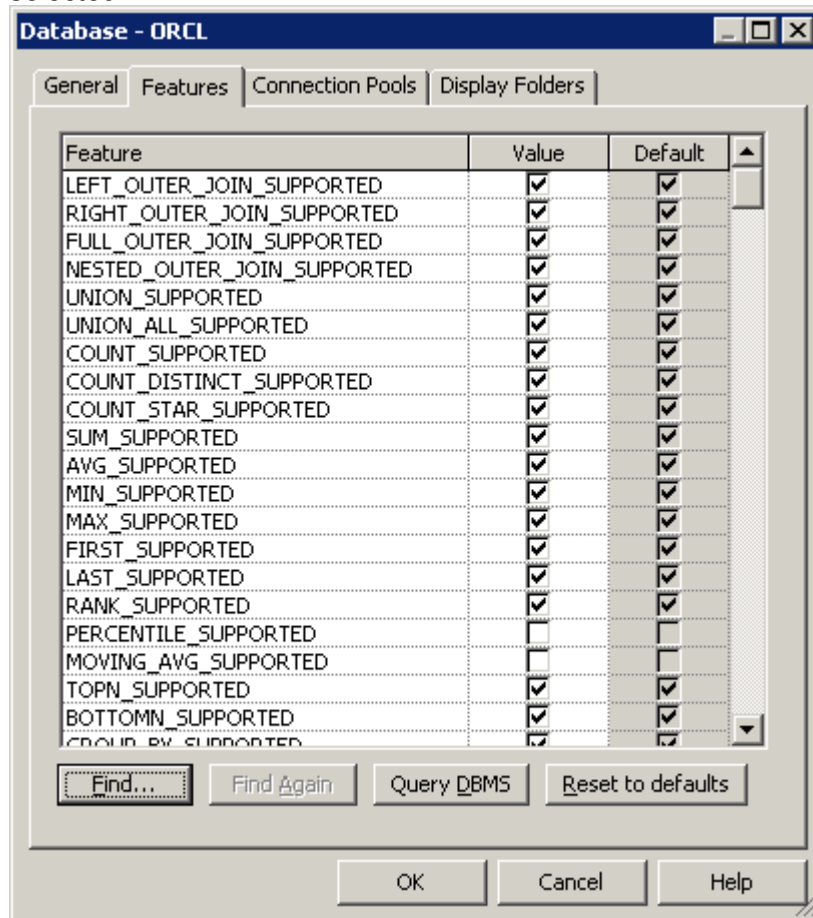


Recall that D1_CALENDAR2 contains time information, D1_CUSTOMER2 contains customer information, and D1_PRODUCTS contains product information for the business model. D1_ORDERS2 contains the invoice-level information needed to create the measures in the business model. D1_CALENDAR2, D1_CUSTOMER2, and D1_PRODUCTS are the dimension tables and D1_ORDERS2 is a fact table. In the next practice, you create keys and joins so that the four tables form a star schema.

3. Examine the connection pool for the orcl data source.
 - a. Double-click the **SUPPLIER CP** connection pool object. Recall that you provided this name for the connection pool during the import process. Connection pools regulate access to the data source. Every data source must have at least one connection pool. A connection pool provides connections for multiple concurrent data source requests (queries), reducing the immediate overhead of connecting to a data source. Connection pools automatically queue connection requests when they exceed connection pool limits. You can create more than one connection pool for a single data source to give certain users more immediate access to data over others.
 - b. Call interface is the application program interface with which to access the data source; Oracle Call Interface (OCI) in this example.
 - c. Maximum connections is the maximum number of connections allowed for this connection pool. The default is 10. Each connection consumes about 1 MB of memory.

- d. Data source name is configured to access the database to which you want to connect. This value is set automatically when you import the tables to the physical layer.
 - e. Username and password are also configured automatically during import depending on the parameters set for the data source.
 - f. Enable connection pooling allows a single database connection to remain open for a specified time (in minutes) for use by future query requests. Connection pooling saves the overhead of opening and closing a new connection for every query. If you do not select this option, each query sent to the database opens a new connection.
 - g. For more information about connection pool parameters, consult the *System Administrator's Guide for Oracle Business Intelligence Enterprise Edition* or click the Help button.
 - h. Click **OK** to close the Connection Pool dialog box.
4. Examine the properties of the database object in the Physical layer.
- a. Right-click the **orcl** database object and select **Properties**. You can also double-click the object.
 - b. Click the **General** tab, if not selected by default. This tab provides general information about the data source, such as the database name and database type. Notice that the database type is set to **Oracle 11g/Exadata**.
 - c. Click the **Features** tab. This tab lists features that, when selected or deselected, determine the SQL that Oracle BI Server will issue for this database. This features table is set to the database's default values during the schema import process. You can turn off any of these features if there is a reason to do so. Oracle BI Server will adjust the SQL that it sends to the database accordingly and will compensate for the deselected features with its own functionality. Notice that turning a feature on when the default is "off" may or may not cause that feature to be used in the generated SQL. To use a feature, Oracle BI Server needs to know how the feature is implemented in that database platform. If it does not know, it will not use it, even though the feature is

selected.



- d. Click the **Find** button.
- e. In the Find field, enter **INTERSECT** and click **OK**.
- f. Notice that the **INTERSECT_SUPPORTED** feature is supported on this database platform.
- g. Click the **Connection Pools** tab. This tab displays all connection pools associated with this data source. In this example, there is only one connection pool, SUPPLIER CP.
- h. Click the **Display Folders** tab. You could use this tab to create folders to organize the information in the Physical layer.
- i. Click **Cancel** to close the Database properties dialog box.
5. Examine the properties of a physical table object in the Physical layer.
 - a. Right-click **D1_CALENDAR2** and select **Properties**. The Physical Table properties dialog box opens.
 - b. Click the **General** tab.
 - c. Notice that it is possible to rename the object by using the Name field. Do not rename the object now.
 - d. Notice that the Cacheable check box is selected by default. This determines that queries that hit this table will be cached. You could also select Cache persistence time and use the field and drop-down list to determine the cache persistence time. This determines how long the cached queries that include this table as a source should be used to provide information to users. Ensure that the default, **Cache never expires**, is selected. You learn more about caching in the lesson titled "Cache Management."

- e. Notice that you can select a table type from the Table Type drop-down list. Physical Table is the default, and means that the object represents a physical table. You can also select Stored Proc or Select. Stored Proc is used to call a stored procedure. Select can be used when you want to create a SQL statement to represent a physical table. Leave the table type set to Physical Table.
 - f. Notice that the Hint box is editable. This box is editable only when the database is Oracle. A hint specified here is included in all SQL that references this table.
 - g. Notice that it is possible to add a description of the physical object.
 - h. Click the **Columns** tab.
 - i. Notice that all columns and the corresponding column properties are listed on this tab. It is also possible to add new columns or delete existing columns.
 - j. Select the **Keys** and **Foreign Keys** tabs. Notice that no foreign keys are created yet for this table. You learn how to create foreign keys later in this practice.
 - k. Click **Cancel** to close the Physical Table properties dialog box.
6. Examine the properties of a physical column object in the Physical layer.
- a. Expand the **D1_CALENDAR2** table object.
 - b. Double-click any column to open the Physical Column properties dialog box. The physical column properties include the data type of the columns and whether the column is nullable. (The column can contain NULL values.) The Server Administration Tool automatically selects compatible data types based on the data types of the source database.
 - c. Click **Cancel** to close the Physical Column properties dialog box.
7. Update row counts and view data. It is a good idea to update row counts or view data after an import to verify connectivity. Viewing data or updating row count, if successful, tells you that you have everything configured correctly.
- a. Update the row count for all tables by selecting **Tools > Update All Row Counts**. This may take a moment. In this practice, you imported a small number of tables. Note that updating row counts for all tables can take a long time if you have imported many large tables. It is also possible to update row count for a single physical layer object by right-clicking the object and selecting Update Row Count.
 - b. When Update All Row Counts completes, move the cursor over the tables and columns and observe that row count information is now visible, including when the row count was last updated.
 - c. Right-click any table and select **View Data** to view the data for the table.
 - d. Close the View Data dialog box.
8. Save the repository.
- a. Select **File > Save** or click the **Save** button on the toolbar. If the toolbar is not visible, select **Tools > Options > General > Show Toolbar**.
 - b. Click **No** when prompted to check Global Consistency. 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. Because these layers are not defined yet, bypass this check until the other layers in the repository are built.
 - c. Leave the repository open and remain logged in to the Administration Tool for the next practice.

Practice 3-4: Creating Alias Tables

Goal

To assign aliases to physical tables before mapping them to the Business Model and Mapping layer

Scenario

You create aliases for the metadata objects that you imported into the Physical layer of the repository. It is recommended that you use table aliases frequently in the Physical layer to eliminate extraneous joins and to include best practice naming conventions for physical table names.

Outcome

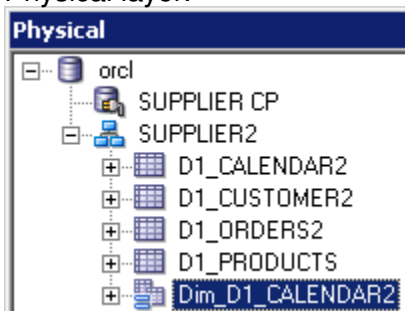
New alias tables for the D1_CALENDAR2, D1_CUSTOMER2, D1_ORDERS2, and D1_PRODUCTS tables

Time

20 minutes

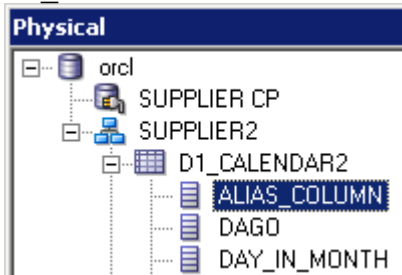
Tasks

1. Create an alias for the D1_CALENDAR2 table.
 - a. Right-click **D1_CALENDAR2** and select **New Object > Alias**. The Alias Physical Dialog box opens.
 - b. Name the alias **Dim_D1_CALENDAR2**. This is a simple naming convention that identifies the table as a dimension table and includes the original table name.
 - c. Click the **Columns** tab. Notice that alias tables inherit all column definitions from the source table.
 - d. Double-click any of the columns listed to open the Physical Column properties dialog box. Notice that the column is read-only and cannot be modified.
 - e. Click **Cancel** to close the Physical Column dialog box.
 - f. Click **OK** to close the Physical Table dialog box. The alias table is added to the Physical layer.

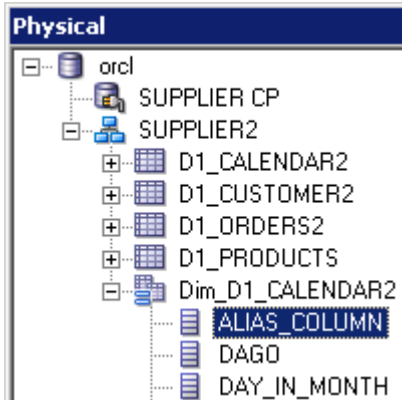


2. Create a new column in the source table and observe the results.
 - a. Right-click the **D1_CALENDAR2** source table and select **New Object > Physical Column**. The Physical Column dialog box opens.
 - b. Name the column **ALIAS_COLUMN**.

- c. Leave the type set to **UNKNOWN** and click **OK**. The column is added to the D1_CALENDAR2 source table.



- d. Expand the **Dim_D1_CALENDAR2** alias table and ensure that the column was automatically added. Creating a new column in a source table automatically creates the same column in all its alias tables.



- e. Double-click **ALIAS_COLUMN** in the Dim_D1_CALENDAR2 alias table. The Physical Column dialog box opens.
 - f. Confirm that the type is **UNKNOWN**.
 - g. Click **Cancel**.
 - h. Return to the D1_CALENDAR2 table and double-click **ALIAS_COLUMN**.
 - i. Change the type to **VARCHAR**.
 - j. Click **OK**.
 - k. Return to the Dim_D1_CALENDAR2 alias table and double-click **ALIAS_COLUMN**.
 - l. Confirm that the type is changed to **VARCHAR**. Modification of a source column forces the same changes to be reflected in the alias columns.
 - m. Click **OK**.
 - n. Return to the D1_CALENDAR2 table, right-click **ALIAS_COLUMN** and select **Delete**.
 - o. Click **Yes** to confirm the deletion.
 - p. Return to the Dim_D1_CALENDAR2 alias table and confirm that ALIAS_COLUMN is deleted. Deletion of a source column automatically deletes the corresponding alias columns.
3. Modify the data types for columns in D1_CALENDAR2 to improve readability of data in Oracle BI analyses.
 - a. Expand **D1_CALENDAR2**.
 - b. Double-click the **YYYYMMDD** column to open the Physical Column dialog box.
 - c. Change the data type from DOUBLE to **INT**.
 - d. Click **OK** to close the Physical Column dialog box.

- e. Repeat and change the data type from DOUBLE to INT for the following D1_CALENDAR2 columns:

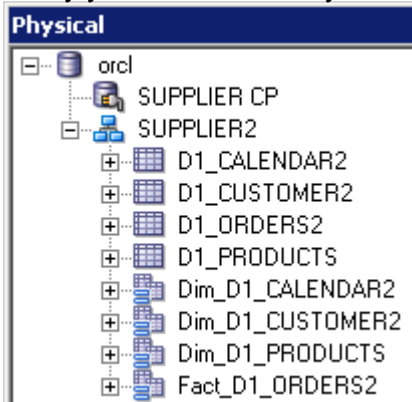
YEAR

MONTHCODE

4. Create alias tables for the three remaining physical tables:

Original Table Name	Alias Table Name
D1_CUSTOMER2	Dim_D1_CUSTOMER2
D1_PRODUCTS	Dim_D1_PRODUCTS
D1_ORDERS2	Fact_D1_ORDERS2

5. Verify your work. The Physical layer should look similar to the following screenshot:



6. Click the **Save** button to save the ABC repository.
7. Click **No** when prompted to check global consistency.
8. Leave the Administration Tool open for the next practice.

Practice 3-5: Defining Keys and Joins

Goal

To define the primary keys, foreign keys, and joins in the Physical layer

Scenario

You have created a new repository, imported the initial tables from the SUPPLIER2 schema into the Physical layer of the repository, and created alias tables. Now you define keys and joins in the Physical Layer of the repository. If the imported database already had primary key-foreign key relationships defined and the primary keys and foreign keys were imported into the repository, then the join conditions would be set up automatically. But that is not always what you want, because foreign key relationships are set in a database for only one purpose, referential integrity, which may not correspond to the purpose of the Administration Tool and BI Server, which is knowing which joins to include in SQL queries. In the SUPPLIER2 schema, primary keys, foreign keys, and joins are not defined and were not imported into the repository. Therefore, you need to define the keys and join conditions manually. You can create physical keys and joins by using either the Physical Diagram or the Joins Manager.



Outcome

Keys and joins are defined on the physical tables.

Time

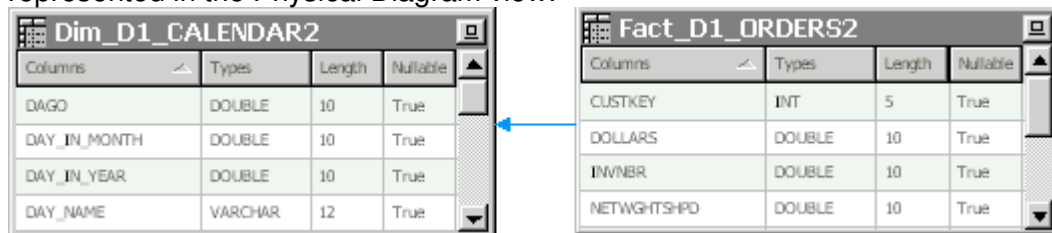
20 minutes

Tasks

1. In this step, you define joins and keys by using the Physical Diagram feature of the Administration Tool.
 - a. Expand the **orcl** database object so that all physical objects are visible.
 - b. Select all four alias tables.
 - c. Right-click one of the four highlighted alias tables and select **Physical Diagram > Object(s) and All Joins** to open the Physical Diagram dialog box. Alternatively, you can click the Physical Diagram button  on the toolbar.
 - d. Rearrange the alias table objects so that they are all visible.
 - e. Click the **New Join** button  on the toolbar.
 - f. Click the **Fact_D1_ORDERS2** table, and then click the **Dim_D1_CALENDAR2** table. The Physical Foreign Key dialog box opens. Please be aware of the following upgrade considerations for Oracle BI EE 11g Release 1 (11.1.1.5): Joins in the Physical and Business Model Diagrams are represented by a line with an arrow at the "one" end of the join, rather than the line with crow's feet at the "many" end of the join that was used in previous releases. When creating joins in the Physical and Business Model Diagrams, you now select the "many" end of the join first, and then select the "one" end of the join. In previous releases, joins in the diagrams were created by selecting the "one" end of the join first.
 - g. Select the columns that join the tables. Select the **Dim_D1_CALENDAR2.YYYYMMDD** column, and then select **Fact_D1_ORDERS2.PERIODKEY**. Ensure that the Expression edit box (at the bottom) contains the following expression:

"orcl".""."SUPPLIER2"."Dim_D1_CALENDAR2"."YYYYMMDD" =
 "orcl".""."SUPPLIER2"."Fact_D1_ORDERS2"."PERIODKEY"

- h. Click **OK**.
- i. Observe the 1:N relationship between Dim_D1_CALENDAR2 and Fact_D1_ORDERS2 represented in the Physical Diagram view:



- j. Repeat the process you to create joins and keys for the other tables. Create the joins using the following expressions as a guide:

"orcl".""."SUPPLIER2"."Dim_D1_CUSTOMER2"."NEWKEY" =
 "orcl".""."SUPPLIER2"."Fact_D1_ORDERS2"."CUSTKEY"

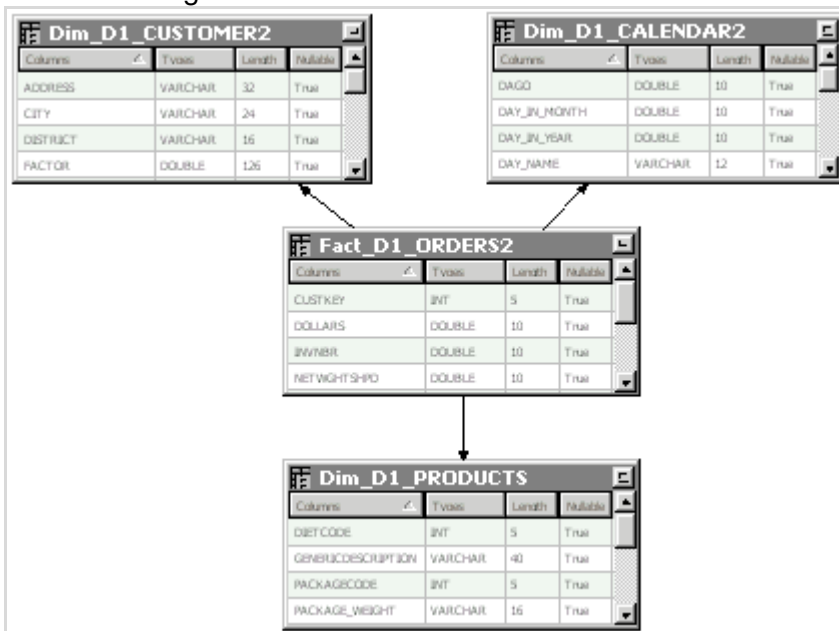
"orcl".""."SUPPLIER2"."Dim_D1_PRODUCTS"."PRODUCTKEY" =
 "orcl".""."SUPPLIER2"."Fact_D1_ORDERS2"."PRODKEY"

- k. You may want to adjust the objects in the Physical Diagram. If so, use the toolbar buttons to zoom in, zoom out, fit the diagram, collapse or expand objects, select

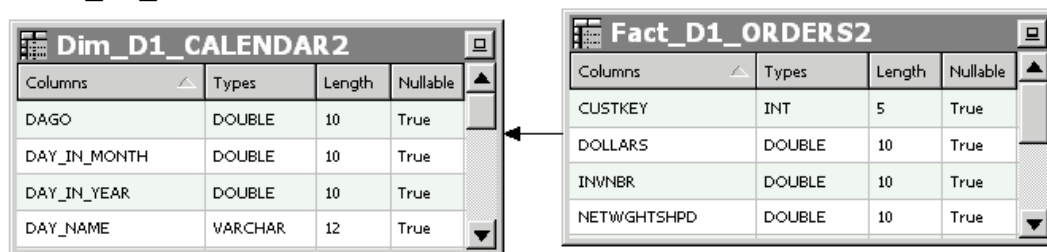
objects, and so forth:



- I. Observe the Physical Diagram and check your work. Your diagram should look similar to the following screenshot:



- m. Close the **Physical Diagram** window.
2. Observe additional options for viewing the physical table diagram.
 - a. Right-click the **Dim_D1_CALENDAR2** table and select **Physical Diagram > Object(s) and Direct Joins**.
 - b. Observe the Physical Diagram. The diagram shows only those objects with direct joins to Dim_D1_CALENDAR2.



- c. Close the **Physical Diagram** window.
- d. Repeat this process for the Fact_D1_ORDERS2 table by selecting **Physical Diagram > Object(s) and Direct Joins** for the Fact_D1_ORDERS2 table. Notice that all the tables are visible in the diagram because all the tables have a direct join relationship with Fact_D1_ORDERS2.
- e. Close the Physical Diagram window.
3. Observe the changes to the physical table properties.
 - a. Expand **Dim_D1_CALENDAR2**. Notice that the YYYYMMDD column now has a key icon. The key was defined when you created the join in the Physical Diagram. Similar keys are defined for the Dim_D1_CUSTOMER2 and Dim_D1_PRODUCTS tables.
 - b. Double-click the **Fact_D1_ORDERS2** table to open the Physical Table properties dialog box.

- c. Click the **Foreign Keys** tab. Notice that the foreign key information is visible. This information was created automatically when you created the join expressions in the Physical Diagram.
- d. Double-click any of the foreign key expressions and notice that the Physical Foreign Key dialog box opens, displaying the join information.
- e. Click **Cancel** to close the Physical Foreign Key dialog box.
- f. Click **Cancel** to close the Physical Table properties dialog box.
4. Explore the Joins Manager, which allows you to examine, edit, and delete all the joins, both physical and logical, in a repository.
 - a. Select **Manage > Joins**. The Joins Manager opens. The joins displayed in the right pane vary depending on the leaf that you select in the left pane. You can view all joins in the repository, in a particular business model, in the Business Model and Mapping layer, in the Physical layer, in the Business Model and Mapping layer for a particular business model, and in the Physical layer for a particular business model. Joins are further divided into logical foreign key, logical join, physical foreign key, and complex join.
 - b. In the left pane, select Business Model and Mapping. Notice that no joins are displayed, because you have not yet created any logical joins in the Business Model and Mapping layer. You do that later in the practices for Lesson 4.
 - c. In the left pane, select **Physical > Physical Foreign Key** to see all physical foreign key joins in the Physical layer. These are the joins that you created earlier in this practice. The Joins Manager displays the join name, the tables in the join, and the join expression.
 - d. Double-click any of the foreign key expressions and notice that the Physical Foreign Key dialog box opens, displaying the join information. Alternatively, you can right-click any join in the list and select **Properties** to open the Physical Foreign Key dialog box. You can edit the join properties by using this dialog box.
 - e. Click **Cancel** to close the Physical Foreign Key dialog box.
 - f. Click any column headings in the right pane to sort the joins by that column.
 - g. Select **Action > New**. Notice that you can create new joins by using the Joins Manager. However, most users tend to create joins with the physical or logical diagrams, as you do in this course.
 - h. Select **Action > Close** to close the Joins Manager.
5. Click the **Save** button to save the ABC repository.
6. Click **No** when prompted to check global consistency.
7. Leave the repository and the Administration Tool open for the next practice.

Congratulations! You have successfully created a new repository, imported a table schema from an external data source into the Physical layer, and defined keys and joins.

Practices for Lesson 4: Building the Business Model and Mapping Layer of a Repository

Chapter 4

Overview of Practices for Lesson 4

Lesson Overview

In these practices, you will create the Business Model and Mapping layer of a repository.

Practice 4-1: Creating the Business Model

Goal

To create a business model in the Business Model and Mapping layer of the repository

Scenario

In the previous practice, you created the Physical layer of the repository. You are now ready to begin building the business model in the Business Model and Mapping layer of the repository. The Business Model and Mapping layer of the Administration Tool defines the business model of the data and specifies the mapping between the business model and the Physical layer schemas. Business models are also referred to as logical models or dimensional models.

Business models are always dimensional, unlike objects in the Physical layer, which reflect the organization of the data sources. The Business Model and Mapping layer can contain one or more business models. Each business model contains logical tables, columns, and joins.

There are two main categories of logical tables: fact and dimension. Logical fact tables contain the measures by which ABC gauges its business operations and performance. Logical dimension tables contain the data used to qualify the facts. This practice assumes that a business model has already been designed on paper. You know what measures are important to ABC, what ABC employees compare measures to, and how the company likes to analyze its data. The goal of this practice is to capture this information in a business model in the Business Model and Mapping layer of the repository.

Outcome

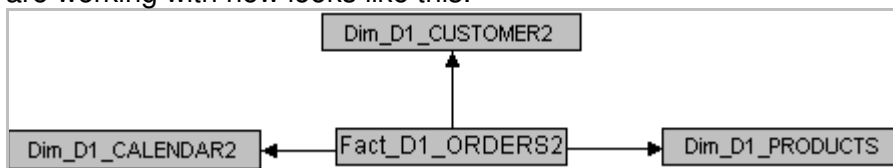
In the Business Model and Mapping layer, the SupplierSales business model is created with the following logical tables: Dim-Time, Dim-Customer, Dim-Product, and Fact-Sales.

Time

20 minutes

Tasks

The business model that ABC has defined is based on Sales, Product, Customer, and Time data. These data elements will be used to model the initial application. The physical model you are working with now looks like this:



1. Return to the ABC repository, which should still be open in offline mode. If the repository is not open, follow these steps to open it.
 - a. Select **Start > Programs > Oracle Business Intelligence > BI Administration**.
 - b. Select **File > Open > Offline**.
 - c. Double-click **ABC.rpd** to open the repository file.
 - d. Enter **welcome1** as the repository password.
2. Create a business model in the Business Model and Mapping layer.

- a. Right-click the white space of the Business Model and Mapping layer and select **New Business Model**.
- b. In the Name field, enter **SupplierSales**.
- c. Click **OK**. The new Business Model and Mapping folder appears in the Business Model and Mapping layer. The red symbol on the business model indicates that it is not yet enabled for querying. You enable the business model for querying after the Presentation layer is defined and the repository passes a global consistency check.

3. Create the logical tables.

- a. In the Physical layer, expand the **orcl** database object and expand the **SUPPLIER2** schema.
- b. Drag the following four alias tables simultaneously from the Physical layer onto the SupplierSales business model.

Dim_D1_CALENDAR2

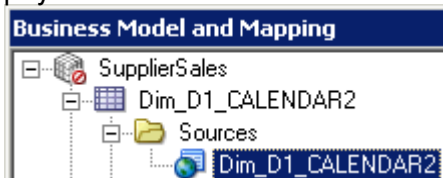
Dim_D1_CUSTOMER2

Dim_D1_PRODUCTS

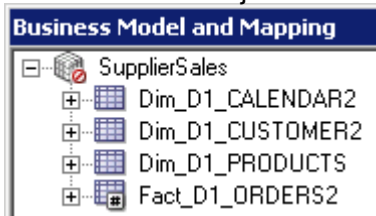
Fact_D1_ORDERS2

This action creates logical tables in the business model with logical columns.

- c. Expand the logical tables and notice that a Sources folder is created for each logical table. In each Sources folder, there is a logical table source. For example, the logical table source for the Dim_D1_CALENDAR2 logical table is the Dim_D1_CALENDAR2 physical table.



- d. Notice also that logical tables have different table icons. In the Business Model and Mapping layer, a hash sign (#) indicates a fact table. The logical joins are automatically inherited from the joins defined in the Physical layer.



4. Rename the logical tables in the business model to make them more meaningful.
 - a. Double-click the **Dim_D1_CALENDAR2** table in the SupplierSales business model. The Logical Table dialog box opens.
 - b. If necessary, click the **General** tab.
 - c. In the Name field, enter **Dim-Time**.
 - d. Click **OK**.
 - e. Repeat the process and rename the following tables. Alternative methods for renaming include right-clicking an object and selecting **Rename**, or clicking an object twice to make it editable. Notice that a logical table name is purely a business model artifact. Logical table names are not necessarily exposed to users.

Table	Rename to:
Dim_D1_CUSTOMER2	Dim-Customer
Dim_D1_PRODUCTS	Dim-Product
Fact_D1_ORDERS2	Fact-Sales

5. Delete the columns that are not needed for analysis.
 - a. Expand the **Fact-Sales** table and delete the following columns:

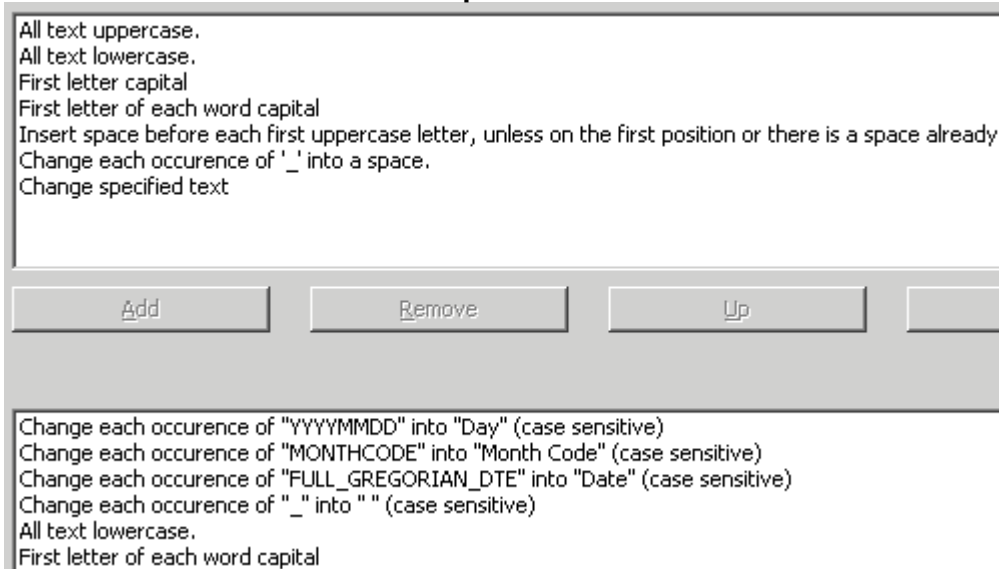
Columns
CUSTKEY
INVNBR
PERIODKEY
PRODKEY

- b. Expand the **Dim-Time** table and delete the following columns:

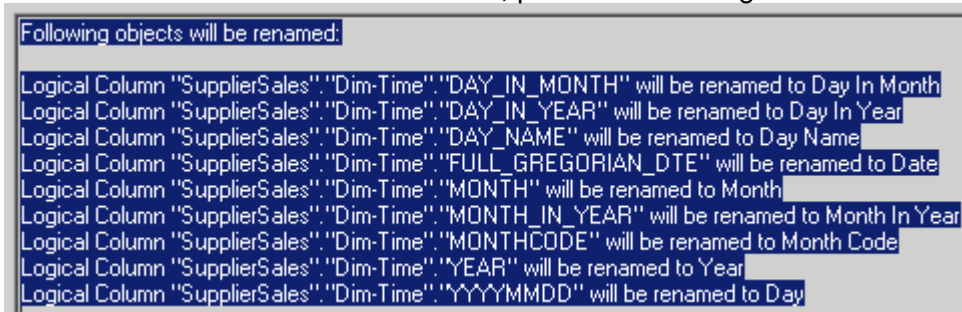
Columns
DAGO
MAGO
ORDERDAY_IN_YEAR
WEEK_NUM_IN_YEAR
YAGO

- c. Expand the **Dim-Product** table and delete the **PGCODE** column.
 - d. Expand the **Dim-Customer** table and delete the **REPNO** and **FACTOR** columns.
6. Use the Rename Wizard to rename logical columns in the Dim-Time logical table. You use the Rename Wizard utility for renaming Business Model or Presentation layer objects. You can use it to replace text strings, change all letters to lowercase, use uppercase for the first letter of words, and so on. You can preview the new names before committing the changes. It is primarily used on Business Model logical columns after importing physical objects into the middle layer. The names that you give logical columns in the business model can be exposed via the Presentation layer to end-user tools such as the Analysis Editor. However, it is possible to override logical column names in the Presentation layer, as you will see later in this course.
 - a. Select **Tools > Utilities > Rename Wizard** and click **Execute**.
 - b. At the bottom of the middle pane, click **Business Model and Mapping**.
 - c. In the middle pane, expand **SupplierSales > Dim-Time**.
 - d. Use Shift + click to select all columns and click **Add** to add the columns to the right pane.
 - e. Click **Next**. Notice that Logical Column is selected.
 - f. Click **Next** again.
 - g. Select **Change specified text**.
 - h. In the Find field, enter **YYYYMMDD**.
 - i. In the "Replace with field," enter **Day**.
 - j. Select **Case Sensitive**.
 - k. Click **Add**.

- l. Select **Change specified text**.
- m. In the Find field, enter **MONTHCODE**.
- n. In the "Replace with field," enter **Month Code**.
- o. Select **Case Sensitive**.
- p. Click **Add**.
- q. Select **Change specified text**.
- r. In the Find field, enter **FULL_GREGORIAN_DTE**.
- s. In the "Replace with field," enter **Date**.
- t. Select **Case Sensitive**.
- u. Click **Add**.
- v. Select **Change specified text**.
- w. In the Find field, enter an underscore.
- x. In the "Replace with field," enter a space.
- y. Click **Add**.
- z. Select **All text lowercase** and click **Add**.
- aa. Select **First letter of each word capital** and click **Add**.

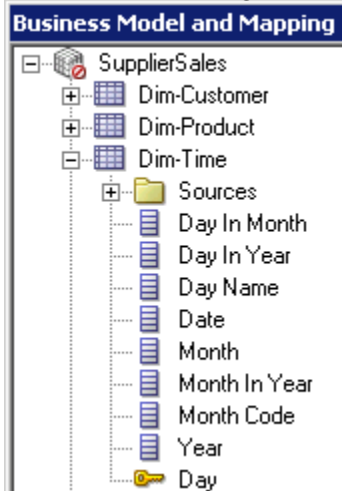


- bb. Click **Next**.
- cc. In the Rename Wizard – Finish window, preview the changes before committing.



- dd. Click **Finish**.

ee. Examine the changes in the repository and verify that they are as expected.



ff. Save the repository.

gg. Do not check global consistency.

7. Rename columns in the remaining logical tables according to the tables below. You can use the Rename Wizard or rename the columns manually.

Fact-Sales columns	Rename as:
DOLLARS	Dollars
NETWGHTSHPD	Net Weight Shipped
UNITORDD	Units Ordered
UNITSHPD	Units Shipped

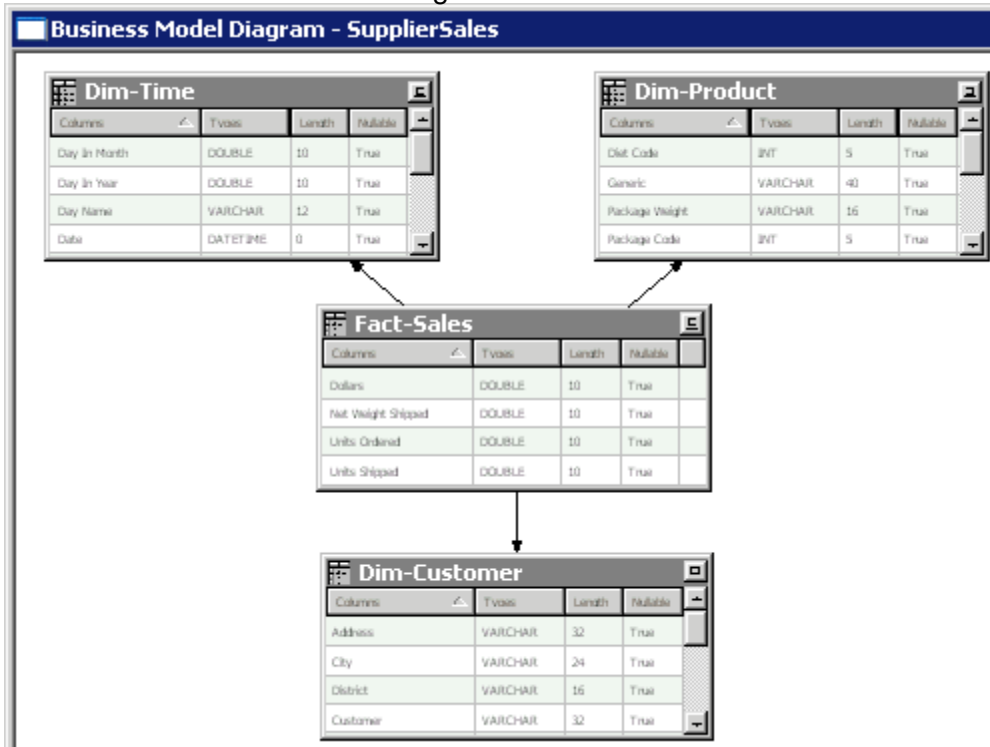
Dim-Customer columns	Rename as:
ADDRESS	Address
CITY	City
DISTRICT	District
NAME	Customer
NEWKEY	Customer Key
PHONE	Phone
REGION	Region
ROUTECD	Route Code
SALESREP	Sales Rep
STATE	State
ZIP_CODE	Zip Code

	Rename as:
DIETCODE	Diet Code
GENERICDESCRIPTION	Generic

PACKAGE_WEIGHT	Package Weight
PACKAGECODE	Package Code
PRODUCTKEY	Product Key
SPECIFICDESCRIPTIN	Specific
SUBTYPECODE	Subtype Code
SUPPLIERCODE	Supplier Code
TYPECODE	Type Code

8. In this step, you verify the logical table keys. For a business model to be valid, each logical dimension table must have a logical key. Logical keys can be composed of one or more logical columns. The logical key defines the lowest level (the most detailed level) of information of any source in the logical table. In this example, the logical table keys were built automatically when you dragged the tables from the Physical layer to the business model.
 - a. Double-click the **Dim-Customer** logical table in the business model to open the Logical Table dialog box.
 - b. Click the **Keys** tab.
 - c. Confirm that the Customer Key column is defined as the key for this table.
 - d. Click **OK** to close the Logical Table dialog box.
 - e. Repeat these steps and ensure that the Day column is defined as the logical table key for the Dim-Time table and Product Key is defined as the logical table key for the Dim-Product table.
9. In this step, you verify the logical table joins. Another requirement for a valid business model is that the logical tables must be joined via logical joins. Logical joins express the cardinality relationships between the logical tables. Logical fact tables are always at the “many” end of these cardinality relationships. Logical joins help Oracle BI Server understand the relationships between the various pieces of the business model. When a query is sent to Oracle BI Server, the server figures out how to construct physical queries by examining how the logical model is structured. Examining logical joins is an integral part of this process.
 - a. Right-click the **SupplierSales** business model and select **Business Model Diagram > Whole Diagram**. The Logical Table Diagram window opens.
 - b. Rearrange the table icons so that they are all visible. Right-click the white space and adjust the zoom factor, if desired. Recall that the logical joins are automatically inherited from the joins defined in the Physical layer. Later in this course you learn how to build logical joins using the Business Model Diagram. The Business Model Diagram

should look similar to the following screenshot.



- c. Double-click one of the join connectors to open the Logical Join dialog box.
 - d. Leave the default values as they are, but notice which properties you can set in the Logical Join dialog box: name, business model, tables, driving table, join type, and cardinality. Also, notice which properties you cannot set: the join expression and the join columns.
 - e. Do not change the default values. Click **OK**. Typically, when defining logical joins, you leave the defaults as they are and click OK.
 - f. Close the Business Model Diagram window. The join relationships determine which tables are the logical dimension tables and which is the logical fact table. Recall that a fact table is always on the "many" side of a logical join. You now have a logical star schema consisting of one logical fact table, Fact-Sales, and three logical dimension tables: Dim-Time, Dim-Product, and Dim-Customer.
10. Save the repository.
 11. Click **No** when prompted to check global consistency.
 12. Leave the repository open for the next practice.

Practice 4-2: Creating Simple Measures

Goal

To examine the logical-to-physical column mappings and to create simple measures

Scenario

The SupplierSales business model is now defined in the Business Model and Mapping layer. In this practice, you review the logical-to-physical table and column mappings to better understand the relationships that exist between logical tables and their logical table sources. You then create measures by setting aggregation rules for logical columns. Then you check the physical tables referenced by the business model.

Outcome

Measures defined in the Fact-Sales logical table

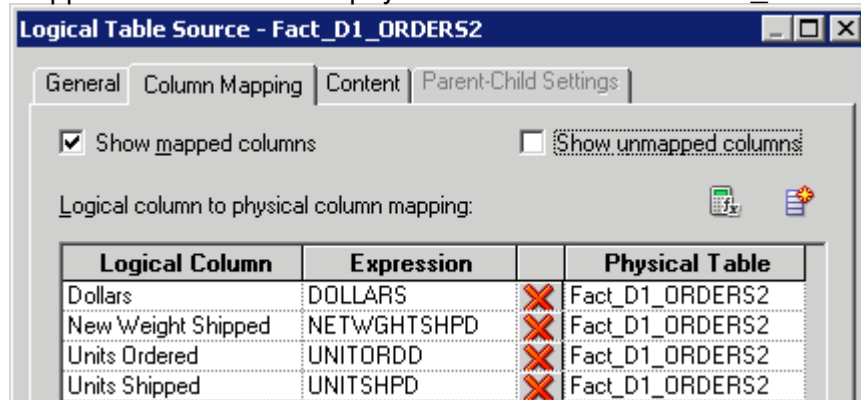
Time

5 minutes

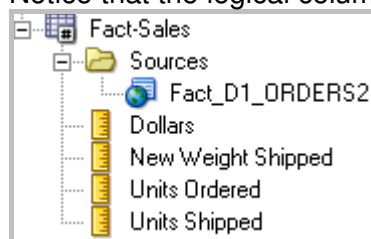
Tasks

1. In this step, you examine a logical table source (LTS). Each logical table in a business model has a subfolder called Sources that contains the logical table sources. Logical table sources contain the mappings from the logical columns in the Business Model and Mapping (BMM) layer to the physical columns in the Physical layer. Every logical column maps directly (or indirectly via another logical column) to a column or columns in the Physical layer.
 - a. Expand the **Sources** folder of the Fact-Sales logical table. Notice that the name of the logical table source for the Fact-Sales table is Fact_D1_ORDERS2. This logical table source was created automatically during the process of dragging the Fact_D1_ORDERS2 physical table to the business model.
 - b. Double-click the **Fact_D1_ORDERS2** logical table source to open the Logical Table Source dialog box.
 - c. Click the **General** tab. By default, the logical table source name corresponds to the name of the physical table that was dragged from the Physical layer, but this name could be changed to something more meaningful. For this exercise, leave the name as is. Notice also that the path to the physical table is identified in the “Map to these tables:” area.
 - d. Click the **Column Mapping** tab to review how the logical columns are mapped to the Physical layer. If necessary, adjust the column widths or drag the entire dialog box window to make it larger or smaller. You should see three column headings: Logical Column, Expression, and Physical Table. For example, the Dollars logical column is

mapped to the DOLLARS physical column in the Fact-D1_ORDERS2 physical table.



- e. The Content tab is discussed later in this course. Click **Cancel** to close the dialog box.
2. In this step, you create some measures by defining default aggregation rules on logical columns defined in the Fact-Sales table. Measures are typically data that is additive, such as total dollars or total quantities. The Fact-Sales logical fact table contains the measures in your business model. You aggregate some of its logical columns by summing the column data.
 - a. Double-click the **Dollars** logical column in the Fact-Sales table. The Logical Column dialog box opens.
 - b. Click the **Aggregation** tab.
 - c. Set the default aggregation rule for Dollars to **Sum**.
 - d. Click **OK**. Notice that the Dollars icon is changed to indicate that an aggregation rule is defined.
 - e. Use Ctrl + click to select **Net Weight Shipped**, **Units Ordered**, and **Units Shipped**.
 - f. Right-click and select **Set Aggregation**.
 - g. Set the default aggregation rule to **Sum**.
 - h. Click **OK**. You can use this method to set the same aggregation rule for multiple columns at once.
 - i. Notice that the logical column icons are changed for all four columns.



3. In this step, you check which physical tables are referenced by a business model. In most situations, there are tables that are included in the Physical layer, but not in the Business Model and Mapping layer. Only the tables referenced in the Business Model and Mapping layer, that is, only the tables included in logical table sources, are used in queries. All other physical tables are ignored when Oracle BI Server queries the physical database.
 - a. Select the **SupplierSales** business model.
 - b. Click the **Physical Diagram** icon on the toolbar.
 - c. The Physical Diagram displays all the physical tables currently referenced by the SupplierSales business model.

- d. Double-click the **connector** between the Dim_D1_CUSTOMER2 and the Fact_D1_ORDERS2 tables. Notice that the physical join is displayed. You can determine that this is a physical join because there is a foreign key join expression.
- e. Click **Cancel** to close the join dialog box.
- f. Close the Physical Diagram.
- g. Select **Tools > Options**.
4. Save the repository.
5. Do not check global consistency.
6. Leave the repository open for the next practice.

Congratulations! You have successfully built a business model in the Business Model and Mapping layer of a repository and created business measures.

Practices for Lesson 5: Building the Presentation Layer of a Repository

Chapter 5

Overview of Practices for Lesson 5

Lesson Overview

In these practices, you will create the Presentation layer of an Oracle Business Intelligence repository.

Practice 5-1: Creating the Presentation Layer

Goal

To create the Presentation layer of a repository

Scenario

You have created the initial SupplierSales business model in the repository. You now create the Presentation layer of the repository. The Presentation layer exposes the business model objects in the Oracle BI Analysis Editor so that users can build analyses to analyze their data.

Outcome

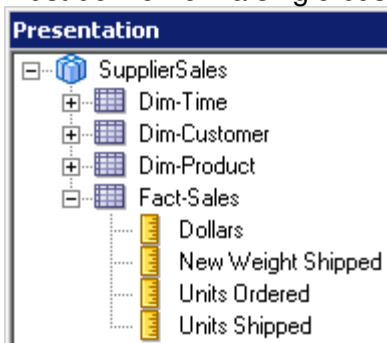
In the Presentation layer of the repository, there is a SupplierSales subject area.

Time

10 minutes

Tasks

1. In this step, you create the Presentation layer for the SupplierSales business model. The Presentation layer has three types of objects: subject area, presentation table, and presentation column. In the Oracle BI Analysis Editor the subject area appears as a subject area, the presentation table appears as a folder, and the presentation columns appear as columns in the folders. The ABC repository should still be open in the Administration Tool from the previous practice.
 - a. Ensure that the Presentation layer is visible. If it is not, select **View > Presentation**.
 - b. Drag the **SupplierSales** business model into the Presentation layer. A subject area appears with the name SupplierSales.
 - c. Expand **SupplierSales** in the Presentation layer. When you create presentation objects by dragging a business model to the Presentation layer, the business model becomes a subject area, the logical tables become presentation tables, and the logical columns become presentation columns. Notice that all objects within a subject area must derive from a single business model.

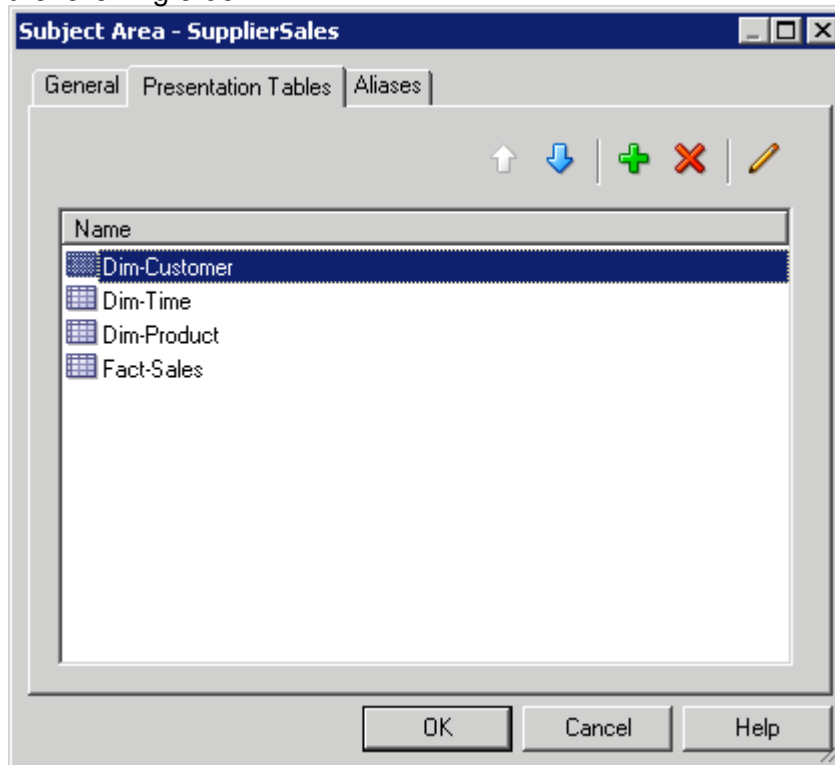


2. In this step, you examine the properties of a subject area.
 - a. Double-click **SupplierSales** in the Presentation layer to open the Subject Area dialog box.
 - b. Click the **General** tab. You use this tab to create or edit a subject area.
 - c. Notice that it is possible to change the name of the subject area. For the purpose of these lessons, leave the name as SupplierSales. This is the name that appears as a subject area in the Analysis Editor. Also, as you will see in later lessons, the name of

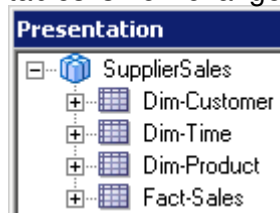
the subject area can be used in the FROM clause of a logical SQL statement. Therefore, short names are desirable. The tool prevents you from giving the same name to a subject area and a presentation table. It is also advisable to avoid using characters (\$, %) that may cause illegal SQL syntax with particular client tools.

- d. Click the **Permissions** button. This dialog box is used to assign user and application role permissions to this repository object. Permissions are discussed in more detail in the lesson titled “Security.”
- e. Click **Cancel** to close the Permissions dialog box.
- f. The custom display name is used if you are planning to present the name in local languages. You can ignore this for the purposes of this practice. You learn more about using custom display names in the lesson titled “Localizing Oracle BI Metadata”.
- g. Notice that the business model is grayed out and cannot be modified. This is because all objects within a subject area derive from a single business model and cannot span multiple business models. After the business model is set, it cannot be changed and the tool prevents you from including objects from other business models.
- h. Notice that “export logical keys” is deselected by default. This is irrelevant to users of the Analysis Editor, but may be advantageous to some third-party query and reporting tools. If selected, columns in the Presentation layer that are key columns in the Business Model and Mapping layer will be presented as key columns to an ODBC client and will have a key icon in the Presentation layer.
- i. Notice that the implicit fact column is not assigned. If you set an implicit fact column, this column is added to a query when it contains columns from two or more dimension tables and no measures. It is used to specify a default join path between dimension tables when there are several possible alternatives. You learn more about configuring implicit fact columns in the lesson titled “Setting an Implicit Fact Column”.
- j. In the Description field, enter something similar to **Analyze Sales and Shipment Data**. This information is visible under the corresponding subject area in Analysis Editor. You confirm this in the next set of practices.
- k. Click the **Presentation Tables** tab.

- I. Use the **Up** and **Down** buttons or drag objects to rearrange the presentation tables into the following order:



- m. Click the **Aliases** tab. If you change the name of a subject area, the tool automatically creates an alias using the previous name. You can use this tab to specify or delete an alias for a subject area.
- n. Click **OK** to close the Subject Area dialog box.
- o. Expand the **SupplierSales** subject area and notice that the order of the presentation tables is now changed in the Presentation layer.



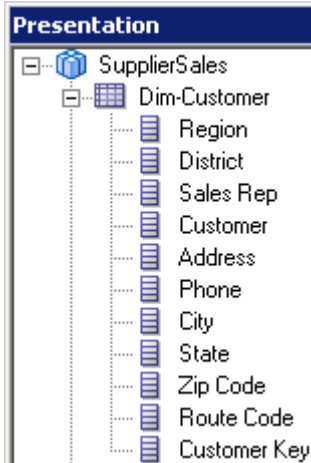
3. In this step, you explore the properties of a presentation table. You can use presentation tables to hold columns from multiple logical tables, thereby reducing the number of apparent logical tables. You can also use presentation tables to organize logical columns into smaller groupings, thereby increasing the number of apparent logical tables. For example, you might have a hundred measures in a logical fact table. You may want to create multiple presentation tables as containers for those measures and thus present them in an organized way (for example, all dollar measures in one folder and all unit measures in another folder). These measure folders could be nested within a single folder in the Analysis Editor.
 - a. Double-click the **Dim-Customer** presentation table in the Presentation layer. The Presentation Table dialog box opens.
 - b. Click the **General** tab.

- c. Notice that, by default, the presentation table name is the same as the logical table name. For Analysis Editor, the name can be anything, except that it should not contain single quotation marks, double quotation marks, or the “%” sign. For client tools that generate SQL, it is advisable to avoid names that might violate valid SQL syntax. For example, avoid SQL key words, spaces, single and double quotation marks, and other characters such as “\$” or “%.” A presentation table name cannot be the same as the subject area name or any logical column name in that catalog. As in a subject area, changing a presentation table name does not have any effect on the logical table name in the Business Model and Mapping layer. The Alias tab keeps a record of any changes.
- d. Type **Customer Data** in the Description field. This will show up as a “tool tip” in the Analysis Editor when the user places the cursor over the object. You confirm this in the next set of practices.
- e. Click the **Columns** tab.
- f. Change the order of the columns, using the **Up** and **Down** buttons or by dragging, into the following order:

Columns
Region
District
Sales Rep
Customer
Address
Phone
City
State
Zip Code
Route Code
Customer Key

- g. Click **OK** to close the Presentation Table dialog box.
4. In this step, you explore the properties of a presentation column. Presentation columns can come from multiple logical tables in a business model. By default, a presentation column uses the same name as its corresponding logical column in the Business Model and Mapping layer. If you rename the column in the Business Model and Mapping layer, corresponding presentation columns are automatically renamed wherever they appear in the Presentation layer. The reverse is not true. If you rename a presentation column, it does not impact the corresponding logical column in the Business Model and Mapping layer. However, as with subject areas and tables, the repository stores an alias for the column using the previous name.
 - a. Expand the **Dim-Customer** presentation table.

- b. Confirm that the columns are now in the order you specified in the previous step.



- c. Double-click the **District** column. The Presentation Column dialog box opens.
- d. Click the **General** tab.
- e. Deselect **Use Logical Column Name**. The Name field can now be edited.
- f. Change the column name by entering **Sales District** in the Name field.
- g. Click the **Aliases** tab and observe that the original logical column name (alias) is stored for this presentation column.
- h. Return to the **General** tab.
- i. Which logical column does this presentation column map to?
-
- j. Which business model does this presentation column map to?
-
- k. Which logical table does this presentation column map to?
-
- l. Click the **Edit** button. The Logical Column dialog box opens.
- m. Click the **General** tab in the Logical Column dialog box.
- n. Click the **Column Source** tab.
- o. Which physical table and column does this presentation column map to?
-
- p. Is this the original physical table that you imported, or the alias table that you created?
-
- q. Click **OK** to close the Logical Column window.
- r. Click **OK** to close the Presentation Column window.
5. In this step, you simplify the content by deleting unnecessary presentation columns. You may not want to expose all the logical columns of a business model in a subject area. You can delete columns from the Presentation layer safely without affecting the existence of the corresponding logical columns in the Business Model and Mapping Layer. For example, key columns in presentation tables can be deleted unless the client tools require that key information be provided.

- a. In the Presentation layer, in the Dim-Customer table, delete the **Customer Key** column.
 - b. Click **Yes** to confirm the deletion.
 - c. In the Dim-Product table, delete the **Product Key** column.
 - d. Click **Yes** to confirm the deletion.
 - e. In the Business Model and Mapping layer, expand **Dim-Customer** and **Dim-Product** and confirm that the Customer Key and Product Key logical columns are not deleted.
6. Rename presentation tables.
- a. Rename the dimension presentation tables, but not the Fact-Sales table:

Presentation table	Rename as:
Dim-Customer Customer	Customer
Dim-Time Time	Time
Dim-Product Product	Product
 - b. Double-click the **Customer** presentation table to open the Presentation Table properties dialog box.
 - c. Click the **Columns** tab and notice that changing the presentation table name does not impact the column mappings.
 - d. Click the **Aliases** tab and notice that the previous name for the presentation table has been stored.
 - e. Click **OK** to close the Presentation Table dialog box.
 - f. Notice also that changing the presentation table names in the Presentation layer has no impact on the logical table names in the Business Model and Mapping layer.
7. Save the repository.
8. Click **No** when prompted to check global consistency.
9. Leave the repository open for the next practice.

Congratulations! You have successfully built the Presentation layer of a repository.

Solutions 5-1: Creating the Presentation Layer

Answers

- 4.i. Which logical column does this presentation column map to?
District
- 4.j. Which business model does this presentation column map to?
SupplierSales
- 4.k. Which logical table does this presentation column map to?
Dim-Customer
- 4.o. Which physical table and column does this presentation column map to?
Dim_D1_CUSTOMER2.DISTRICT
- 4.p. Is this the original physical table that you imported, or the alias table that you created?
Alias table

Practices for Lesson 6: Testing and Validating a Repository

Chapter 6

Overview of Practices for Lesson 6

Lesson Overview

In these practices, you will test and validate a repository and make it available for queries.

Practice 6-1: Testing the Repository

Goal

To test the repository by generating some queries, retrieving the results, and examining the query log

Scenario

You finished building the initial business model, and now you must test the repository before continuing. You begin by checking the repository for errors by using the consistency check option. You then test the repository by using the Analysis Editor to run queries. Finally, you examine the query log file to verify the SQL generated by Oracle BI Server.

Outcome

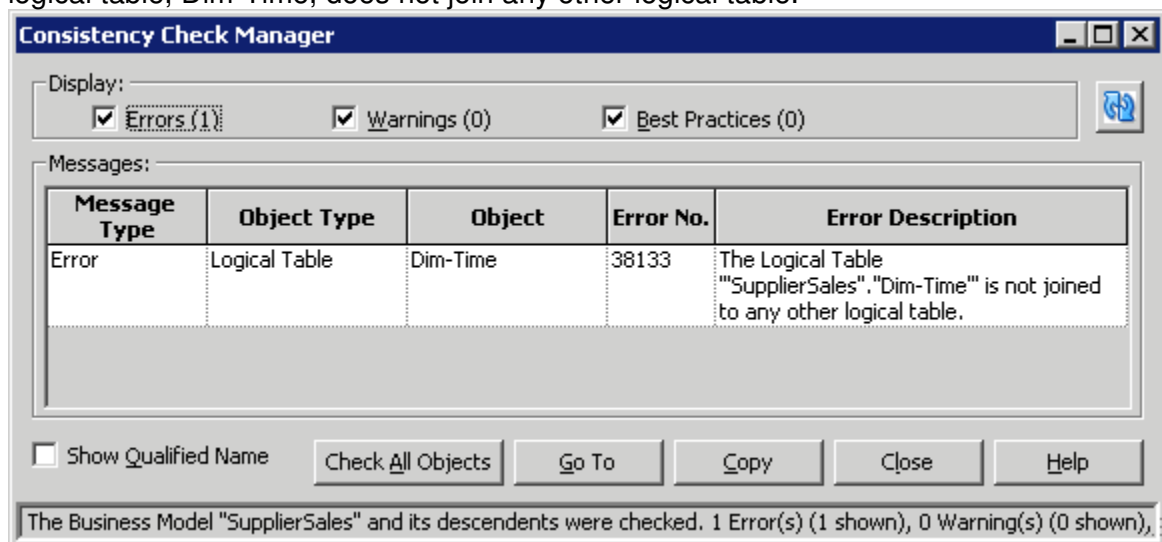
A tested and verified repository file

Time

45 minutes

Tasks

1. Modify the business model to generate an inconsistent business model due to a missing logical join.
 - a. In the Business Model and Mapping layer, right-click **SupplierSales** and select **Business Model Diagram > Whole Diagram**.
 - b. Select the **join connection** between the Dim-Time and Fact-Sales logical tables.
 - c. Right-click and select **Delete** to create a condition in the business model where an undefined join condition exists.
 - d. Click **Yes** to confirm the delete.
 - e. Close the Business Model Diagram window.
 - f. In the Business Model and Mapping layer, right-click **SupplierSales** and select **Check Consistency**. The Consistency Check Manager appears and displays an error message for the SupplierSales business model. The error description explains that the logical table, Dim-Time, does not join any other logical table.



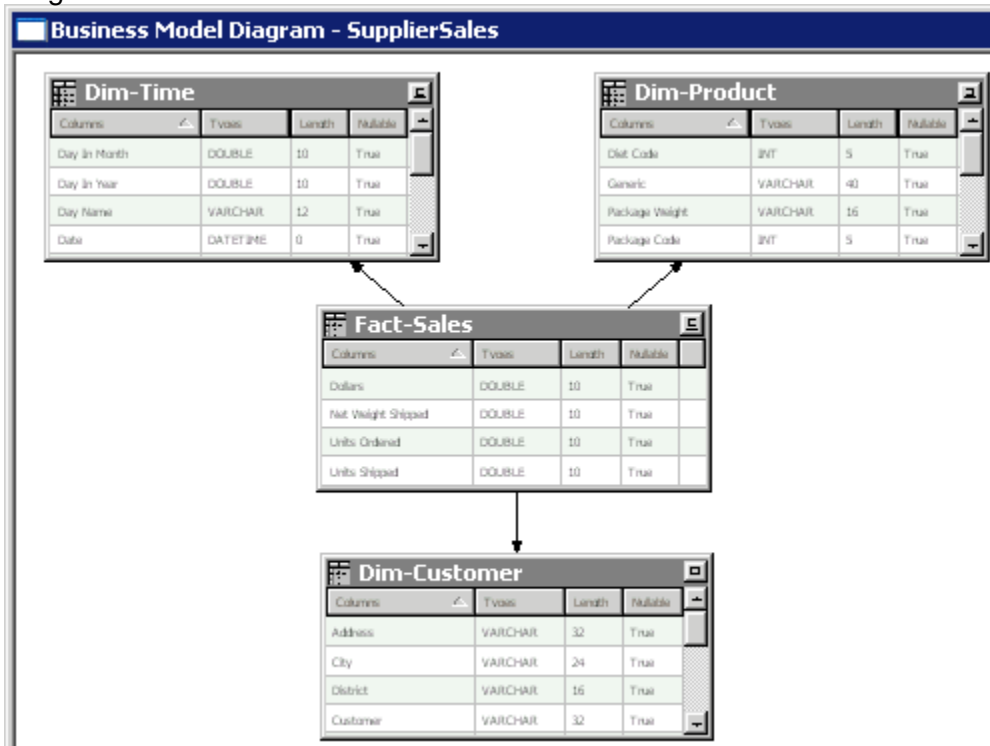
- g. The consistency check provides three types of messages:

Error messages indicate errors that must 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, a warning message about a disabled join may be the result of the administrator intentionally disabling a join (for example, by eliminating a circular join condition).

Best Practices messages provide information about conditions but do not indicate an inconsistency (for example, “fact table does not contain a logical key”).

- h. Notice there are buttons that allow you to perform other tasks, such as going to the object in the repository or copying the error message.
- i. Click **Close** to close Consistency Check Manager.
2. Use the Business Model Diagram to repair the logical join.
- a. Right-click the **SupplierSales** business model and select **Business Model Diagram > Whole Diagram**.
- b. Click the **New Join** icon on the tool bar.
- c. Select **Fact-Sales** and then **Dim-Time** in the diagram. The order is important. The Logical Join dialog box opens. Recall that you examined this dialog box in the practices for the lesson titled “Building the Business Model and Mapping Layer of a Repository.”
- d. Click **OK** to close the Logical Join dialog box.
- e. The logical join between Dim-Time and Fact-Sales is re-created in the Business Model Diagram.



- f. Close the Business Model Diagram.

3. Perform a global consistency test to check the repository for inconsistencies. A consistent repository has met the following requirements:

All logical columns are mapped directly or indirectly to one or more physical columns.

All logical dimension tables have a logical key.

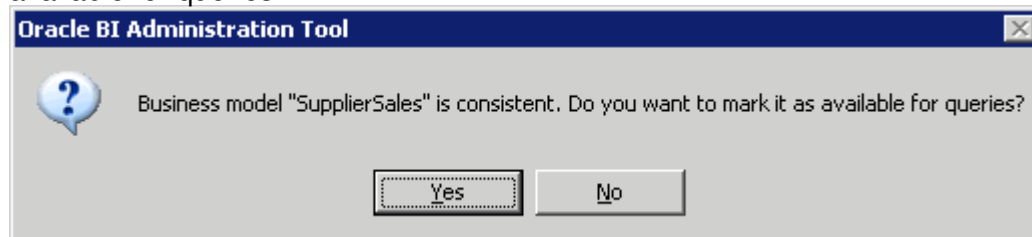
All logical tables have a logical join relationship to another logical table.

There are at least two logical tables in the business model: one is a logical fact table, the other is logical dimension table. Both tables may map to the same physical table.

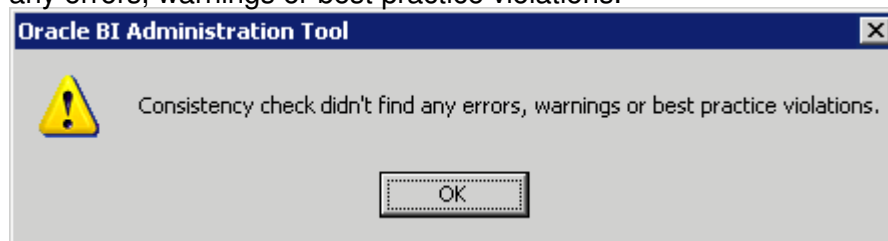
There are no circular logical join relationships.

A presentation catalog exists for the business model.

- a. Select **File > Check Global Consistency**. A message appears indicating that the SupplierSales business model is consistent and asks whether you want to mark it as available for queries.

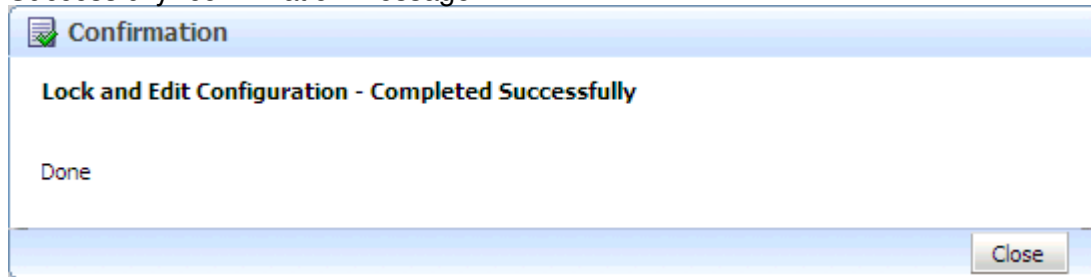


- b. Click **Yes**. You should receive the following message: "Consistency check didn't find any errors, warnings or best practice violations."

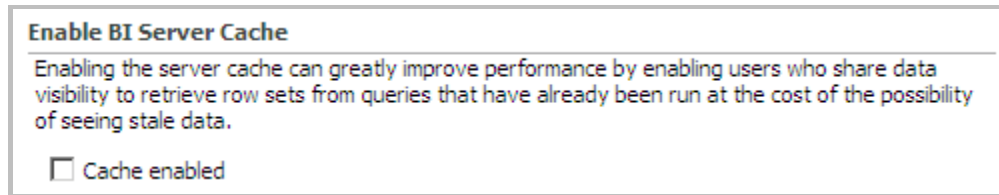


- c. Click **OK** to close the message. The SupplierSales business model folder has changed from unavailable for queries (red icon) to available for queries (green icon).
- d. Save the repository.
- e. Click **No** when prompted to check global consistency, because you just checked consistency.
- f. Select **File > Close** to close the repository.
- g. Leave the Administration Tool open.
4. Use Fusion Middleware Control to disable cache. Caching is typically not used during development, except to test the cache. You learn more about caching in the lesson titled "Cache Management."
 - a. Return to **Fusion Middleware Control**, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, select **Capacity Management** and then the **Performance** subtab.
 - e. Locate the **Enable BI Server Cache** section. Cache is enabled by default.
 - f. Click **Lock and Edit Configuration**.

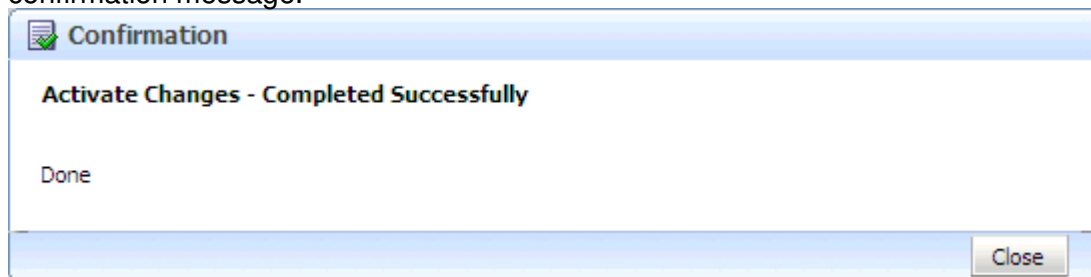
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



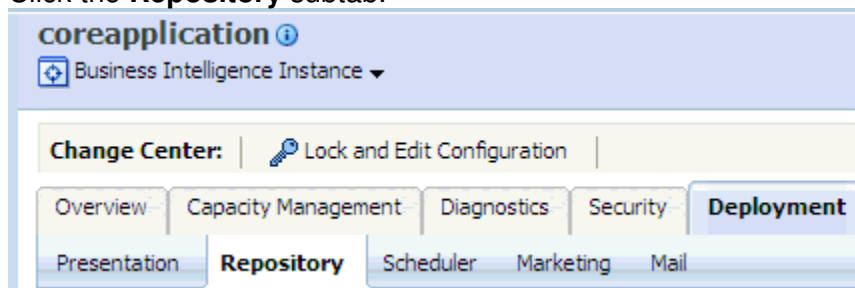
- h. Deselect **Cache enabled**.



- i. Click **Apply**.
- j. Click **Activate Changes**.
- k. Allow processing to complete.
- l. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.

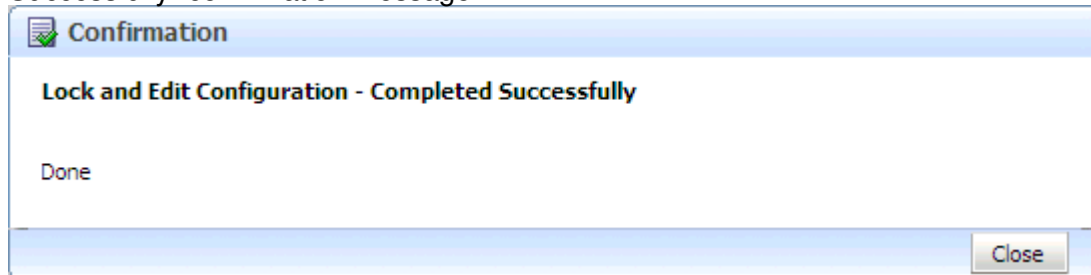


- m. Do not click **Restart to apply recent changes** yet. You do that after uploading the repository in the next set of steps.
5. Use Fusion Middleware Control to upload the repository.
- a. In the right pane, click the **Deployment** tab.
- b. Click the **Repository** subtab.

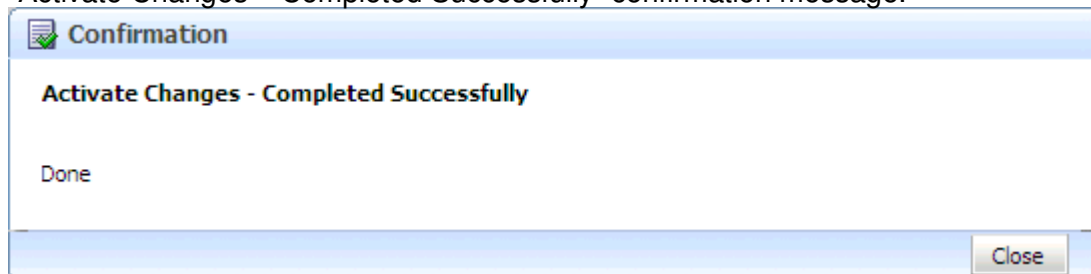


- c. Click **Lock and Edit Configuration**.

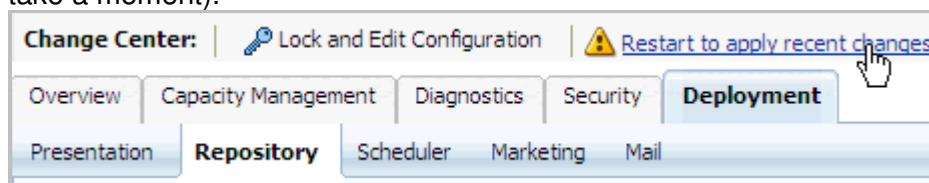
- d. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- e. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- f. By default, the Choose file dialog box should open to the default repository directory. If not, browse to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository**.
- g. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- h. Enter **welcome1** in the Repository Password and Confirm Password fields.
- i. Click **Apply**. Notice that Default RPD now displays ABC with an extension (for example, ABC_BI0002).
- j. Click **Activate Changes**.
- k. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



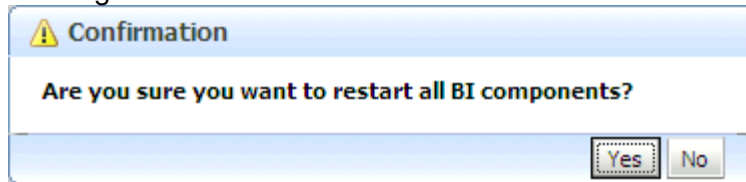
- l. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



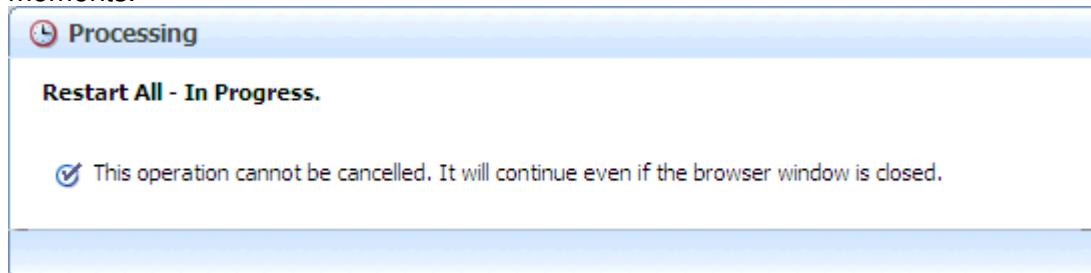
- m. On the Overview page, click **Restart**.



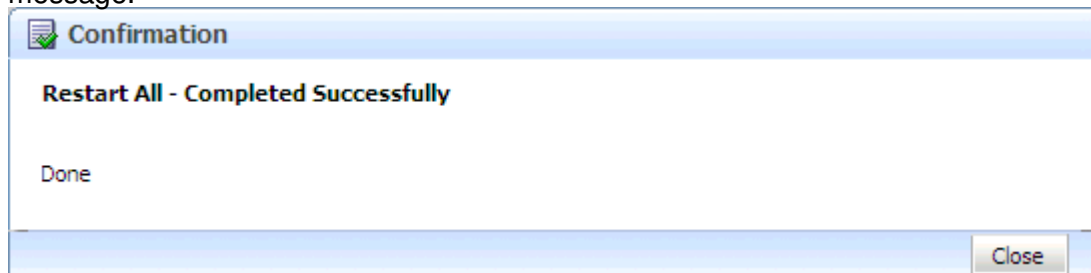
- n. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



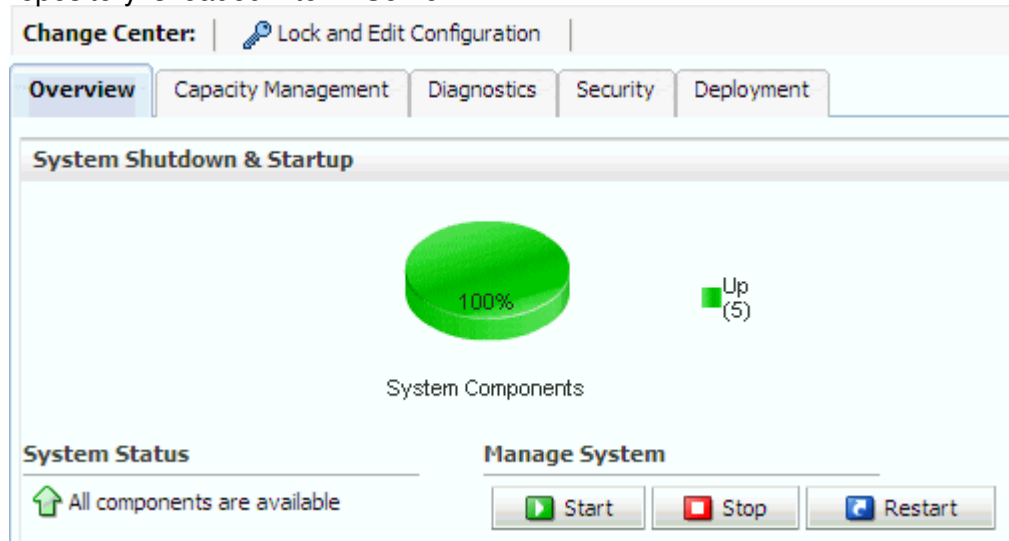
- o. Allow the Restart All – In Progress processing to complete. This may take a few moments.



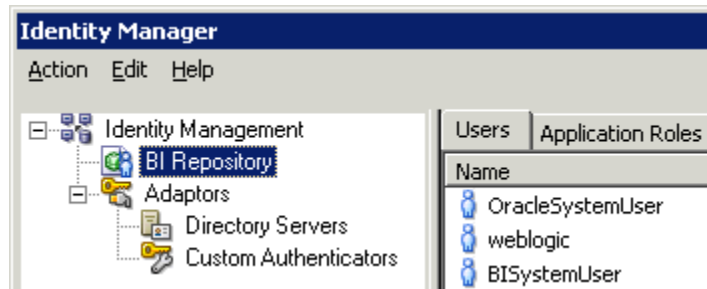
- p. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- q. Confirm that System Components = 100%. Cache is now disabled and the ABC repository is loaded into BI Server.

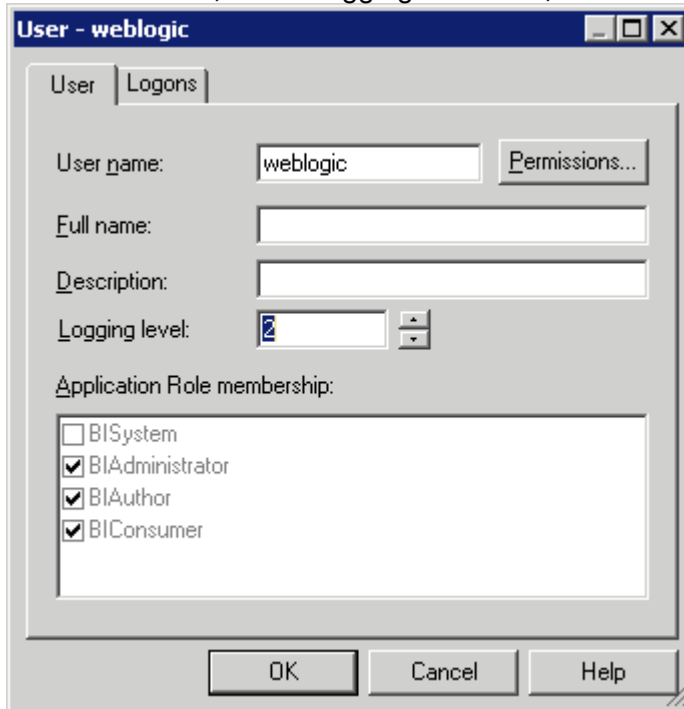


- r. Leave Fusion Middleware Control open.
6. In this step, you enable query logging for the weblogic user. To test a repository, you must build some analyses, retrieve the results, and examine the query log. You log query Practice at the individual-user level. Logging is intended for testing, debugging, and technical support. In production mode, logging is normally disabled because query logging can impact performance by producing very large log files.
- a. Return to the Administration Tool.
 - b. Select **File > Open > Online** to open the ABC repository in online mode.
 - c. Enter **welcome1** as the repository password.
 - d. Enter **weblogic** as the username and **welcome1** as the user password.
 - e. Click **Open** to open the repository in online mode.
 - f. Select **Manage > Identity** to open Identity Manager.
 - g. In the left pane, select **Identity Management > BI Repository**.
 - h. Click the **Users** tab. In online mode, by default, no users are retrieved, because the list of users might be very large.
 - i. Select **Action > Set Online User Filter** to specify the set of users you want to retrieve. The filter is empty by default, which means that no users are retrieved. You can enter * to retrieve all users, or enter a combination of characters for a specific set of users, such as A* to retrieve all users whose names begin with the letter A. The filter is not case-sensitive.
 - j. Enter * and click **OK** to retrieve all users. A list of users appears in the right pane.



- k. In the right pane, double-click **weblogic**. The User dialog box opens.

- l. On the User tab, in the Logging level field, set the value to **2**.

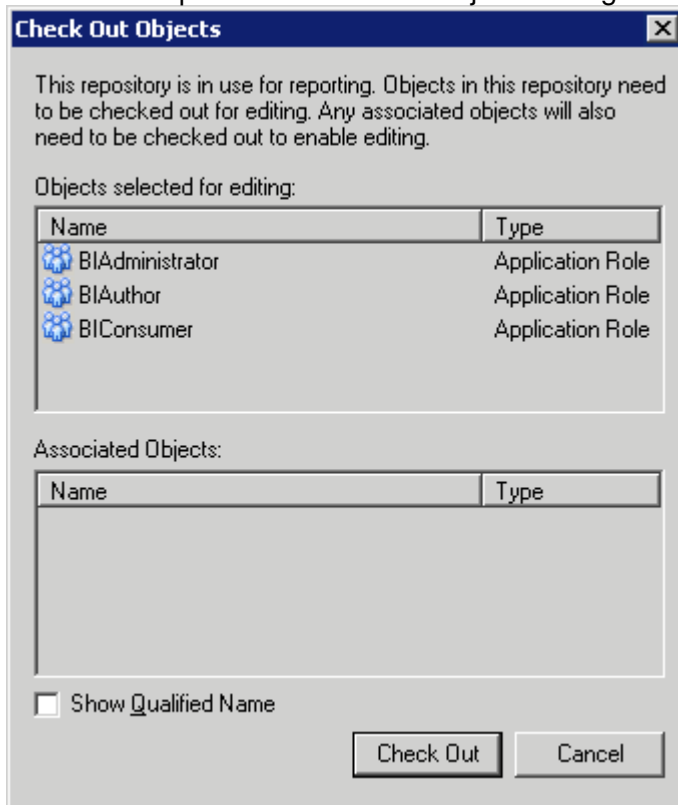


The 'User - weblogic' dialog box has two tabs: 'User' and 'Logons'. The 'User' tab is active. It contains the following fields and controls:

- User name:** A text box containing 'weblogic' and a 'Permissions...' button.
- Full name:** An empty text box.
- Description:** An empty text box.
- Logging level:** A dropdown menu with '2' selected.
- Application Role membership:** A list box containing four roles: BISystem, BIAdministrator, BIAuthor, and BIConsumer. The last three are checked.

At the bottom are 'OK', 'Cancel', and 'Help' buttons.

- m. Click **OK** to open the Check Out Objects dialog box.



The 'Check Out Objects' dialog box contains the following information:

This repository is in use for reporting. Objects in this repository need to be checked out for editing. Any associated objects will also need to be checked out to enable editing.

Objects selected for editing:

Name	Type
BIAdministrator	Application Role
BIAuthor	Application Role
BIConsumer	Application Role

Associated Objects:

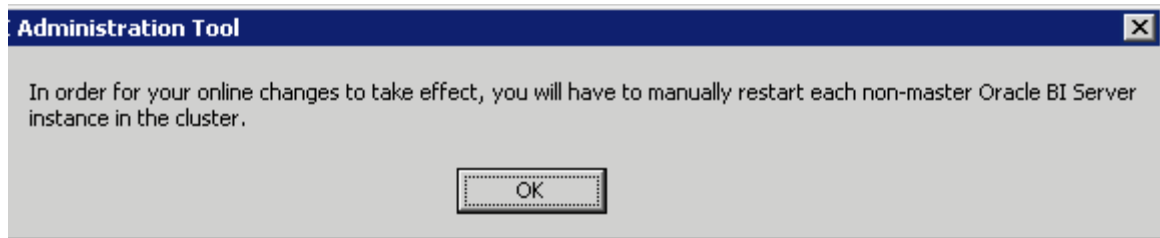
Name	Type
------	------

☐ Show Qualified Name

At the bottom are 'Check Out' and 'Cancel' buttons.

- n. Click **Check Out**. When you are working in a repository open in online mode, you are prompted to check out objects when you attempt to perform various operations.
- o. Select **Action > Close** to close the Security Manager window. Additional security-related topics are addressed in more detail in the lesson titled "Security."

- p. Select **File > Check In Changes** or click the **Check In Changes** icon on the toolbar.
- q. Save the repository. There is no need to check consistency.
- r. Select **File > Close** to close the repository.
- s. Click **OK** when you receive the following message: “In order for your online changes to take effect, you will have to manually restart each non-master Oracle BI Server instance in the cluster.”



- t. Leave the Administration Tool open.
- 7. Open Analysis Editor to execute queries and test the SupplierSales business model.
 - a. Return to the Oracle Business Intelligence browser tab where you signed out of Analysis Editor, and click **here** to sign in.

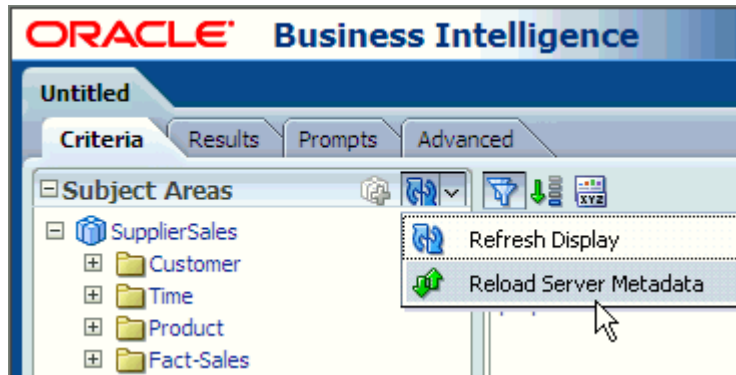
Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- b. Sign in as **weblogic** with password **welcome1**.
- c. In the Create section, click **Analysis** to open the Select Subject Area window. Notice that the description that you created for the SupplierSales subject area is visible.

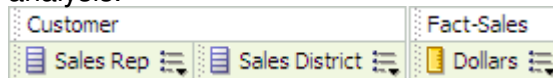


- d. Click **SupplierSales** to open Analysis Editor.
- e. Place the cursor over the **Customer** folder and ensure that the description you entered in an earlier practice appears as a “tool tip.”

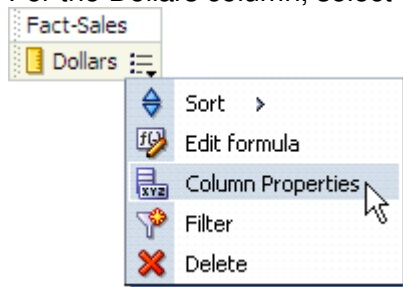
- f. Select **Reload Server Metadata**.



8. Create a new analysis and format the columns.
- Expand **Customer** and double-click the **Sales Rep** and **Sales District** columns to add the columns to the analysis.
 - Expand the **Fact-Sales** table and double-click the **Dollars** column to add it to the analysis.

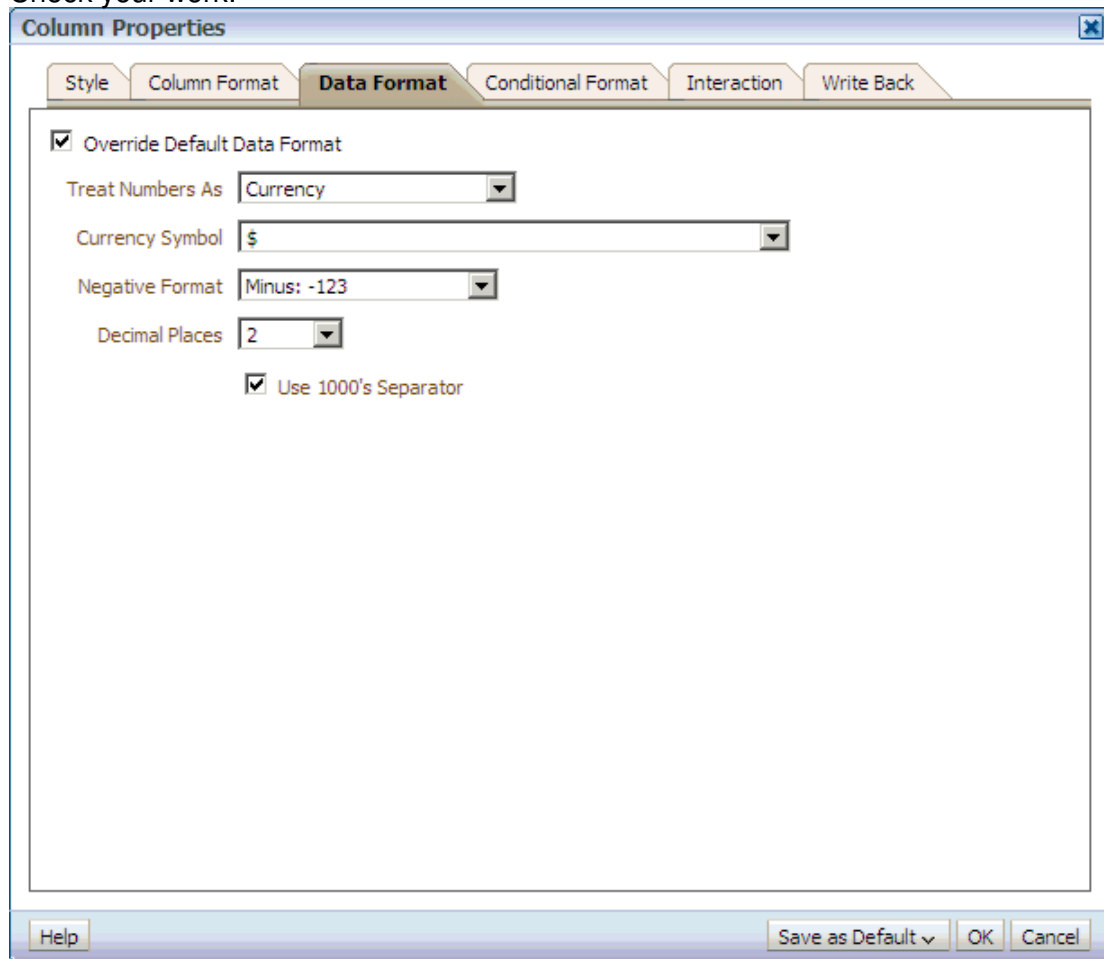


- c. For the Dollars column, select **Column Properties**.



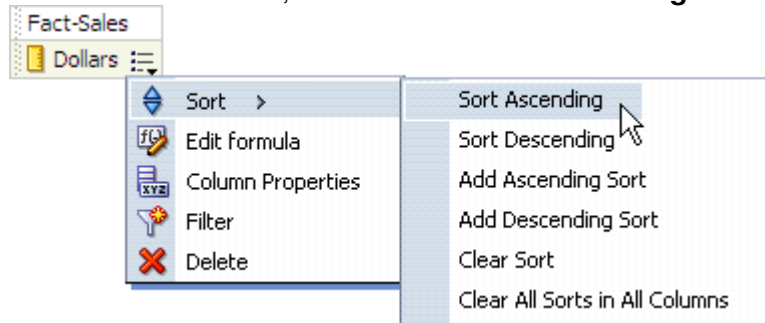
- In the Column Properties dialog box, click the **Data Format** tab.
- Click **Override Default Data Format**.
- In the Treat Number As field, select **Currency**.
- In the Currency Symbol field, select **\$**.
- In the Decimal Places field, select **2**.
- Select the **Use 1000's Separator** check box.

j. Check your work:



k. Select **Save as Default** > **Save as the system-wide default for “Fact-Sales”.Dollars”**.

l. In the Dollars column, select **Sort** > **Sort Ascending**.



9. Create filters for the analysis and view the results.

- Expand the **Time** table.
- Double-click **Year** to add it to the analysis.
- For the Year column, select **Filter** to open the New Filter dialog box.

- d. In the Value field, enter **2009** and click **OK**.

- e. For the Sales District column, select **Filter**.
 f. Click the **down arrow** next to the Value field.
 g. Select **MidAtlantic** from the list, and click **OK**.
 h. Notice that both filters are now added to the Filters area:

Year is equal to / is in 2009
 AND Sales District is equal to / is in MidAtlantic

- i. Delete the **Sales District** and **Year** columns from the analysis so that these columns will not be displayed in the analysis results.
 j. Your analysis now includes two columns:

Customer	Fact-Sales
Sales Rep	Dollars

And two filters:

Year is equal to / is in 2009
 AND Sales District is equal to / is in MidAtlantic

- k. Click the **Results** tab to view the results. Verify that the Dollars column is formatted and sorted as expected. The results show total dollars for each sales rep in the MidAtlantic sales district for the year 2009.

Sales Rep	Dollars
GEORGE MASUR	\$97,129.88
DALE FAIRWEATHER	\$167,932.96
WALLY RAISANEN	\$190,607.12
PAULA MADISON	\$467,039.54

10. In this step, you create a new analysis showing the dollars for each sales district in the East region for the year 2009.

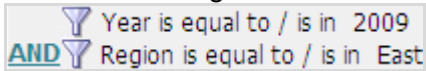
- a. Select **New > Analysis > SupplierSales**.
 b. Click **OK** when prompted with the following message: "Are you sure you want to navigate away from this page?".
 c. Create the following request:

Customer	Fact-Sales
Sales District	Dollars

- d. Sort the **Dollars** column in ascending order.
- e. Click the **Filter** button on the Filters pane header.



- f. Select **More Columns**. This is another method for setting filters.
- g. Add the following filters for **Year** and **Region**:



- h. Click **Results**.

Sales District	Dollars
Florida	\$406,139.91
MidAtlantic	\$922,709.50
UpperSouth	\$1,513,814.30
Yankee	\$3,220,998.40

11. Examine the query log.

- a. Click Administration.
- b. Click **OK** when prompted with the message: "Are you sure you want to navigate away from this page?"
- c. In the Session Management section, click **Manage Sessions** to open the Manage Sessions page.
- d. In the Cursor Cache section, locate your query. It should look similar to the screenshot:

2505	weblogic	1	Finished	0s	Close View Log	4/14/2011 4:22:57 PM GMT+00:00	<pre>SET VARIABLE QUERY_SRC_CD='Report';SELECT 0 s_0, "SupplierSales"."Customer"."Sales District" s_1, "SupplierSales"."Fact-Sales"."Dollars" s_2 FROM "SupplierSales" WHERE (('Time"."Year" = 2009) AND ("Customer"."Region" = 'East')) ORDER BY 1, 3 ASC NULLS LAST, 2 ASC NULLS LAST</pre>
------	----------	---	----------	----	---	--------------------------------	---

- e. Click **View Log**.

- f. If necessary, scroll to the bottom of the file and then scroll up to locate the last query executed by **weblogic**. The log file should look similar to the screenshot:

```
139c0000] [username: weblogic]
##### [[
----- SQL Request:
SET VARIABLE QUERY_SRC_CD='Report';SELECT
    0 s_0,
    "SuppliersSales"."Customer"."Sales District" s_1,
    "SuppliersSales"."Fact-Sales"."Dollars" s_2
FROM "SuppliersSales"
WHERE
((("Time"."Year" = 2009) AND ("Customer"."Region" = 'East'))
ORDER BY 1, 3 ASC NULLS LAST, 2 ASC NULLS LAST

]]
[2011-04-14T16:22:57.000+00:00] [OracleBIServerComponent] [TRACE:2]
[USER-23] [] [ecid: 84e31caac17e33a1:10a2bbdc:12f4a44f468:
-8000-00000000000003175] [tid: 1ba8] [requestid: 139c000f] [sessionid:
139c0000] [username: weblogic] ----- General Query Info:
[[
Repository: Star, Subject Area: SuppliersSales, Presentation:
SuppliersSales

]]
[2011-04-14T16:22:57.000+00:00] [OracleBIServerComponent] [TRACE:2]
[USER-18] [] [ecid: 84e31caac17e33a1:10a2bbdc:12f4a44f468:
-8000-00000000000003175] [tid: 1ba8] [requestid: 139c000f] [sessionid:
139c0000] [username: weblogic] ----- Sending query to
database named orcl (id: <<5089>>), connection pool named SUPPLIER CP:
[[
WITH
SAWITH0 AS (select sum(T90.DOLLARS) as c1,
    T76.DISTRICT as c2
from
    D1_CALENDAR2 T58 /* Dim_D1_CALENDAR2 */ ,
    D1_CUSTOMER2 T76 /* Dim_D1_CUSTOMER2 */ ,
    D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */
where ( T58.YEAR = 2009 and T58.YYYYMMDD = T90.PERIODKEY and
T76.NEWKEY = T90.CUSTKEY and T76.REGION = 'East' )
group by T76.DISTRICT)
select distinct 0 as c1,
    D1.c2 as c2,
    D1.c1 as c3
from
    SAWITH0 D1
order by c3, c2

]]
```

- g. Locate the **SQL Request** section. This section contains the logical SQL issued by the query.
- h. 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.
- i. Locate the **Sending query to database named orcl** section, just below the General Query Info section. This section identifies the physical SQL issued by the query, connection pool, and data source to which Oracle BI Server connects.
- j. Click the browser **back button** to return to the Manage Session page.
- k. Click **Back** to return to the Administration page.
- l. Click **Home** to return to the Home page.
- m. Sign out of Oracle BI.

12. Use another method to examine the query log.
 - a. Return to **Fusion Middleware Control**, which should still be open.
 - b. Click the **Diagnostics** tab.
 - c. Click the **Log Messages** subtab.
 - d. Scroll to the bottom of the window to the **View / Search Log Files** section.
 - e. Click **Server Log**.
 - f. In the Log Messages screen, leave the data range set to **Most Recent, 1 Days**.
 - g. Deselect all message types except for **Trace**.
 - h. In the Message field, enter **sending query to database**.

Log Messages

☐ Search

☐ Selected Targets (1)

Date Range:

* Message Types: ☐ Incident Error ☐ Error ☐ Warning ☐ Notification ☒ Trace ☐ Unknown

Message:

- i. Click **Search**.
- j. Select the last message in the list. This is the most recent query sent to the database.
- k. In the bottom pane, click the **Collapse Pane** button (arrow on the right side) to view the log message. Your results should look similar to the screenshot.

```

----- Sending query to database named orcl (id: <<5089>>), connection pool named SUPPLIER CP:
WITH
SAWITH0 AS (select sum(T90.DOLLARS) as c1,
               T76.DISTRICT as c2
from
  D1_CALENDAR2 T58 /* Dim_D1_CALENDAR2 */ ,
  D1_CUSTOMER2 T76 /* Dim_D1_CUSTOMER2 */ ,
  D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */
where ( T58.YEAR = 2009 and T58.YYYYMMDD = T90.PERIODKEY and T76.NEWKEY = T90.CUSTKEY and T76.REGION = 'East' )
group by T76.DISTRICT)
select distinct 0 as c1,
               D1.c2 as c2,
               D1.c1 as c3
from
  SAWITH0 D1
order by c3, c2

```

- l. Click the **Restore Pane** button.
- m. Throughout the course you can use either method to check the query log.
- n. Leave Enterprise Manager open.

Congratulations! You have successfully used the Consistency Check Manager, Oracle BI Analysis Editor, and the query log to test and check the repository.

Practices for Lesson 7: Managing Logical Table Sources

Chapter 7

Overview of Practices for Lesson 7

Lesson Overview

In these practices, you will manage logical table sources in and Oracle BI repository.

Practice 7-1: Enhancing the Product Dimension

Goal

To import normalized tables with additional product information into the Physical layer of the repository

Scenario

There are product tables that store detailed information about ABC's products. You want to add these tables to the Product dimension in the Business Model and Mapping layer. You import these tables into the repository and create keys and foreign key joins for the tables.

Outcome

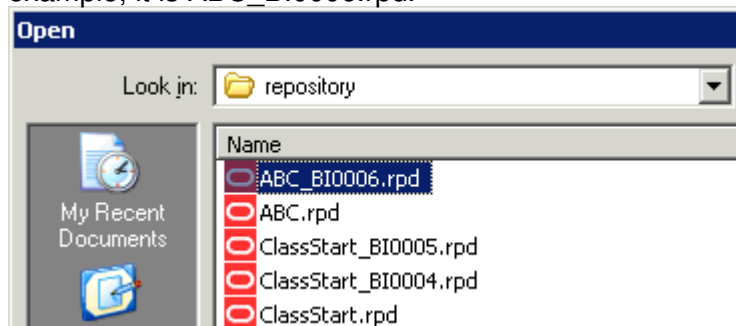
D1_PRICELIST, D1_PROD_DIET_TYPES, D1_PRODUCT_SUBTYPE, D1_PRODUCT_TYPE, and D1_SUPPLIERS tables imported into the Physical layer with associated keys and joins

Time

10 minutes

Tasks

1. Replace the existing ABC repository with the ABC repository version that contains the security information.
 - a. Return to the Oracle BI Administration Tool, which should still be open. If not, select **Start > Programs > Oracle Business Intelligence > BI Administration**.
 - b. Select **File > Open > Offline**.
 - c. Select the **ABC repository with the highest number extension**. In the screenshot example, it is ABC_BI0006.rpd.



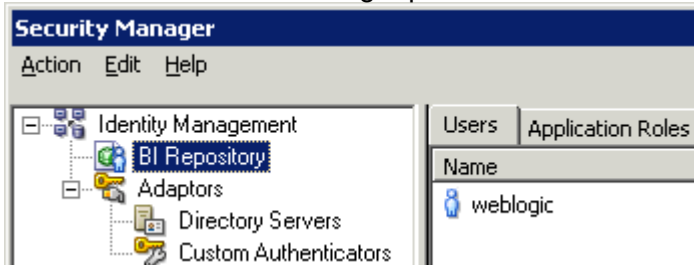
- d. Click **Open**. You should receive a message that this repository can only be opened as read-only.



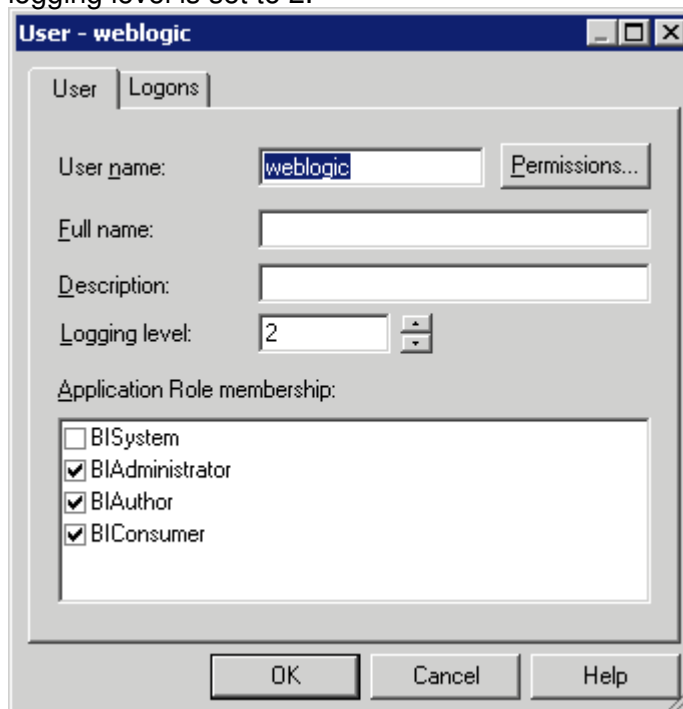
Recall that this is the repository that is currently uploaded into Oracle BI Server memory. You want to use this version of the repository because it has the user- and logging-level information that you defined earlier. In offline mode, users do not appear

in Identity Manager unless you have first modified them in the Administration Tool in online mode, as you did in the previous set of practices.

- e. Click **Yes** to open the Open Offline dialog box. The Repository Password dialog box opens.
- f. Enter **welcome1** as the repository password.
- g. Click **OK** to open the repository in read-only mode.
- h. Select **File > Save As** to open the Save As dialog box.
- i. Select **ABC.rpd** to enter it in the File name field.
- j. Click **Save**.
- k. You should receive the following message: "ABC.rpd already exists. Do you want to replace it?".
- l. Click **Yes** to open the ABC repository in offline mode. This action replaces the existing ABC repository with the ABC repository that contains the security information.
- m. Select **Manage > Identity** to open Identity Manager.
- n. Select **BI Repository** in the left pane.
- o. Click the **Users** tab in the right pane and confirm that the **weblogic** user is visible.



- p. Double-click **weblogic** to open the User dialog box. On the User tab, confirm that the logging level is set to 2.



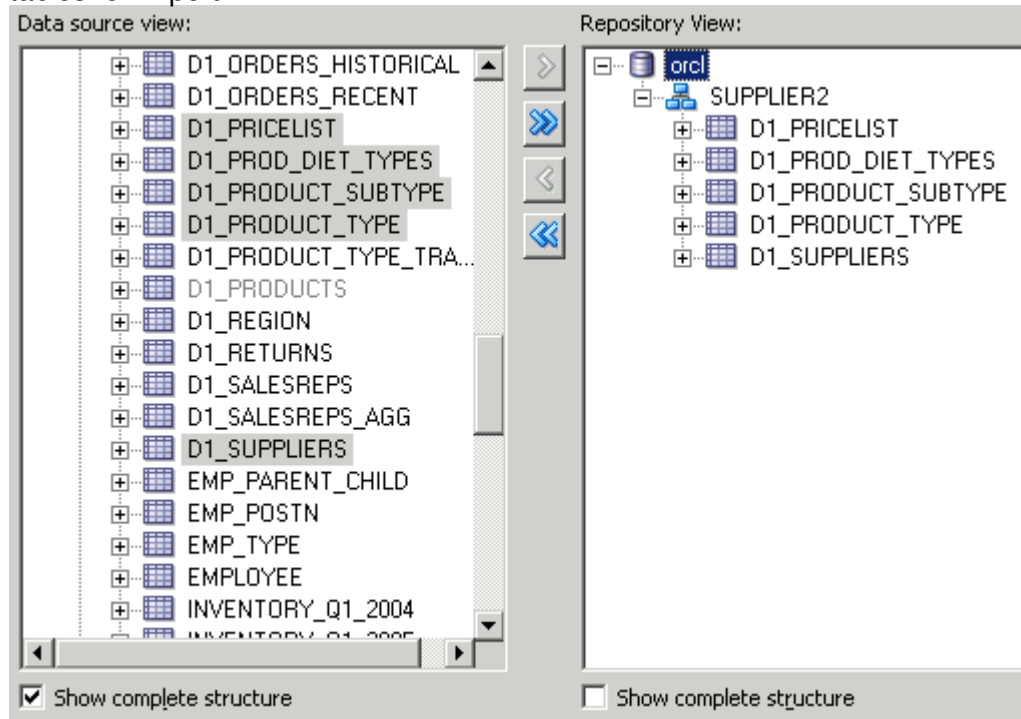
- q. Click **Cancel** to close the User dialog box.

- r. Select **Action > Close** to close Identity Manager. The offline ABC repository now has a weblogic user with a logging level set to 2. This will allow you to check the query log as you complete the remaining practices in this course.
- 2. In this step, you import additional product tables that store product code, pricing, and supplier information. The product dimension is an example of information stored physically in a normalized table structure. Data warehouse design writers such as Ralph Kimball refer to this as “snowflaking a dimension.” Many database administrators regard this as good database design, so this is a very common practice. So far, you have only included the information in the root product table in the logical subject area. After import, you can include information from the other product tables.
 - a. In the Physical layer, expand the **orcl** database object.
 - b. Right-click the **SUPPLIER CP** connection pool and select **Import Metadata**.
 - c. Accept the defaults in the Select Metadata Types window and click **Next**.
 - d. In the Data source view pane, expand **SUPPLIER2**.
 - e. Select the following tables for import:

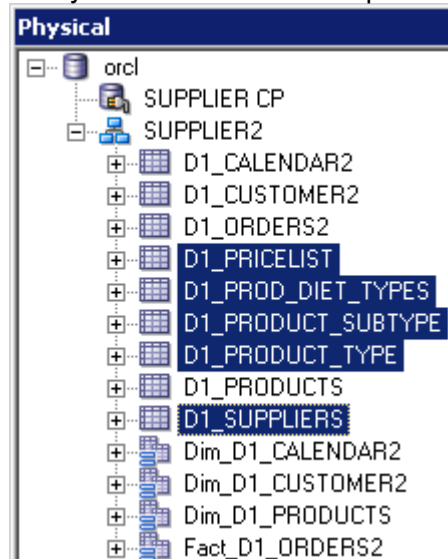
Table Name
D1_PRICELIST
D1_PROD_DIET_TYPES
D1_PRODUCT_SUBTYPE
D1_PRODUCT_TYPE
D1_SUPPLIERS

- f. Confirm that only the five tables are selected and that no higher level objects are selected.
- g. Click the **Import Selected** button to move the tables to the Repository View pane.
- h. Expand **SUPPLIER2** in the Repository View and verify that the new product tables are visible. Deselect **Show complete structure** in the Repository view to see only the

tables for import.



- i. Click **Finish**.
- j. Verify that the tables are imported into the Physical layer.



- k. Update row counts for the new tables to confirm connectivity.
3. Create the following aliases for the imported product tables:

Table	Alias
D1_PRICELIST	Dim_D1_PRICELIST
D1_PROD_DIET_TYPES	Dim_D1_PROD_DIET_TYPES
D1_PRODUCT_SUBTYPE	Dim_D1_PRODUCT_SUBTYPE
D1_PRODUCT_TYPE	Dim_D1_PRODUCT_TYPE

Practice 7-2: Creating Multiple Sources for a Logical Table Source (Manual)

Goal

To add the information from the price libgvgst table to the Product dimension

Scenario

You have imported the product tables that store detailed information about ABC's products into the Physical layer of the repository, and configured keys and joins for the tables. So far, the Dim-Product logical table in the Business Model and Mapping layer has only information from the root product table: Dim_D1_PRODUCTS. You now add the information from the price list table to the Dim-Product logical table. This will simplify the data structure, in effect, creating a denormalized logical table.

Outcome

In the Business Model and Mapping layer, the Dim_D1_PRICELIST physical table is added to the existing logical table source for the Dim-Product logical table. The Price logical column is added to the Dim-Product logical table and mapped to the appropriate physical column.

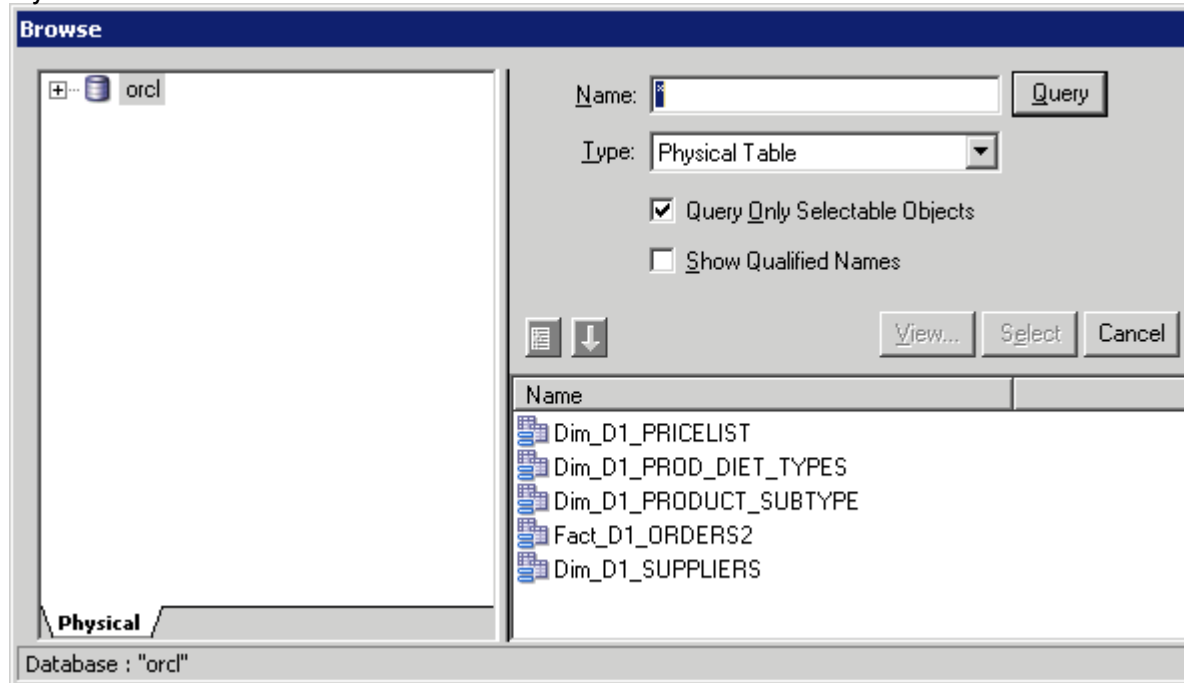
Time

10 minutes

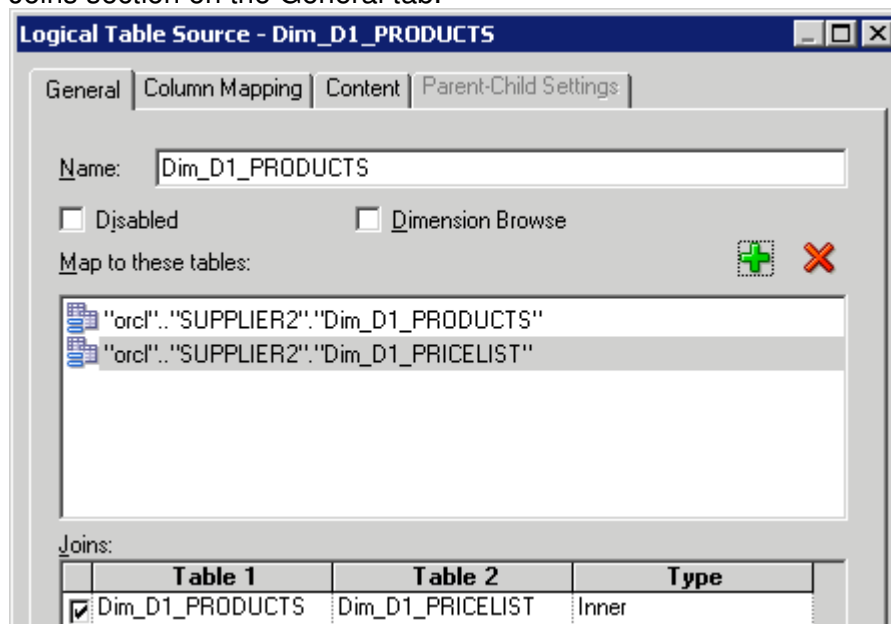
Tasks

1. In this step, you add the information from the price list table to the existing logical table source for the Dim-Product logical table. There are two methods to add multiple sources for an existing logical table source. In this practice, you use the Properties window of an existing logical table source, which is a manual process and requires several steps. In the next practice, you use a more automated process.
 - a. In the Business Model and Mapping layer, expand the **Dim-Product** logical table and then the **Sources** folder, and then double-click the **Dim_D1_PRODUCTS** logical table source to view the properties.
 - b. Click the **General** tab and click **Add**. The Browse window automatically includes only those tables that are joined directly to the table already in the logical table source. In this case, it includes all tables that join to Dim_D1_PRODUCTS. Only tables that join to tables included in the logical source can be added to the logical source. For example, notice that Dim_D1_PRODUCT_TYPE is not visible in the browse list. This is because it does not have a direct join relationship with Dim_D1_PRODUCTS in the Physical

layer.



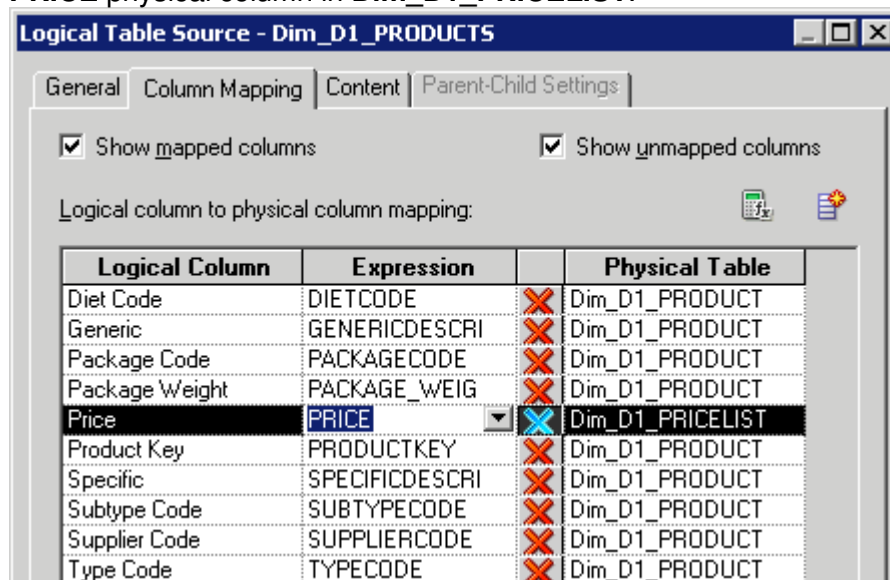
- c. In the Browse window, select the **Dim_D1_PRICELIST** table and click the **Select** button. The table is added to the logical table source and the join is displayed in the Joins section on the General tab.



- d. Select the **join** in the Joins section. The View Details button becomes active.
- e. Click **View Details** to open the Complex Join dialog box and view the read-only details of the join.
- f. Click **Cancel** to close the dialog box.
- g. To change a join to an outer join, you could use the drop-down list in the Type column. This allows you to change the join type from inner to three kinds of outer joins. For the purpose of these practices, leave the type as Inner. You can think of the tables in a logical table source as being like a database view. When it formulates physical SQL,

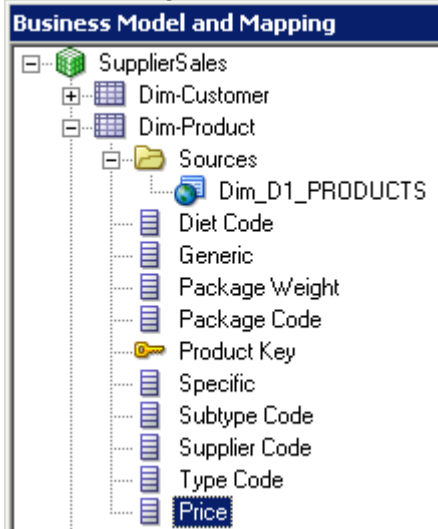
Oracle BI Server will leave out the tables in this “view” that are not needed to satisfy the logical query (join elimination), but only if the join type is Inner. When the join type is Outer, however, Oracle BI Server will always include the tables.

- h. Click **OK** to close the Logical Table Source dialog box.
2. In this step, you create a new logical column based on the modified logical table source. In the previous step you used the manual method to add a physical table to a logical table source. That method does not add any logical columns to the logical table, nor does it change the logical-to-physical mapping of any existing column. Now that the physical table that stores the pricing information has been added to the Dim-Product logical table source, you create a new logical column and map it to the appropriate physical table and column.
 - a. In the Business Model and Mapping layer, right-click **Dim-Product** and select **New Object > Logical Column**.
 - b. Enter **Price** in the Name field and click **OK**.
 - c. Double-click the **Dim_D1_PRODUCTS** logical table source to open its properties dialog box.
 - d. Click the **Column Mapping** tab.
 - e. If necessary, select the **Show unmapped columns** check box. Notice that the column you just created, Price, is not mapped to any physical column.
 - f. Use the drop-down list in the Expression field to map the **Price** logical column to the **PRICE** physical column in **Dim_D1_PRICELIST**.



- g. Click **OK** to close the Logical Table Source dialog box.

- h. The Price logical column is added to the Dim-Product logical table.



- i. Save the repository without checking consistency.

Practice 7-3: Creating Multiple Sources for a Logical Table Source (Automated)

Goal

To add the information from the additional product tables to the Product dimension

Scenario

You have manually added information from the price list table to the Dim-Product logical table. Now you add information from the other product tables to the Dim-Product logical table using a more automated method. This automated method adds multiple physical tables to the existing logical table source for the Dim-Product logical table and simultaneously adds logical columns to the Dim-Product logical table.

Outcome

In the Business Model and Mapping layer, the D1_PROD_DIET_TYPES, Dim_D1_PRODUCT_SUBTYPE, Dim_D1_PRODUCT_TYPE, and D1_SUPPLIERS physical tables are added to the logical table source for the Dim-Product logical table. Also, the DIET_TYPE, ITEMSUBTYPE, ITEMTYPE, and ITEMSUPPLIER logical columns are added to the Dim-Product logical table and mapped to the appropriate physical columns.

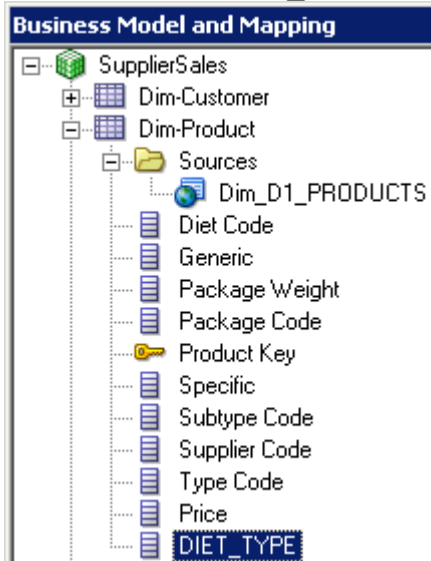
Time

10 minutes

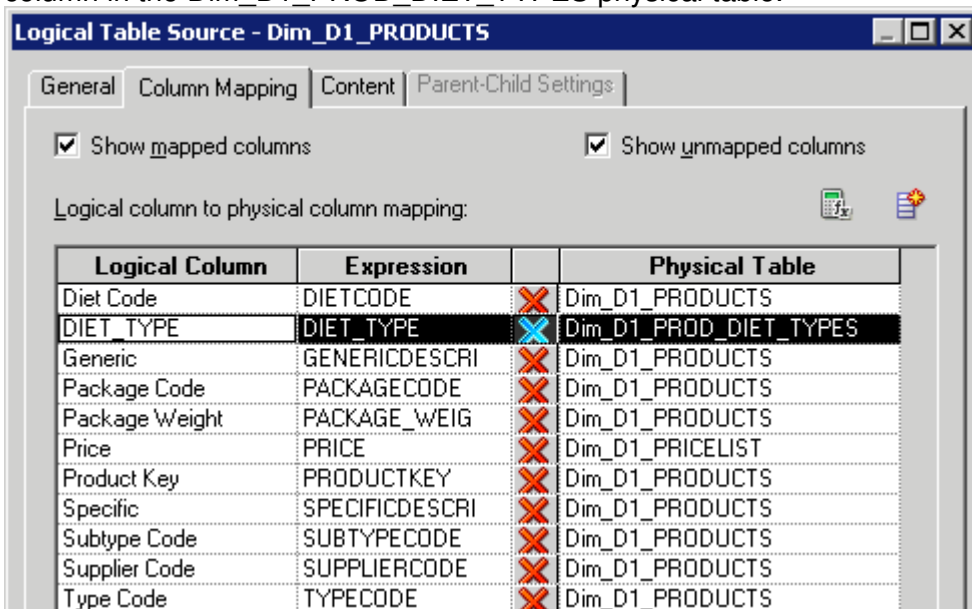
Tasks

1. In this step, you use an automated method to add new logical table sources and logical columns to the Dim-Product logical table.
 - a. In the Business Model and Mapping layer, expand **Dim-Product > Sources** so that the **Dim_D1_PRODUCTS** logical table source is visible.
 - b. In the Physical layer, expand the **Dim_D1_PROD_DIET_TYPES** table and select the **DIET_TYPE** column.
 - c. From the Physical layer, drag the **DIET_TYPE** column onto the **Dim_D1_PRODUCTS** logical table source in the Dim-Product logical table in the Business Model and Mapping layer.

- d. Confirm that the **DIET_TYPE** column is added to the Dim-Product logical table.



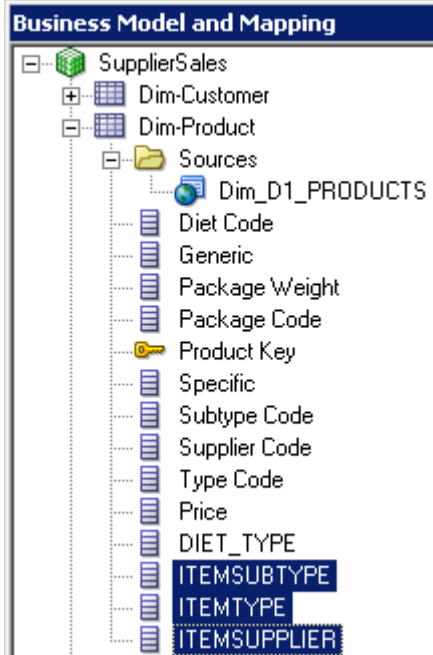
- e. Double-click the **Dim_D1_PRODUCTS** logical table source to view its properties.
 f. Click the **Column Mapping** tab.
 g. Notice that the DIET_TYPE logical column is mapped to the DIET_TYPE physical column in the Dim_D1_PROD_DIET_TYPES physical table.



- h. Click **OK** to close the Logical Table Source properties dialog box.
 2. Repeat the steps to map three more tables and columns to the Dim-Product logical table.
 a. Drag the following columns from the Physical layer to the Dim_D1_PRODUCTS logical table source. Be sure to drag to the Dim_D1_PRODUCTS logical table source, not the Dim-Product logical table.

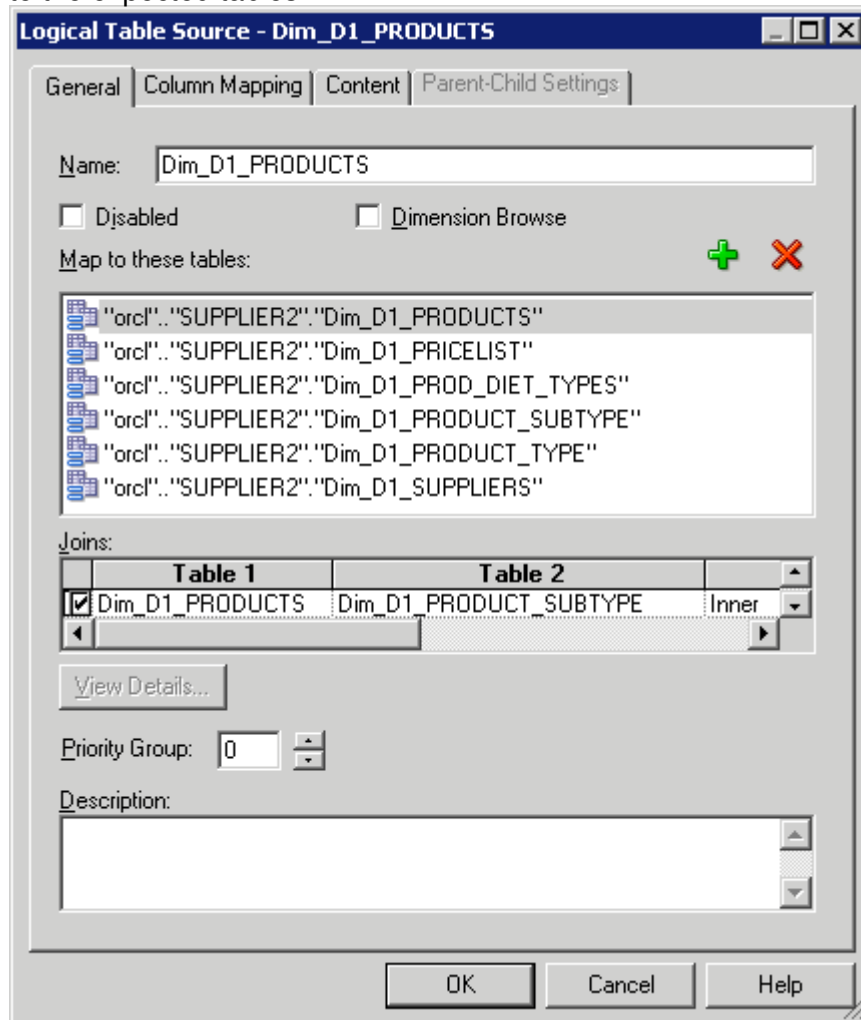
Physical Table	Physical Column
Dim_D1_PRODUCT_SUBTYPE	ITEMSUBTYPE
Dim_D1_PRODUCT_TYPE	ITEMTYPE
Dim_D1_SUPPLIERS	ITEMSUPPLIER

- b. Confirm that the three columns are added to the Dim-Product logical table.



- c. Open the **Dim_D1_PRODUCTS** logical table source properties dialog box, click the **General** tab, and verify that the Dim_D1_PRODUCTS logical table source now maps

to the expected tables.



- d. Click the **Column Mapping** tab to verify the logical column to physical column mappings.

Logical Column	Expression		Physical Table
Diet Code	DIETCODE	✗	Dim_D1_PRODUCTS
Generic	GENERICDESCRIP	✗	Dim_D1_PRODUCTS
Package Weight	PACKAGE_WEIGHT	✗	Dim_D1_PRODUCTS
Package Code	PACKAGECODE	✗	Dim_D1_PRODUCTS
Product Key	PRODUCTKEY	✗	Dim_D1_PRODUCTS
Specific	SPECIFICDESCRIP	✗	Dim_D1_PRODUCTS
Subtype Code	SUBTYPECODE	✗	Dim_D1_PRODUCTS
Supplier Code	SUPPLIERCODE	✗	Dim_D1_PRODUCTS
Type Code	TYPECODE	✗	Dim_D1_PRODUCTS
Price	PRICE	✗	Dim_D1_PRICELIST
DIET_TYPE	DIET_TYPE	✗	Dim_D1_PROD_DIET_TYPES
ITEMSUBTYPE	ITEMSUBTYPE	✗	Dim_D1_PRODUCT_SUBTYPE
ITEMTYPE	ITEMTYPE	✗	Dim_D1_PRODUCT_TYPE
ITEMSUPPLIER	ITEMSUPPLIER	✗	Dim_D1_SUPPLIERS

- e. Click **OK** to close the Logical Table Source properties dialog box.

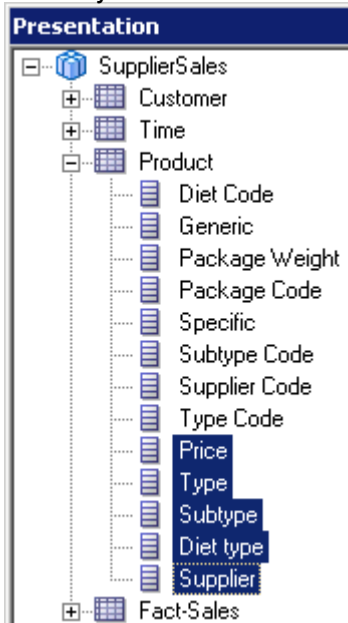
3. Rename the new logical columns in the Dim-Product logical table:

From...	To...
DIET_TYPE	Diet Type
ITEMSUBTYPE	Subtype
ITEMTYPE	Type
ITEMSUPPLIER	Supplier

4. Add the new product information to the SupplierSales presentation catalog.
- a. Drag the five new columns from the Business Model and Mapping layer onto the **Product** table in the **SupplierSales** presentation catalog and reorder the columns in the Presentation layer as follows:

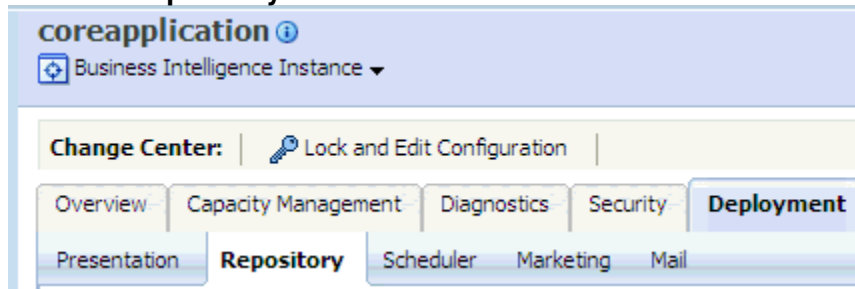
Column Name
Price
Type
Subtype
Diet Type
Supplier

- b. Check your work:

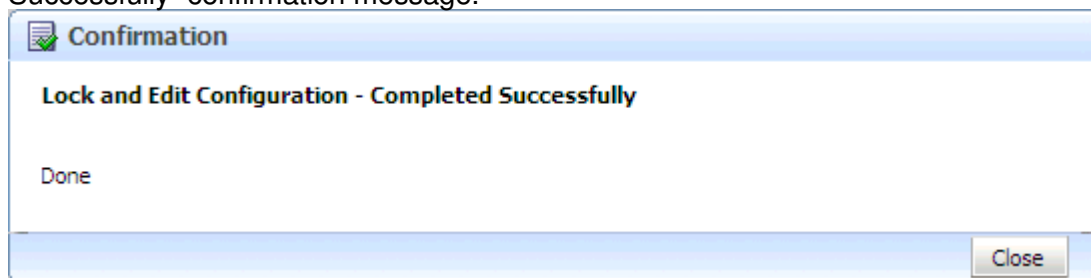


- c. Save the repository.
- d. Check consistency. You should receive a “Consistency check didn’t find any errors, warnings or best practice violations.” message.
- e. Click **OK** to close the Consistency Check Manager message. If you receive any error or warning messages, fix them before proceeding.
- f. Close the repository.
- g. Leave the Administration Tool open.

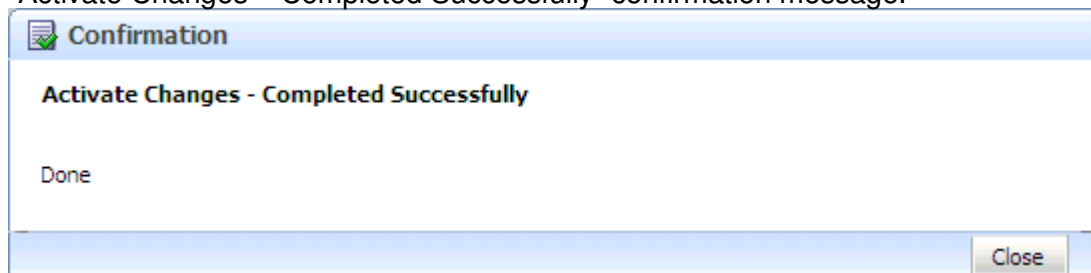
5. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.



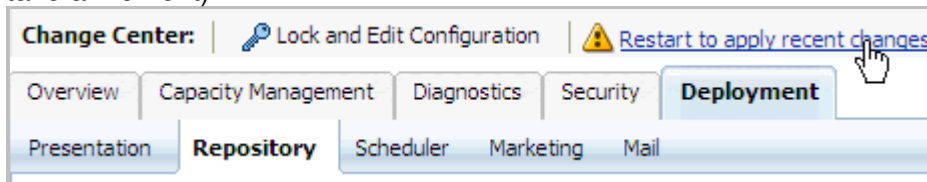
- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with a new extension (for example, ABC_BI0007).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



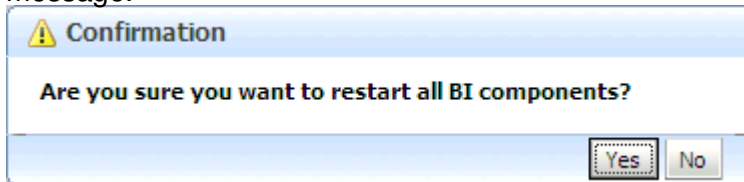
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



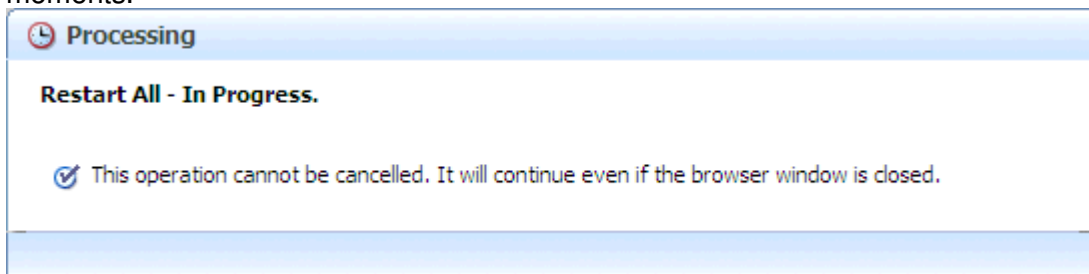
- p. On the Overview page, click **Restart**.



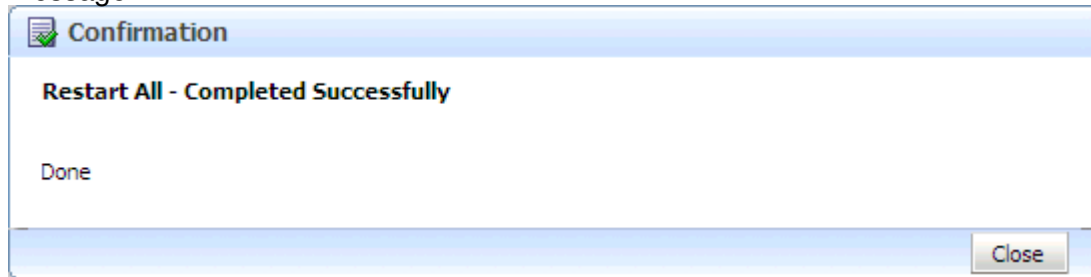
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



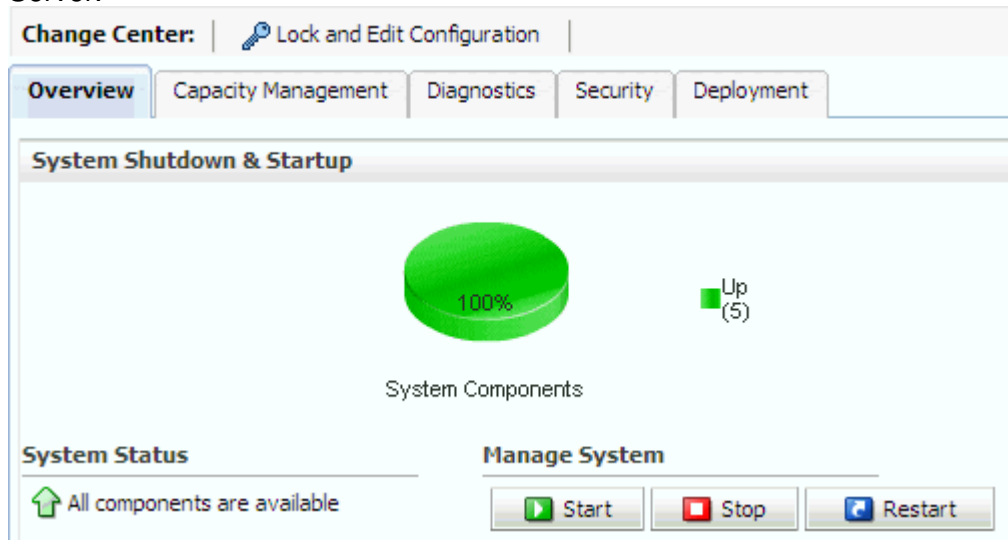
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message



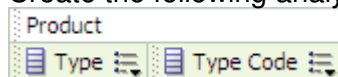
- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Enterprise Manager open.
6. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out of Analysis Editor and click **here** to sign in.

Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- Sign in as **weblogic** with password **welcome1**.
- In the Create section, click **Analysis** to open the Select Subject Area window.
- Click **SupplierSales** to open Analysis Editor.
- Create the following analysis:



- f. Click the **Results** tab to view the results.

Type	Type Code
Baking	100
Beef	101
Beverage	102
Bread	103
Cereal	104
Cheese	105
Condiments	106
Dessert	107
Entree	108
Frozen	109
Grains	110
Lamb	111
Non-food	112
Pasta	113
Pork	114
Poultry	115
Rice	116

7. Examine the query log to determine which table or tables have been accessed for this query. You can click the **Administration** link or use Fusion Middleware Control. The following instructions are for Fusion Middleware Control.
- Return to **Fusion Middleware Control**, which should still be open.
 - Click the **Diagnostics** tab.
 - Click the **Log Messages** subtab.
 - Scroll to the bottom of the window to the **View / Search Log Files** section.
 - Click **Server Log**.
 - On the Log Messages screen, leave the data range set to **Most Recent, 1 Days**.
 - Deselect all message types except for **Trace**.
 - In the Message field, enter **sending query to database**.

Log Messages

☐ Search

Selected Targets (1)

Date Range:

* Message Types: ☐ Incident Error ☐ Error ☐ Warning ☐ Notification ☒ Trace ☐ Unknown

Message:

- Click **Search**.
- Select the last message in the list. This is the most recent query sent to the database.
- In the bottom pane, click the **Collapse Pane button** (arrow on the right side) to view the log message.
- Examine the query log. Notice that three tables, Dim_D1_PRODUCTS, Dim_D1_PRODUCT_SUBTYPE, and Dim_D1_PRODUCT_TYPE, have all been accessed, despite the fact that the Dim_D1_PRODUCT_TYPE table contains both

columns. The log should look similar to the following screenshot:

```
WITH
SAWITH0 AS (select distinct T99.TYPECODE as c1,
      T359.ITEMTYPE as c2
from
      D1_PRODUCTS T99 /* Dim_D1_PRODUCTS */ ,
      D1_PRODUCT_SUBTYPE T355 /* Dim_D1_PRODUCT_SUBTYPE */ ,
      D1_PRODUCT_TYPE T359 /* Dim_D1_PRODUCT_TYPE */
where ( T99.SUBTYPECODE = T355.SUBTYPECODE and T355.TYPECODE = T359.TYPECODE ) )
select distinct 0 as c1,
      D1.c1 as c2,
      D1.c2 as c3
from
      SAWITH0 D1
order by c3, c2
```

- m. Why are all the three tables included in the query? All three tables are included because of the join conditions. The only way the query can access the Dim_D1_PRODUCT_TYPE table is through the D1_PRODUCTS table. In the next practice you learn how to instruct Oracle BI Server to directly access the Dim_D1_PRODUCT_TYPE table via a second logical table source.
- n. Click the **Restore Pane button** in the upper right corner to restore the Log Messages pane.
- o. Leave Fusion Middleware Control open.
- p. Sign out of Analysis Editor.

Practice 7-4: Adding a New Logical Table Source

Goal

To add a second logical table source to the Product dimension

Scenario

You examine the physical sources for the Dim-Product logical table and discover that the Type Code and Type columns are mapped to different physical tables. You also discover that the information for both columns is stored in a common physical table, Dim_D1_PRODUCT_TYPE.

In order to model the most economical method for Oracle BI Server to generate queries against these two columns, you add a second logical table source to the Dim-Product logical table so that Oracle BI Server queries only one table for the Type Code and Type columns.

Outcome

In the Business Model and Mapping layer, Type is added as second logical table source for the Dim-Product logical table.

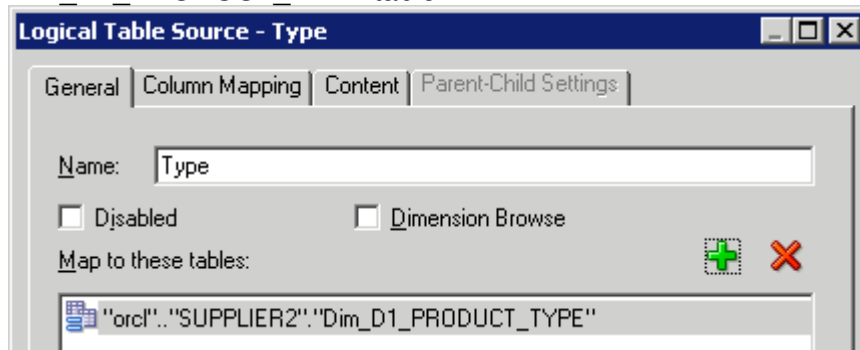
Time

10 minutes

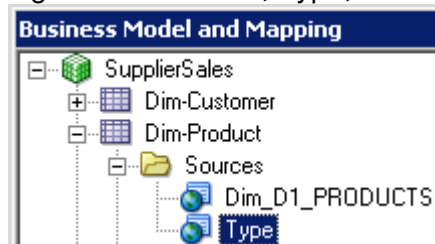
Tasks

1. Return to the Administration Tool, which should still be open, and open the **ABC** repository file in offline mode with the repository password **welcome1**.
2. Examine the existing column mappings for the Type and Type Code columns.
 - a. In the Business Model and Mapping layer, expand the **Sources** folder of the Dim-Product logical table
 - b. Double-click the **Dim_D1_PRODUCTS** logical table source to open the Logical Table Source dialog box.
 - c. Click the **Column Mapping** tab.
 - d. Notice that the Type Code logical column is mapped to the TYPECODE physical column in Dim_D1_PRODUCTS.
 - e. Notice that the Type logical column is mapped to the ITEMTYPE physical column in Dim_D1_PRODUCT_TYPE.
 - f. Click **Cancel** to close the Logical Table Source dialog box.
3. Determine which physical table stores information for both Type Code and Type.
 - a. In the Physical layer, expand the **Dim_D1_PRODUCT_TYPE** physical table.
 - b. Verify that this table stores information for both ITEMTYPE and TYPECODE.
4. Model a new mapping for the Type Code logical column by creating a second logical table source for the Dim-Product logical table.
 - a. In the Business Model and Mapping layer, right-click the **Dim-Product** logical table and select **New Object > Logical Table Source**.
 - b. On the **General** tab, enter **Type** in the Name field.
 - c. Click the **Add** button (green plus sign).
 - d. In the Browse dialog box, double-click the **Dim_D1_PRODUCT_TYPE** physical table to select it. The Type logical table source is now mapped to the

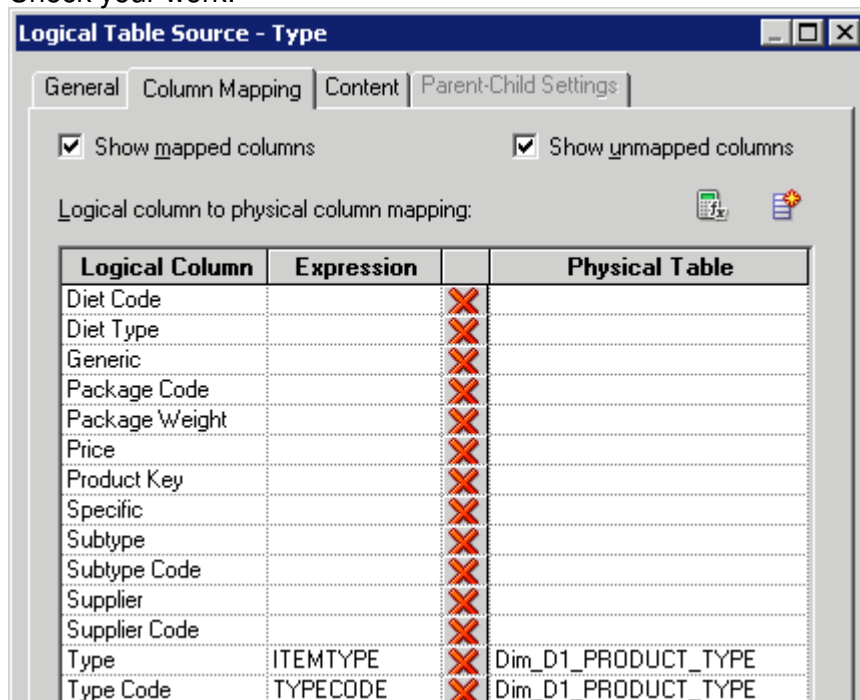
Dim_D1_PRODUCT_TYPE table.



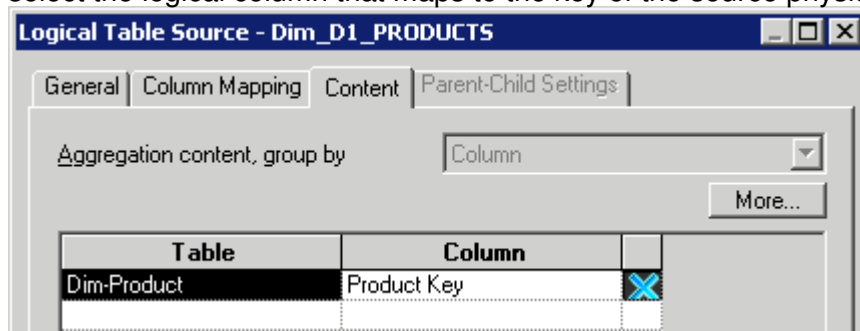
- e. Click **OK** to close the Logical Table Source properties dialog box. Notice that the new logical table source, Type, is added to the Sources folder.



- f. Double-click the **Type** logical table source to view its properties.
- g. Click the **Column Mapping** tab. Make sure that **Show unmapped columns** is selected.
- h. Use the Expression field to map the **Type Code** logical column to the **Dim_D1_PRODUCT_TYPE.TYPECODE** physical column.
- i. Repeat to map the **Type** logical column to the **Dim_D1_PRODUCT_TYPE.ITEMTYPE** physical column.
- j. Check your work:

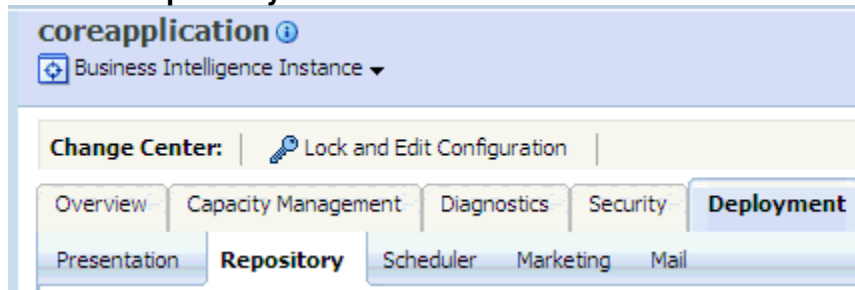


- k. Click **OK** to close the Logical Table Source dialog box. There are now two logical table sources for the Dim-Product logical table and the Type Code logical column maps to physical columns in both the Dim_D1_PRODUCT_TYPE and Dim_D1_PRODUCTS tables.
5. Specify the aggregation content for the logical table sources. To use a source correctly, Oracle BI Server has to know what each source contains in terms of the business model. Therefore, you must define aggregation content for each logical table source.
 - a. Double-click the **Dim_D1_PRODUCTS** logical table source.
 - b. Click the **Content** tab.
 - c. Notice that for “Aggregation content, group by” column is selected and grayed out. Although you have the option to specify aggregation content by column or logical level, it is recommended that you use logical levels exclusively. However, for the purpose of demonstrating results in this practice, you define the content by columns. Later, in the “Using Aggregates” lesson, you learn how to define content using logical levels.
 - d. In the Table field, select **Dim-Product**.
 - e. In the Column field, select **Product Key**. When multiple logical columns could be used, select the logical column that maps to the key of the source physical table.

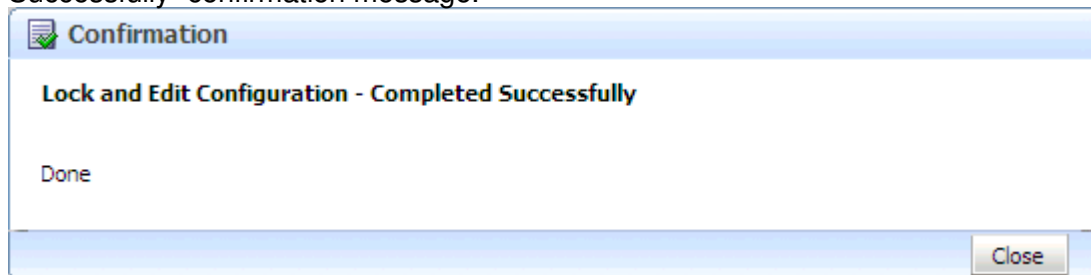


- f. Click **OK** to close the Logical Table Source dialog box.
- g. Double-click the **Type** logical table source.
- h. On the Content tab, set table to **Dim-Product** and column to **Type Code**.
- i. Click **OK** to close the Logical Table Source dialog box.
- j. Save the repository.
- k. Click **Yes** to check consistency. If you receive any error or warning messages, fix them before proceeding.
- l. If the repository is consistent, close the repository.
- m. Leave the Administration Tool open.
6. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.

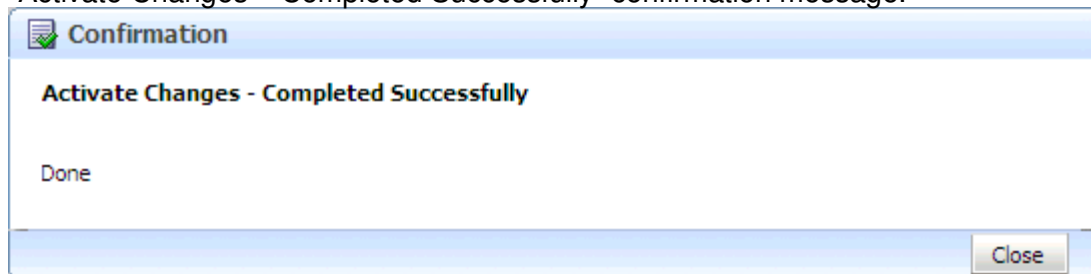
- e. Click the **Repository** subtab.



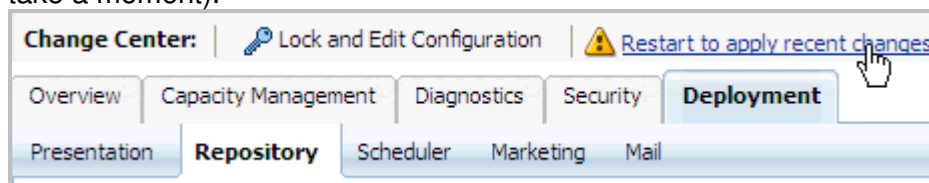
- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository**.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays ABC with an extension (for example, ABC_BI0008).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



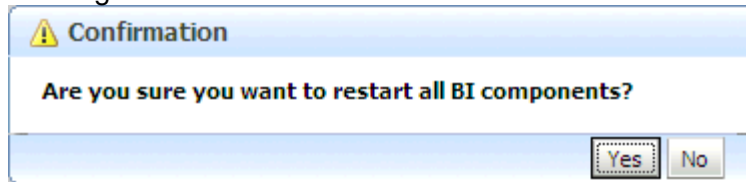
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



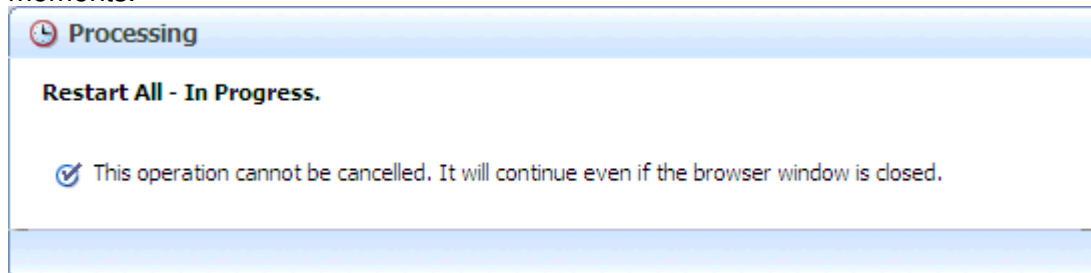
- p. On the Overview page, click **Restart**.



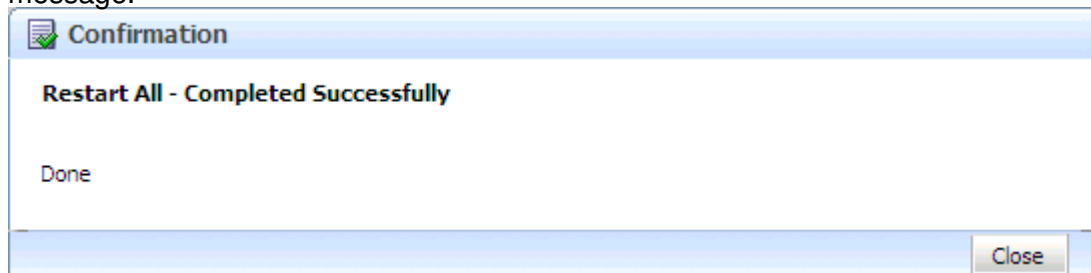
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



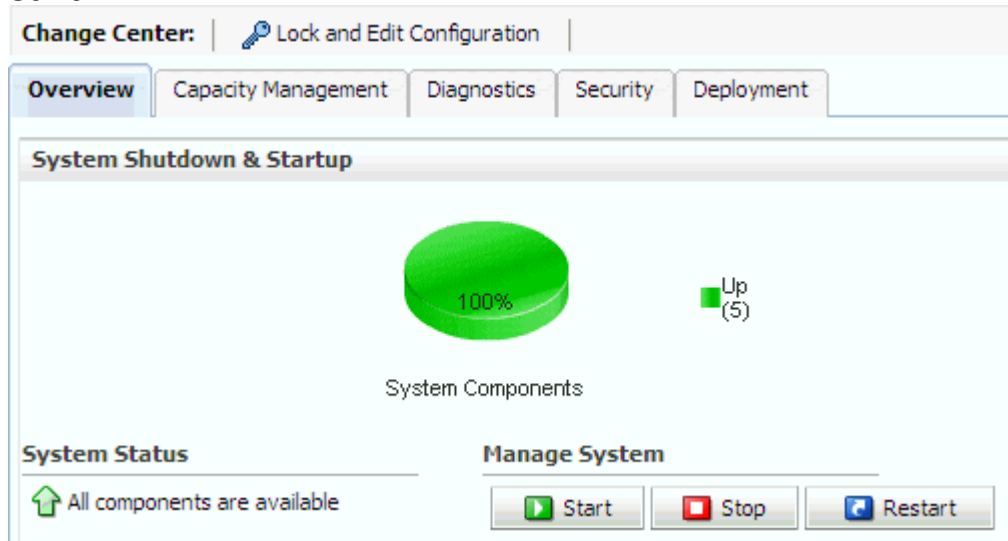
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



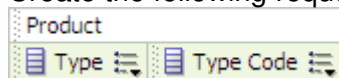
- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
7. Open Analysis Editor to execute queries and test the SupplierSales business model.
 - a. Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.

Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- b. Sign in as **weblogic** with password **welcome1**.
 - c. In the Create section, click **Analysis** to open the Select Subject Area window.
 - d. Click **SupplierSales** to open Analysis Editor.
8. Create the same request as in the previous practice to check your work.
 - a. Create the following request:



- b. Click the **Results** tab to view the results.

Type	Type Code
Baking	100
Beef	101
Beverage	102
Bread	103
Cereal	104
Cheese	105
Condiments	106
Dessert	107
Entree	108
Frozen	109
Grains	110
Lamb	111
Non-food	112
Pasta	113
Pork	114
Poultry	115
Rice	116

9. Examine the query log to determine which table or tables have been accessed for this query. You can click the **Administration** link or use Fusion Middleware Control. The following instructions are for Fusion Middleware Control.
 - a. Return to the **Diagnostics** tab in Fusion Middleware Control, which should still be open.
 - b. Click the **Log Messages** subtab.
 - c. Scroll to the bottom of the window to the **View / Search Log Files** section.
 - d. Click **Server Log**.
 - e. On the Log Messages screen, leave the data range set to **Most Recent, 1 Days**.
 - f. Deselect all message types except for **Trace**.
 - g. In the Message field, enter **sending query to database**.

Log Messages

Selected Targets (1)

Date Range:

* Message Types: ☐ Incident Error ☐ Error ☐ Warning ☐ Notification ☒ Trace ☐ Unknown

Message:

- h. Click **Search**.
- i. Select the last message in the list. This is the most recent query sent to the database.
- j. In the bottom pane, click the **Collapse Pane button** (arrow on the right side) to view the log message.
- k. Examine the query log. Verify that only one table, Dim_D1_PRODUCT_TYPE, is accessed by the query. Because you defined the aggregation content in the logical table sources for the Dim-Product logical table, Oracle BI Server executes its query against the most “economical” source, which in this example is the

Dim_D1_PRODUCT_TYPE table. To provide the best performance, BI Server bypasses the physical joins and accesses the table directly rather through multiple tables as in the previous example. The log should look similar to the following screenshot:

```
SAWITH0 AS (select T502.TYPECODE as c1,  
                  T502.ITEMTYPE as c2  
from  
    D1_PRODUCT_TYPE T502 /* Dim_D1_PRODUCT_TYPE */ )  
select distinct 0 as c1,  
    D1.c1 as c2,  
    D1.c2 as c3  
from  
    SAWITH0 D1  
order by c3, c2
```

- l. Click the **Restore Pane button** in the upper right corner to restore the Log Messages pane.
- m. Leave Fusion Middleware Control open.
- n. Sign out of Analysis Editor.

Practices for Lesson 8: Adding Calculations to a Fact

Chapter 8

Overview of Practices for Lesson 8

Lesson Overview

In these practices, you will create calculation measures in an Oracle BI repository.

Practice 8-1: Creating Calculation Measures

Goal

To create calculation measures

Scenario

You use two different methods to create measures that contain calculations. First, you create a measure that is derived from other existing logical columns as a way to apply post-aggregation calculations to the measure. Then you create a measure using physical columns as a way to apply pre-aggregation calculations to the measure. In both cases, you use analyses and the query log to verify your results.

Outcome

A Dollars per Units Ordered logical column with post-aggregation calculations derived from existing logical columns. A Price x Units Ordered logical column with pre-aggregation calculations based on physical columns.

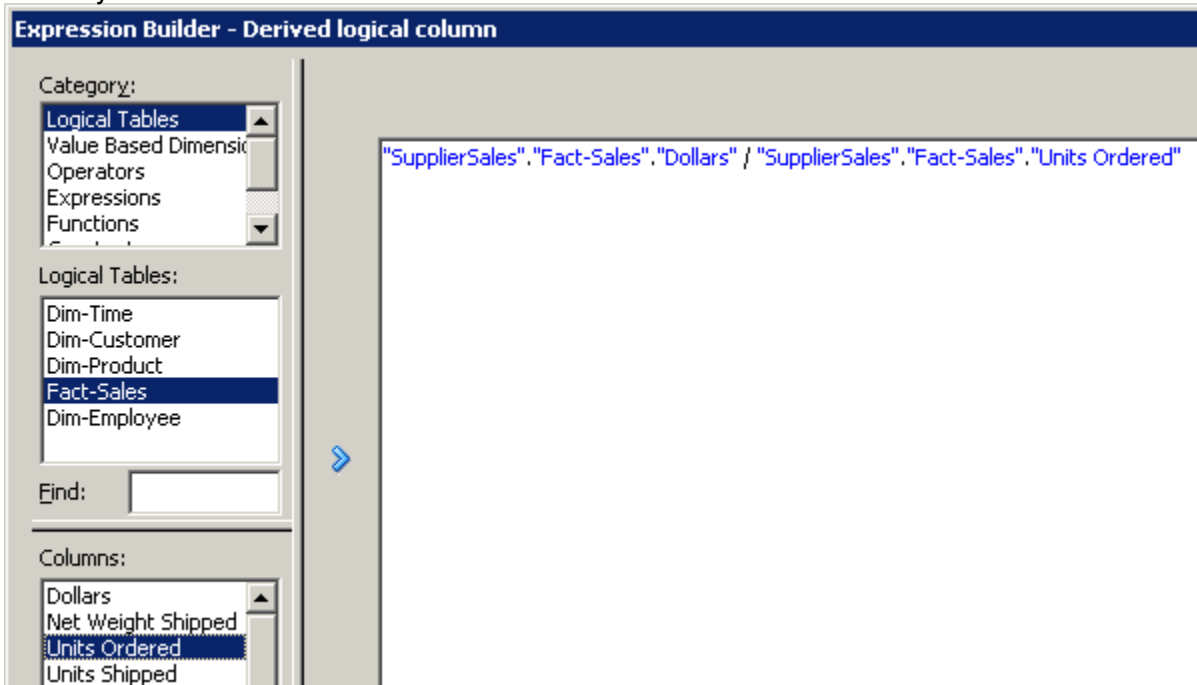
Time

35 minutes

Tasks

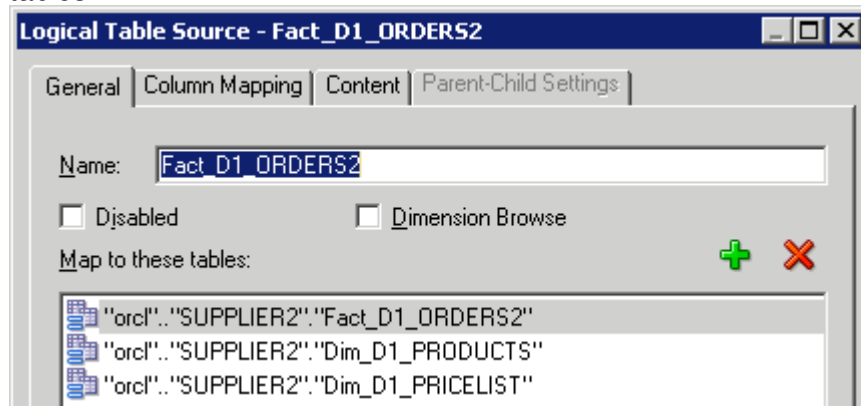
1. In this step, you define a new logical measure that is derived from other existing logical columns as a way to apply post-aggregation calculations to the measure.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with the repository password **welcome1**.
 - b. In the Business Model and Mapping layer, right-click the **Fact-Sales** logical table and select **New Object > Logical Column**.
 - c. On the General tab, name the column **Dollars per Units Ordered**.
 - d. Click the **Column Source** tab.
 - e. Select **Derived from existing columns using an expression**.
 - f. Click the **Edit Expression** button to open Expression Builder.
 - g. Under Category, select **Logical Tables**.
 - h. Under Logical Tables, select **Fact-Sales**.
 - i. Under Columns, select **Dollars**.
 - j. Click the **Insert selected item** arrow to add Dollars to the expression.
 - k. Click the **division sign** on the toolbar.
 - l. Under Columns, double-click the **Units Ordered** logical column to insert it in the formula.

- m. Check your results:



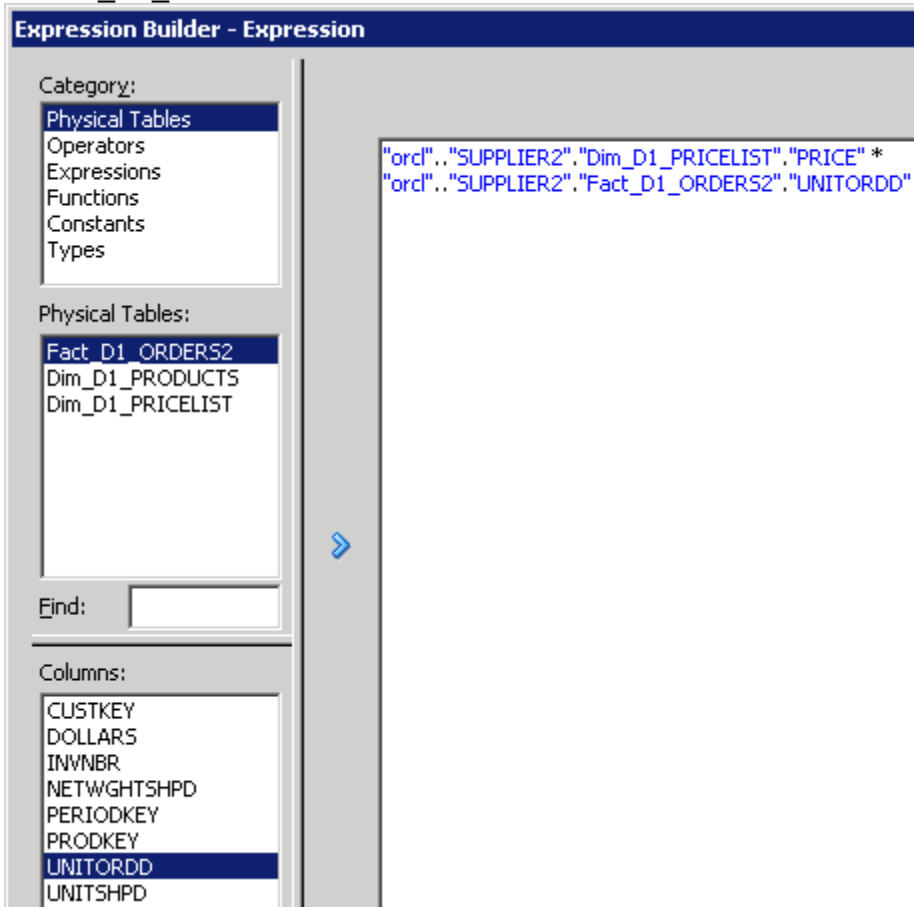
- n. Click **OK** to close the Expression Builder. Notice that the expression is now displayed in the text edit box of the Logical Column dialog box.
- o. In the "Derives from" field, observe the calculation that will be sent to the data source in the physical SQL.
- p. Click **OK** to close the Logical Column dialog box. Dollars per Units Ordered is added to the business model.
- q. Add the new **Dollars per Units Ordered** column to the **Fact-Sales** table in the **SupplierSales** presentation catalog.
2. To the Fact-Sales logical table source, add mappings that are needed to create a measure that uses physical columns to define a calculation formula.
- a. In the Business Model and Mapping layer, expand **Fact-Sales > Sources**.
- b. In the Physical layer, expand **Dim_D1_PRODUCTS**.
- c. Drag **PRODUCTKEY** from Dim_D1_PRODUCTS to **Fact-Sales > Sources > Fact_D1_ORDERS2**. This adds a PRODUCTKEY logical column to the Fact-Sales logical table.
- d. In the Physical layer, expand **Dim_D1_PRICELIST**.
- e. Drag **PRICE** from Dim_D1_PRICELIST to **Fact-Sales > Sources > Fact_D1_ORDERS2**. This adds a PRICE logical column to the Fact-Sales logical table.
- f. Double-click the **Fact_D1_ORDERS2** logical table source to open the Logical Table Source properties dialog box.
- g. Click the **General** tab. Notice that as a result of dragging the physical columns to the logical table source, the Fact_D1_ORDERS2 logical table source now maps to three

tables.



- h. Why is it necessary to map to both product tables? Because of the join relationships. You are going to use the PRICE column in a calculated measure, and Fact_D1_ORDERS2 can only access Dim_D1_PRICELIST.PRICE via Dim_D1_PRODUCTS.
3. Create a measure using physical columns as a way to apply pre-aggregation calculations to the measure.
- Click the **Column Mapping** tab.
 - Click the **Add New Column** button to open the Logical Column dialog box.
 - Name the column **Price x Units Ordered**.
 - Click the **Aggregation** tab and set the aggregation to **Sum**.
 - Click **OK** to close the logical column dialog box.
 - Click the **Edit Expression** button to open the Expression Builder.
 - Under Category, select **Physical Tables**.
 - Under Physical Tables, select **Dim_D1_PRICELIST**.
 - Under Columns, double-click **PRICE** to add it to the formula.
 - Click the **multiplication icon** on the toolbar to add it to the formula.

- k. Add **Fact_D1_ORDERS2.UNITORDD** to the formula.

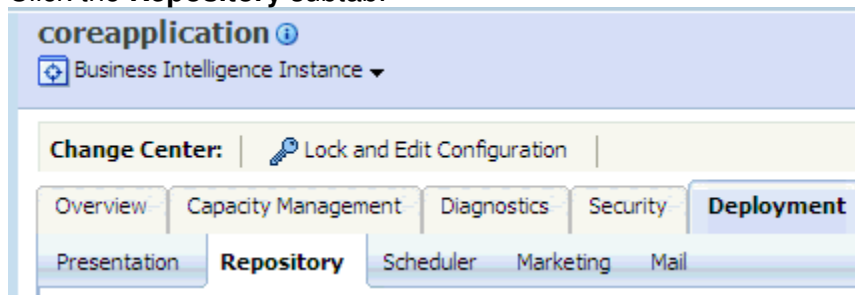


- l. Click **OK** to close the Expression Builder.
- m. Notice that you can see the formula in the Expression column for Price x Units Ordered on the Column Mapping tab. You may need to adjust the column width to see the entire formula.
- n. Click **OK** to close the Logical Table Source properties dialog box.
- o. Double-click the **Price x Units Ordered** logical column.
- p. Click the **Column Source** tab and notice that Price x Units Ordered is not derived from existing columns using an expression, but rather from physical mappings. In the Derives from field, notice that the formula multiplies the physical columns first and then sums the result.

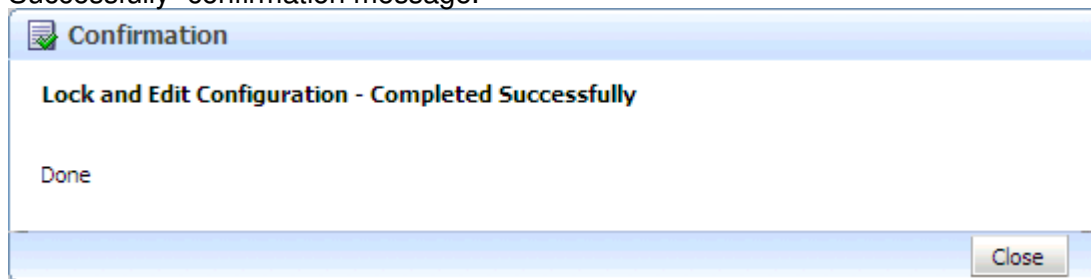
Derives from:
`sum(Fact_D1_ORDERS2.UNITORDD * Dim_D1_PRICELIST.PRICE)`

- q. Click **OK** to close the Logical Column properties dialog box.
- r. Add **Price x Units Ordered** to the **Fact-Sales** presentation table in the **SupplierSales** catalog in the Presentation layer.
- s. Delete the **PRICE** and **PRODUCTKEY** logical columns from the **Fact-Sales** logical table.
- t. Save the repository.
- u. Click **Yes** to check global consistency. Fix any errors or warnings before proceeding.
- v. Close the repository.

- w. Leave the Administration Tool open.
- 4. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.

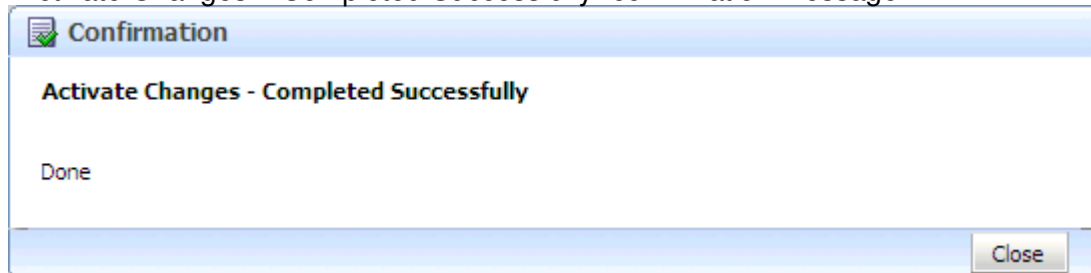


- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.

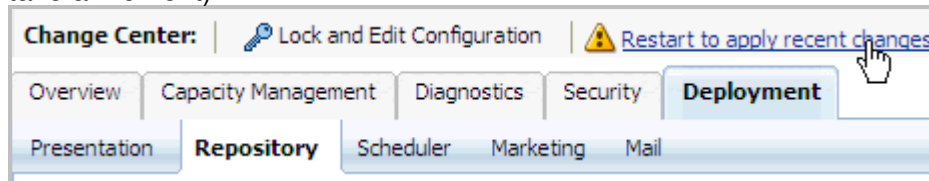


- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to
**D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplicatio
n_obis1\repository.**
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension (for example, ABC_BI0009).
- m. Click **Activate Changes**.

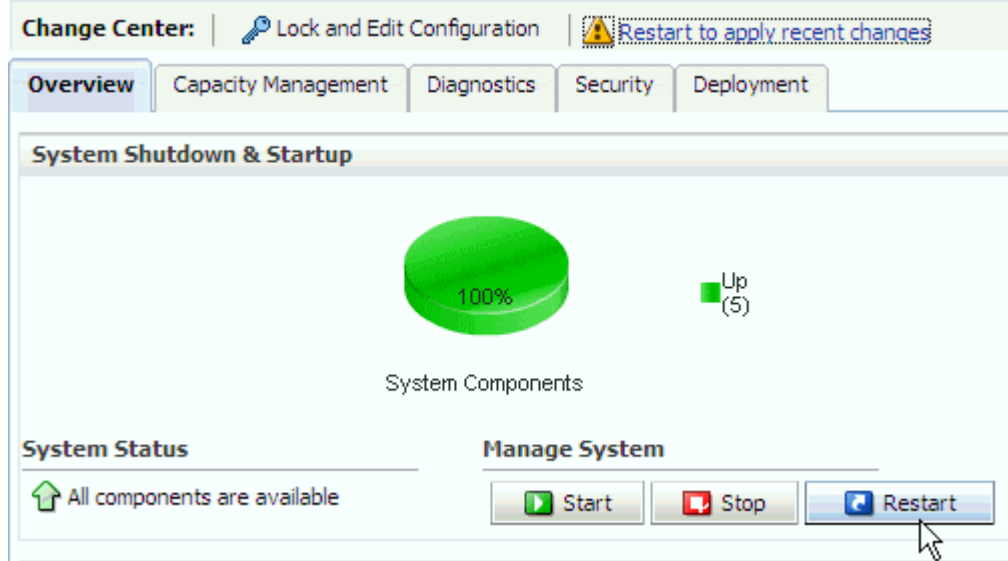
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



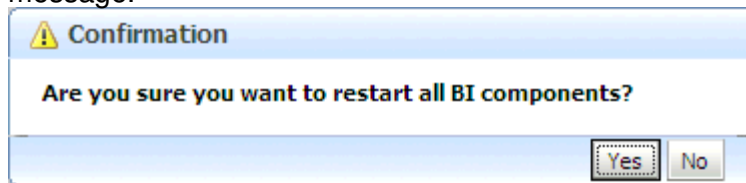
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



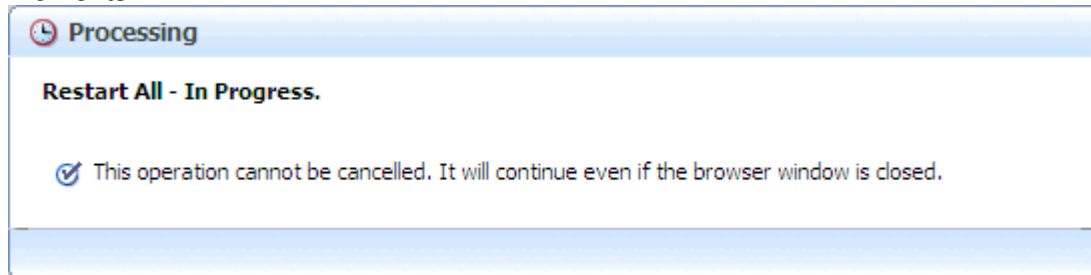
- p. On the Overview page, click **Restart**.



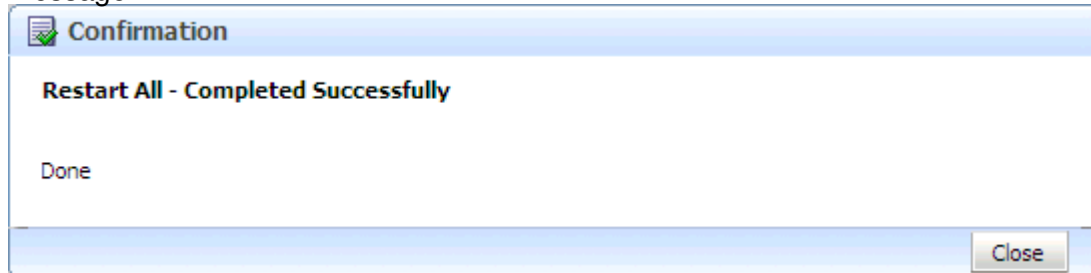
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



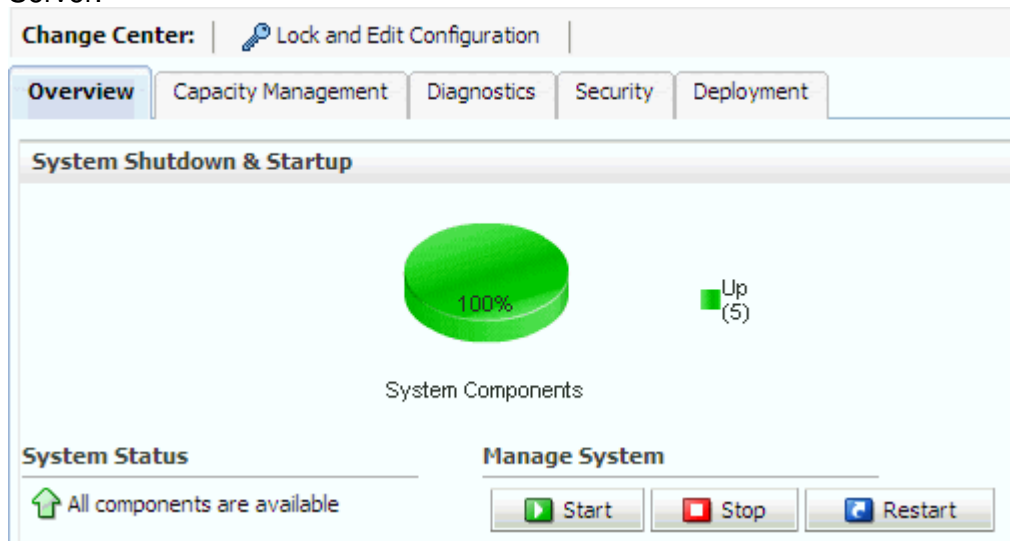
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



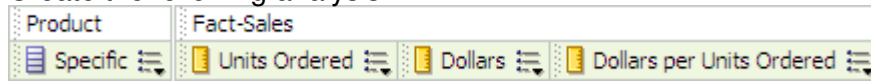
- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
5. Open Analysis Editor to execute queries and test the SupplierSales business model.
 - a. Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.

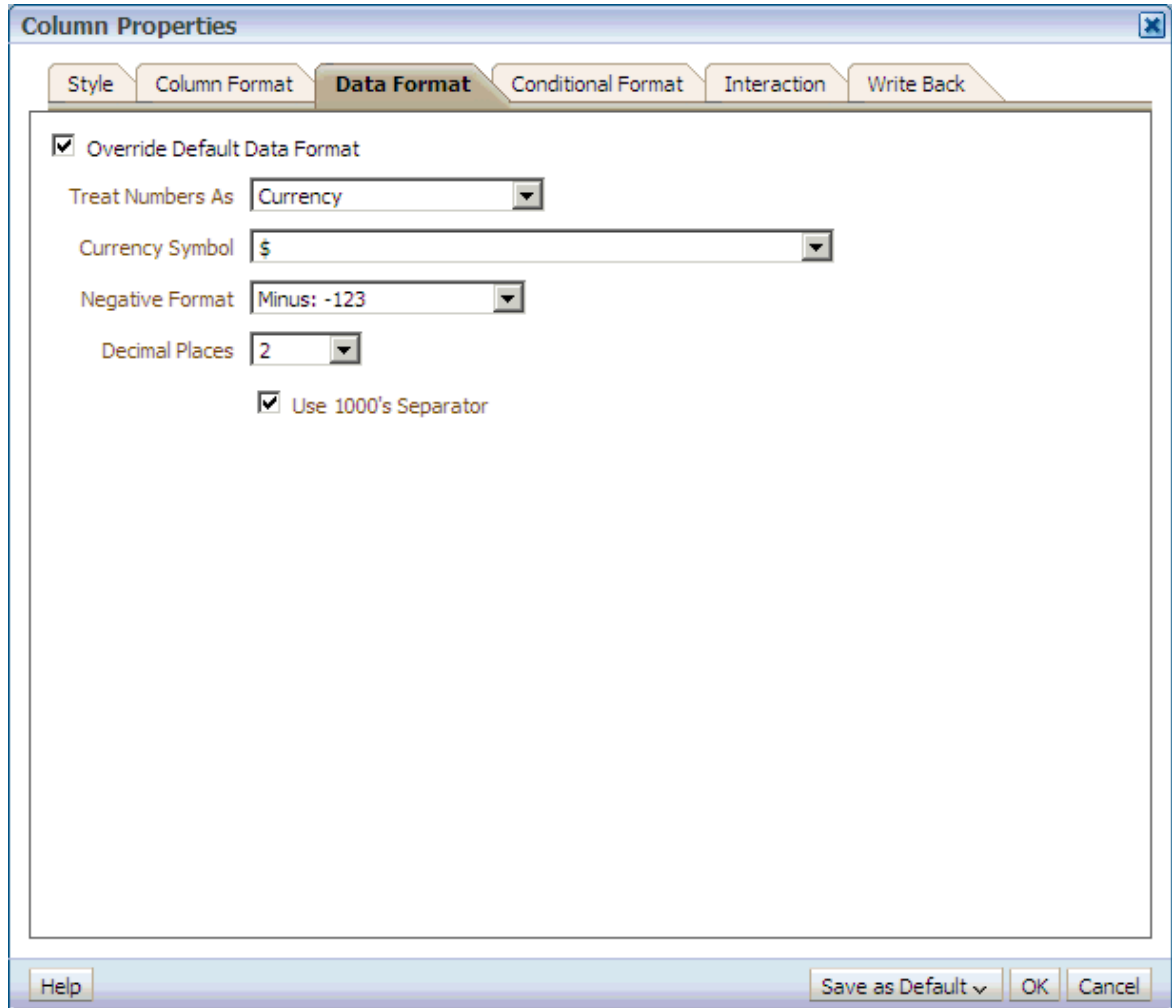
A light blue message box with the text: "Thank you for using Oracle Business Intelligence software. You have successfully signed out." and "To sign in again, click here." with a blue link.
 - b. Sign in as **weblogic** with password **welcome1**.
 - c. In the Create section, click **Analysis** to open the Select Subject Area window.
 - d. Click **SupplierSales** to open Analysis Editor.
 6. Create an analysis to check your work for the Dollars per Units Ordered column.

- a. Create the following analysis:



- b. Select **Column Properties** for **Dollars per Units Ordered**.

- c. Click the **Data Format** tab and set the data format using the following screenshot as a reference:



- d. Save as the default for Dollars per Units Ordered.

- e. Click **Results**.

Specific	Units Ordered	Dollars	Dollars per Units Ordered
*100 Ct Foam Containers 6"x6"x3"	2,531	\$50,192.40	\$19.83
*100 Ct Foam Plastic Plates 8"	9	\$950.07	\$105.56
*12 Ct Tulip Vase 8"	5,669	\$129,295.02	\$22.81
*20 Cu Ft Commercial Upright Freezer Unit w/Child Safety Latch, 4 Shelves	3	\$7,801.01	\$2,600.34
*Chef's Knife 8"	90	\$595.81	\$6.62
*Napkin/Straw/Condiment Holder Side-mount 16" x 6"	6	\$7,348.60	\$1,224.77
*Padded Bar Stool 36" Swivel Base	46	\$2,344.31	\$50.96
*Pizza Keeper 15" x 15"	272	\$14,278.79	\$52.50
*Push Broom 18" Wide	852	\$4,698.00	\$5.51
*Saute Pan 12"	1,368	\$13,433.06	\$9.82
10 Ct Frozen Bread Dough White 2 lb Loaf	336	\$12,248.16	\$36.45
10 Pak Guest Check Tablet w/Tearoff Receipt 100 Sheets	822	\$29,701.40	\$36.13
100 Ct 1/3 lb Hamburger Patties	10,538	\$1,424,403.25	\$135.17

7. Examine the query log.

- a. Click the **Administration** link or return to the **Diagnostics** tab in Fusion Middleware Control and locate your query. The query log should be similar to the following:

```
SAWITH0 AS (select sum(T90.UNITORDD) as c1,
                sum(T90.DOLLARS) as c2,
                T99.SPECIFICDESCRIPTIN as c3
from
    D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */ ,
    D1_PRODUCTS T99 /* Dim_D1_PRODUCTS */
where ( T90.PRODKEY = T99.PRODUCTKEY )
group by T99.SPECIFICDESCRIPTIN)
select distinct 0 as c1,
    D1.c3 as c2,
    D1.c2 / nullif( D1.c1, 0) as c3,
    D1.c2 as c4,
    D1.c1 as c5
from
    SAWITH0 D1
order by c2
```

- b. Notice that UNITORDD and DOLLARS are summed first in the query, and then the division of Dollars by Units Ordered is calculated in the outer query block (D1.c2 / nullif (D1.c1, 0) as c3 in this example). Because you defined the Dollars per Units Ordered calculation using logical columns, the columns are summed first and then the division is calculated.

8. Check your work for the Price x Units Ordered column.

- a. Return to Analysis Editor and create the following new analysis:

Product	Fact-Sales		
Specific	Price	Units Ordered	Price x Units Ordered

- b. Modify the data format for both **Price** and **Price x Units Ordered** using the following screenshot as a reference:

Column Properties

Style | Column Format | **Data Format** | Conditional Format | Interaction | Write Back

☒ Override Default Data Format

Treat Numbers As: Currency

Currency Symbol: \$

Negative Format: Minus: -123

Decimal Places: 2

☒ Use 1000's Separator

Help | Save as Default | OK | Cancel


- c. Click **Results**.

Specific	Price	Units Ordered	Price x Units Ordered
10 Ct Frozen Bread Dough White 2 lb Loaf	\$33.31	336	\$11,192.16
100 Ct 1/3 lb Hamburger Patties	\$144.58	10,538	\$1,523,584.04
100 Ct Asst Children's Snacks 2 oz	\$14.32	7,822	\$112,011.04
100 Ct Beef Bouillon Cubes .5 oz	\$17.23	950	\$16,368.50
12 Pak Frozen Chicken 4 oz	\$29.43	166,060	\$4,887,145.80
12 Pak Frozen Peas 16 oz	\$17.44	164,120	\$2,862,252.80
2 Pak Coconut Cream Beverage Base Imported 14 oz	\$79.10	24	\$1,898.40
2 Pak Frank's Mustard 16 oz	\$20.46	25,860	\$529,095.60
2 Pak Frozen Chicken Wings 5 lbs	\$37.18	20,198	\$750,961.64
2 Pak Frozen Cornish Game Hens 8 oz	\$41.74	115	\$4,800.10
2 Pak Frozen Duck in Orange Sauce 6 oz	\$84.21	281	\$23,663.01
2 Pak Frozen Quail 8 oz	\$44.04	13	\$572.52
2 Pak Honey Glaze for Ham/Pork 16 oz	\$25.65	350	\$8,977.50

9. Examine the query log.

- a. Open the query log. Your results should resemble the following screenshot:

```
SAWITH0 AS (select sum(T90.UNITORDD) as c1,
                sum(T90.UNITORDD * T490.PRICE) as c2,
                T490.PRICE as c3,
                T99.SPECIFICDESCRIPTIN as c4
from
    D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */ ,
    D1_PRODUCTS T99 /* Dim_D1_PRODUCTS */ ,
    D1_PRICELIST T490 /* Dim_D1_PRICELIST */
where ( T90.PRODKEY = T99.PRODUCTKEY and T99.PRODUCTKEY = T490.PROD
group by T99.SPECIFICDESCRIPTIN, T490.PRICE)
select distinct 0 as c1,
    D1.c3 as c2,
    D1.c4 as c3,
    D1.c2 as c4,
    D1.c1 as c5
from
    SAWITH0 D1
order by c3, c2
```

- b. Notice that the multiplication of units ordered and price is calculated first and then summed: (**sum(T90.UNITORDD * T490.PRICE) as c2** in this example). Compare these results with the query results for the calculation that used existing logical columns in the formula, where the columns were summed first and then calculated. What are the advantages and disadvantages of defining a logical column in terms of other logical columns, rather than in terms of the physical sources directly? The advantage of defining a logical column formula based on existing logical columns is that you have to define it only once. When you create formulas based on physical columns, you have to map for each physical source from which it could be derived. However, sometimes you have no choice if you have to use physical columns to apply an aggregation rule after a calculation. What would happen if you deleted a logical column that is used to define the formula of another logical column? The derived column would not be deleted automatically. However, the tool would display an icon  that warns you about this condition. Don't do this! This is just for informational purposes.
- c. Why did deleting the PRICE and PRODUCTKEY logical columns from the Fact-Sales logical table have no impact on the query results for the Price x Units Ordered measure? Because Price x Units Ordered is based on physical columns. The PRICE and PRODUCTKEY columns were added to the Fact-Sales logical table only to create the mappings needed to create the Price x Units Ordered measure.
- d. Sign out of Analysis Editor.

Practice 8-2: Creating Calculation Measures by Using the Calculation Wizard

Goal

To create calculation measures using the Calculation Wizard

Scenario

Using the Calculation Wizard, you create two calculation measures named Cuts and Percent Not Shipped. The Cuts measure calculates the difference between the units ordered and units shipped. The Percent Not Shipped measure calculates what percentage of the units ordered has not been shipped. The calculation measures created by the Calculation Wizard are based on existing logical columns.

Outcome

In the Business Model and Mapping layer, Cuts and Percent Not Shipped are added to the Fact-Sales logical table.

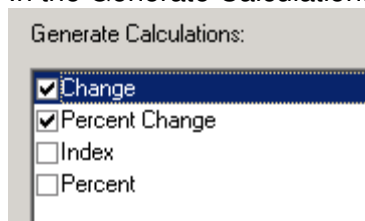
In the Presentation layer, Cuts and Percent Not Shipped are added to the Fact-Sales presentation table.

Time

20 minutes

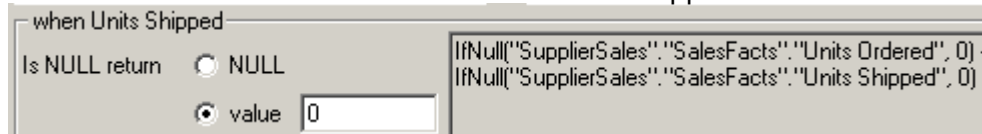
Tasks

1. In this step, you create two calculation measures using the Calculation Wizard.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository file in offline mode with the repository password **welcome1**.
 - b. In the Business Model and Mapping layer, expand the **Fact-Sales** table.
 - c. Right-click the **Units Ordered** column and select **Calculation Wizard**.
 - d. In the Calculation Wizard – Introduction dialog box, click **Next**.
 - e. In the **Choose columns** pane, **Fact-Sales** is selected. The columns that are available to include in the calculation appear in the right pane.
 - f. Select the **Units Shipped** check box.
 - g. Click **Next**.
 - h. In the Generate Calculations section, ensure that the **Change** and **Percent Change** check boxes are both selected.
 - i. In the Generate Calculations section, ensure that **Change** is highlighted.

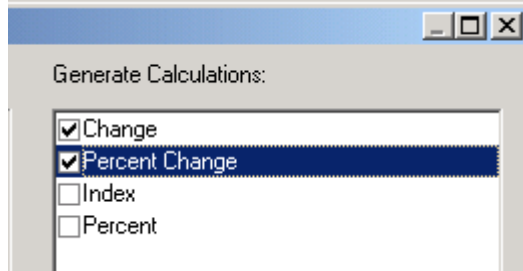


- j. In the Calculation Name field, enter **Cuts**.

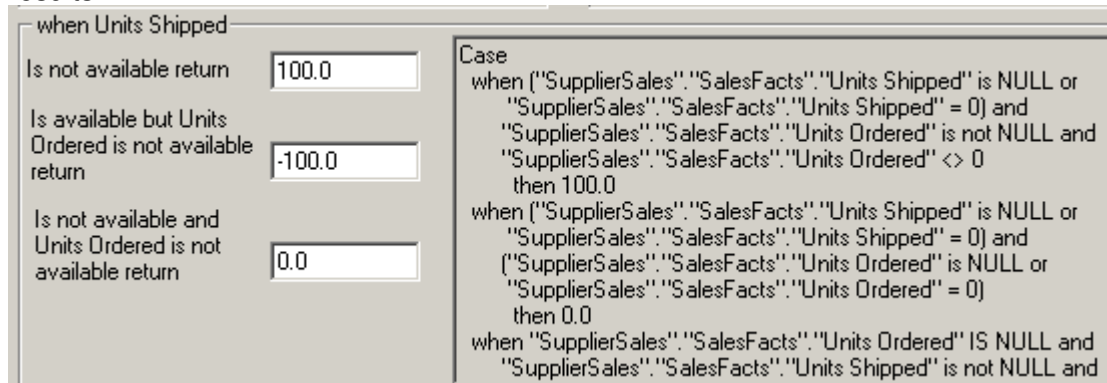
- k. Notice the results that are returned when Units Shipped is NULL:



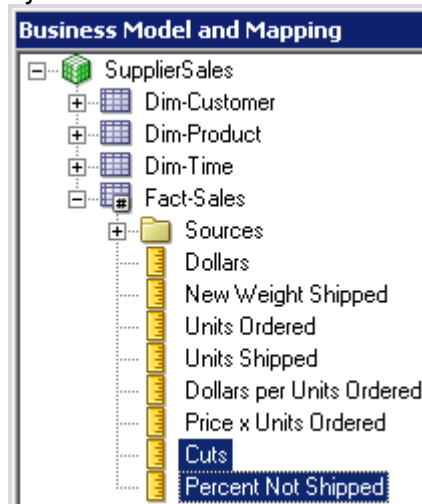
- l. In the Generate Calculations section, select **Percent Change** so that it is highlighted.



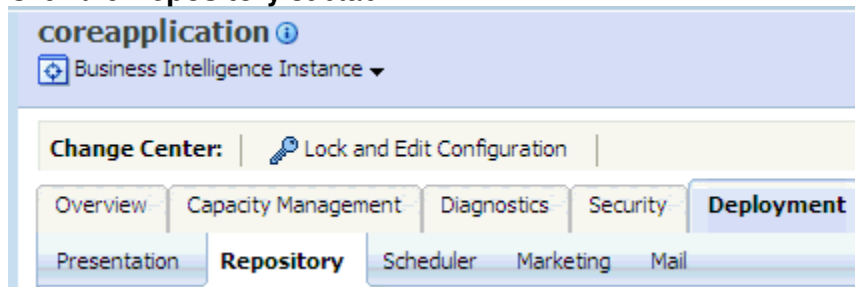
- m. Change the Calculation Name to **Percent Not Shipped**.
- n. Notice the results that are returned when Units Shipped is not available (NULL) or is zero using the following parameters. The following screenshot shows only partial results:



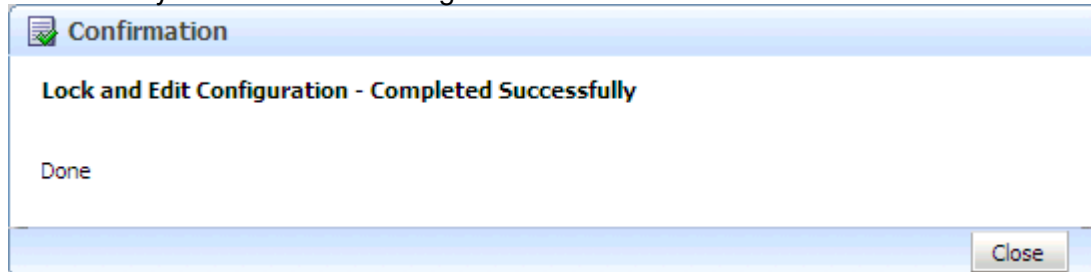
- o. Click **Next** to view the Finish window.
- p. Review the two calculations measures that will be created by the Wizard: **Cuts** and **Percent Not Shipped**.
- q. Click **Finish**.
- r. In the Business Model and Mapping layer, confirm that the two new columns created by the Calculation Wizard are visible in the Fact-Sales table.



- s. Drag the **Cuts** and **Percent Not Shipped** logical columns to the **Fact-Sales** presentation table in the **SupplierSales** presentation catalog.
- t. Save the repository and check global consistency. Fix any errors or warnings before proceeding.
- u. Close the repository.
2. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control Enterprise Manager, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.

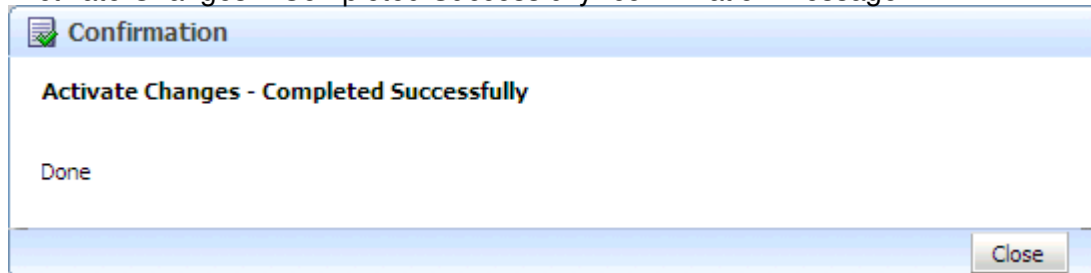


- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.

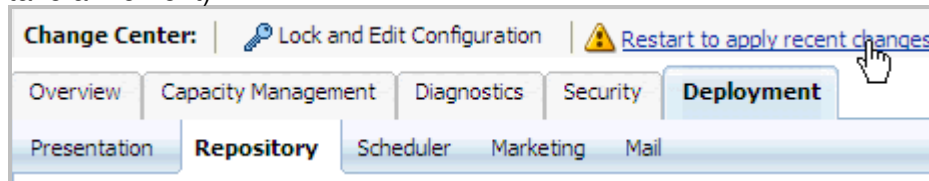


- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension (for example, ABC_BI0007).
- m. Click **Activate Changes**.

- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



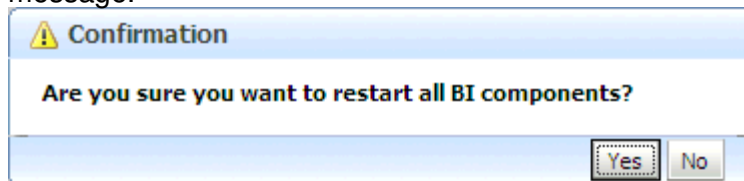
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



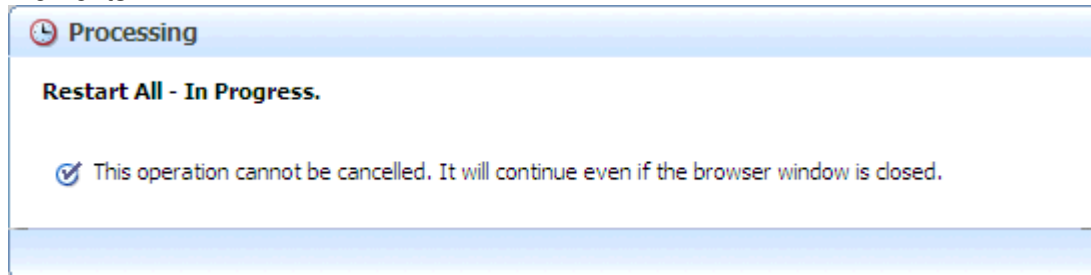
- p. On the Overview page, click **Restart**.



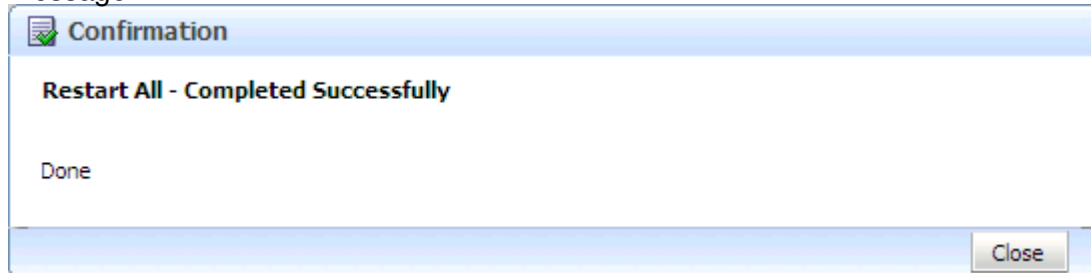
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



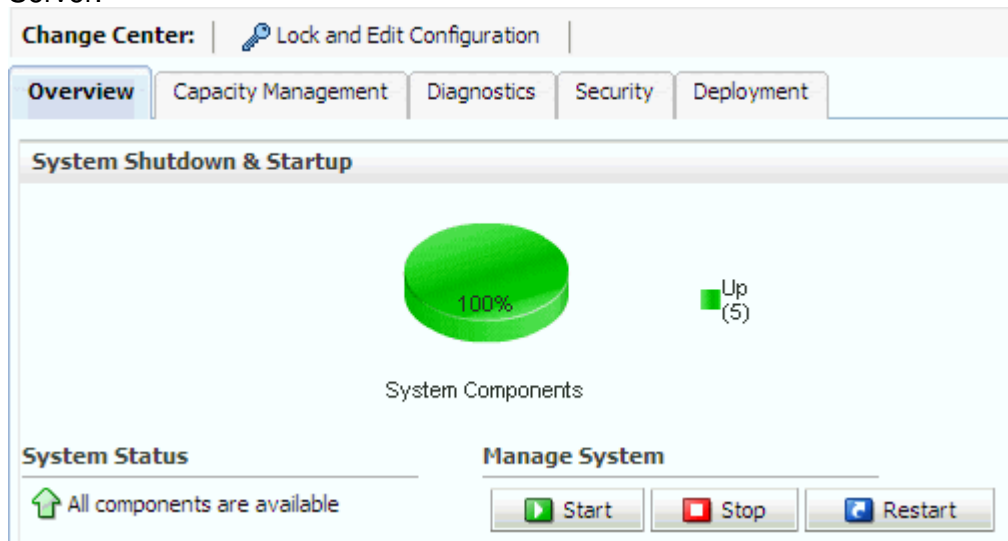
- r. Allow the **Restart All – In Progress** processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Enterprise Manager open.
3. Open Analysis Editor to execute queries and test the SupplierSales business model.
- a. Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.
- Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).
- b. Sign in as **weblogic** with password **welcome1**.
- c. In the Create section, click **Analysis** to open the Select Subject Area window.
- d. Click **SupplierSales** to open Analysis Editor.
4. Create an analysis to check your work for the Calculation Wizard.

- a. Create the following analysis:

Product	Fact-Sales			
Type	Units Ordered	Units Shipped	Cuts	Percent Not Shipped

- b. Modify the data format for Percent Not Shipped using the following screenshot as a guide:

Column Properties

Style Column Format **Data Format** Conditional Format Interaction Write Back

☒ Override Default Data Format

Treat Numbers As Percentage

Negative Format Minus: -123

Decimal Places 2

☐ Use 1000's Separator

- c. Click **Save as Default** and save these settings as the system-wide default for Percent Not Shipped.

- d. Click **Results**.

Type	Units Ordered	Units Shipped	Cuts	Percent Not Shipped
Baking	301,154	299,709	1,445	0.48%
Beef	136,299	135,341	958	0.71%
Beverage	244,208	241,179	3,029	1.26%
Bread	86,365	85,490	875	1.02%
Cereal	57,392	56,737	655	1.15%
Cheese	204,358	203,023	1,335	0.66%
Condiments	522,293	523,070	-777	-0.15%
Dessert	122,389	121,282	1,107	0.91%
Entre	47,783	47,158	625	1.33%
Frozen	13	13	0	0.00%
Grains	2,581	2,570	11	0.43%

5. Examine the query log.

- a. Your query log message should resemble the following screenshot:

```
----- Sending query to database named orcl (id:
<<2397>>), connection pool named SUPPLIER CP: [[
WITH
SAWITH0 AS (select sum(T90.UNITSHPD) as c1,
                sum(T90.UNITORDD) as c2,
                T502.ITEMTYPE as c3
from
    D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */ ,
    D1_PRODUCTS T99 /* Dim_D1_PRODUCTS */ ,
    D1_PRODUCT_SUBTYPE T498 /* Dim_D1_PRODUCT_SUBTYPE */ ,
    D1_PRODUCT_TYPE T502 /* Dim_D1_PRODUCT_TYPE */
where ( T90.PRODKEY = T99.PRODUCTKEY and T99.SUBTYPECODE =
T498.SUBTYPECODE and T498.TYPECODE = T502.TYPECODE )
group by T502.ITEMTYPE)
select distinct 0 as c1,
    D1.c3 as c2,
    nvl(D1.c2 , 0) - nvl(D1.c1 , 0) as c3,
    case when D1.c2 <> 0 and (D1.c1 in (0) or D1.c1 is
null) and not D1.c2 is null then 100.0 when (D1.c2 in (0) or
D1.c2 is null) and (D1.c1 in (0) or D1.c1 is null) then 0.0
when D1.c1 <> 0 and not D1.c1 is null and D1.c2 is null then
-100.0 else (D1.c2 - D1.c1) * 100.0 / nullif( D1.c1, 0) end
as c4,
    D1.c2 as c5,
    D1.c1 as c6
from
    SAWITH0 D1
order by c2
```

- b. Notice that UNITSHPD and UNITORDD columns are summed first...

```
SAWITH0 AS (select sum(T90.UNITSHPD) as c1,
                sum(T90.UNITORDD) as c2,
```

...and then calculated:

```
select distinct 0 as c1,
    D1.c3 as c2,
    nvl(D1.c2 , 0) - nvl(D1.c1 , 0) as c3,
    case when D1.c2 <> 0 and (D1.c1 in (0) or D1.c1 is
null) and not D1.c2 is null then 100.0 when (D1.c2 in (0) or
D1.c2 is null) and (D1.c1 in (0) or D1.c1 is null) then 0.0
when D1.c1 <> 0 and not D1.c1 is null and D1.c2 is null then
-100.0 else (D1.c2 - D1.c1) * 100.0 / nullif( D1.c1, 0) end
as c4,
    D1.c2 as c5,
    D1.c1 as c6
```

This is because the Calculation Wizard used logical columns to build the calculations.

- c. Sign out of Analysis Editor.

Practice 8-3: Creating a Rank Measure

Goal

To create a rank measure

Scenario

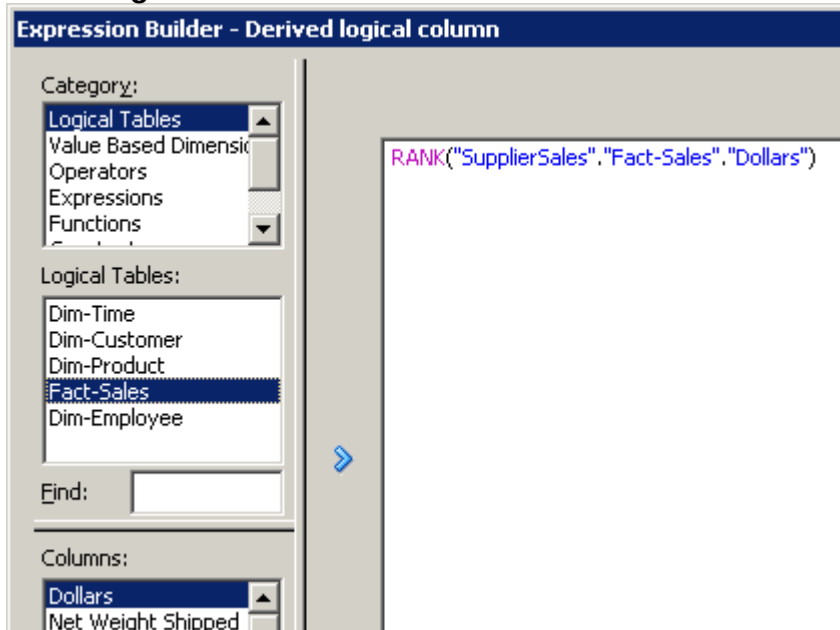
You use the RANK function to calculate rank for the Dollars logical column.

Time

20 minutes

Tasks

1. Create a new rank measure referencing existing logical columns.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository file in offline mode with repository password **welcome1**.
 - b. In the Business Model and Mapping layer, right-click the **Fact-Sales** table and select **New Object > Logical Column**.
 - c. On the General tab, name the column **Rank Dollars**.
 - d. On the Column Source tab select **Derived from existing columns using an expression**.
 - e. Open the Expression Builder.
 - f. Select **Functions > Display Functions > Rank**.
 - g. Double-click **Rank** or click **Insert**.
 - h. Click **<<numExpr>>** in the right pane.
 - i. Select **Logical Tables > Fact-Sales > Dollars** and add it to the formula.



- i.
 - j. Click **OK** to close the Expression Builder.

- k. Check your work:

Logical Column - Rank Dollars

General | Column Source | Aggregation | Levels

Data

Type: Length: ☒ Nullable

Derives from:

Column Source Type

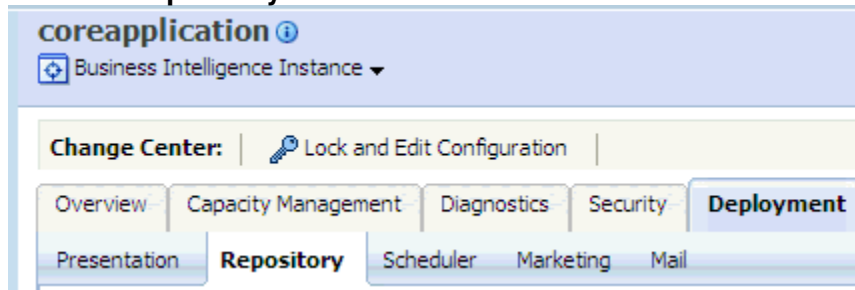
☐ Derived from physical mappings
☐ Show all logical sources

Logical Table Source | Mapped as

☒ Derived from existing columns using an expression

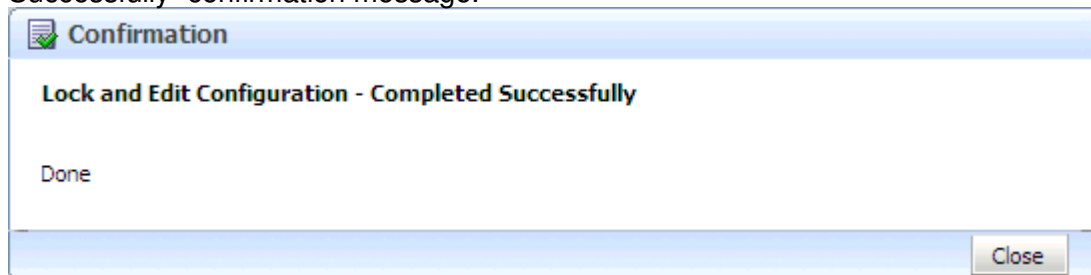
- l. Click **OK** to close the Logical Column dialog box. Rank Dollars is added to the business model.
 - m. Add the **Rank Dollars** measure to the **Fact-Sales** presentation table in the Presentation layer.
 - n. Save the repository and check consistency. Fix any errors or warnings before proceeding.
 - o. Close the repository.
 - p. Leave the Admin Tool open.
2. Use Fusion Middleware Control Enterprise Manager to upload the repository.
- a. Return to Fusion Middleware Control Enterprise Manager, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.

- e. Click the **Repository** subtab.

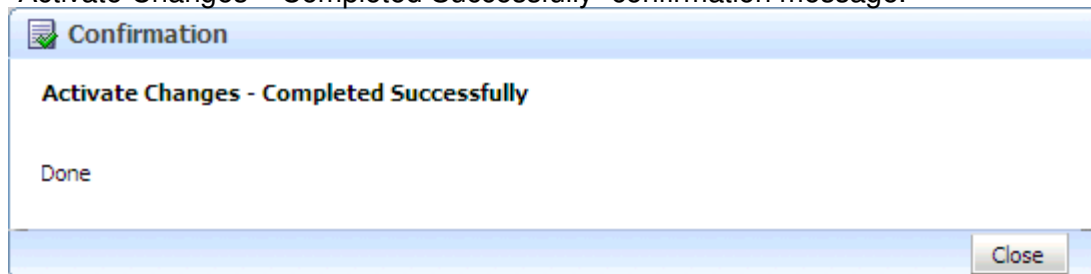


- f. Click **Lock and Edit Configuration**.

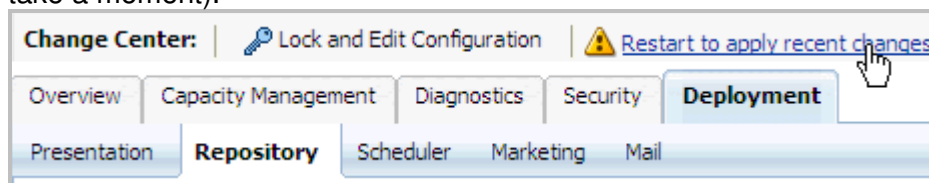
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the Choose file dialog box.
- i. By default, the Choose file dialog box should open to the default repository directory. If not, browse to
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension (for example, ABC_BI0007).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



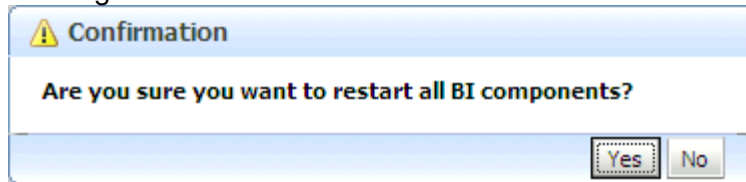
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



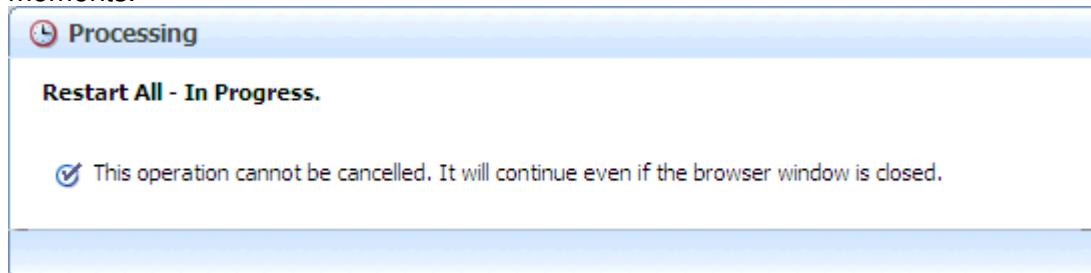
- p. On the Overview page, click **Restart**.



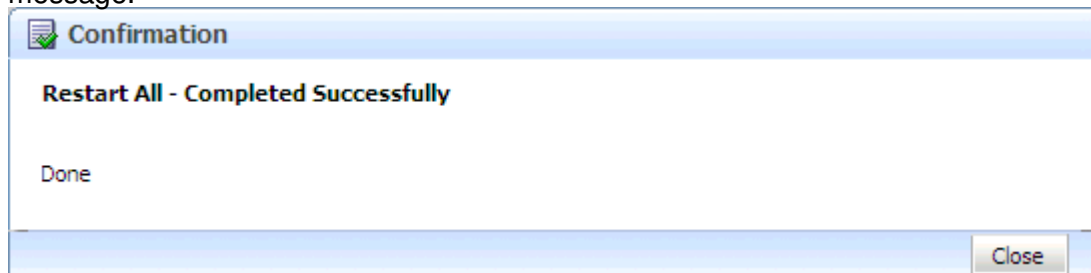
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



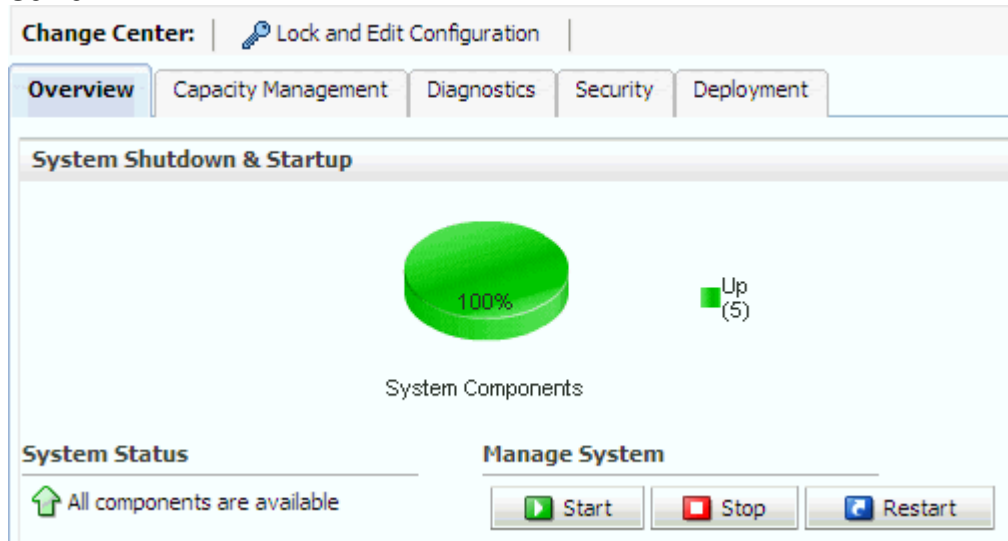
- r. Allow the **Restart All – In Progress** processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



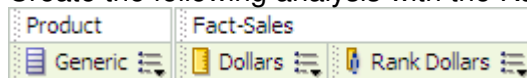
- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
3. Open Analysis Editor to execute an analysis to test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.

Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Click **SupplierSales** to open Analysis Editor.
4. Create an analysis to check your work.
- Create the following analysis with the Rank Dollars column sorted in ascending order:



- b. Click **Results**:

Generic	Dollars	Rank Dollars
American Cheese Slices	\$6,545,550.65	1
Frozen Chicken	\$4,913,525.17	2
Frozen Peas	\$3,183,208.89	3
Canned Tuna	\$1,748,759.40	4
Frozen Lasagna	\$1,625,759.41	5
White Shortening	\$1,598,467.80	6
Fizzy Cola	\$1,499,240.60	7
Frozen Turkey	\$1,479,515.82	8
Hamburger Patties	\$1,424,403.25	9
Frank's Diet Italian	\$1,266,839.57	10

The results show total dollars for each product and how each product ranks in total sales.

- Sign out of Analysis Editor.

Practices for Lesson 9: Working with Logical Dimensions

Chapter 9

Overview of Practices for Lesson 9

Lesson Overview

In these practices, you will build logical dimensions with level-based hierarchies and parent-child hierarchies.

Practice 9-1: Creating Logical Dimension Hierarchies

Goal

To add logical dimension hierarchies to the business model

Scenario

A logical dimension represents a hierarchical organization of logical columns belonging to a single logical dimension table. Logical dimensions can exist in the Business Model and Mapping layer and in the Presentation Layer. Adding logical dimensions to the Presentation layer exposes them to users, which enables users to create hierarchy-based queries. You implement three logical dimensions for ABC: Product, Customer, and Time. Creating logical dimensions with hierarchies allows you to build level-based measures, define aggregation rules that vary by dimension, provide drilldown on charts and tables in analyses and dashboards, and define the content of aggregate sources.

Outcome

In the Business Model and Mapping layer, there are Product, Customer, and Time logical dimensions.

Time

35 minutes

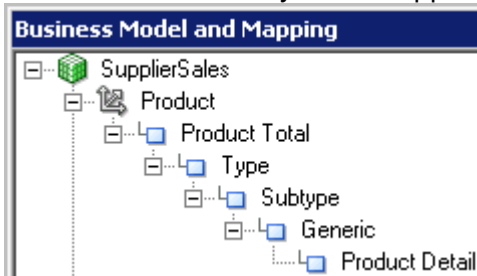
Tasks

1. Create a logical dimension for products.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with the repository password **welcome1**.
 - b. In the Business Model and Mapping layer, right-click **SupplierSales** and select **New Object > Logical Dimension > Dimension with Level-Based Hierarchy**. The Logical Dimension dialog box opens.
 - c. Name the logical dimension **Product**.
 - d. Click **OK**. The new logical dimension appears in the Business Model and Mapping layer. Notice the three-arrow icon.



2. Add the parent level of the hierarchy.
 - a. Right-click the **Product** logical dimension and select **New Object > Logical Level**.
 - b. In the Name field, enter **Product Total**.
 - c. Because this level represents the grand total for products, select the **Grand total level** check box. Notice that when you do this, the **Supports rollup to higher level of aggregation** field is grayed out and protected.
 - d. Click **OK**. The new level appears as a child of the Product logical dimension.
3. Add the child levels of the hierarchy.

- a. Right-click the **Product Total** level and select **New Object > Child Level**.
- b. In the Name field, enter **Type**.
- c. Click **OK**. The Type level appears as a child of the Product Total level.
- d. Repeat the above steps to add further child levels: **Subtype**, **Generic**, and **Product Detail**. Your hierarchy should appear as follows:



4. In this step, you associate columns from the logical dimension table with levels in the logical dimension hierarchy, starting from top to bottom. Here are some guidelines for associating columns with levels:

Not all columns in the dimension table must be associated explicitly with a level; the ones that are not will be associated with the lowest level implicitly.

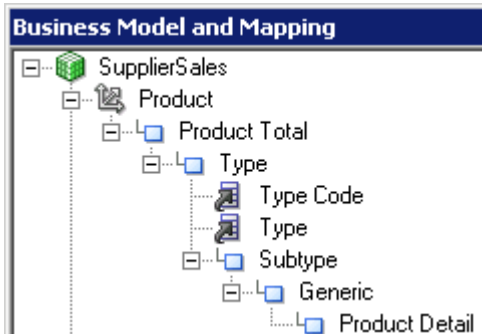
No columns can be associated with more than one level (although it may be part of the level key of a lower level).

If a column pertains to more than one level, associate it with the highest level it belongs to.

No level except the Grand Total level can exist without at least one column being associated with it.

The Detail level (lowest level) must have the column that is the logical key of the dimension table associated with it and it must be the key for that level.

- a. In the Business Model and Mapping layer, expand the **Dim-Product** table, select the **Type** column, and drag it up to the **Type** level.
- b. Drag the logical column **Type Code** onto the **Type** level in the Product logical dimension.

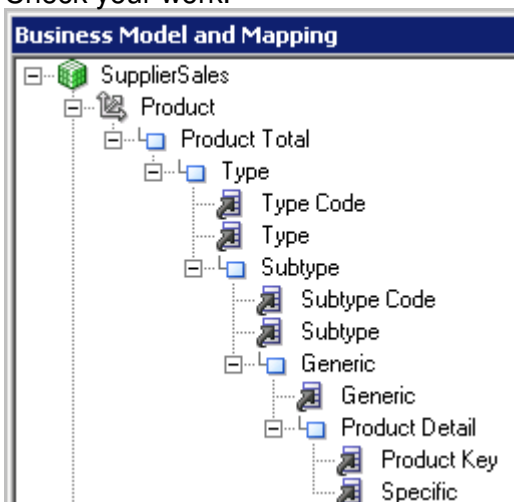


- c. Continue dragging logical columns from the Dim-Product logical table to the **Product** logical dimension levels:

Logical Column	Level
Subtype	Subtype
Subtype Code	Subtype
Generic	Generic

Specific	Product Detail
Product Key	Product Detail

- d. Check your work.



Any column not associated explicitly with a level is associated implicitly with the detail level. In this example, Package Weight and Supplier (among others) are by default associated with the Product Detail level. The logical dimension table key column, Product Key in this example, must be associated explicitly with the lowest level, Product Detail.

- e. Double-click the **Type** column in the Dim-Product logical table. The Logical Column dialog box opens.
 - f. Click the **Levels** tab. This tab identifies the logical dimensions and logical levels associated with this logical column. These values were set when you dragged the column into the Product logical dimension hierarchy.
 - g. Click the **drop-down list** for the **Type** logical level. Notice that this is another method for associating logical columns with dimensions and logical levels. You use this method later in the next practice. Leave the logical level set to **Type**.
 - h. Click **Cancel** to close the Logical Column dialog box.
5. In this step, you specify the level keys for the Type level in the hierarchy. Each logical level (except the top level defined as the Grand Total level) must have one or more attributes that compose a key level. The key level defines the unique elements in each logical level. The logical key for a logical table has to be associated with the lowest level of a logical dimension hierarchy and has to be the level key for that level.
- a. In the Product logical dimension, double-click the **Type** level. The Logical Level properties dialog box opens.
 - b. Click the **Keys** tab.
 - c. Enter **Type** in the Key Name field.
 - d. In the Columns field, select **Dim-Product.Type**.
 - e. Leave the Description field blank.
 - f. Select **Use for Display**.
 - g. Select the next row.
 - h. Enter **Type Code** in the Key Name field.
 - i. In the Columns field, select **Dim-Product.Type Code**.

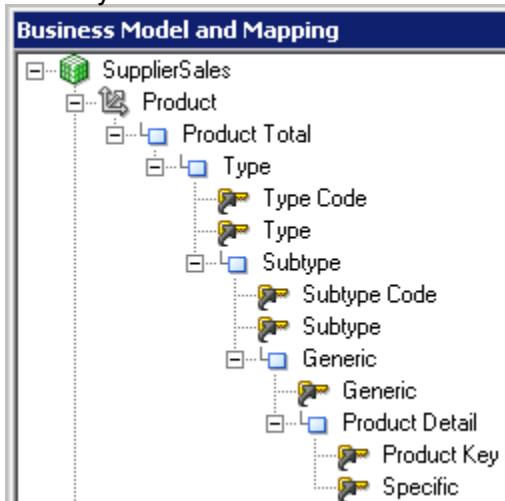
- j. Leave the Description field blank.
- k. Leave the **Use for Display** deselected. “Use for Display” is explained in the next set of steps.

Key Name	Columns	Description	Use for Display
Type	Dim-Product.Type		<input checked="" type="checkbox"/>
Type Code	Dim-Product.Type Code		<input type="checkbox"/>

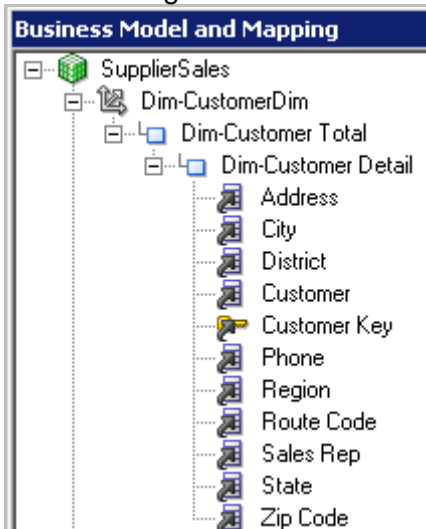
- l. Click **OK**. Notice that both level columns now display a key icon.
- 6. Set the level keys for the Subtype level using a different method.
 - a. In the Product logical dimension, right-click the **Subtype** column (not the level) and select **New Logical Level Key**.
 - b. Ensure that the **Subtype** check box is selected.
 - c. Ensure that the **Use for Display** check box is selected.
 - d. Click **OK**.
 - e. Right-click the **Subtype Code** column and select **New Logical Level Key**.
 - f. Ensure that the **Subtype Code** check box is selected.
 - g. Deselect the **Use for Display** check box.
 - h. Click **OK**. Explanation: Subtype is selected for display and Subtype Code is not. Later, when a user drills down in an analysis or a dashboard, the default drill is to the level key that has “Use for Display” selected in the next lowest level. Based on this example, when a user drills down from the Type level (the next highest level), the default is to drill down to the Subtype column, not the Subtype Code column.
- 7. Continue setting the following level keys for the remaining levels:

Dimensional Level	Key	Use for Display
Generic	Generic	Yes
Product Detail	Specific	Yes
Product Detail	Product Key	No

8. Check your final results.



9. In this step, you use another method to create the Customer logical dimension and levels.
 - a. Right-click the **Dim-Customer** logical table and select **Create Logical Dimension > Dimension with Level-Based Hierarchy**. A new logical dimension named Dim-CustomerDim is created with two levels, Dim-Customer Total and Dim-Customer Detail. The Dim-Customer Detail level is populated with all columns from the Dim-Customer logical dimension table.

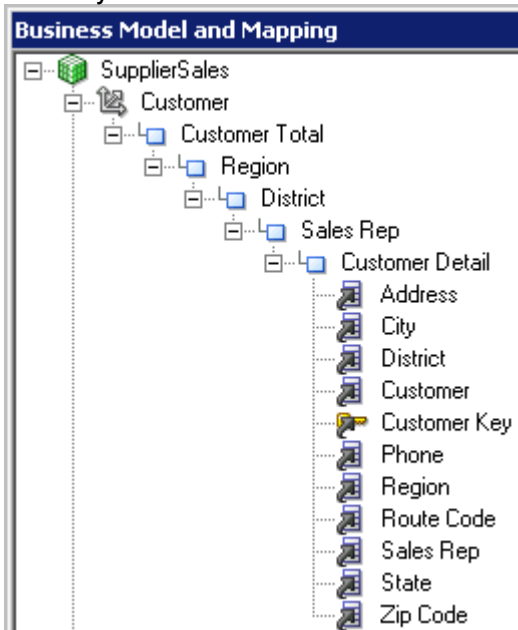


- b. Rename the logical dimension and the levels:

Old Name	New Name
Dim-CustomerDim	Customer
Dim-Customer Total	Customer Total
Dim-Customer Detail	Customer Detail

- c. Right-click the **Customer Detail** level and select **New Object > Parent Level**.
 - d. Name the parent level **Sales Rep** and click **OK**.
 - e. Create a **District** level as a parent of **Sales Rep**.
 - f. Create a **Region** level as a parent of **District**.
 - g. Right-click the **Customer** logical dimension and select **Expand All**.

h. Check your work.



i. Add columns to the hierarchy by dragging columns from the Customer Detail level (not the Dim-Customer logical table) to the other Customer hierarchy levels. This is a useful method when business models are large. It eliminates the need to scroll to logical tables to locate columns.

Column	Level
Region	Region
District	District
Sales Rep	Sales Rep

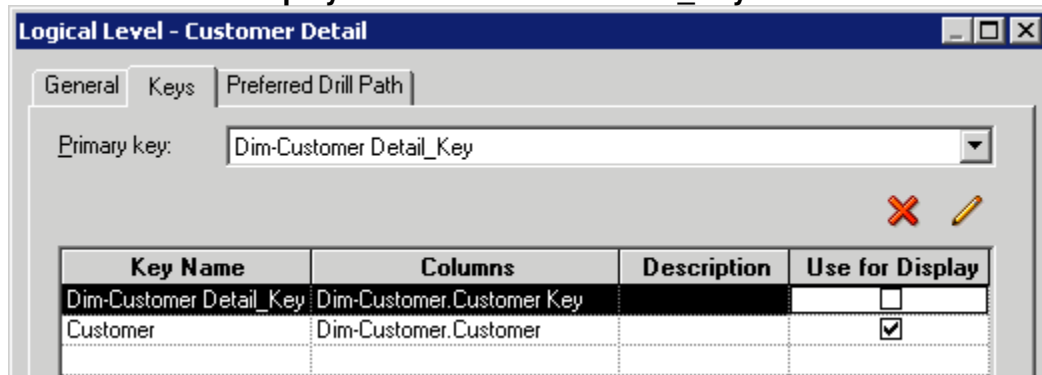
j. Create the following keys for each level. Notice that Customer Key is already identified as a key for the Customer Detail level.

Level	Key	Use for Display
Region	Region	Yes
District	District	Yes
Sales Rep	Sales Rep	Yes
Customer Detail	Customer	Yes

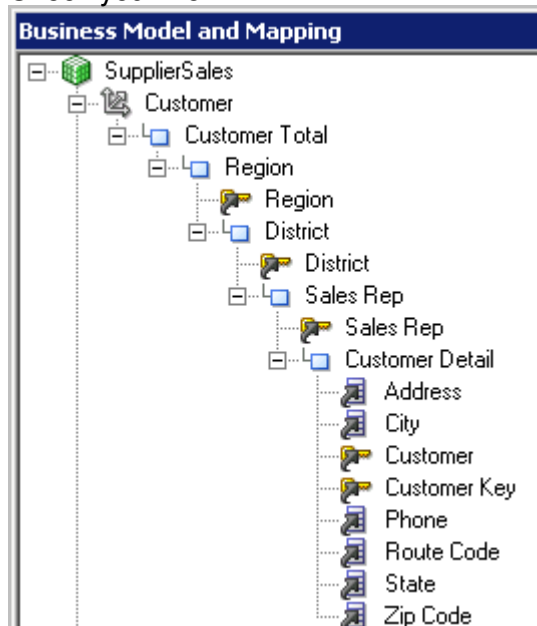
k. Double-click the **Customer Detail** level.

l. Click the **Keys** tab.

- m. Deselect **Use for Display** for **Dim-Customer Detail_Key**.



- n. Click **OK**.
o. Check your work:



10. Create a preferred drill path for the Customer Detail level. You can use a preferred drill path to identify the drill path to use when users drill down through data in analyses. You should use this only to specify a drill path that is outside the normal drill path defined by the dimensional level hierarchy. It is most commonly used to drill from one dimension to another.
- Double-click the **Customer Detail** level to open the Logical Level dialog box.
 - Click the **Preferred Drill Path** tab.
 - Click **Add**.
 - In the Browse dialog box, double-click **Type** in the right pane to add it as the preferred drill path in the Logical Level dialog box.
 - Click **OK**.
 - Save the repository without checking consistency.
11. In this step, you can use either of the two methods described in the previous steps to create the Time logical dimension level-based hierarchy.
- Create a logical dimension named **Time** based on the Time logical dimension table.
 - Create the following levels:

Level	Grand Total Level	Supports rollup to higher level of aggregation
Time Total	Yes	Protected
Year	No	Yes
Quarter	No	Yes
Month	No	Yes
Time Detail	No	Yes

12. There is no logical column that you can associate with the Quarter level, so you must create one and map it to the MONTH_IN_YEAR column with a formula using a CASE statement.
 - a. Expand **Dim-Time > Sources** and double-click the **Dim_D1_CALENDAR2** logical table source to open the Logical Table Source dialog box.
 - b. Click the **Column Mapping** tab.
 - c. Click the **Add new column** button.
 - d. On the General tab, name the column **Quarter**.
 - e. Click **OK**.
 - f. On the Column Mapping tab, ensure that **Show unmapped columns** is selected.
 - g. Ensure that the **Quarter** column is selected.
 - h. Click the **Edit Expression** button to open the Expression Builder for the **Quarter** column.
 - i. Use fully qualified column names and build the following formula. Or, you can copy and paste this formula from the **Quarter.txt** file in D:\PracticeFiles.

```

CASE
WHEN "ORCL".""."SUPPLIER2"."Dim_D1_CALENDAR2"."MONTH_IN_YEAR" < 4 THEN
1
WHEN "ORCL".""."SUPPLIER2"."Dim_D1_CALENDAR2"."MONTH_IN_YEAR" < 7 THEN
2
WHEN "ORCL".""."SUPPLIER2"."Dim_D1_CALENDAR2"."MONTH_IN_YEAR" < 10
THEN 3
ELSE 4
END

```

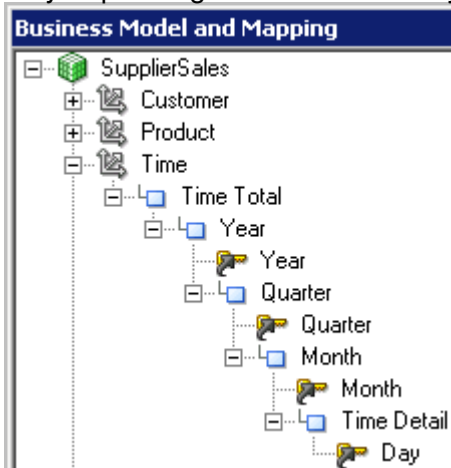
- j. Click **OK** to close the Expression Builder. The expression is displayed in the Expression field in the Logical Table Source dialog box.
 - k. Click **OK** to close the Logical Table Source dialog box. The Quarter column is displayed in the business model.
13. Add logical columns to the Time logical dimension levels:

Logical Column	Level
Year	Year
Quarter	Quarter
Month	Month
Day	Time Detail

14. Create the keys for each child level in the Time logical dimension. Notice that additional configuration of the key for the Quarter level is completed in a subsequent step.

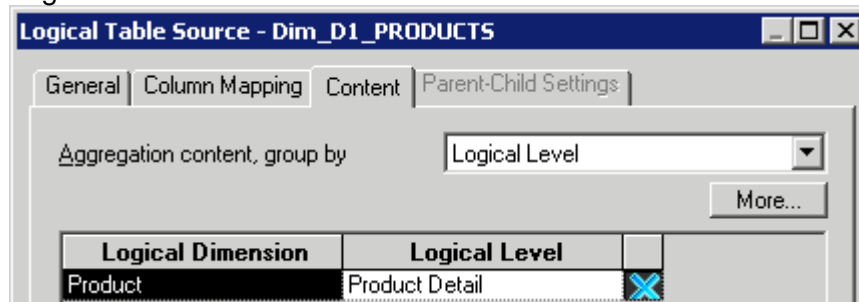
Level	Key	Use for Display
Year	Year	Yes
Quarter	Quarter	Yes
Month	Month	Yes
Time Detail	Day	Yes

15. Setting the key for Quarter involves an additional step. Quarters (1, 2, 3, 4) occur each year. However, these numbers do not uniquely identify a quarter, except when combined with Year. Therefore, you must make Year part of the Quarter level.
 - a. Double-click the **Quarter** level in the Time logical dimension.
 - b. Click the **Keys** tab.
 - c. Select the **Quarter** key.
 - d. Click the **Edit** button.
 - e. Click the **Add** Button.
 - f. In the Browse dialog box, expand the **Dim-Time** logical table.
 - g. Select **Year** and click **OK**.
 - h. Click **OK** to close the **Logical Level Key** dialog box.
 - i. Click **OK** to close the **Logical Level** dialog box.
 - j. Check your work. It should look similar to the following screenshot. Your results may vary depending on which method you used to build the hierarchy.



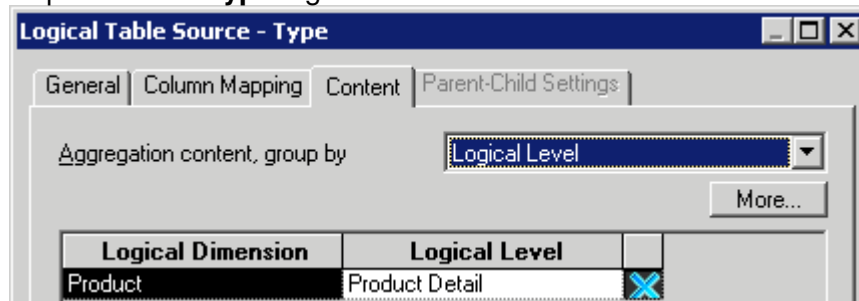
16. Set logical levels for the logical table sources.
 - a. Expand **Dim-Product > Sources** and double-click the **Dim_D1_PRODUCTS** logical table source to open the Logical Table Source dialog box.
 - b. Click the **Content** tab.
 - c. Recall that in the practices for the “Managing Logical Table Sources” lesson you set the Aggregation content to Column. Now that you have created logical dimension hierarchies, change **Aggregation content, group by** to **Logical Level** and set the

Logical Level to **Product Detail**.



d. Click **OK** to close the Logical Table Source dialog box.

e. Repeat for the **Type** logical table source.



f. Expand **Dim-Time > Sources** and double-click the **Dim_D1_CALENDAR2** logical table source to open the Logical Table Source dialog box.

g. Click the **Content** tab.

h. Depending on which method you used to create the Time logical dimension hierarchy, the logical level may or may not be set. Set it to **Time Detail** and click **OK**.

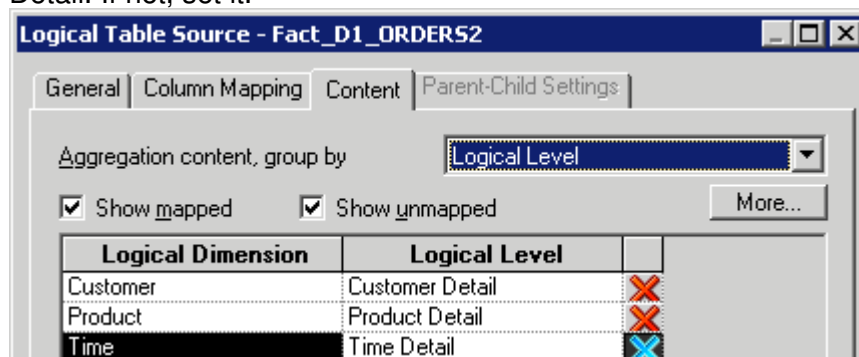
i. Expand **Fact-Sales > Sources** and double-click the **Fact_D1_ORDERS2** logical table source.

j. Click the **Content** tab.

k. Set the logical level to **Product Detail** for the **Product** logical dimension.

l. Set the logical level to **Time Detail** for the **Time** logical dimension.

m. The logical level for the Customer logical dimension should already be set to Customer Detail. If not, set it.

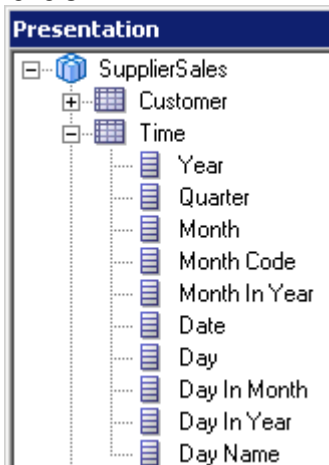


n. Click **OK** to close the Logical Table Source dialog box.

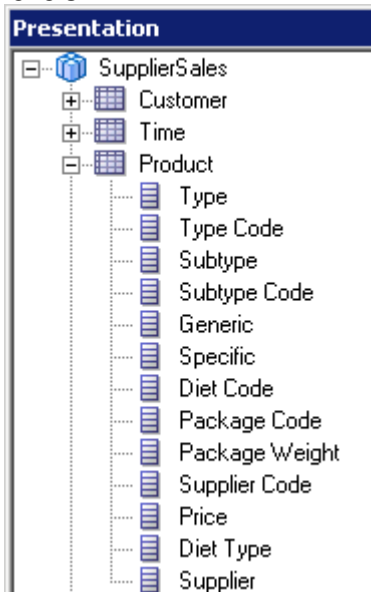
17. Modify the Presentation layer.

a. Drag the **Quarter** logical column from the **Dim-Time** logical table to the **Time** presentation table in the SupplierSales subject area to make it available for analyses.

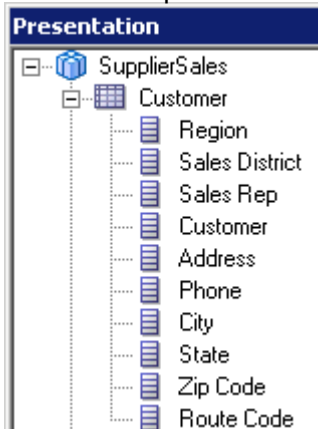
- b. Reorder the columns in the Time presentation table to match the logical dimension levels.



- c. Reorder the columns in the Product presentation table to match the logical dimension levels.

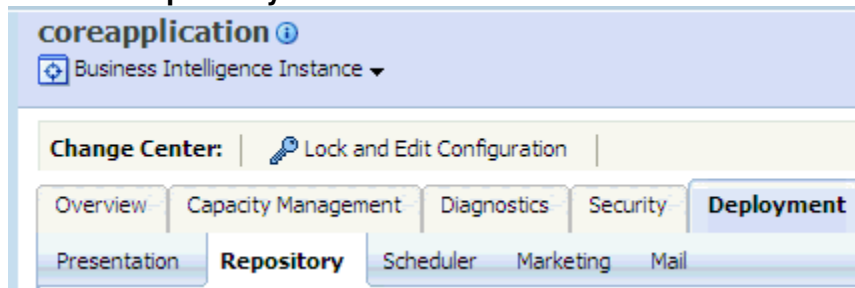


- d. Verify the column order in the Customer presentation table. You modified the columns in an earlier practice.

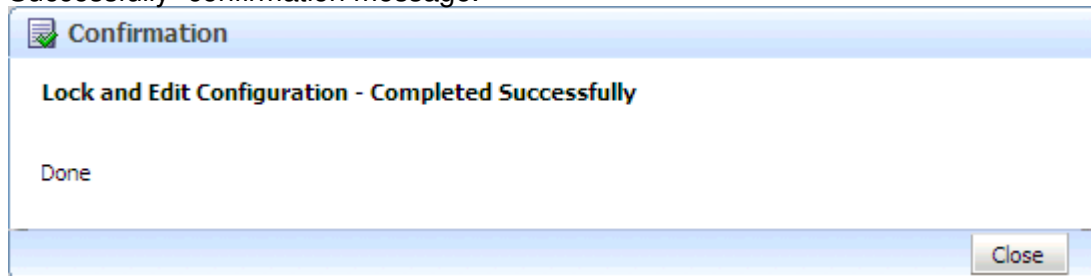


- e. Save the repository.

- f. Check consistency. Fix any consistency errors or warnings before proceeding.
 - g. Close the repository.
 - h. Leave the Administration Tool open for the next practice.
18. Use Fusion Middleware Control to upload the repository.
- a. Return to Fusion Middleware Control Enterprise Manager, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.

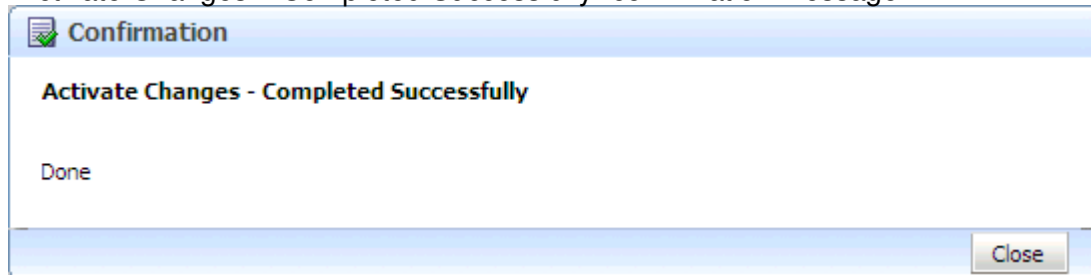


- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.

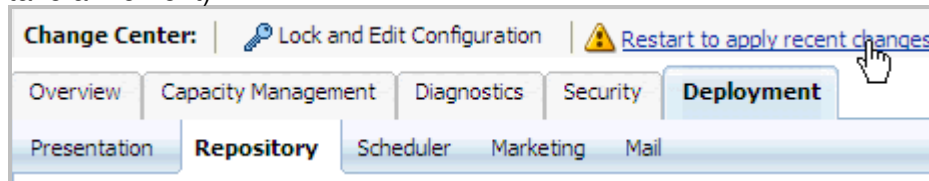


- h. In the Upload BI Server Repository section, click **Browse** to open the File Upload dialog box.
- i. By default, the File Upload dialog box should open to the default repository directory. If not, browse to
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension, for example, ABC_BI0007.
- m. Click **Activate Changes**.

- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



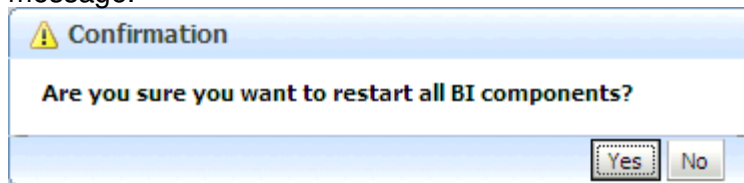
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



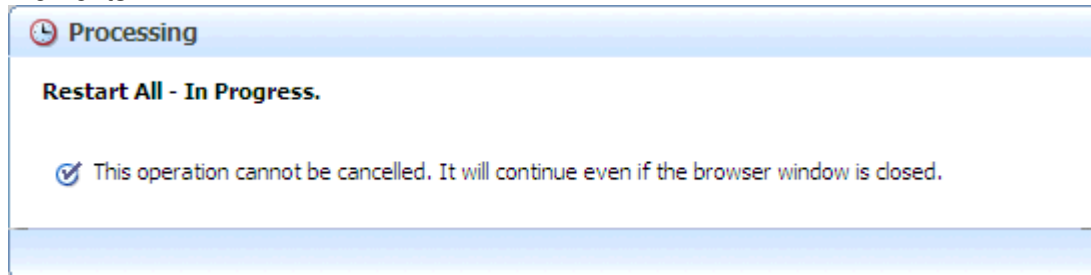
- p. On the Overview page, click **Restart**.



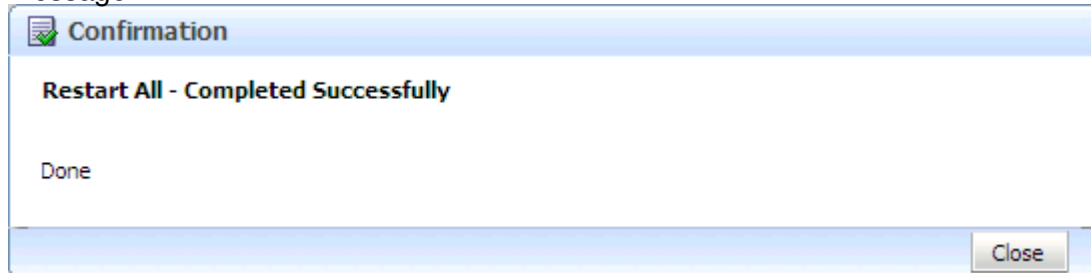
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



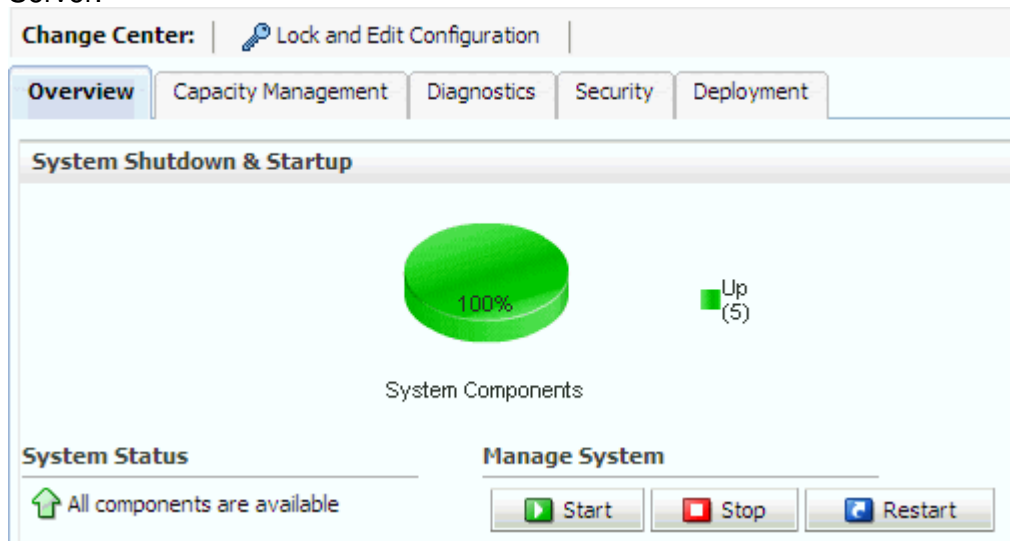
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
19. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.
- Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).
- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Click **SupplierSales** to open Analysis Editor.
20. Create an analysis to test the Time logical dimension:

- a. Create the following analysis:

Time	Fact-Sales
Year	Dollars

- b. Click **Results**.
c. Drill down on **2008** and ensure that you can see dollars data by quarter.

Year	Quarter	Dollars
2008	1	\$11,829,308.46
	2	\$12,207,160.59
	3	\$12,390,129.35
	4	\$12,601,988.23

- d. Drill down on any quarter and ensure that you can see dollars data by month.
e. Drill down on any month and ensure that you can see dollars data by day. Your results should look similar to the following screenshot:

Year	Quarter	Month	Day	Dollars
2008	1	February	20080202	\$186,902.59
			20080203	\$231,248.77
			20080204	\$268,628.95
			20080205	\$95,719.53
			20080206	\$297,776.28
			20080207	\$120,794.67
			20080209	\$128,237.69
			20080210	\$189,540.39
			20080211	\$290,734.13
			20080212	\$254,689.24

21. Create the following new analysis to test the Customer logical dimension:

Customer	Fact-Sales
Region	Dollars

- a. Click **Results**.
b. Verify that you can drill down from the Region level to Sales District > Sales Rep > Customer > Type. Recall that you set Type as the preferred drill path for the Customer Detail level. Your results should look similar to the following screenshot:

Region	Sales District	Sales Rep	Customer	Type	Dollars
Central	Gulf	MARY SILVER	Alley Dog	Baking	\$33,636.33
				Beef	\$18,099.85
				Beverage	\$13,305.29
				Bread	\$16,194.60

22. Create the following analysis to test the Product logical dimension:

Product	Fact-Sales
Type	Dollars

- a. Click **Results**.
b. Verify that you can drill down from the Type level to Subtype > Generic > Specific. Your results should look similar to the following screenshot:

Type	Subtype	Generic	Specific	Dollars
Baking	Balsamic Vinegar	Balsamic Vinegar	Balsamic Vinegar	\$29,155

- c. Sign out of Analysis Editor.

Practice 9-2: Creating Level-Based Measures

Goal

To create level-based measures

Scenario

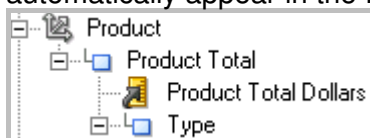
Now that you have created level-based logical dimension hierarchies, you create level-based measures that calculate total dollars at various levels in the hierarchies.

Time

30 minutes

Tasks

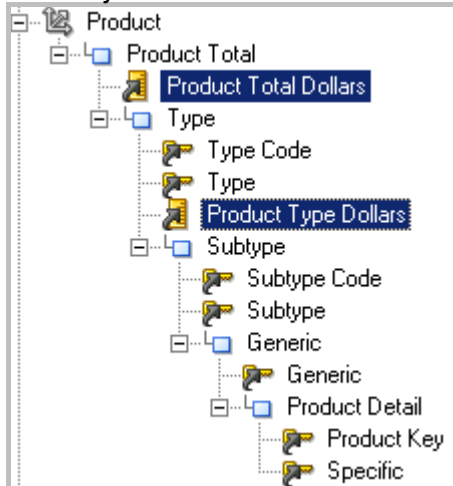
1. Create level-based measures for the Product logical dimension.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the Business Model and Mapping layer, right-click the **Fact-Sales** table and select **New Object > Logical Column**.
 - c. On the General tab, in the Name field, enter **Product Total Dollars**.
 - d. Click the **Column Source** tab.
 - e. Select **Derived from existing columns using an expression**.
 - f. Open the Expression Builder.
 - g. In the Expression Builder, add **Logical Tables > Fact-Sales > Dollars** to the expression. Recall that the Dollars column already has a default aggregation rule of **Sum**.
 - h. Click **OK**.
 - i. Click the **Levels** tab.
 - j. For the **Product** logical dimension, select **Product Total** from the Level drop-down list to specify that this measure should be calculated at the grand total level in the product hierarchy.
 - k. Click **OK**. Notice that setting the level for the logical column causes the measure to automatically appear in the Product logical dimension.



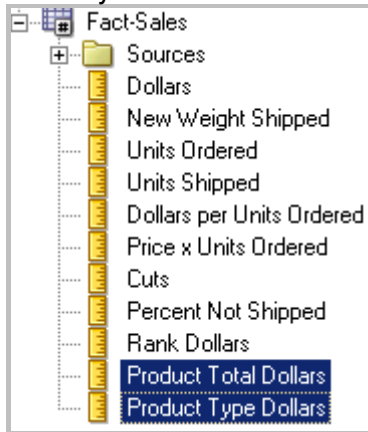
- l. Repeat the steps to create a second level-based measure:

Name	Dimension	Level
Product Type Dollars	Product	Type

- m. Check your final results for the Product logical dimension:



- n. Check your final results in the Fact-Sales logical table:



- o. Expose the new columns to users by dragging **Product Total Dollars** and **Product Type Dollars** from the **Fact-Sales** logical table to the **Fact-Sales** presentation table in the SupplierSales catalog in the Presentation layer.
- p. Save the repository.
- q. Check consistency. Fix any errors or warnings before proceeding.
- r. Leave the repository open. You check your work in Analysis Editor in Practice 9-5.

Practice 9-3: Creating Share Measures

Goal

To create a share measure using level-based measures


Scenario

Now that you have created level-based measures, you use them to create a share measure for products.

Time

10 minutes

Tasks

1. Create a new share measure derived from existing logical columns.
 - a. In the Business Model and Mapping layer, right-click **Fact-Sales** and select **New Object > Logical Column**.
 - b. On the General tab, name the logical column **Product Share**.
 - c. On the Column Source tab, select **Derived from existing columns using an expression**.
 - d. Open the Expression Builder.
 - e. Select **Functions > Mathematic Functions > Round**.
 - f. Click **Insert selected item**. The function appears in the edit box.
 - g. Click **SourceNumber** in the formula.
 - h. Enter **100*** followed by a space.
 - i. Insert **Logical Tables > Fact-Sales > Dollars**.
 - j. Using the toolbar, click the **Division**  button. Another set of angle brackets appears, **<<expr>>**.
 - k. Click **<<expr>>**.
 - l. Insert **Logical Tables > Fact-Sales > Product Total Dollars**. Recall that this is the total level-based measure for the Product logical dimension hierarchy.
 - m. Click between the last set of angle brackets, **<<Digits>>**, and enter **1**. This represents the number of digits of precision with which to round the integer.
 - n. Check your work:

```
Round(100* "SupplierSales"."Fact-Sales"."Dollars" /  
      "SupplierSales"."Fact-Sales"."Product Total Dollars" , 1)
```

This share measure will allow you to run an analysis to show how sales of a specific product compares to overall sales for all products.
 - o. Click **OK** to close the Expression Builder.
 - p. Click **OK** to close the Logical Column properties dialog box.
 - q. The **Product Share** logical column is added to the business model.
 - r. Add the **Product Share** measure to the Fact-Sales presentation table.
 - s. Save the repository.
 - t. Check consistency. If there are errors or warnings, correct them before you proceed.
 - u. Leave the repository open. You check your work in Analysis Editor in Practice 9-5.

Practice 9-4: Creating Dimension-Specific Aggregation Rules

Goal

To create a measure with dimension-specific aggregation rules

Scenario

ABC wants a measure called Average Daily Dollars, which sums dollar amounts over the Customer and Product dimensions and divides by the number of days in the Time dimension. This measure can be used to compare the average daily dollar amount from month-to-month when the number of order days in each month varies.

Outcome

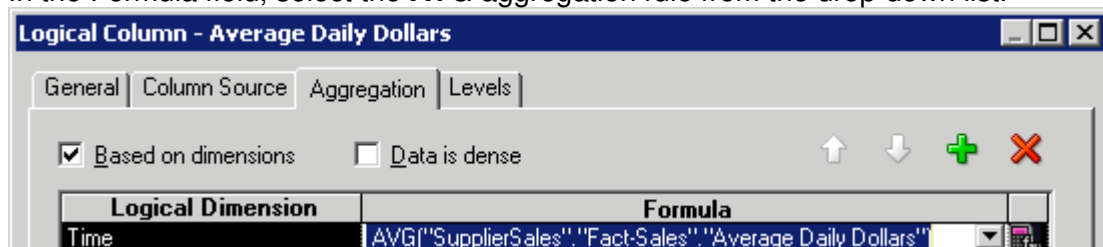
A new measure called Average Daily Dollars with dimension-specific aggregation rules

Time

15 minutes

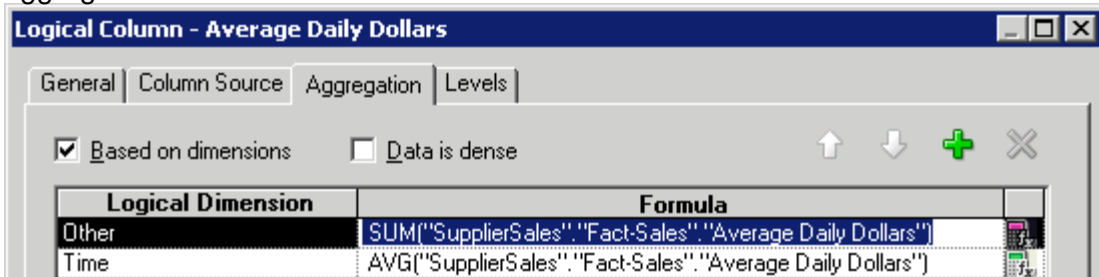
Tasks

1. Create a new measure with dimension-specific aggregation rules.
 - a. In the Business Model and Mapping layer, right-click **Fact-Sales** and select **New Object > Logical Column**.
 - b. On the General tab, name the logical column **Average Daily Dollars**.
 - c. Click the **Column Source** tab.
 - d. Ensure that **Show all logical sources** is selected.
 - e. Double-click the **Fact_D1_ORDERS2** logical table source.
 - f. Click the **Column Mapping** tab.
 - g. Ensure that **Show unmapped columns** is selected.
 - h. Use the Expression column to map **Average Daily Dollars** to the **Fact_D1_ORDERS2.DOLLARS** physical column.
 - i. Click **OK** to close the Logical Table Source dialog box.
 - j. Click the **Aggregation** tab.
 - k. Select **Based on dimensions**.
 - l. In the Browse dialog box, select the **Time** dimension and click **OK**.
 - m. In the Formula field, select the **AVG** aggregation rule from the drop-down list.



- n. Click the **Add** button.
- o. Select **Other** and click **OK**.
- p. Select the **SUM** aggregation rule for **Other**.

- q. With **Other** selected in the Logical Dimension column, use the **Up** button to change the aggregation rule order.



- r. Click **OK** to close the Logical Column dialog box.
- s. Drag the **Average Daily Dollars** measure to the **Fact-Sales** presentation table.
- t. Save the repository.
- u. Check Global Consistency. Fix errors and warnings before you proceed.
- v. Leave the repository open. You check your work in Analysis Editor in Practice 9-5.

Practice 9-5: Creating Presentation Hierarchies

Goal

To create presentation hierarchies that expose logical dimension hierarchies in Oracle BI analyses.

Scenario

Presentation hierarchies and presentation levels provide an explicit way to expose the multidimensional model in Oracle BI analyses and dashboards. When presentation hierarchies and levels are defined in the Presentation layer, roll-up information is displayed in the Analysis Editor navigation pane, providing users with important contextual information. Most importantly, users can create hierarchy-based queries using these objects.

Outcome

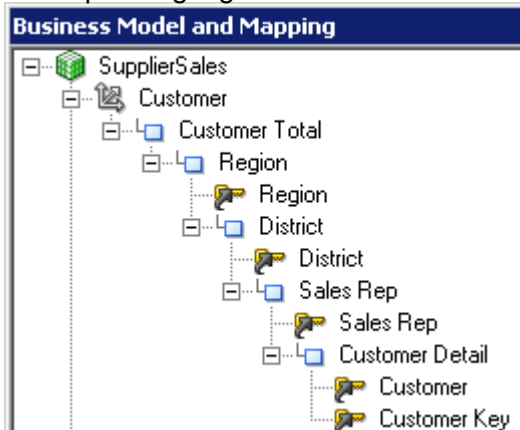
Presentation hierarchies are added to the Customer, Time, and Product presentation tables.

Time

15 minutes

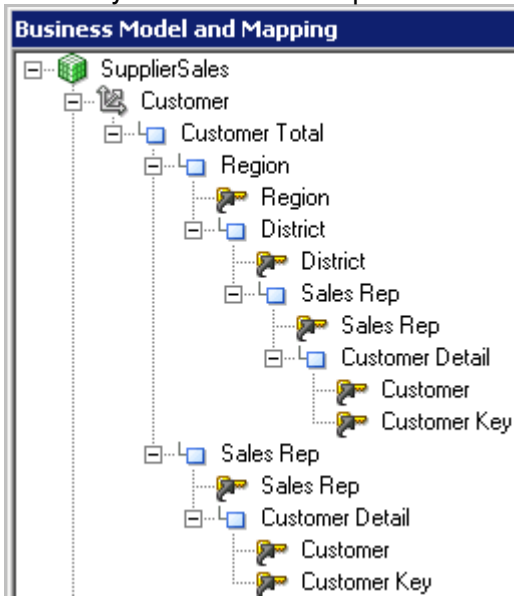
Tasks

1. Modify the customer logical dimension to include multiple hierarchies.
 - a. Expand the **Customer** logical dimension.
 - b. Delete all columns from the Customer Detail level except for **Customer** and **Customer Key**. Notice that deleting columns from the hierarchy does not delete the corresponding logical columns.



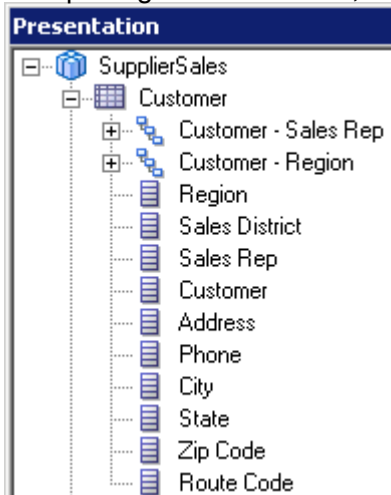
- c. Right-click the **Customer Total** level and select **New Object > Shared Level as Child**.

- d. In the Browse dialog box, select **Sales Rep** and click **OK**. This creates a second hierarchy named Sales Rep in the Customer logical dimension:

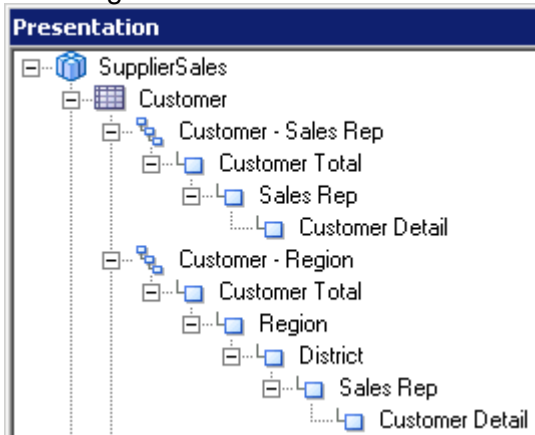


2. Create presentation hierarchies.

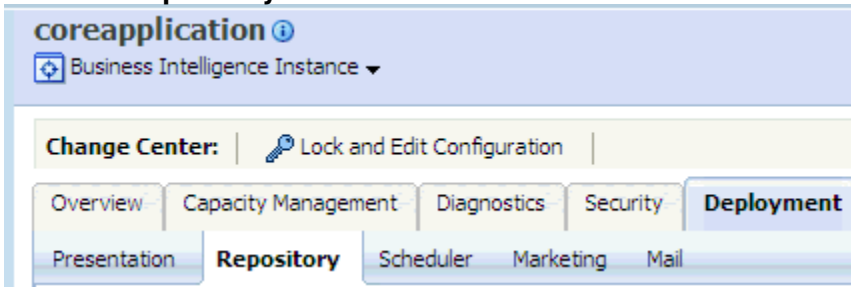
- a. Drag the **Customer** logical dimension from the Business Model and Mapping layer to the **Customer** presentation table. Notice that two separate presentation hierarchies are created, one for each logical dimension hierarchy. For logical dimensions that contain multiple logical hierarchies, multiple separate presentation hierarchies are created:



- b. Expand the presentation hierarchies and notice that each contains the drill path defined in the logical dimension.

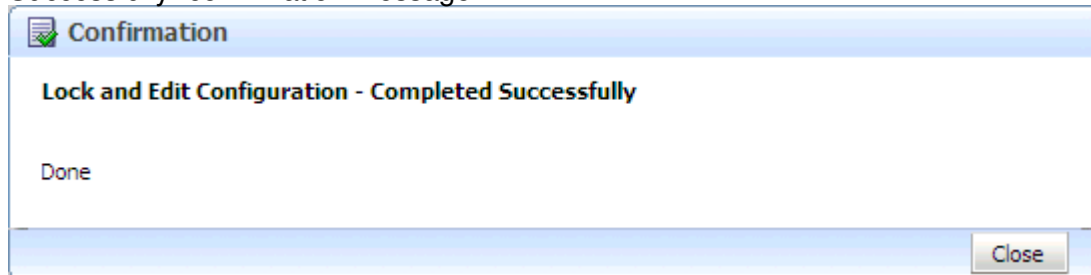


- c. Drag the **Product** logical dimension to the **Product** presentation table.
 - d. Drag the **Time** logical dimension to the **Time** presentation table. Notice that only one presentation hierarchy is created for the Product and Time logical dimensions. For logical dimensions that contain one logical hierarchy, one presentation hierarchy is created.
 - e. Save the repository.
 - f. Check consistency. Fix any errors or warnings before proceeding.
 - g. Close the repository.
 - h. Leave the Administration Tool open.
3. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
<http://localhost:7001/em>.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.

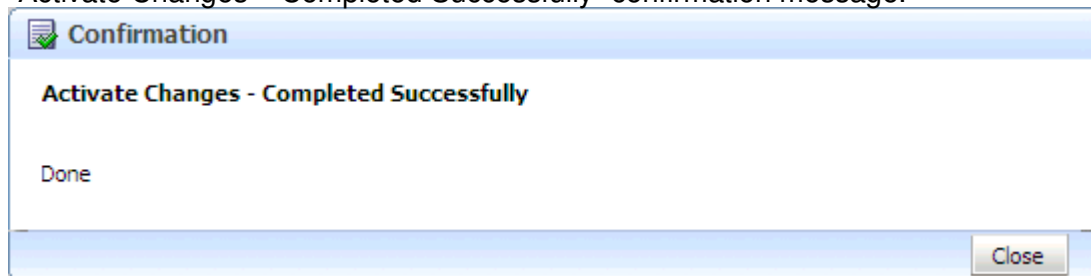


- f. Click **Lock and Edit Configuration**.

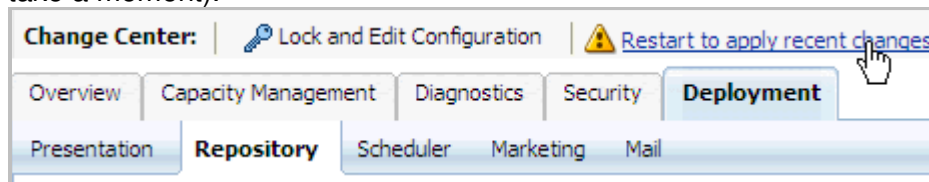
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the “Choose file” dialog box.
- i. By default, the “Choose file” dialog box should open to the default repository directory. If not, browse to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository**.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension, for example, ABC_BI0007.
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



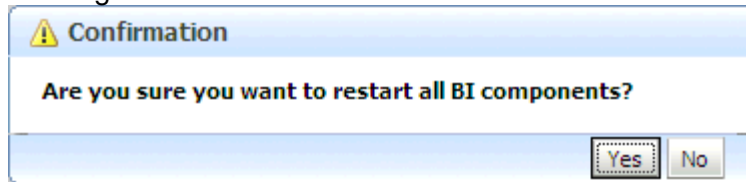
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



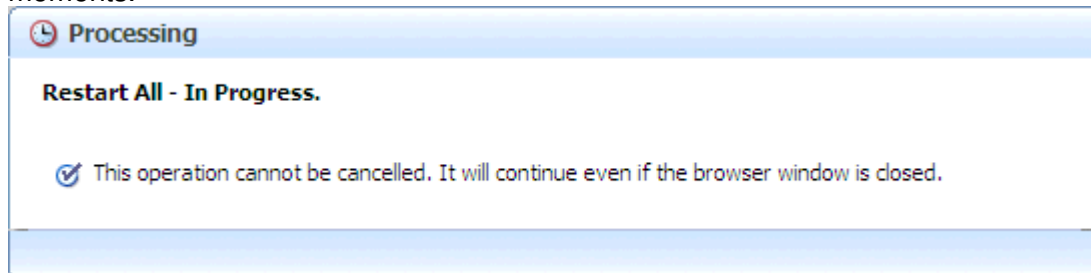
- p. On the Overview page, click **Restart**.



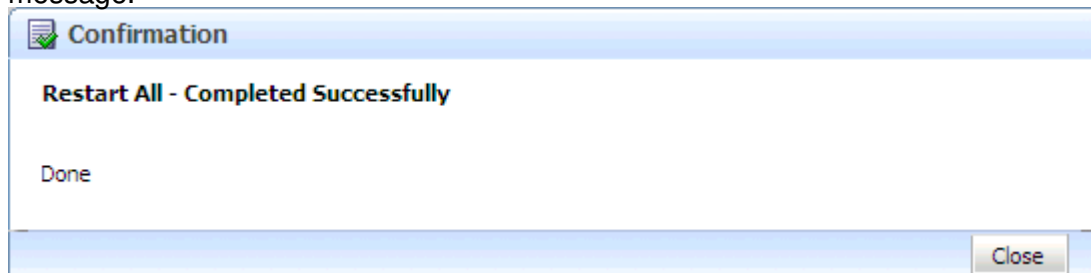
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



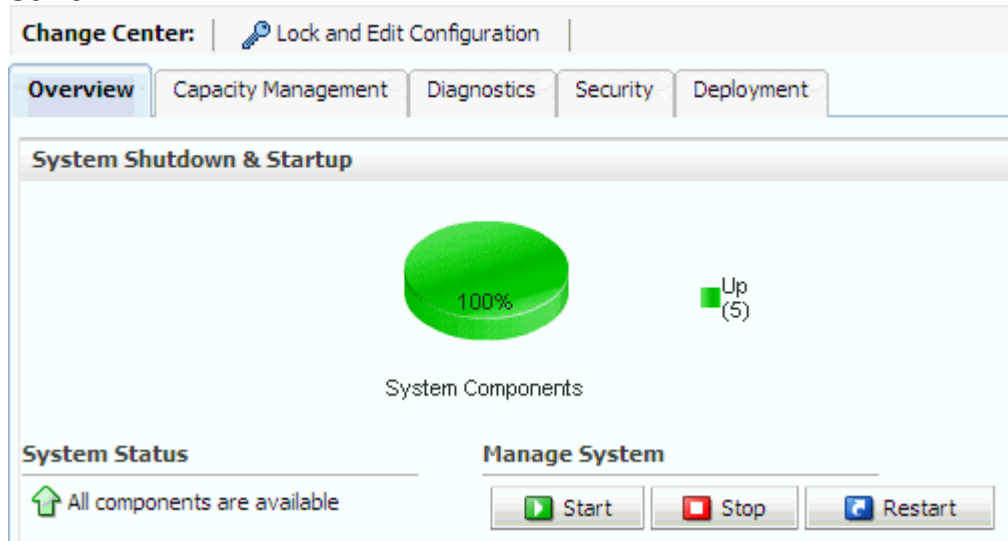
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.

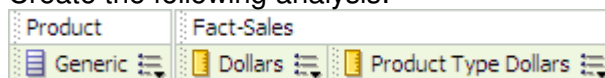


- u. Leave Fusion Middleware Control open.
4. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.

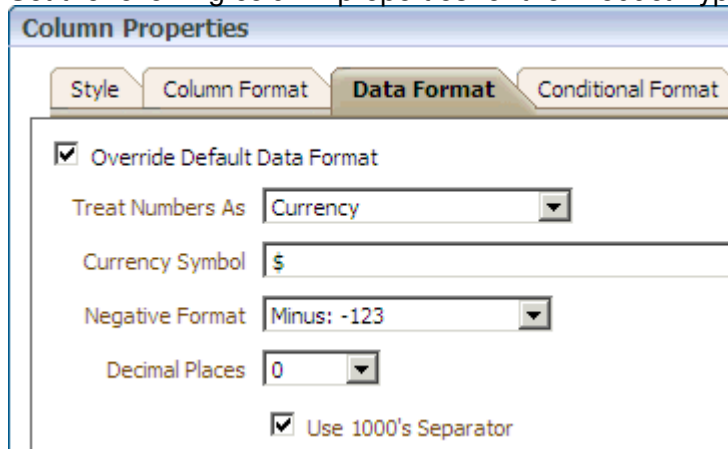
Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Select the **SupplierSales** subject area.
5. Create an analysis to test the level-based measures.

- a. Create the following analysis:



- b. Set the following column properties for the Product Type Dollars column:



- c. Click **Save as Default > Save as the system-wide default for “Fact-Sales”. “Product Type Dollars”**.

- d. Click **Results**. Notice that **Product Type Dollars** returns dollars grouped by Type even though the query is at a different level than Type; Generic in this example.

Generic	Dollars	Product Type Dollars
American Cheese Slices	\$6,545,550.65	\$7,201,383
Apple Dumplings	\$23,923.76	\$2,259,388
Apple Juice	\$1,230,174.71	\$4,470,714
Apple Sauce	\$995,921.24	\$9,123,306
Baked Beans	\$513,499.11	\$5,035,506
Balsamic Vinegar	\$29,154.67	\$5,275,980
Bar Glasses	\$9,000.82	\$4,809,399
Bar Stools	\$2,344.31	\$4,809,399
Barley	\$3,607.48	\$40,566
Beef Bouillon	\$22,356.69	\$5,275,980

- e. Drill down on a value in the **Generic** column. Notice that drilldown is to the **Specific** column by default. Recall that the Specific level in the Product hierarchy had two columns associated with it: Product Key and Specific. Because you set "Use for Display" for the Specific column, and did not set it for the Product Key column, Generic drills to Specific by default. When a user drills down from the next highest level, the default is to drill down to the column with "Use for Display" selected.

Generic	Specific	Dollars	Product Type Dollars
American Cheese Slices	2 Pak American Cheese Slices 16 slices	\$6,545,551	\$7,201,383

6. Create an analysis to test the share measure.

- a. Create the following analysis:

Product	Fact-Sales
Generic	Dollars
	Product Share

- b. Set the following column properties for the Product Share column:

Column Properties

Style Column Format **Data Format** Conditional Format

☒ Override Default Data Format

Treat Numbers As: Percentage

Negative Format: Minus: -123

Decimal Places: 2

☐ Use 1000's Separator

- c. Select **Save as Default > Save as the system-wide default for Fact-Sales.Product Share**.
- d. Sort **Product Share** in descending order.
- e. Click **Results**:

Generic	Dollars	Product Share
American Cheese Slices	\$6,545,551	10.10%
Frozen Chicken	\$4,913,525	7.60%
Frozen Peas	\$3,183,209	4.90%
Canned Tuna	\$1,748,759	2.70%
Frozen Lasagna	\$1,625,759	2.50%
White Shortening	\$1,598,468	2.50%
Fizzy Cola	\$1,499,241	2.30%

The results show total dollars for each product and the percent of total sales for each product sorted in descending order.

- f. Create a similar analysis to view results for Product Type:

Product	Fact-Sales
Type	Dollars Product Share

- g. Click **Results**.

Type	Dollars	Product Share
Condiments	\$9,123,306	14.10%
Poultry	\$7,304,207	11.30%
Cheese	\$7,201,383	11.10%
Baking	\$5,275,980	8.20%
Beef	\$5,088,480	7.90%
Vegetable	\$5,035,506	7.80%
Non-food	\$4,809,399	7.40%
Beverage	\$4,470,714	6.90%

The results show total dollars for each product type and the percent of total sales for each product type sorted in descending order.

7. Create an analysis to test the Average Daily Dollars measure with a dimension-specific aggregation rule.

- a. Create the following analysis and filter:

Customer	Fact-Sales
Region	Dollars Average Daily Dollars

Month Code is equal to / is in 200901

- b. Set the following column properties for the Average Daily Dollars column:

Column Properties

Style

Column Format

Data Format

Conditional Format

☒ Override Default Data Format

Treat Numbers As Currency

Currency Symbol \$

Negative Format Minus: -123

Decimal Places 0

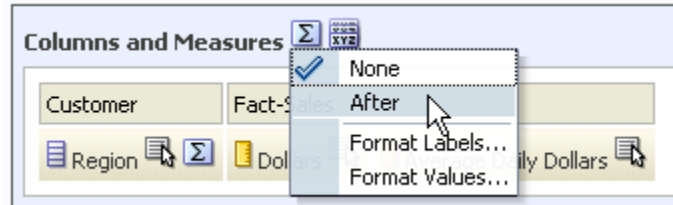
☒ Use 1000's Separator

- c. Click **Results**.

Region	Dollars	Average Daily Dollars
Central	\$954,306	\$38,172
East	\$1,646,242	\$65,850
West	\$1,775,737	\$71,029

- d. Click the **Edit View** button for the Table view.

- e. In the Layout area, click the Total button for “Columns and Measures” and select **After**.



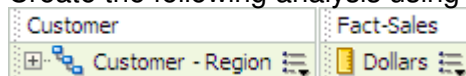
- f. Check your results:

Region	Dollars	Average Daily Dollars
Central	\$954,306	\$38,172
East	\$1,646,242	\$65,850
West	\$1,775,737	\$71,029
Grand Total	\$4,376,285	\$175,051

- g. Considering that there are 25 order days in the first month of 2009, use the Windows calculator to check that the results of the analysis are correct for average daily dollars.

8. Create analyses to test presentation hierarchies.

- a. Create the following analysis using the Customer – Region presentation hierarchy:



- b. Click **Results**.

- c. Expand the hierarchy levels to view data at different levels.

	Dollars
Customer - Region	
[-] Customer Total	\$64,612,461
[-] Central	\$13,423,387
[-] Gulf	\$843,693
[-] MARY SILVER	\$843,693
Alley Dog	\$304,899
Papa Pete's Pizza	\$538,794
[+] LowerMidWest	\$4,753,536
[+] MidWest	\$2,452,673
[+] Texas	\$4,125,482
[+] UpperMidWest	\$1,248,003
[+] East	\$25,460,351
[+] West	\$25,728,722

- d. Build another analysis using the **Customer – Sales Rep** presentation hierarchy and view the results. Notice that the drill path is different than the drill path for the Customer – Region hierarchy.

	Dollars
Customer - Sales Rep	
[-] Customer Total	\$64,612,461
[-] ALAN ZIFF	\$1,527,930
Chang's Mongolian Grill	\$331,217
Globus Office	\$212,182
Half-Shell Restaurant	\$880,096
Times On Bay	\$104,436
[+] ANDREW TAYLOR	\$594,773
[+] ANN JOHNSON	\$2,445,038
[+] ANNE WILLIAMS	\$865,118
[+] BARBARA JENSEN	\$477,633

- e. Sign out of Analysis Editor.

Practice 9-6: Creating Parent-Child Hierarchies

Goal

To create a parent-child hierarchy

Scenario

A parent-child hierarchy is a hierarchy of members that all have the same type. This contrasts with level-based hierarchies, where members of the same type occur only at a single level of the hierarchy. The most common real-life occurrence of a parent-child hierarchy is an organizational reporting hierarchy chart, where the following all apply:

Each individual in the organization is an employee.

Each employee, apart from the top-level managers, reports to a single manager.

The reporting hierarchy has many levels.

In relational tables, the relationships between different members in a parent-child hierarchy are implicitly defined by the identifier key values in the associated base table. However, for each Oracle BI Server parent-child hierarchy defined on a relational table, you must also explicitly define the inter-member relationships in a separate parent-child relationship table.

Outcome

A parent-child hierarchy is added to the business model.

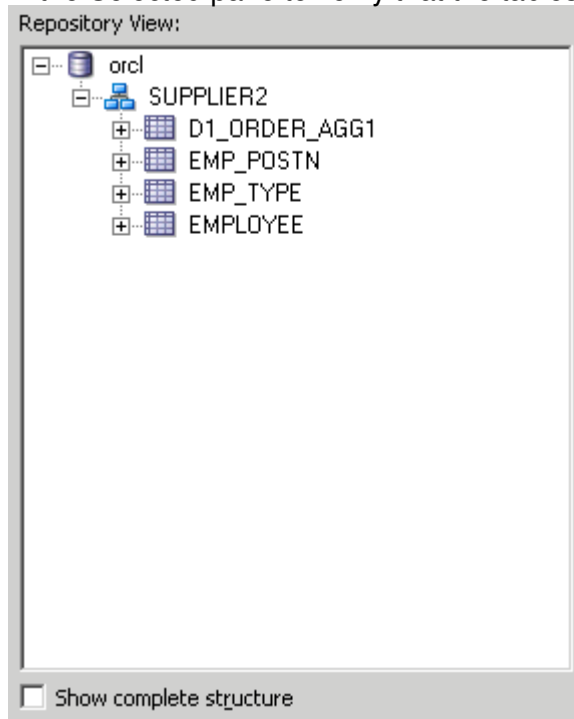
Time

40 minutes

Tasks

1. Import employee tables and an aggregate fact table.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the Physical layer, expand **orcl**.
 - c. Right-click the **SUPPLIER CP** connection pool and select **Import Metadata**.
 - d. Accept the defaults in the Select Metadata Types window and click **Next**.
 - e. In the Select Metadata Objects window, expand the **SUPPLIER2** schema in the Available pane.
 - f. Select the following tables:
D1_ORDER_AGG1
EMP_POSTN
EMP_TYPE
EMPLOYEE
 - g. Click the **Import selected** arrow to add the tables to the Selected pane.

- h. When import completes, deselect “Show complete structure” and expand **SUPPLIER2** in the Selected pane to verify that the tables are added.



- i. Click **Finish** to add the tables to the repository.
 - j. Expand **SUPPLIER2** in the Physical layer and verify that the tables are imported.
2. Explore the new tables.
 - a. Expand the **EMPLOYEE** table. Each row in the table contains two identifying keys, one to identify the member itself (**EMPLOYEE_KEY**) and the other to identify the "parent" of the member (**MGR_ID**). To see the data for this table, right-click the table and select **View Data**.
 - b. Expand the **EMP_POSTN** table. This table contains employee position information.
 - c. Expand the **EMP_TYPE** table. This table contains employee type information.
 - d. Expand the **D1_ORDER_AGG1** table. This table contains sales facts aggregated to the Sales Rep, Product Type, and Month levels.
 - e. Notice that the table missing from the Physical layer is the parent-child relationship table, which you create later in this practice.
3. Create aliases for the tables:

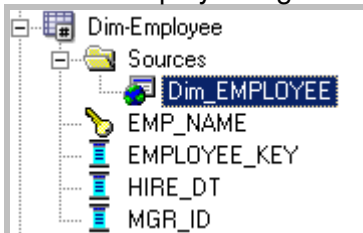
Table	Alias
D1_ORDER_AGG1	Fact D1 ORDER AGG1
EMP_POSTN	Dim_EMP_POSTN
EMP_TYPE	Dim_EMP_TYPE
EMPLOYEE	Dim_EMPLOYEE

4. Use the Physical Diagram to create the following physical joins for the alias tables:

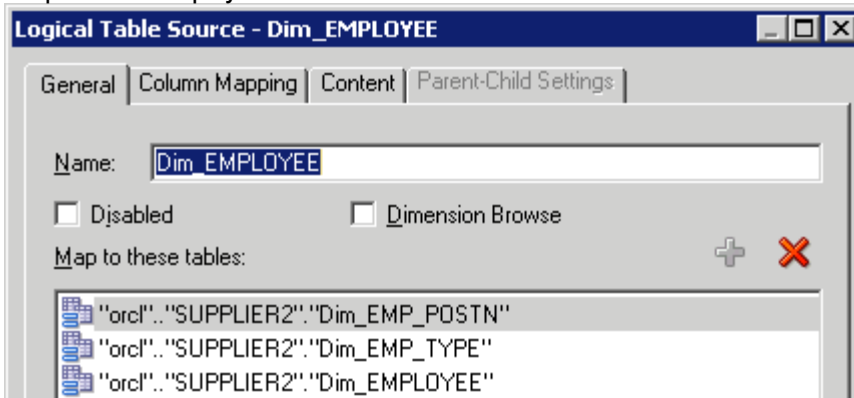
Dim_EMP_TYPE.TYPE_KEY = Dim_EMPLOYEE.EMP_TYPE_KEY
 Dim_EMP_POSTN.POSTN_KEY = DIM_EMPLOYEE.EMP_POSTN_KEY



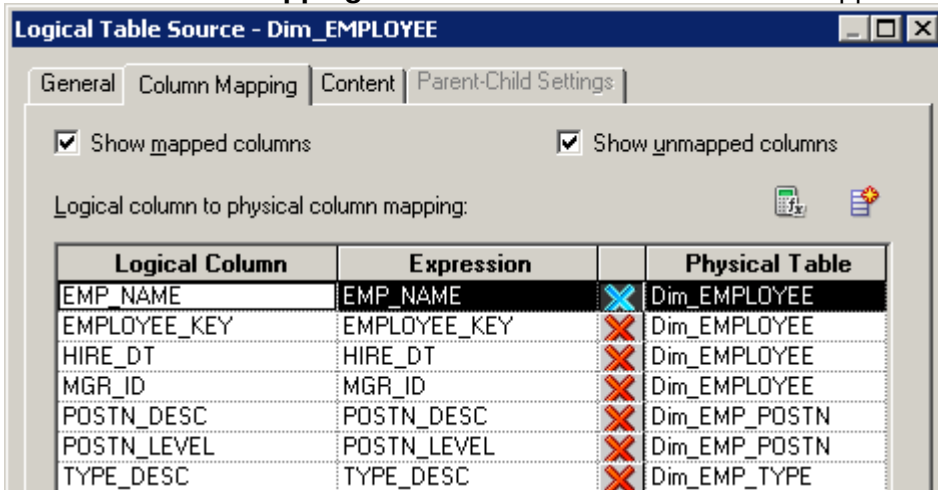
5. Add logical tables and columns for the Employee dimension to the SupplierSales business model.
 - a. Right-click the **SupplierSales** business model and select **New Object > Logical Table**.
 - b. Name the logical table **Dim-Employee** and click **OK**.
 - c. Expand **Dim_EMPLOYEE** in the Physical layer.
 - d. Drag the following columns from **Dim_EMPLOYEE** to **Dim-Employee**:
 - EMP_NAME**
 - EMPLOYEE_KEY**
 - HIRE_DT**
 - MGR_ID**
 - e. Notice that this creates logical columns and a Dim_EMPLOYEE logical table source for the Dim-Employee logical table.



- f. Drag the **TYPE_DESC** column from **Dim_EMP_TYPE** to the **Dim_EMPLOYEE** logical table source (not the Dim-Employee logical table).
- g. Drag the **POSTN_DESC** and **POSTN_LEVEL** columns from **Dim_EMP_POSTN** to the **Dim_EMPLOYEE** logical table source (not the Dim-Employee logical table).
- h. Double-click the **Dim_EMPLOYEE** logical table source to open the Logical Table Source properties dialog box.
- i. Click the **General** tab and notice that the Dim_EMPLOYEE logical table source now maps to three physical alias tables.



- j. Click the **Column Mapping** tab to view how the columns are mapped:



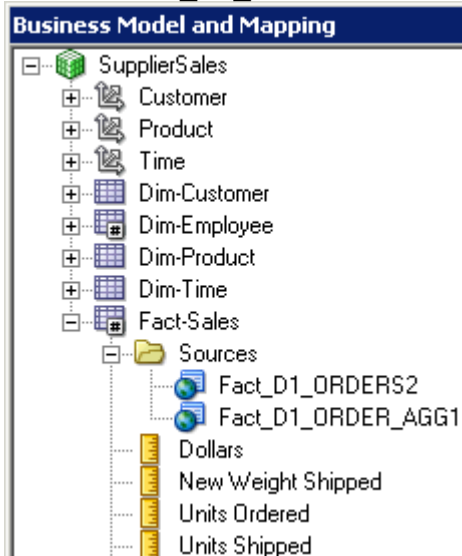
- k. Click **OK** to close the Logical Table Source dialog box.

- l. Rename the logical columns:

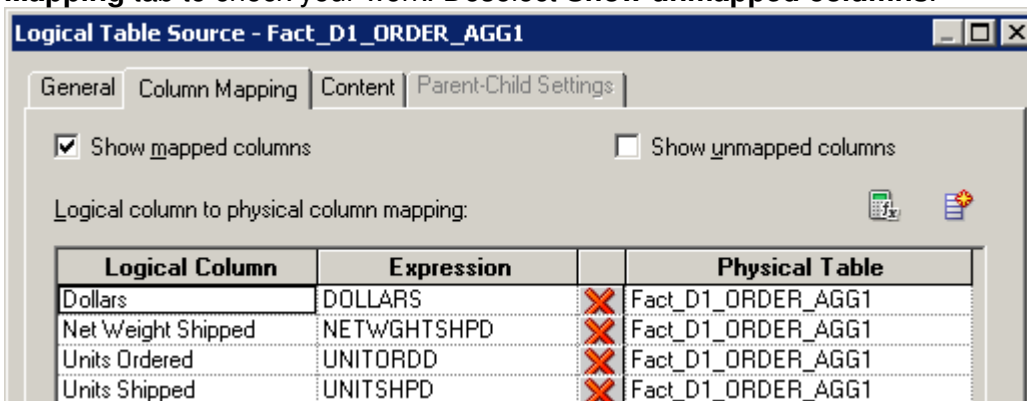
Logical column	Rename to:
EMP_NAME	Employee Name
EMPLOYEE_KEY	Employee ID
HIRE_DT	Hire Date
MGR_ID	Manager ID
TYPE_DESC	Employee Type
POSTN_DESC	Position
POSTN_LEVEL	Position Level

6. Set the key for the Dim-Employee logical table.
 - a. Double-click the **Dim-Employee** logical table.
 - b. Click the **Keys** tab.
 - c. In the Key Name column, enter **Employee ID**.
 - d. In the Columns field, use the drop-down list to select **Employee ID**.
 - e. Click **OK**.
7. Create a new logical source and columns within the existing logical fact table.
 - a. In the Physical layer, expand the **Fact_D1_ORDER_AGG1** table.
 - b. In the Business Model and Mapping layer, expand **Fact-Sales**.
 - c. Drag the **DOLLARS**, **NETWGHTSHPD**, **UNITORDD**, and **UNITSHPD** columns one at a time from Fact_D1_ORDER_AGG1, and drop each onto their corresponding Fact-Sales logical columns: **Dollars**, **Net Weight Shipped**, **Units Ordered**, and **Units Shipped**. This creates a new Fact_D1_ORDER_AGG1 logical table source and corresponding column mappings.

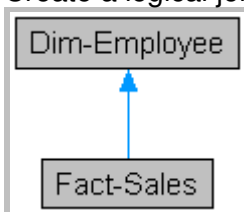
- d. Expand the **Fact-Sales > Sources** folder. Notice that there are now two logical table sources: **Fact_D1_ORDERS2** and **Fact_D1_ORDER_AGG1**.



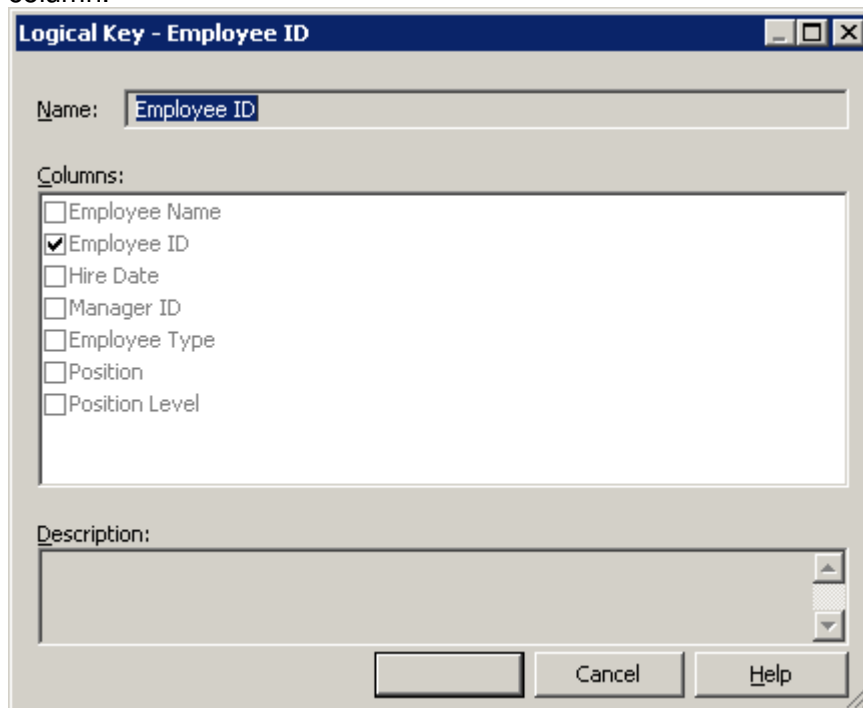
- e. Double-click the **Fact_D1_ORDER_AGG1** logical table source and click the **Column Mapping** tab to check your work. Deselect **Show unmapped columns**.



- f. Click **OK** to close the Logical Table Source dialog box.
- g. Double-click the **Fact_D1_ORDERS2** logical table source and click the **Column Mapping** tab. Notice that these four logical columns, Dollars, Units Shipped, Units Ordered, and Net Weight Shipped, now map to their corresponding columns in both the **Fact_D1_ORDERS2** table and the **Fact_D1_ORDER_AGG1** table.
- h. Click **OK** to close the Logical Table Source dialog box.
8. Create a logical join for the **Dim-Employee** table.
- In the **SupplierSales** business model, select **Dim-Employee** and **Fact-Sales**.
 - Right-click either of the highlighted tables and select **Business Model diagram > Selected Tables Only** to open the Business Model Diagram.
 - Create a logical join between **Dim-Employee** and **Fact-Sales**.



- d. Close the Business Model Diagram.
9. Create a parent-child logical dimension based on the Dim-Employee logical table.
 - a. Right-click the **Dim-Employee** logical table and select **Create Logical Dimension > Dimension with Parent-Child Hierarchy** to open the Logical Dimension dialog box.
 - b. On the General tab, name the logical dimension **Employee**.
 - c. Click **Browse** next to Member Key. The browse window shows the logical dimension table in the business model with its corresponding keys.
 - d. Click **View** to view the key column. Verify that the key points to the **Employee ID** column.



- e. Click **Cancel**.
- f. Click **OK** in the Browse window.
- g. Click **Browse** next to Parent Column.
- h. The **Browse** window shows the columns, other than the primary key, in the logical table that you selected in the previous step.
- i. Deselect **Show Qualified name**.
- j. Select **Manager ID** as the Parent Column for the parent-child hierarchy and click **OK**.
- k. Do not close the **Logical Dimension - Employee** dialog box.
10. Define the parent-child settings. At this point, if the logical table that you selected was not from a relational table source, you could click OK in the Logical Dimension dialog box to finish the process of creating the dimension. However, because the logical table you selected is from a relational table source, you must continue the dimension definition process to set up a parent-child relationship table for the hierarchy. For each parent-child hierarchy defined on a relational table, you must also explicitly define the inter-member relationships in a separate parent-child relationship table. In the process of creating the parent-child relationship table, you may choose one of the following options: 1. Select a previously-created parent-child relationship table. 2. Use a wizard that will generate scripts to create and populate the parent-child relationship table. In the next set of steps, you use a wizard to generate scripts to create and populate the parent-child relationship table. Later in




this practice you perform the steps to select a previously-created parent-child relationship table.

- a. In the Logical Dimension dialog box, select **Parent-Child Settings** to display the Parent-Child Table Settings dialog box. Notice that at this point the Parent-Child Relationship table is not defined.

Parent-Child Relationship Table Settings

In order to enable querying of all ancestors and descendants of a given repository object in addition to direct parent and children, a parent-child relationship table association is necessary with the relational logical table source of the hierarchy.

Select a relational parent-child table source to view its column details.

Logical Table	Logical Table Source	Parent-Child Relationship Table
Dim-Employee	Dim_EMPLOYEE	  

Parent-Child Relationship Table Column Details

Member Key:

Parent Key:

Relationship Distance:

Leaf Node Identifier:

OK Cancel Help

- b. Click the **Create Parent-Child Relationship Table button** (blue asterisk) to start the wizard. As you will see, the wizard generates SQL scripts for creating and populating the parent-child relationship table. At the end of the wizard, Oracle BI Server stores the scripts into directories chosen during the wizard session. The scripts, when executed, will make the parent-child relationship table available to the repository.
- c. In the Generate Parent-Child Relationship Table - Script Location window, enter **EMP_PARENT_CHILD_Create** as the name for the DDL script to create the parent-child table.
- d. In the Location field, accept the default location where the script will be stored.
- e. Enter **EMP_PARENT_CHILD_Populate** as the name for the DDL script to populate the parent-child table.

- f. In the Location field, accept the default location where the script will be stored.

DDL Script to Create Parent-Child Relationship Table

Name: EMP_PARENT_CHILD_Create

Location: d:\bi\instances\instance1\bfoundation\OracleB: Browse...

DDL Script to Populate Parent-Child Relationship Table

Name: EMP_PARENT_CHILD_Populate

Location: d:\bi\instances\instance1\bfoundation\OracleB: Browse...

- g. Click **Next**.
- h. Enter **EMP_PARENT_CHILD** as the name for the parent-child relationship table.

Provide details for the parent-child relationship table that will be created/populated on executing the scripts generated in this wizard.

Name: EMP_PARENT_CHILD

Description

Physical Location

Data Source: orcl

Catalog/Schema: SUPPLIER2 Browse...

Logical Associations

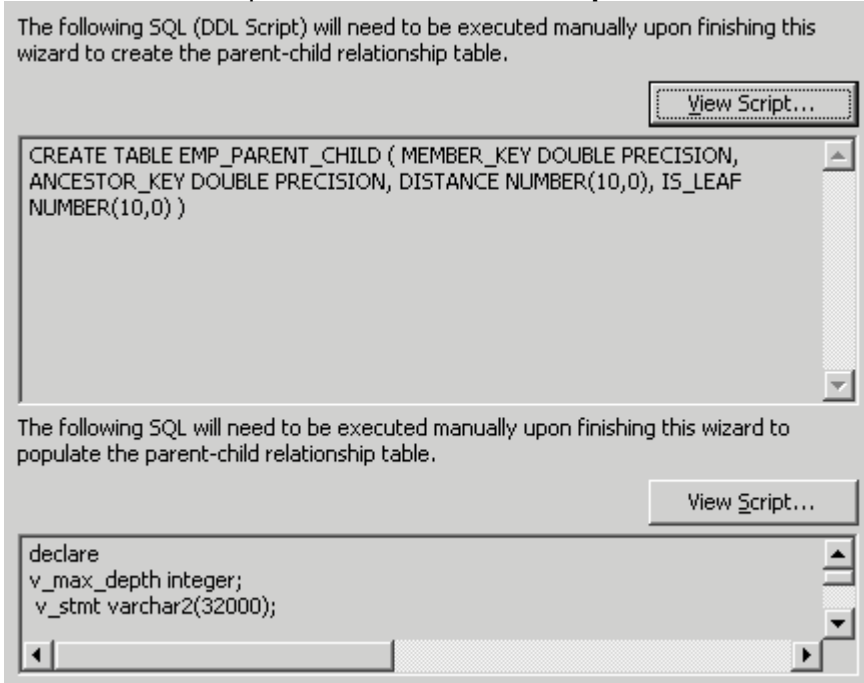
Dimension: Logical Dimension

Logical Table: Dim-Employee

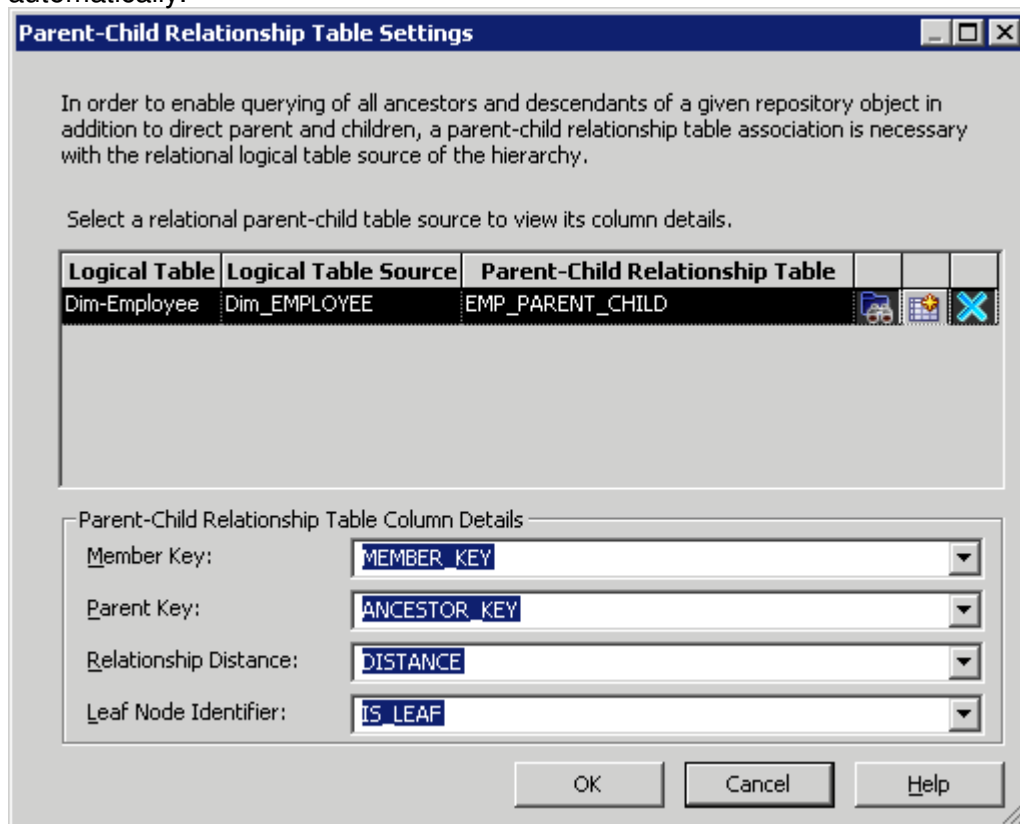
Logical Table Source: Dim_EMPLOYEE

- i. Accept the defaults for the physical location and logical associations, and click **Next**.

- j. In the Preview Script window, click **View Script** to view either or both of the scripts.



- k. Click **Finish** to close the wizard.
- l. Notice that **EMP_PARENT_CHILD** is now defined as the parent-child table for the logical table source.
- m. Notice also that the parent-child table column details have been populated automatically.



EMP_PARENT_CHILD table has already been created and populated in the database.
The EMP_PARENT_CHILD physical layer object now points to the
EMP_PARENT_CHILD physical table in the database.

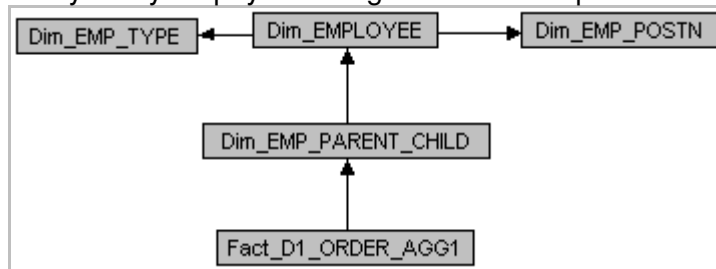
11. Modify Physical layer objects. After adding the parent-child relationship table to the Physical layer, you must make some modifications in both the Physical layer and the Business Model and Mapping layer.

- a. Create an alias for the **EMP_PARENT_CHILD** table named **Dim_EMP_PARENT_CHILD**.
- b. Select the following tables and open the Physical Diagram:
Dim_EMP_PARENT_CHILD
Dim_EMP_POSTN
Dim_EMP_TYPE
Dim_EMPLOYEE
Fact_D1_ORDER_AGG1
- c. Create the following new join relationships in the Physical layer:

```
"orcl"."SUPPLIER2"."Dim_EMPLOYEE"."EMPLOYEE_KEY" =
"orcl"."SUPPLIER2"."Dim_EMP_PARENT_CHILD"."ANCESTOR_KEY"

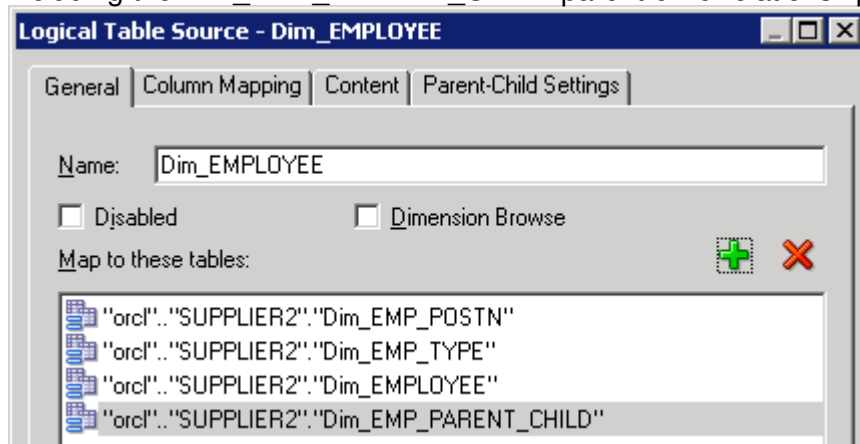
"orcl"."SUPPLIER2"."Dim_EMP_PARENT_CHILD"."MEMBER_KEY" =
"orcl"."SUPPLIER2"."Fact_D1_ORDER_AGG1"."SREP_KEY"
```

- d. Verify that your physical diagram relationships look similar to the following screenshot:



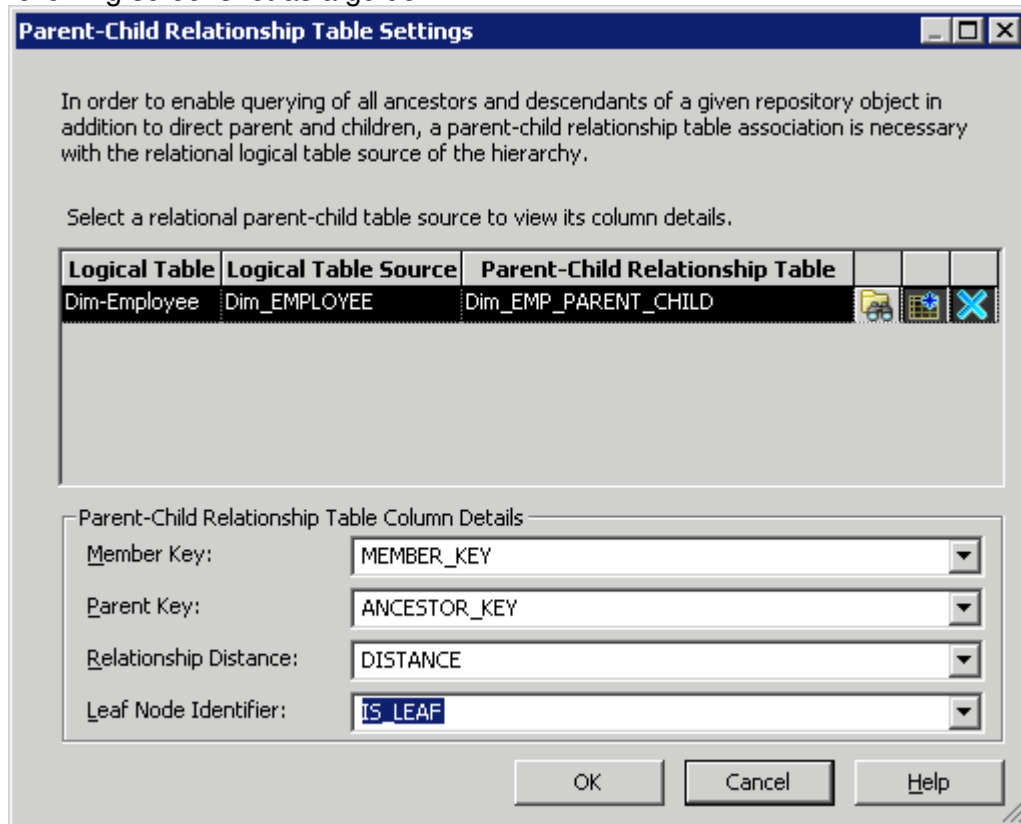
- e. Close the Physical Diagram.
12. Modify business model objects.
 - a. In the Physical layer, expand **Dim_EMP_PARENT_CHILD**.
 - b. In the Business Model and Mapping layer, expand **Dim-Employee > Sources**.
 - c. Drag the **DISTANCE** column from Dim_EMP_PARENT_CHILD to the Dim_EMPLOYEE logical table source. This action creates a new logical column and maps the Dim_EMPLOYEE logical table source to the Dim_EMP_PARENT_CHILD table.
 - d. Double-click the **Dim_EMPLOYEE** logical table source.
 - e. Click the **General** tab.

- f. Notice that the Dim_EMPLOYEE logical table source is now mapped to four tables including the Dim_EMP_PARENT_CHILD parent-child relationship table.

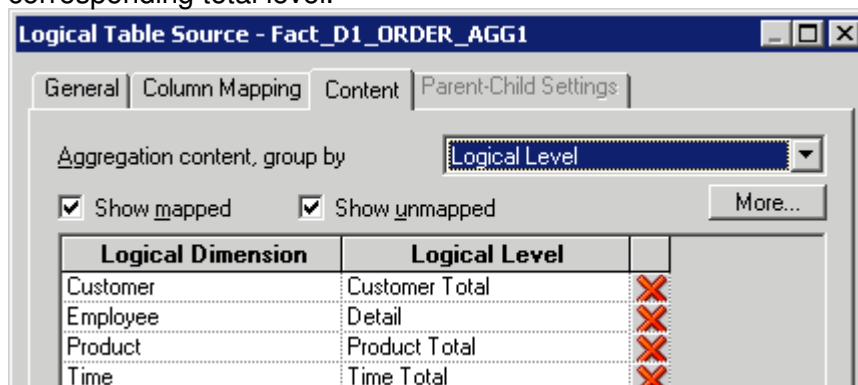


- g. Click **OK** to close the Logical Table Source dialog box.
- h. Rename the **DISTANCE** logical column to **Distance**.
13. This next set of steps demonstrates the second technique for defining the parent-child relationship table, which is to select a previously-created, existing parent-child relationship table. In this example, you select the Dim_EMP_PARENT_CHILD alias table.
- Double-click the **Employee** parent-child logical dimension to open the Logical Dimension dialog box.
 - On the General tab, click **Parent-Child Settings** to open the Parent-Child Table Settings dialog box.
 - Notice that, as expected, the parent-child table is the table generated by the wizard: EMP_PARENT_CHILD.
 - To set the parent-child table to an existing parent-child relationship table, click the **Select Parent-Child Relationship Table** button.
 - In the Select Parent-Child Relationship Table dialog box, select the **Dim_EMP_PARENT_CHILD** alias that you created.
 - The parent-child table is now set to the alias table instead of the original physical table generated by the wizard. Recall that you defined your physical joins using the alias table.

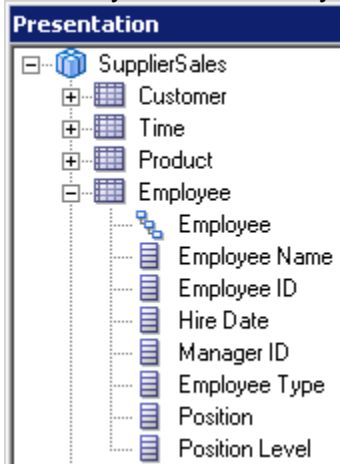
- g. In the Parent-Child Table Column Details section, set the appropriate columns. Use the following screenshot as a guide:



- h. Click **OK** to close the Parent-Child Relationship Table Settings dialog box.
- i. Click **OK** to close the Logical Dimension dialog box.
14. Set aggregation content levels for the Fact-Sales logical table sources. Because you added a new logical dimension, you must set the logical levels.
- Expand **Fact-Sales > Sources**.
 - Double-click the **Fact_D1_ORDERS2** logical table source to open the Logical Table Source dialog box.
 - On the Content tab, set the logical level for the Employee logical dimension to **Total**.
 - Click **OK** to close the Logical Table Source dialog box.
 - Double-click the **Fact_D1_ORDER_AGG1** logical table source.
 - On the Content tab, set the Employee level to **Detail** and all other logical levels to the corresponding total level.

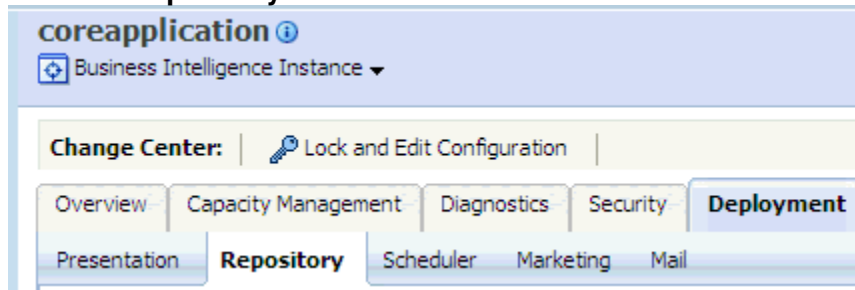


- g. Click **OK** to close the Logical Table Source dialog box.
- 15. Make the parent-child logical dimension available for queries.
 - a. Drag the **Dim-Employee** logical table to the SupplierSales subject area.
 - b. Rename the **Dim-Employee** presentation table **Employee**.
 - c. Move the **Employee** presentation table above the Fact-Sales presentation table.
 - d. Expand the **Employee** presentation table and notice that the Employee presentation hierarchy is automatically added to the presentation table.

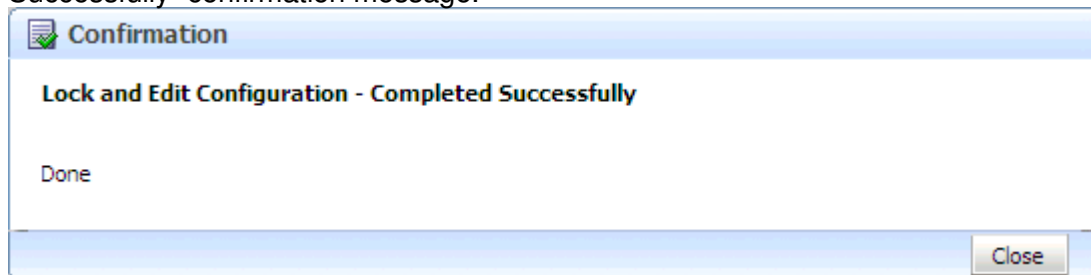


- e. Double-click the **Employee** presentation hierarchy to open the Presentation Hierarchy dialog box.
- f. On the **Display Columns** tab, select **Employee ID**.
- g. Click the **red X** to remove this display column.
- h. Click the **Add** button (green plus sign).
- i. Select **Employee Name** as a display column.
- j. Click **OK** to close the Presentation Hierarchy dialog box.
- k. Save the repository.
- l. Check consistency. Fix any errors or warnings before proceeding.
- m. Close the repository.
- n. Leave the Administration Tool open.
- 16. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
<http://localhost:7001/em>.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.

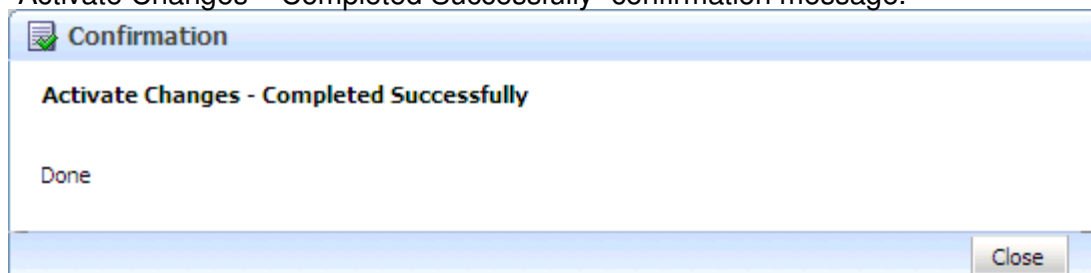
- e. Click the **Repository** subtab.



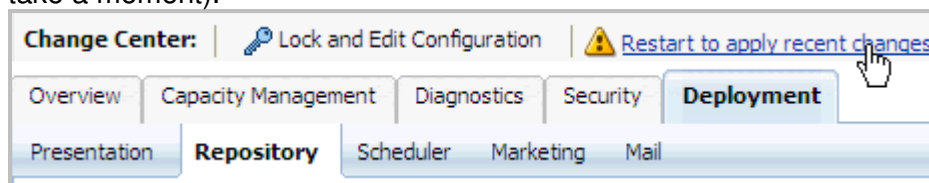
- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the “Choose file” dialog box.
- i. By default, the “Choose file” dialog box should open to the default repository directory. If not, browse to
D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension, for example, ABC_BI0007.
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



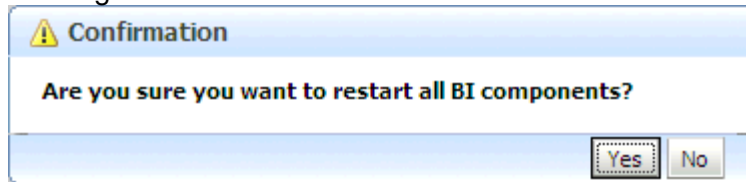
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



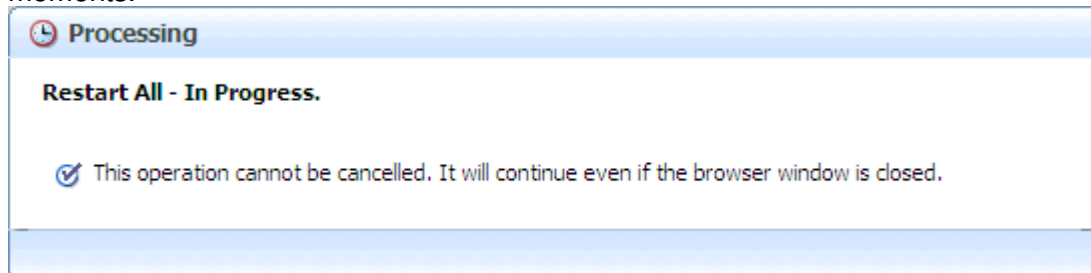
- p. On the Overview page, click **Restart**.



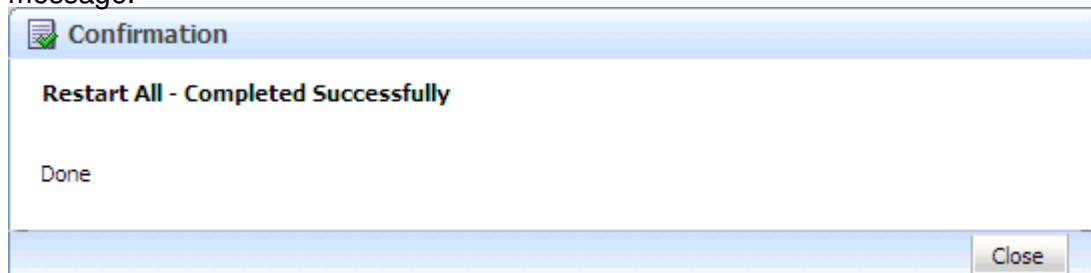
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



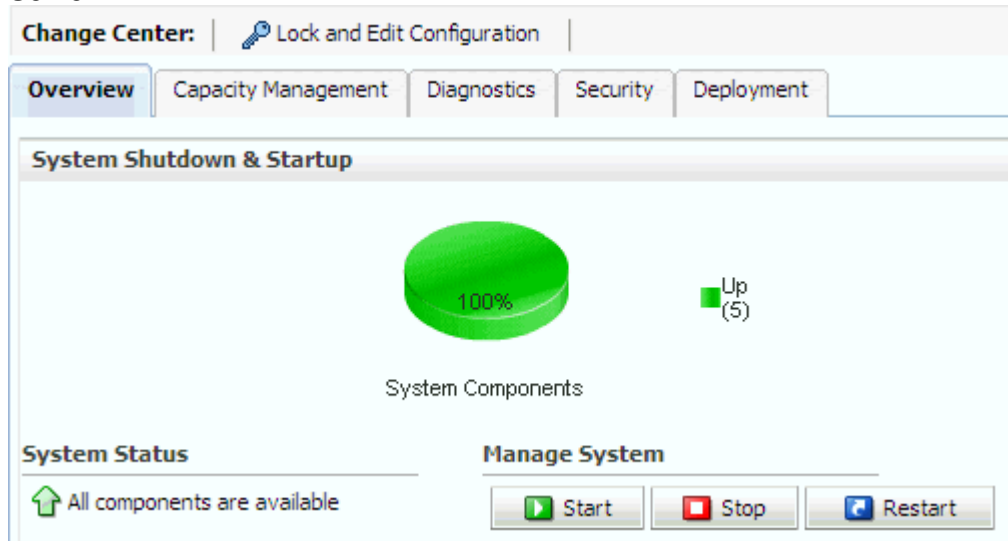
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



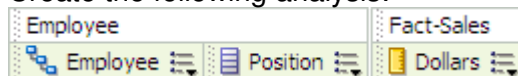
- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
17. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.

Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).

- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Select the **SupplierSales** subject area.
18. Create an analysis to check your work.
- Create the following analysis:



- Check **Results**. Expand the pivot table to view data at different levels of the hierarchy. Notice that the Dollars measure rolls up through each level.

		Dollars
Employee	Position	
<input type="checkbox"/> PAULA MADISON	Sales VP	\$63,132,455
<input checked="" type="checkbox"/> GEORGE MASUR	Sr. Supervisor	\$601,455
<input checked="" type="checkbox"/> LYLE IRWIN	Sr. Manager	\$12,991,176
<input type="checkbox"/> MARY SILVER	Manager	\$18,355,438
<input checked="" type="checkbox"/> JOSE CRUZ	Supervisor	\$7,243,603
<input checked="" type="checkbox"/> KATHY LOBO	Supervisor	\$5,169,673
<input type="checkbox"/> LILLIAN BAYER	Supervisor	\$5,127,514
DALE AREND	Associate	\$1,111,591
DONALD KIMBRIEL	Senior	\$689,577
TIM ALLEN	Associate	\$873,659
TRACIE BELL	Associate	\$113,121
<input checked="" type="checkbox"/> MAYNARD WAGNER	Manager	\$29,367,508

- Leave Oracle BI open for the next practice.

Practices for Lesson 10: Using Aggregates

Chapter 10

Overview of Practices for Lesson 10

Lesson Overview

In these practices, you will add aggregate tables to the business model to improve performance.

Practice 10-1: Using Aggregate Tables

Goal

To use aggregated data tables to improve performance of summarized queries

Scenario

ABC wants to add aggregate tables that store pre-computed results that are aggregated measures over a set of dimensional attributes. You must specify the level of aggregation for each source using logical levels. The necessary aggregate tables are already in the database:

MONTHS contains one row for each year and month combination, which can be considered an aggregation of the Time dimension to the Month level.

D1_PRODUCT_TYPE is already part of your model and contains one row for each product type, which can be considered an aggregation of the Product dimension to the Product Type level.

D1_ORDER_AGG1 is already part of your model and contains sales facts aggregated to the Product Type, and Month levels.

Outcome

In the Physical layer, there are new physical sources for the aggregate tables. In the Business Model and Mapping layer, aggregate content is defined for the logical table sources.

Time

45 minutes

Tasks

1. Import the **MONTHS** physical aggregate table from the database into the Physical layer of the repository. You need both aggregate fact and aggregate dimension tables because you must create logical dimension sources at the same level of detail as the fact sources.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the Physical layer, expand **orcl**.
 - c. Right-click the **SUPPLIER CP** connection pool and select **Import Metadata**.
 - d. Accept the defaults in the Select Metadata Types screen and click **Next** to open the Select Metadata Objects screen.
 - e. Expand the **SUPPLIER2** schema.
 - f. In the **Data source view** pane, select the **MONTHS** table for import.
 - g. Click the **Import selected** button to add the table to the Repository View pane.
 - h. Expand **SUPPLIER2** in the Repository View pane and verify that **MONTHS** is added.
 - i. Click **Finish** to add the tables to the repository.
 - j. Expand **SUPPLIER2** in the Physical Layer and confirm that **MONTHS** is added to the repository.
 - k. Expand **MONTHS**.
 - l. Double-click the **YEAR** column to open the Physical Column dialog box.
 - m. Change the column type to **INT**.
 - n. Repeat for **MONTHCODE**.
 - o. Create a **Dim_MONTHS_AGG** alias tables for **MONTHS**.
-

2. Create an alias for the product type table.
 - a. Right-click **D1_PRODUCT_TYPE** and select **New Object > Alias**.
 - b. On the General tab, name the alias **Dim_D1_PRODUCT_TYPE_AGG**.
 - c. Click **OK** to close the Physical Table dialog box.
3. View the data in the aggregate tables.
 - a. Right-click **Fact_D1_ORDER_AGG1** and select **View Data**. Fact_D1_ORDER_AGG1 contains sales facts aggregated to the Product Type (TYPEKEY), and Month (PERKEY) levels. It contains 10212 rows, compared to Fact_D1_ORDERS2, which contains 351636 rows.

DOLLARS	NETWGHTSHPD	PERKEY	SALESREP	SREP_KEY	TYPEKEY
751.22	461.09	200902.0	BARBARA JENSEN	5.0	112
1676.12	1758.03	200903.0	BARBARA JENSEN	5.0	112
5691.81	1100.25	200904.0	BARBARA JENSEN	5.0	112
44.01	53.0	200802.0	BARBARA JENSEN	5.0	113
30.65	52.0	200803.0	BARBARA JENSEN	5.0	113
39.17	43.0	200804.0	BARBARA JENSEN	5.0	113
10.97	21.0	200805.0	BARBARA JENSEN	5.0	113
21.94	42.0	200806.0	BARBARA JENSEN	5.0	113
8.71	10.0	200807.0	BARBARA JENSEN	5.0	113

- b. Close the **View Data** window.
- c. View data for the **Dim_MONTHS_AGG** table. The Dim_MONTHS_AGG table contains one row for each year and month combination, which is an aggregation of the Time dimension to the Month level. It contains 16 rows, compared to Dim_D1_CALENDAR2, which contains 474 rows.

MAGO	MONTH_IN_YEAR	MONTHCODE	MONTHNAME	QUARTER	QUARTERDESC	YAGO
200712.0	1	200801.0	January	1.0	Q1_2008	200701.0
200801.0	2	200802.0	February	1.0	Q1_2008	200702.0
200802.0	3	200803.0	March	1.0	Q1_2008	200703.0
200803.0	4	200804.0	April	2.0	Q2_2008	200704.0
200804.0	5	200805.0	May	2.0	Q2_2008	200705.0
200805.0	6	200806.0	June	2.0	Q2_2008	200706.0
200806.0	7	200807.0	July	3.0	Q3_2008	200707.0
200807.0	8	200808.0	August	3.0	Q3_2008	200708.0
200808.0	9	200809.0	September	3.0	Q3_2008	200709.0

- d. View data for the **Dim_D1_PRODUCT_TYPE_AGG** table. Dim_D1_PRODUCT_TYPE_AGG contains one row for each product type, which is an aggregation of the Product dimension to the Type level. It contains 21 rows, compared to Dim_D1_PRODUCTS, which contains 192 rows.

ITEMTYPE	TYPECODE
Baking	100
Beef	101
Beverage	102
Bread	103
Cereal	104
Cheese	105
Condiments	106
Dessert	107
EnrTe	108
Frozen	109

4. Create physical joins between the aggregate fact and aggregate dimension tables.
 - a. In the Physical layer, select the following three tables:
Dim_D1_PRODUCT_TYPE_AGG
Dim_MONTHS_AGG
Fact_D1_ORDER_AGG1

- b. Click the **Physical Diagram** icon on the toolbar.
- c. Use the **New Join** button and create the following joins:
`Dim_MONTHS_AGG.MONTHCODE = Fact_D1_ORDER_AGG1.PERKEY`
`Dim_D1_PRODUCT_TYPE_AGG.TYPECODE = Fact_D1_ORDER_AGG1.TYPEKEY`
- d. Check your work:



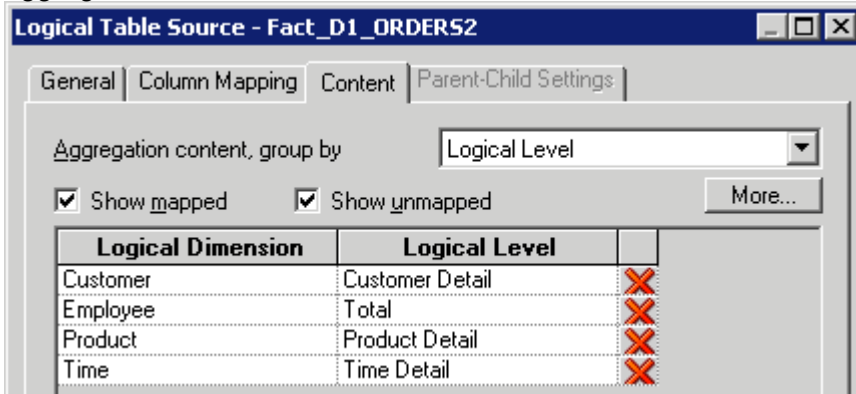
- e. Close the Physical Diagram.
5. Specify the aggregation content of the logical table source for the Fact-Sales logical table so that Oracle BI Server knows what level of data is stored in the aggregate tables.
 - a. In the Business Model and Mapping layer, expand **Fact-Sales > Sources**.
 - b. Double-click the **Fact_D1_ORDER_AGG1** logical table source to open its properties dialog box. Recall that you created this logical table source in the practices for the previous lesson.
 - c. Click the **Column Mapping** tab. Recall that the four measure columns, Dollars, Units Ordered, Units Shipped, and New Weight Shipped, map to both the Fact_D1_ORDERS2 and Fact_D1_ORDER_AGG1 tables.
 - d. Click the **Content** tab.
 - e. In the “Aggregation content, group by” field, ensure that the value is **Logical Level**. This is the default.
 - f. Use the drop-down lists in the Logical Level field to specify the aggregation content as follows:

Logical Dimension	Logical Level	
Customer		X
Employee	Detail	X
Product	Type	X
Time	Month	X

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 “know” to access the aggregate tables instead of the detail tables. For example, if a user queries for total sales by month, the server will access the Fact_D1_ORDER_AGG1 aggregate fact table and the corresponding aggregate dimension table, Dim_MONTHS_AGG. If a user queries for a level lower than the levels specified here (for example, day instead of month), the server will access the detail tables (Fact_D1_ORDERS2, and Dim_D1_CALENDAR2). If a user queries for higher level (year instead of month), the aggregate tables will be used as well, because when a query is run against a logical level or above, the aggregate tables are used.

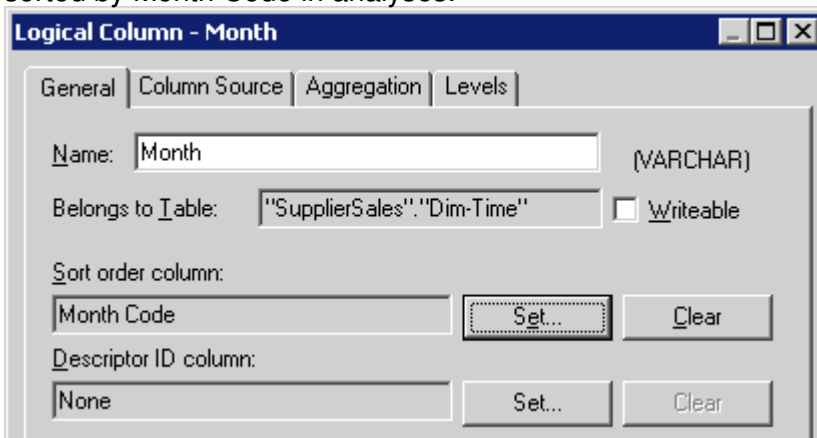
- g. Click **OK** to close the Logical Table Source dialog box for Fact_D1_ORDER_AGG1.

- h. Specify the content for the remaining fact logical table source, Fact_D1_ORDERS2. You are doing this because it is good practice to set the levels for the detail source to the lowest levels in the hierarchies. This is because you want the server to access the detail tables when queries are against levels lower than those specified for the aggregate tables. It is also a good practice to specify the content of all sources for documentation purposes, as another administrator could interpret the lack of an aggregation content statement as an inadvertent omission of information.



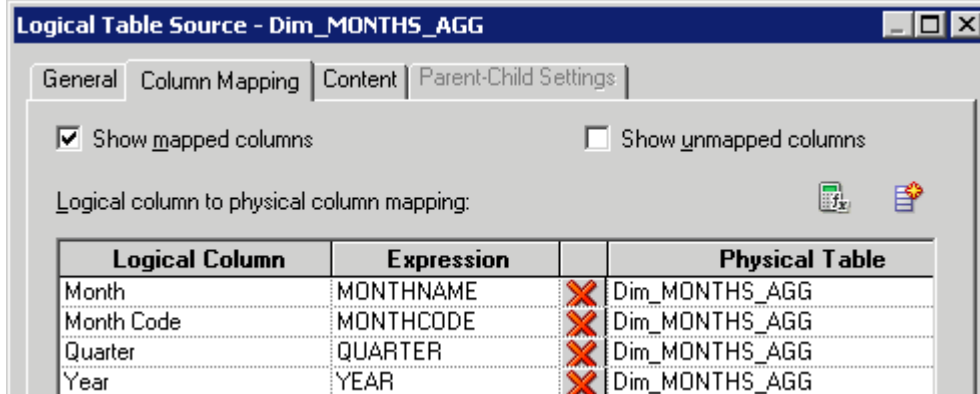
Please be aware that the name of the detail level for the Customer logical dimension may be different from the screenshot, depending on which method you used to create the Customer logical dimension in the previous set of practices. Notice also that Employee is a parent-child logical dimension, so the logical level is set to Total.

- i. Click **OK** to close the Logical Table Source dialog box.
6. Create a new source within the Dim-Time logical table that points to the Dim_MONTHS_AGG aggregate table.
 - j. In the Physical layer, expand the **Dim_MONTHS_AGG** table.
 - k. In the Business Model and Mapping layer, expand **Dim-Time > Sources**.
 - l. Drag the **MONTHCODE**, **MONTHNAME**, **QUARTER**, and **YEAR** columns one at a time from Dim_MONTHS_AGG onto the corresponding Dim-Time logical columns to create a new logical table source, Dim_MONTHS, and the corresponding column mappings (drag MONTHNAME to MONTH).
 - m. Double-click the **Month** column to open the Logical Column dialog box.
 - n. On the General tab, set the sort-order column to **Month Code**, so that Month is always sorted by Month Code in analyses.



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

- p. Double-click the **Dim_MONTHS_AGG** logical table source to open the Logical Table Source dialog box.
- q. Click the **Column Mapping** tab to check your work.



- r. Leave the Logical Table Source dialog box open.
7. Specify the aggregation content for the Dim_MONTHS_AGG logical table source for the Dim-Time logical table so that Oracle BI Server knows what level of data is stored in the aggregate table. Recall that the Dim_MONTHS_AGG table contains data aggregated at the month level.
 - a. Select the **Content** tab in the Dim_MONTHS logical table source.
 - b. In the “Aggregation content, group by” field, ensure that the value is **Logical Level**.
 - c. For the Time logical dimension, set the logical level to **Month**:

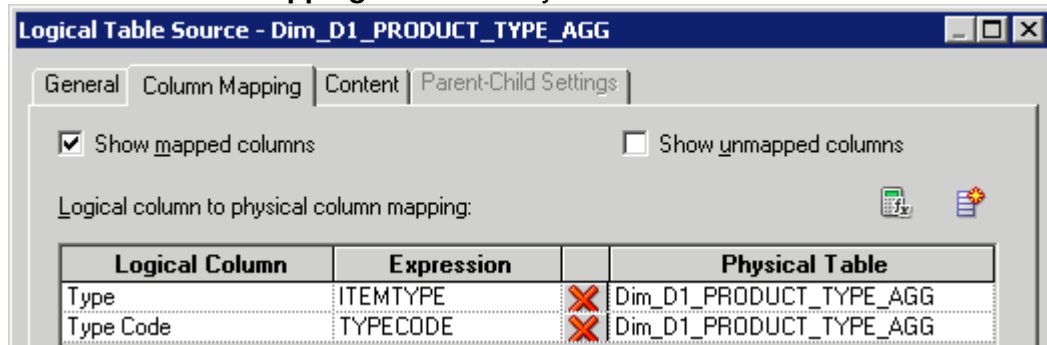


- d. Click **OK**.
- e. Confirm that the logical level is set to **Time Detail** for the remaining logical table source, Dim_D1_CALENDAR2. It is best practice to set the levels for the detail source to the lowest logical level in the hierarchy.

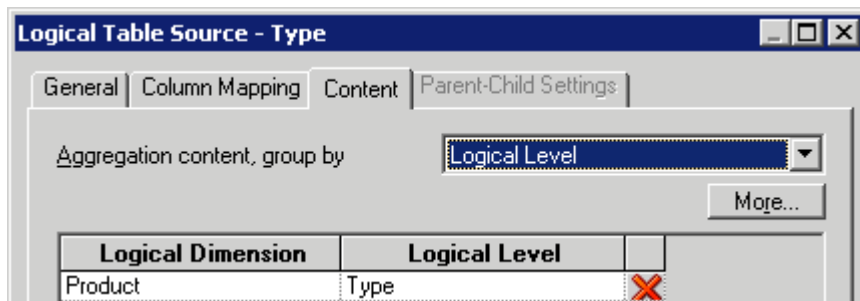


- f. Click **OK** to close the Logical Table Source dialog box.
8. Create a new source within the Dim-Product logical table that points to the Dim_D1_PRODUCT_TYPE_AGG aggregate table.
 - a. In the Physical layer, expand the **Dim_D1_PRODUCT_TYPE_AGG** table.
 - b. In the Business Model and Mapping layer, expand **Dim-Product > Sources**.

- c. Drag the **ITEMTYPE** and **TYPECODE** columns one at a time from Dim_D1_PRODUCT_TYPE_AGG onto the corresponding Dim-Product logical columns, Type and Type Code, respectively, to create a new logical table source, Dim_D1_PRODUCT_TYPE_AGG, and the corresponding column mappings.
- d. Double-click the **Dim_D1_PRODUCT_TYPE_AGG** logical table source to open the Logical Table Source dialog box.
- e. Click the **Column Mapping** tab to check your work.

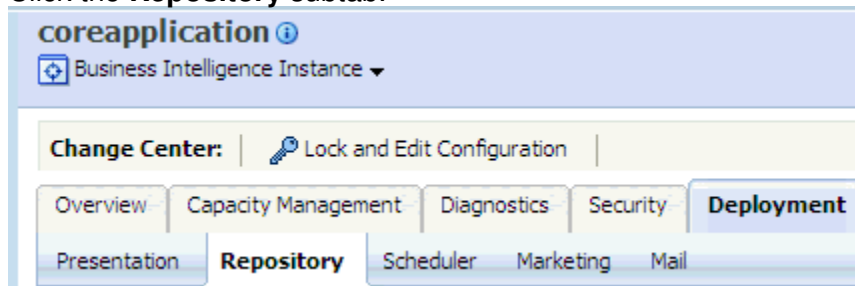


- f. Leave the Logical Table Source dialog box open.
9. Set the aggregation content for the logical table sources for the Dim-Product logical table. Use the following screenshots as a guide.

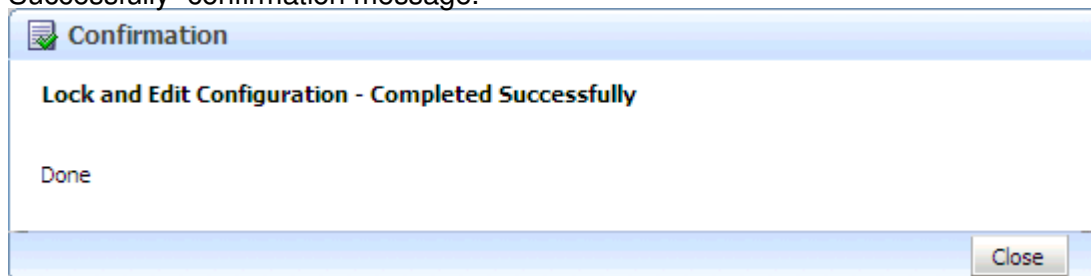


10. Remove an unnecessary key from Dim-Product.
 - a. Double-click **Dim-Product** to open the Logical Table dialog box.

- b. Click the **Keys** tab.
 - c. Select **Dim_D1_PRODUCT_TYPE_AGG_Key**. This key was created when you dragged columns from the **Dim_D1_PRODUCT_TYPE_AGG** physical table.
 - d. Click the red **X** to delete the key. This key is not needed and will generate consistency check errors if not removed.
 - e. Click **OK** to close the Logical Table dialog box.
11. Save the repository.
12. Check global consistency.
13. Fix any errors or warnings before you continue.
15. Close the repository. Leave the Administration Tool open.
16. Notice that you do not need to change the Presentation layer. You made changes in the business model that impact how the queries are processed and which sources are accessed. However, the user interface remains the same, so there is no need to change the Presentation layer. It will automatically use the new sources.
17. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
http://localhost:7001/em.
 - b. If your session has timed out, log in as **weblogic/welcome1**.
 - c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
 - d. In the right pane, click the **Deployment** tab.
 - e. Click the **Repository** subtab.

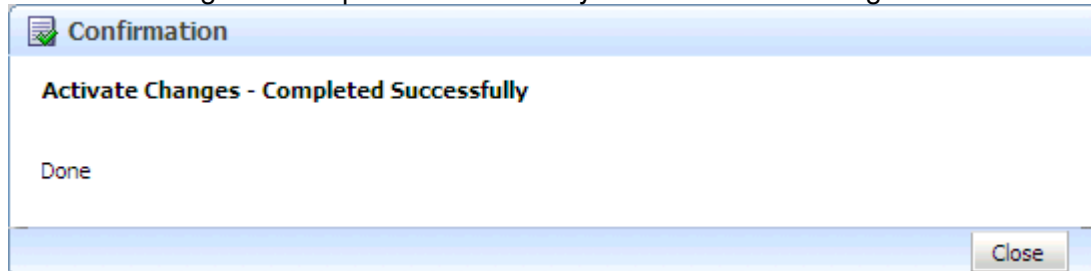


- f. Click **Lock and Edit Configuration**.
 - g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.

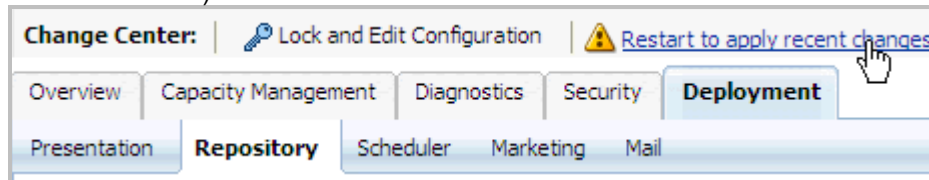


- h. In the Upload BI Server Repository section, click **Browse** to open the “Choose file” dialog box.
 - i. By default, the “Choose file” dialog box should open to the default repository directory. If not, browse to
**D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplicatio
n_obis1\repository.**

- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension (for example, ABC_BI0007).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



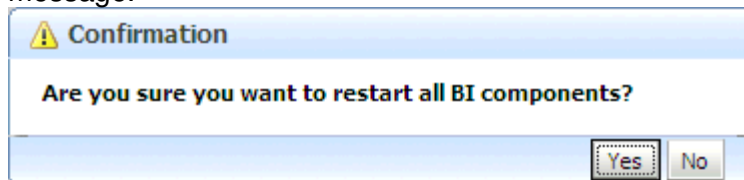
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



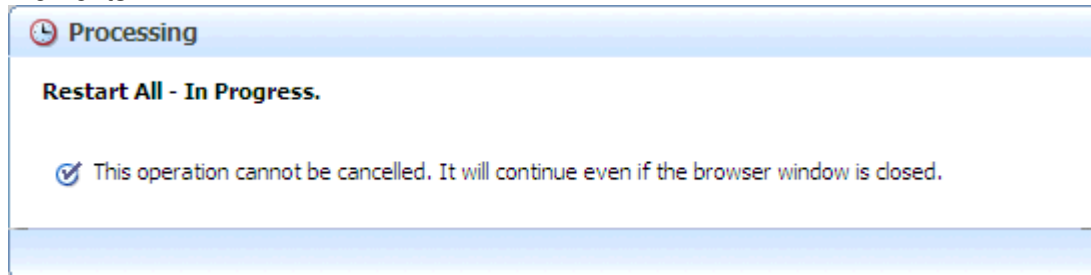
- p. On the Overview page, click **Restart**.



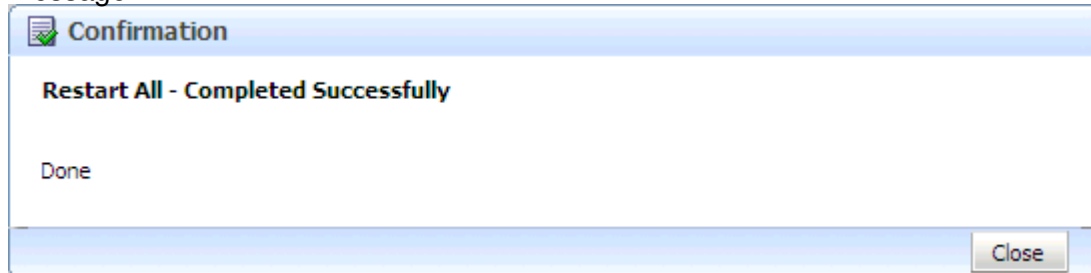
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



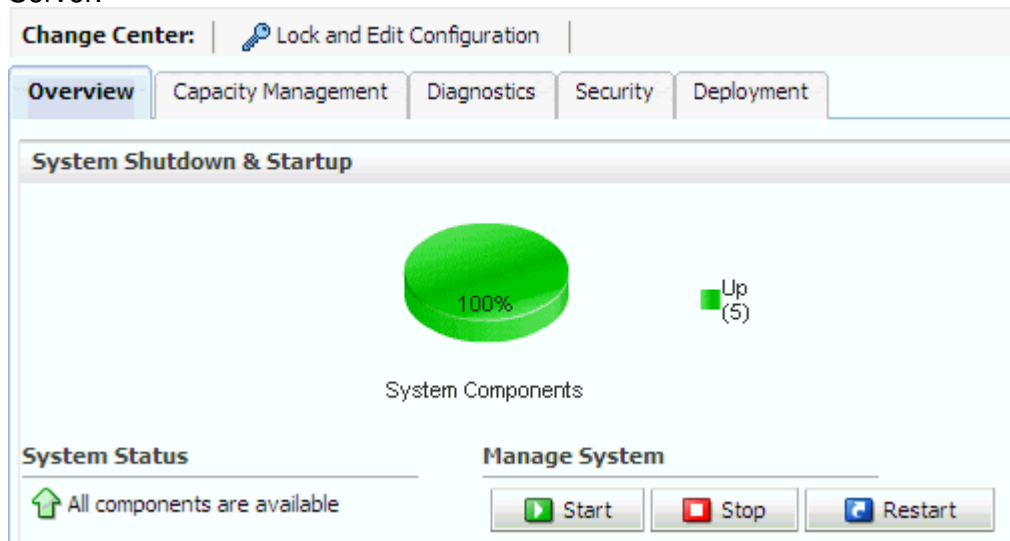
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
18. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.
- Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).
- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Click **SupplierSales** to open Analysis Editor.
19. Create and run analyses to test your work.

- a. Create the following analysis and filter:

Time	Fact-Sales
Month	Dollars

Year is equal to / is in 2008

- b. Click **Results**.

Month	Dollars
January	\$3,595,669
February	\$3,945,187
March	\$3,975,774
April	\$3,907,292
May	\$4,061,558
June	\$3,994,531
July	\$4,062,212
August	\$4,242,611
September	\$3,810,263
October	\$4,596,372
November	\$3,655,169
December	\$3,997,616

- c. Leave Analysis Editor open.
- d. Inspect the query log and confirm that the query uses the Fact_D1_ORDER_AGG1 aggregate fact table and the related D1_MONTHS_AGG aggregate dimension table.

```

----- Sending query to database named orcl (id: <<1801>>),
WITH
SAWITH0 AS (select sum(T600.DOLLARS) as c1,
      T720.MONTHNAME as c2,
      T720.MONTHCODE as c3
from
      MONTHS T720 /* Dim_MONTHS_AGG */ ,
      D1_ORDER_AGG1 T600 /* Fact_D1_ORDER_AGG1 */
where ( T600.PERKEY = T720.MONTHCODE and T720.YEAR = 2008 )
group by T720.MONTHCODE, T720.MONTHNAME)
select distinct 0 as c1,
      D1.c2 as c2,
      D1.c3 as c3,
      D1.c1 as c4
from
      SAWITH0 D1
order by c3

```

- e. Return to Analysis Editor and create a new analysis.

Time	Fact-Sales
Year	Dollars

- f. Click **Results**.

Year	Dollars
2008	\$47,844,253
2009	\$15,288,202

- g. Inspect the query log.

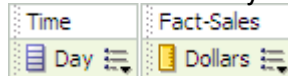
```

----- Sending query to database named orcl (id: <<2123>>),
WITH
SAWITH0 AS (select sum(T600.DOLLARS) as c1,
                T720.YEAR as c2
from
    MONTHS T720 /* Dim_MONTHS_AGG */ ,
    D1_ORDER_AGG1 T600 /* Fact_D1_ORDER_AGG1 */
where ( T600.PERKEY = T720.MONTHCODE )
group by T720.YEAR)
select distinct 0 as c1,
    D1.c2 as c2,
    D1.c1 as c3
from
    SAWITH0 D1
order by c2

```

Notice that the query still uses the same aggregate tables. This is because Year is at a higher level than Month in the Time logical dimension hierarchy, so the aggregate tables are still used.

- h. Create a new analysis.



- i. Click **Results**.

Day	Dollars
20080102	\$26,036
20080103	\$12,210
20080105	\$140,140
20080106	\$85,079
20080107	\$399,278
20080108	\$151,171
20080109	\$107,925
20080110	\$110,789

- j. Inspect the query log.

```

----- Sending query to database named orcl (id: <<2361>>),
WITH
SAWITH0 AS (select sum(T90.DOLLARS) as c1,
                T58.YYYYMMDD as c2
from
    D1_CALENDAR2 T58 /* Dim_D1_CALENDAR2 */ ,
    D1_ORDERS2 T90 /* Fact_D1_ORDERS2 */
where ( T58.YYYYMMDD = T90.PERIODKEY )
group by T58.YYYYMMDD)
select distinct 0 as c1,
    D1.c2 as c2,
    D1.c1 as c3
from
    SAWITH0 D1
order by c2

```

Notice that the detail fact table, Fact_D1_ORDERS2, and the detail dimension table, Dim_D1_CALENDAR2, are accessed instead of the aggregate tables. This is because the requested data is at a lower level than what is contained in the aggregate tables. Therefore, the aggregate tables do not contain the data and the detail tables are used in the query.

- k. Create a new analysis to test product data:



I. Click **Results**.

Type	Dollars
Baking	\$4,925,520.80
Beef	\$4,916,016.20
Beverage	\$4,398,107.48
Bread	\$1,578,743.45
Cereal	\$1,309,071.41
Cheese	\$7,140,616.10
Condiments	\$9,105,121.39
Dessert	\$2,208,426.80
Entre	\$1,807,794.11

- m. Inspect the query log and confirm that the query uses the Fact_D1_ORDER_AGG1 aggregate fact table and the related Dim_D1_PRODUCT_TYPE_AGG aggregate dimension table.

```
----- Sending query to database named orcl (id: <<2706>>),  
WITH  
SAWITH0 AS (select sum(T600.DOLLARS) as c1,  
T732.ITEMTYPE as c2  
from  
D1_PRODUCT_TYPE T732 /* Dim_D1_PRODUCT_TYPE_AGG */ ,  
D1_ORDER_AGG1 T600 /* Fact_D1_ORDER_AGG1 */  
where ( T600.TYPEKEY = T732.TYPECODE )  
group by T732.ITEMTYPE)  
select distinct 0 as c1,  
D1.c2 as c2,  
D1.c1 as c3  
from  
SAWITH0 D1  
order by c2
```

Practice 10-2: Setting the Number of Elements

Goal

To set the number of elements for logical dimension levels

Scenario

In this practice, you set the number of elements for logical dimension levels and observe the results. The number of elements is used by Oracle BI Server when picking aggregate sources. Setting the number of elements is only necessary when there are two or more aggregate sources that could be accessed by an Oracle BI query. Aggregate fact sources are accessed based on a combination of the fields selected as well as the number of elements of the levels in the logical dimensions to which they map. The number does not have to be exact, but ratios of numbers from one logical level to another should be accurate. By adjusting the number of elements, you can alter the aggregate fact source selected by the Oracle BI Server.

Time

30 minutes

Tasks

1. Import an additional aggregate table into the Physical layer of the repository.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the Physical layer, expand **orcl**.
 - c. Right-click the **SUPPLIER CP** connection pool and select **Import Metadata**.
 - d. Accept the defaults in the Select Metadata Types screen and click **Next** to open the Select Metadata Objects screen.
 - e. Scroll to the **SUPPLIER2** schema and expand it.
 - f. In the **Data source view** pane, select the **D1_ORDER_AGG2** table for import.
 - g. Click the **Import selected** button to add the table to the Repository View pane.
 - h. Expand **SUPPLIER2** in the Repository View pane and verify that **D1_ORDER_AGG2** is added.
 - i. Click **Finish** to add the table to the repository.
 - j. Expand **SUPPLIER2** in the Physical Layer and confirm that the **D1_ORDER_AGG2** table is added to the repository.
 - k. Create a **Fact_D1_ORDER_AGG2** alias table for **D1_ORDER_AGG2**.
2. View the data in the aggregate tables.
 - a. View the data for **Fact_D1_ORDER_AGG2**. **Fact_D1_ORDER_AGG2** contains sales facts aggregated to the Product (PRODKEY) and Month (PERKEY) levels. It has 25,373 rows.

DISTKEY	DOLLARS	NETWGHTSHPD	PERKEY	PRODKEY
California	454.79	1505.0	200803.0	1105
California	563.82	2123.0	200804.0	1105
California	550.16	1896.0	200805.0	1105
California	404.96	1371.0	200806.0	1105
California	504.82	1574.38	200807.0	1105
California	411.98	1744.0	200808.0	1105
California	227.76	630.88	200809.0	1105
California	395.52	1466.0	200810.0	1105
California	342.02	1494.0	200811.0	1105
California	377.93	1318.5	200812.0	1105
California	511.49	1744.0	200901.0	1105
California	349.39	1083.06	200902.0	1105

Compare this to Fact_D1_ORDER_AGG1, which contains sales facts aggregated to the Product Type (TYPEKEY) and Month (PERKEY) levels.

DOLLARS	NETWGHTSHPD	PERKEY	SALESREP	SREP_KEY	TYPEKEY
751.22	461.09	200902.0	BARBARA JENSEN	5.0	112
1676.12	1758.03	200903.0	BARBARA JENSEN	5.0	112
5691.81	1100.25	200904.0	BARBARA JENSEN	5.0	112
44.01	53.0	200802.0	BARBARA JENSEN	5.0	113
30.65	52.0	200803.0	BARBARA JENSEN	5.0	113
39.17	43.0	200804.0	BARBARA JENSEN	5.0	113
10.97	21.0	200805.0	BARBARA JENSEN	5.0	113
21.94	42.0	200806.0	BARBARA JENSEN	5.0	113
8.71	10.0	200807.0	BARBARA JENSEN	5.0	113

3. Create physical joins between the aggregate fact and dimension tables.

- a. In the Physical layer, select the following three tables:

Dim_D1_PRODUCTS

Dim_MONTHS_AGG

Fact_D1_ORDER_AGG2

- b. Click the **Physical Diagram** icon on the toolbar.

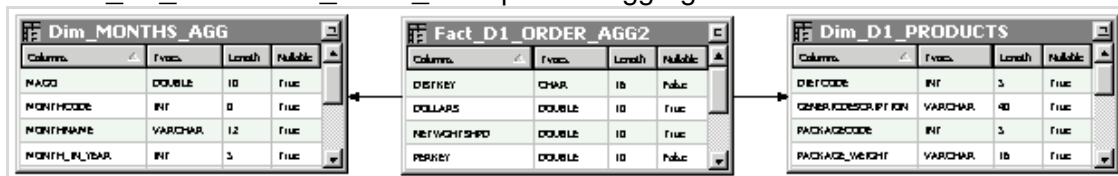
- c. Rearrange the tables to make them visible in the Physical Diagram.

- d. Use the **New foreign key** button and create the following joins:

Dim_MONTHS_AGG.MONTHCODE = Fact_D1_ORDER_AGG2.PERKEY

Dim_D1_PRODUCTS.PRODUCTKEY = Fact_D1_ORDER_AGG2.PRODKEY

- e. Check your work. Notice that Fact_D1_ORDER_AGG2 joins to the Dim_D1_PRODUCTS product detail table, whereas Fact_D1_ORDER_AGG1 joined to the Dim_D1_PRODUCT_TYPE_AGG product aggregate table:



- f. Close the Physical Diagram.

4. Create a new logical source and columns within the existing logical fact table.

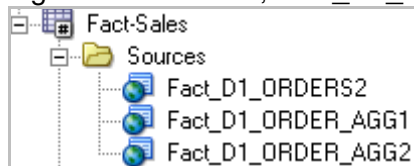
- a. In the Physical layer, expand the **Fact_D1_ORDER_AGG2** table.

- b. In the Business Model and Mapping layer, expand **Fact-Sales**.

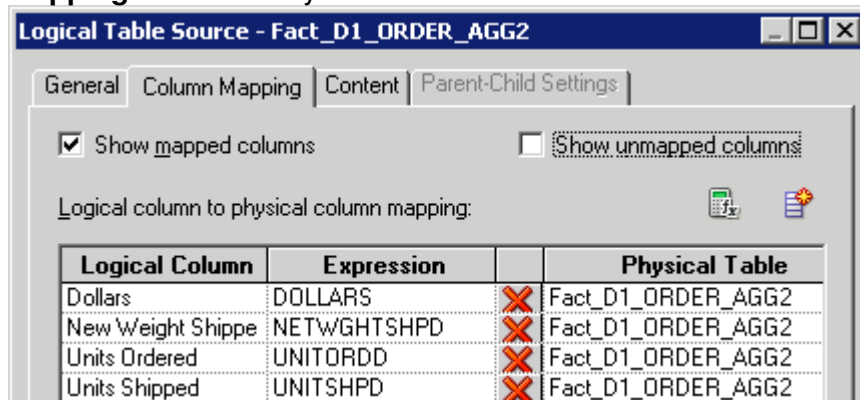
- c. Drag the **DOLLARS**, **NETWGHTSHPD**, **UNITSHPD**, and **UNITORDD** columns one at a time from Fact_D1_ORDER_AGG2 and drop each onto their corresponding Fact-Sales logical columns: **Dollars**, **Net Weight Shipped**, **Units Shipped**, and **Units**

Ordered. This creates a new Fact_D1_ORDER_AGG2 logical table source and corresponding column mappings.

- d. Expand **Fact-Sales > Sources**. Notice that there are now two aggregate logical table sources, Fact_D1_ORDER_AGG1 and Fact_D1_ORDER_AGG2, and one detail logical table source, Fact_D1_ORDERS2.



- e. Double-click the **Fact_D1_ORDER_AGG2** logical table source and click the **Column Mapping** tab to check your work.



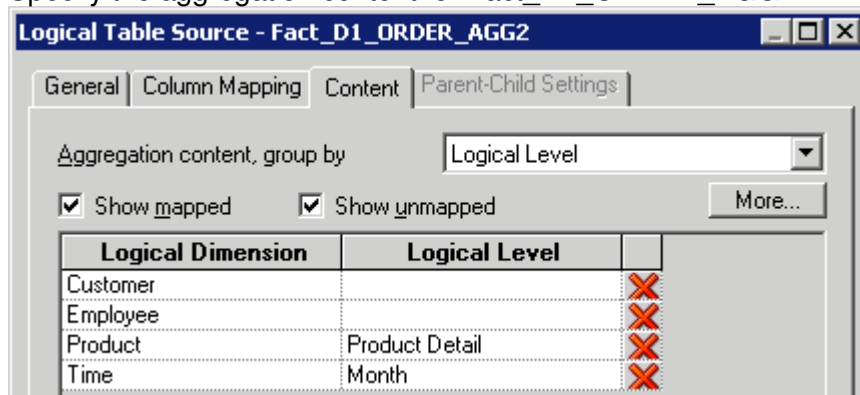
Notice that these four logical columns, Dollars, Units Shipped, Units Ordered, and Net Weight Shipped, now map to their corresponding columns in three tables:

Fact_D1_ORDERS2

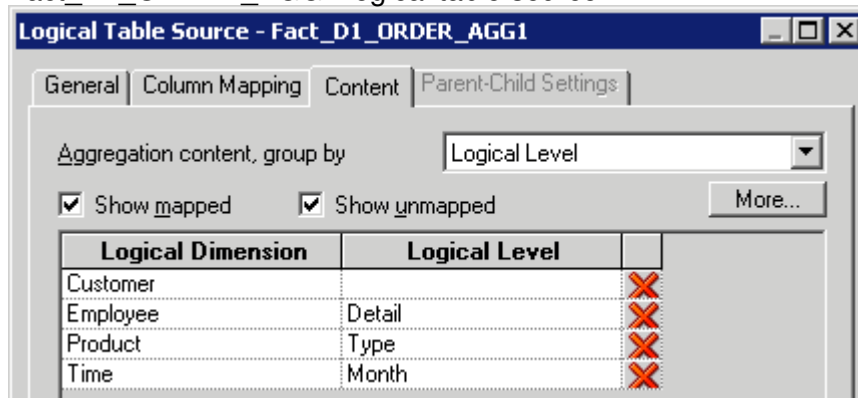
Fact_D1_ORDER_AGG1

Fact_D1_ORDER_AGG2

5. Specify the aggregation content of the new logical table source so that Oracle BI Server knows what level of data is stored in the aggregate tables.
- Click the **Content** tab.
 - Specify the aggregation content for Fact_D1_ORDER_AGG2.



- c. Compare the aggregation content for Fact_D1_ORDER_AGG2 with that of the Fact_D1_ORDER_AGG1 logical table source.



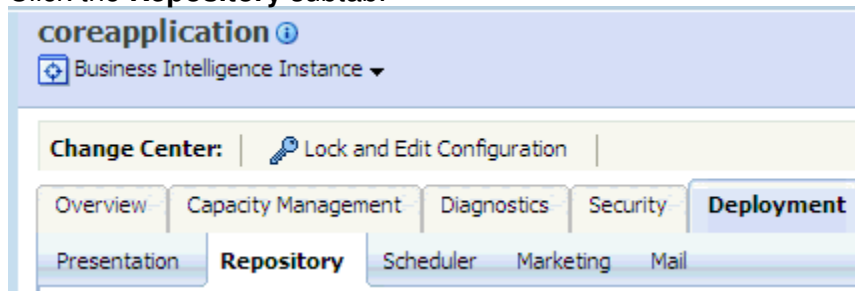
Notice that both aggregate logical table sources have aggregation content defined for the same logical dimensions, Product and Time, but the levels are different for the Product logical dimension.

- d. Click **OK** to close the Logical Table Source dialog box.
6. In this step you set the number of elements in the logical dimensions. Aggregate fact sources are selected based on a combination of the fields selected as well as the number of elements of the levels in the logical dimensions to which they map. This number is used by the Oracle BI Server when picking aggregate sources. Setting the number of elements is only necessary when there are two or more aggregate sources that could be accessed by an Oracle BI query. The number does not have to be exact, but ratios of numbers from one logical level to another should be accurate.
 - a. Expand the **Product** logical dimension.
 - b. Double-click the **Product Total** level and select the **General** tab. Because this is the grand total level, the number of elements is automatically set to one.
 - c. Click **Cancel**.
 - d. Double-click the **Type** level and select the **General** tab.
 - e. Set the number of elements at this level to **22**. One way to determine the number of elements at each level is by updating row counts and then observing the row count number for the corresponding column in the Physical layer.
 - f. Continue to set the number of elements for the levels in the **Product** logical dimension:

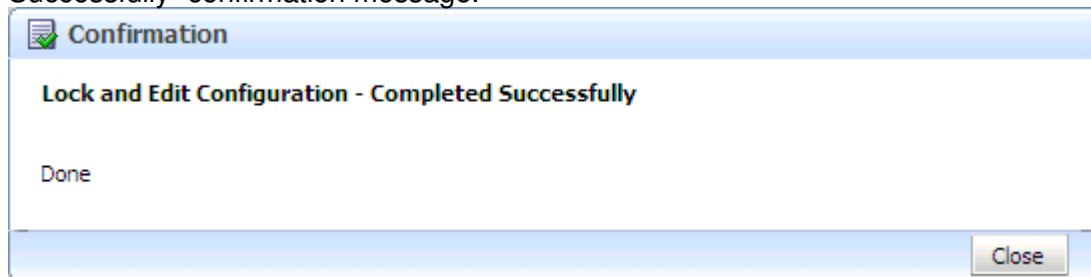
SubType: 159
Generic: 186
Product Detail: 192
 - g. Set the number of elements for the levels in the **Time** logical dimension:

Year: 2
Quarter: 6
Month: 16
Time Detail: 474
 - h. Save the repository
 - i. Check consistency. Fix any errors or warnings before proceeding.
 - j. Close the repository. Leave the Admin Tool open.
 7. Use Fusion Middleware Control to upload the repository.
 - a. Return to Fusion Middleware Control, which should still be open in your browser. If it is not open, enter the following URL:
<http://localhost:7001/em>.

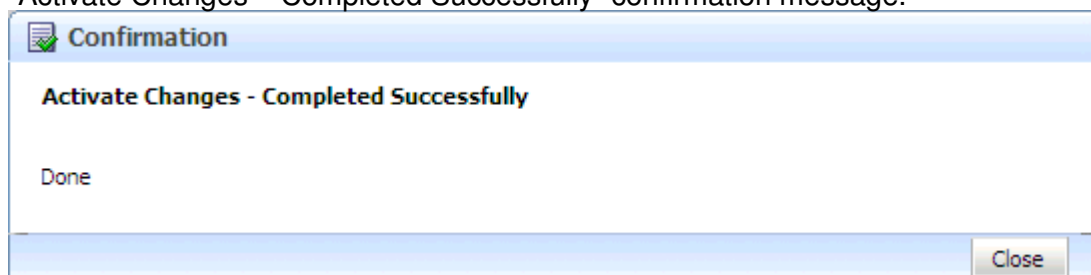
- b. If your session has timed out, log in as **weblogic/welcome1**.
- c. In the left pane, expand **Business Intelligence** and select **coreapplication**.
- d. In the right pane, click the **Deployment** tab.
- e. Click the **Repository** subtab.



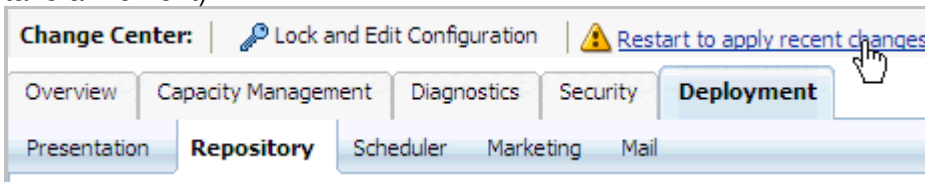
- f. Click **Lock and Edit Configuration**.
- g. Click **Close** when you receive the “Lock and Edit configuration – Completed Successfully” confirmation message.



- h. In the Upload BI Server Repository section, click **Browse** to open the “Choose file” dialog box.
- i. By default, the “Choose file” dialog box should open to the default repository directory. If not, browse to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplication_obis1\repository**.
- j. Select **ABC.rpd** and click **Open**. You can also double-click the repository to open it.
- k. Enter **welcome1** in the Repository Password and Confirm Password fields.
- l. Click **Apply**. Notice that Default RPD now displays **ABC** with an extension (for example, ABC_BI0007).
- m. Click **Activate Changes**.
- n. Allow Active Changes processing to complete. Click **Close** when you receive the “Activate Changes – Completed Successfully” confirmation message.



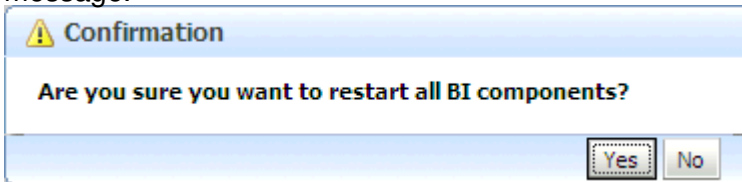
- o. Click **Restart to apply recent changes** to navigate to the Overview page (this may take a moment).



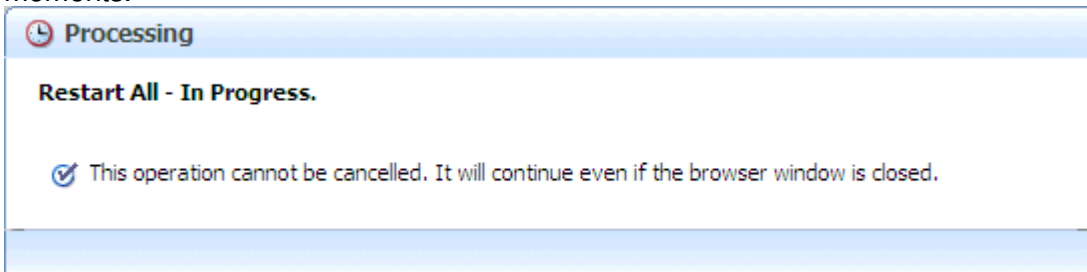
- p. On the Overview page, click **Restart**.



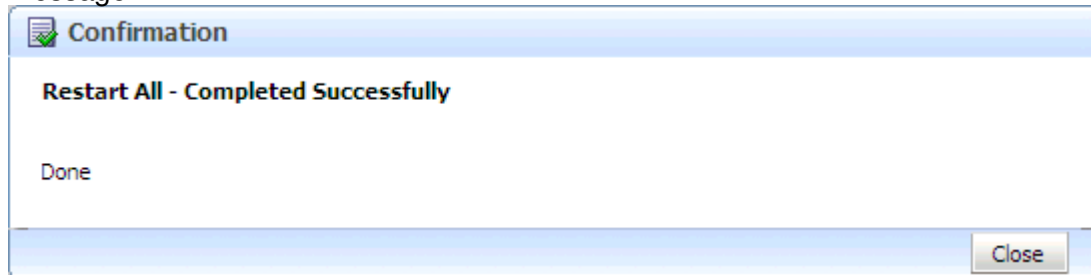
- q. Click **Yes** when you receive the “Are you sure you want to restart all BI components?” message.



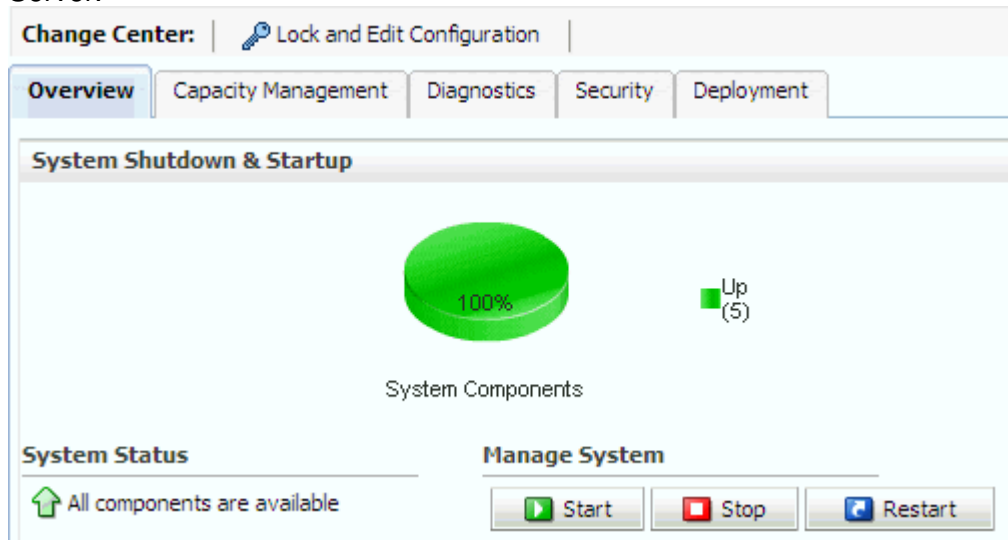
- r. Allow the Restart All – In Progress processing to complete. This may take a few moments.



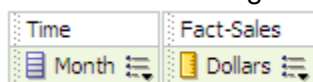
- s. Click **Close** when you receive the “Restart All – Completed Successfully” confirmation message.



- t. Confirm that System Components = 100%. The ABC repository is loaded into BI Server.



- u. Leave Fusion Middleware Control open.
8. Open Analysis Editor to execute queries and test the SupplierSales business model.
- Return to the Oracle Business Intelligence browser tab where you signed out and click **here** to sign in.
- Thank you for using Oracle Business Intelligence software. You have successfully **signed out**.
To sign in again, click [here](#).
- Sign in as **weblogic** with password **welcome1**.
 - In the Create section, click **Analysis** to open the Select Subject Area window.
 - Click **SupplierSales** to open Analysis Editor.
9. Create and run analyses to check your work.
- Create the following analysis and filter.



Year is equal to / is in 2008

b. Click **Results**

Month	Dollars
January	\$3,595,669
February	\$3,945,187
March	\$3,975,774
April	\$3,907,292
May	\$4,061,558
June	\$3,994,531
July	\$4,062,212
August	\$4,242,611
September	\$3,810,263
October	\$4,596,372
November	\$3,655,169
December	\$3,997,616

c. Inspect the query log.

```
----- Sending query to database named orcl (id: <<1867>>),  
WITH  
SAWITH0 AS (select sum(T600.DOLLARS) as c1,  
    T720.MONTHNAME as c2,  
    T720.MONTHCODE as c3  
from  
    MONTHS T720 /* Dim_MONTHS_AGG */ ,  
    D1_ORDER_AGG1 T600 /* Fact_D1_ORDER_AGG1 */  
where ( T600.PERKEY = T720.MONTHCODE and T720.YEAR = 2008 )  
group by T720.MONTHCODE, T720.MONTHNAME)  
select distinct 0 as c1,  
    D1.c2 as c2,  
    D1.c3 as c3,  
    D1.c1 as c4  
from  
    SAWITH0 D1  
order by c3
```

Notice that despite the fact that both aggregate logical table sources have their aggregation content defined for the same logical dimensions, Product and Time, the query uses Fact_D1_ORDER_AGG1. This is because Oracle BI Server determined it was more economical to access Fact_D1_ORDER_AGG1 instead of Fact_D1_ORDER_AGG2 based on the levels. For example, to access Fact_D1_ORDER_AGG2, it calculates 3072 potential rows: 16 months * 192 products. To access Fact_D1_ORDER_AGG1, it calculates 352 potential rows: 16 months * 22 product types. Therefore, Fact_D1_ORDER_AGG1 is the more economical aggregate source because its aggregation content is defined at the Type level for the Product logical dimension.

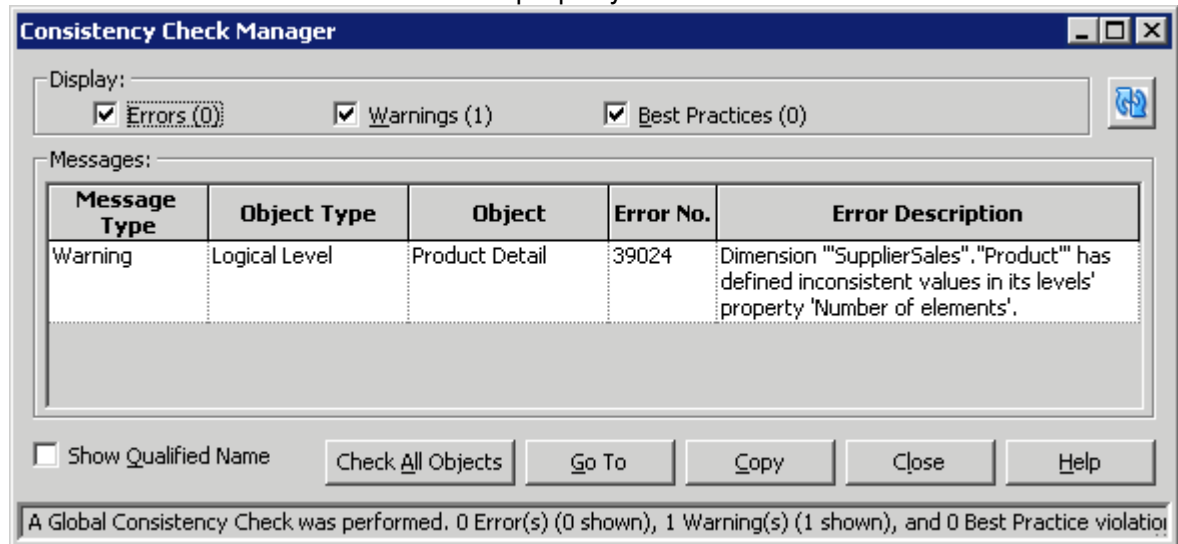
d. Sign out of Analysis Editor.

10. Adjust the number of elements to alter the aggregate fact source selected by Oracle BI Server.

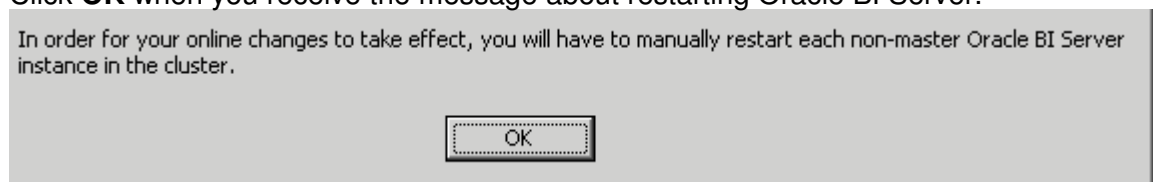
- Return to the Administration Tool and open the repository in **online** mode.
- Enter **welcome1** as the repository password and user password and click **Open**.
- In the Business Model and Mapping layer, expand the **Product** logical dimension.
- Double-click the **Type** level.
- Click **Check Out**.
- Click the **General** tab and change the number of elements at this level from **22** to **220**.
- Click **OK** to close the Logical Level dialog box. Now if you run the same analysis (Month, Dollars), changing the number of elements at this level increases the potential

rows for Fact_D1_ORDER_AGG1 from 352 to 3520 (16 months * 220 product types), which is now higher than the potential rows for Fact_D1_ORDER_AGG2, which is 3072. Thus the same query will access Fact_D1_ORDER_AGG2 instead of Fact_D1_ORDER_AGG1. Please notice that you are doing this only to demonstrate a teaching point. This is not a correct modeling technique.

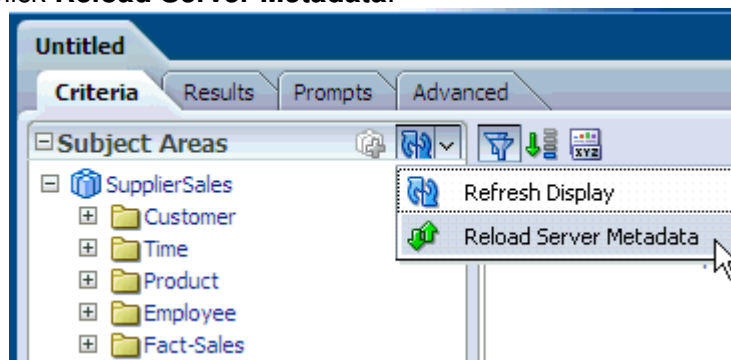
- h. Check in the changes.
- i. Check consistency. For the purpose of this teaching point you can ignore the warning about inconsistent values in the levels' property and click **Close**.



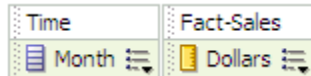
- j. Save the repository.
- k. Close the repository.
- l. Click **OK** when you receive the message about restarting Oracle BI Server.



- m. Leave the Administration Tool open.
11. Create an analysis to check your work.
- a. Return to Oracle BI and sign in as **weblogic/welcome1**.
 - b. Click **Analysis > SupplierSales**.
 - c. Click **Reload Server Metadata**.



- d. Create the same analysis and filter as earlier in this practice:



Year is equal to / is in 2008

- e. Click **Results**.

Month	Dollars
January	\$3,595,669
February	\$3,945,187
March	\$3,975,774
April	\$3,907,292
May	\$4,061,558
June	\$3,994,531
July	\$4,062,212
August	\$4,242,611
September	\$3,810,263
October	\$4,596,372
November	\$3,655,169
December	\$3,997,616

- f. Inspect the query log.

```

----- Sending query to database named orcl (id: <<6924>>),
WITH
SAWITH0 AS (select sum(T762.DOLLARS) as c1,
                T720.MONTHNAME as c2,
                T720.MONTHCODE as c3
            from
                MONTHS T720 /* Dim_MONTHS_AGG */ ,
                D1_ORDER_AGG2 T762 /* Fact_D1_ORDER_AGG2 */
            where ( T720.MONTHCODE = T762.PERKEY and T720.YEAR = 2008 )
            group by T720.MONTHCODE, T720.MONTHNAME)
select distinct 0 as c1,
                D1.c2 as c2,
                D1.c3 as c3,
                D1.c1 as c4
            from
                SAWITH0 D1
            order by c3

```

Notice that the query now uses Fact_D1_ORDER_AGG2 instead of Fact_D1_ORDER_AGG1. This is because Oracle BI Server determined it was more economical to access Fact_D1_ORDER_AGG2 instead of Fact_D1_ORDER_AGG1 based on the levels. To access Fact_D1_ORDER_AGG2, it calculates 3072 potential rows: 16 months * 192 products. To access Fact_D1_ORDER_AGG1, it calculates 3520 potential rows: 16 months * 220 product types. Therefore, Fact_D1_ORDER_AGG2 is now the more economical aggregate source. The important point is that setting the number of elements in the logical dimension hierarchies helps Oracle BI Server determine the most economical source to access when there are multiple aggregate sources.

- g. Sign out of Analysis Editor.

Practices for Lesson 11: Using Partitions and Fragments

Chapter 11

Overview of Practices for Lesson 11

Lesson Overview

In these practices, you will model partitions in an Oracle BI repository.

Practice 11-1: Modeling a Value-Based Partition

Goal

To model a value-based partition

Scenario

ABC wants to store its fact data in two separate partitions, one for recent data and one for historical data. Each partition contains the same columns. Only the data values are different. The historical data partition stores invoice fact data up to and including December 31, 2008. The recent data partition stores invoice fact data after December 31, 2008.

Outcome

In the Physical layer, there are two new physical sources: D1_ORDERS_RECENT and the D1_ORDERS_HISTORICAL. In the Business Model and Mapping layer, logical table sources are modified for the Fact-Sales logical table.

Time

30 minutes

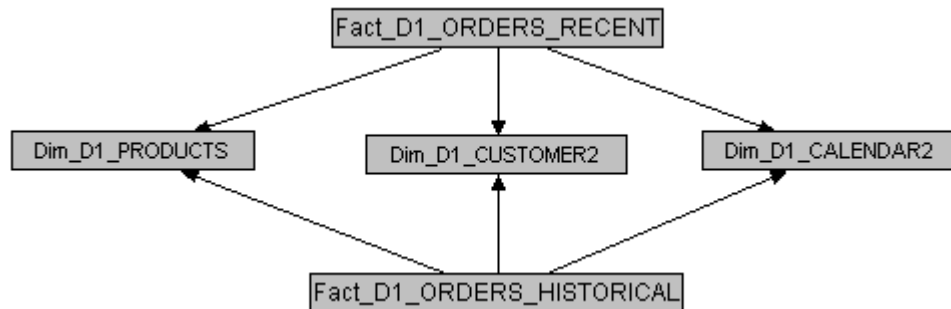
Tasks

1. Import two value-based partitioned sources with fact data into the Physical layer.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the Physical layer, expand **orcl**.
 - c. Right-click the **SUPPLIER CP** connection pool and select **Import Metadata**.
 - d. Accept the defaults in the Select Metadata Types window and click **Next** to open the Select Metadata Objects window.
 - e. Expand the **SUPPLIER2** schema.
 - f. In the Data source view pane, select the **D1_ORDERS_RECENT** and the **D1_ORDERS_HISTORICAL** tables for import.
 - g. Click the **Import selected** button to add the tables to the Repository View pane.
 - h. Expand **SUPPLIER2** in the Repository View pane and confirm that the D1_ORDERS_RECENT and the D1_ORDERS_HISTORICAL tables are added.
 - i. Click **Finish** to add the tables to the repository.
 - j. Expand **SUPPLIER2** in the Physical Layer and confirm that the D1_ORDERS_RECENT and the D1_ORDERS_HISTORICAL tables are added to the repository. D1_ORDERS_HISTORICAL has fact data for all of 2008. D1_ORDERS_RECENT has fact data for the first four months of 2009.
 - k. Update row counts for the two new tables to ensure connectivity.
D1_ORDERS_RECENT: 80223 rows
D1_ORDERS_HISTORICAL: 271413 rows
 - l. Double-click **PERIODKEY** in D1_ORDERS_RECENT to open the Physical column dialog box.
 - m. Change the Type to **INT**.
 - n. Repeat for **PERIODKEY** in D1_ORDERS_HISTORICAL.

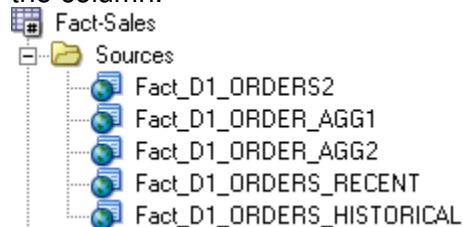
- o. Create the following aliases:
Fact_D1_ORDERS_RECENT
Fact_D1_ORDERS_HISTORICAL
- p. Use the Physical Diagram to create the following physical joins:

```
Dim_D1_CALENDAR2.YYYYMMDD = Fact_D1_ORDERS_RECENT.PERIODKEY
Dim_D1_PRODUCTS.PRODUCTKEY = Fact_D1_ORDERS_RECENT.PRODKEY
Dim_D1_CUSTOMER2.NEWKEY = Fact_D1_ORDERS_RECENT.CUSTKEY
```

```
Dim_D1_CALENDAR2.YYYYMMDD = Fact_D1_ORDERS_HISTORICAL.PERIODKEY
Dim_D1_PRODUCTS.PRODUCTKEY = Fact_D1_ORDERS_HISTORICAL.PRODKEY
Dim_D1_CUSTOMER2.NEWKEY = Fact_D1_ORDERS_HISTORICAL.CUSTKEY
```

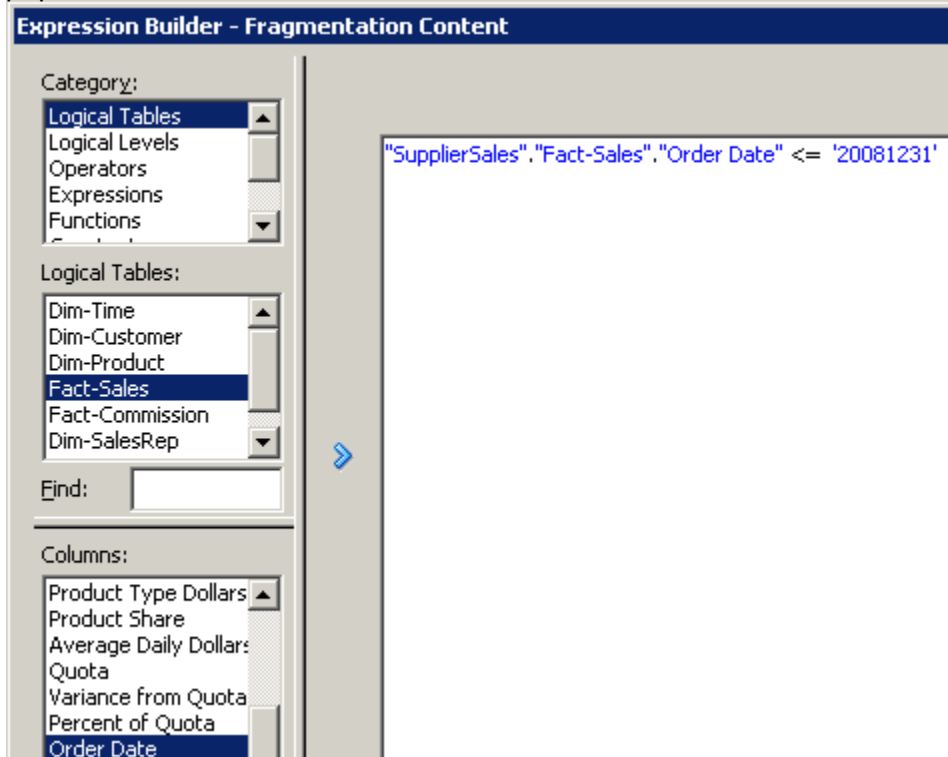


2. Create two new logical table sources for the Fact-Sales logical table.
 - a. Expand **SupplierSales > Fact-Sales > Sources** in the Business Model and Mapping layer.
 - b. Expand **Fact_D1_ORDERS_RECENT** in the Physical layer.
 - c. Drag the **DOLLARS** column from Fact_D1_ORDERS_RECENT to the **Dollars** column in Fact-Sales. This creates a new logical table source named Fact_D1_ORDERS_RECENT and maps the column.
 - d. Repeat the steps for the **DOLLARS** column in **Fact_D1_ORDERS_HISTORICAL**. This creates a new logical table source named Fact_D1_ORDERS_HISTORICAL and maps the column.

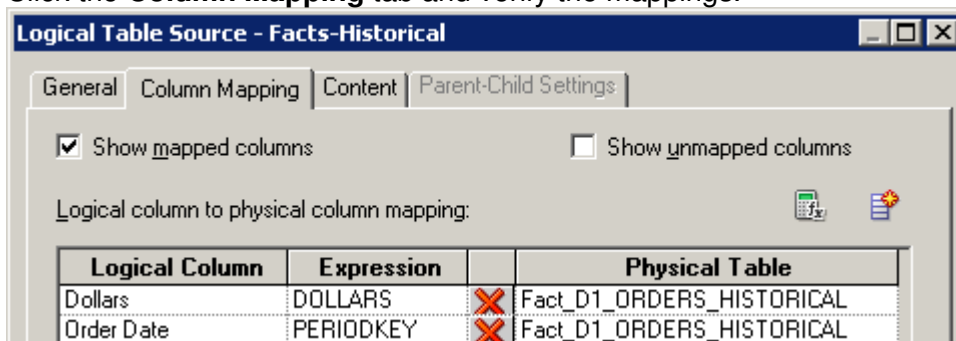


3. Create a new logical column.
 - a. Right-click **Fact-Sales** and select **New Object > Logical Column**.
 - b. Name the column **Order Date**.
 - c. Drag the **PERIODKEY** column from **Fact_D1_ORDERS_RECENT** to the **Order Date** column in **Fact-Sales**.
 - d. Drag the **PERIODKEY** column from **Fact_D1_ORDERS_HISTORICAL** to the **Order Date** column in **Fact-Sales**.
 - e. Drag **Order Date** to the **Fact-Sales** presentation table.
4. Create fragmentation content for the two new logical table sources.

- a. Double-click **Fact_D1_ORDERS_HISTORICAL** in the Sources folder and click the **General** tab.
- b. Rename the logical table source to **Facts-Historical**.
- c. Click the **Content** tab.
- d. Click the button next to the Fragmentation content field to open Expression Builder.
- e. Select **Logical Tables > Fact-Sales**, and then double-click the **Order Date** column. The expression is added to the expression builder.
- f. Click "**<=**" on the toolbar to add it to the formula and then add '**20081231**' to the formula. You are hard-coding this formula so that this logical table source will be used for queries where Order Date (PERIODKEY) is equal to or less than December 31, 2008 (20081231). In the next lesson you learn how to use variables to dynamically populate formulas.

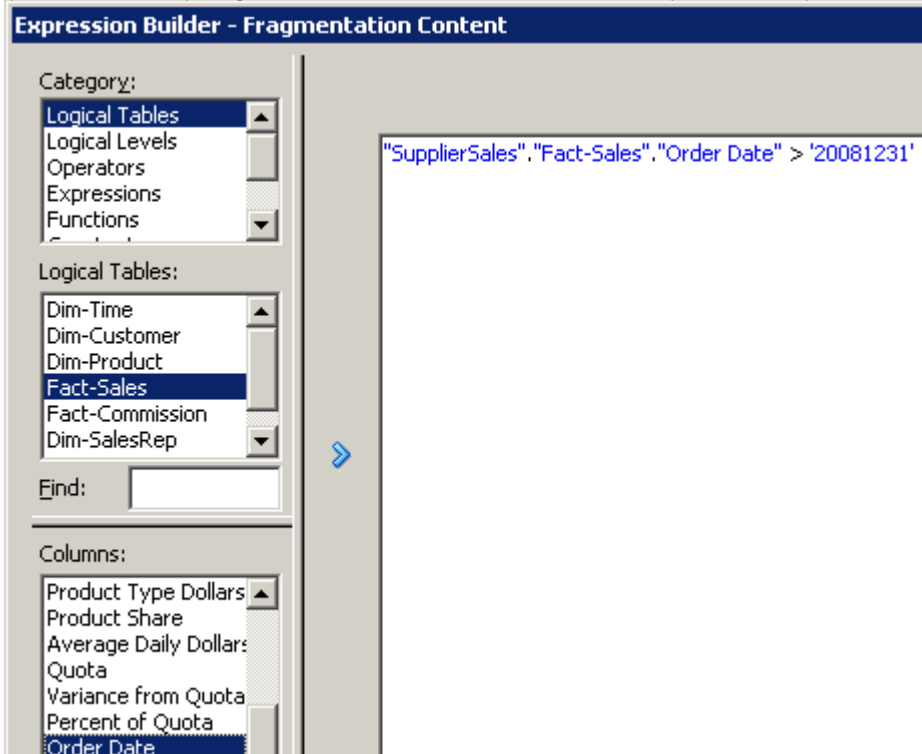


- g. Click **OK** to close the Expression Builder. Notice that the expression is added to the Fragmentation content field.
- h. Select **This source should be combined with others at this level**.
- i. Click the **Column Mapping** tab and verify the mappings.

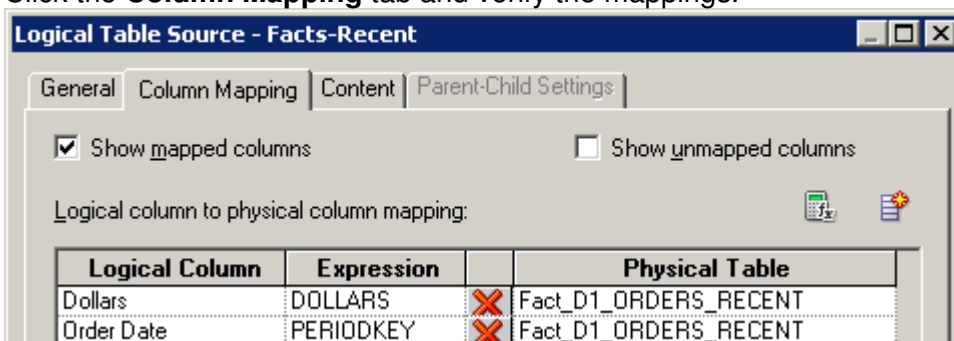


- j. Click **OK** to close the Logical Table Source dialog box.

- k. Double-click **Fact_D1_ORDERS_RECENT** in the Sources folder and click the **General** tab.
- l. Rename the logical table source to **Facts-Recent**.
- m. Click the **Content** tab.
- n. Open the Expression Builder and create the following expression:
SupplierSales."Fact-Sales**".**"Order Date**" > '20081231'**. You are hard-coding this formula so that this logical table source will be used for queries where Order Date (PERIODKEY) is greater than December 31, 2008 (20081231).

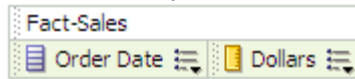


- o. Click **OK**.
- p. Select **This source should be combined with others at this level**.
- q. Click the **Column Mapping** tab and verify the mappings.



- r. Click **OK**.
- s. Notice that, other than adding the Order Date column to the Fact-Sales presentation table, you did not have to change the Presentation layer. The Order Date column is added for testing purposes.
- t. Save the repository.
- u. Check consistency. Fix any errors or warnings before you proceed.

- v. Close the repository.
5. Create and run analyses to test your work.
 - a. Use Fusion Middleware Control to upload the ABC repository and restart Oracle BI components. If you need help, refer to steps from earlier practices.
 - b. Return to Analysis Editor.
 - c. Create an analysis.



- d. Click **Results**.

Order Date	Dollars
20080102	\$26,036
20080103	\$12,210
20080105	\$140,140
20080106	\$85,079
20080107	\$399,278
20080108	\$151,171
20080109	\$107,925
20080110	\$110,789

Notice that Order Date records are returned for both before and after 20081231. Click the **Display Maximum** button at the bottom of the table to confirm.

- e. Leave Analysis Editor open.
- f. Inspect the query log. Confirm that the query references both partition tables. Both partition tables are used because you did not apply a filter in the query. Notice that a separate select statement is sent to each table and Oracle BI Server performs a UNION ALL to combine the results.

```

----- Sending query to database named orcl (id: <<1680>>), connection pool named SUPPLIER CP:
WITH
SAWITH0 AS ((select T801.PERIODKEY as c2,
  T801.DOLLARS as c3
from
  D1_ORDERS_RECENT T801 /* Fact_D1_ORDERS_RECENT */
union all
select T792.PERIODKEY as c2,
  T792.DOLLARS as c3
from
  D1_ORDERS_HISTORICAL T792 /* Fact_D1_ORDERS_HISTORICAL */)),
SAWITH1 AS (select sum(D3.c3) as c1,
  D3.c2 as c2
from
  SAWITH0 D3
group by D3.c2)
select distinct 0 as c1,
  D2.c2 as c2,
  D2.c1 as c3
from
  SAWITH1 D2
order by c2

```

Pa

Partitioning can create complexity in the query environment. Queries must access multiple tables and then consolidate the results. One of the important benefits of Oracle BI Server is that it can do this type of navigation and consolidation automatically and invisibly to the end user.

- g. Return to Analysis Editor and add a filter where **Order Date** is less than or equal to **20081231**.

New Filter

Column: Order Date

Operator: is less than or equal to

Value: 20081231

☐ Protect Filter

☐ Convert this filter to SQL

Buttons: Add More Options, Clear All, Help, OK, Cancel

- h. Run the analysis again.
- i. Verify that your results display only Order Dates that are less than or equal to 20081231.

Order Date	Dollars
20080102	\$26,036
20080103	\$12,210
20080105	\$140,140
20080106	\$85,079
20080107	\$399,278
20080108	\$151,171
20080109	\$107,925
20080110	\$110,789

- j. Leave Analysis Editor open.
- k. Inspect the query log and confirm that the query accessed only one partition, **Fact_D1_ORDERS_HISTORICAL**.

```

----- Sending query to database named orcl (id: <<2404>>),
WITH
SAWITH0 AS (select sum(T792.DOLLARS) as c1,
                T792.PERIODKEY as c2
from
    D1_ORDERS_HISTORICAL T792 /* Fact_D1_ORDERS_HISTORICAL */
where ( T792.PERIODKEY <= 20081231 )
group by T792.PERIODKEY)
select distinct 0 as c1,
    D1.c2 as c2,
    D1.c1 as c3
from
    SAWITH0 D1
order by c2

```

6. Test that your results are correct when Order Date is greater than 20081231.

- a. Return to the analysis and edit the filter to show records where Order Date is greater than 20081231.

Column: Order Date
Operator: is greater than
Value: 20081231
Buttons: Add More Options, Clear All, Help, OK, Cancel
Options: ☐ Protect Filter, ☐ Convert this filter to SQL

- b. Run the analysis.
c. Verify that your result only displays records where Order Date is greater than 20081231.

Order Date	Dollars
20090102	\$143,693
20090104	\$324,001
20090105	\$172,597
20090106	\$112,132
20090107	\$170,791
20090108	\$171,649
20090109	\$119,321
20090111	\$302,795
20090112	\$136,086

- d. Inspect the query log and confirm that the query being issued accesses the correct partition, Fact_D1_ORDERS_RECENT.

```
----- Sending query to database named orcl (id: <<2650>>),  
WITH  
SAWITH0 AS (select sum(T801.DOLLARS) as c1,  
                T801.PERIODKEY as c2  
from  
    D1_ORDERS_RECENT T801 /* Fact_D1_ORDERS_RECENT */  
where ( 20081231 < T801.PERIODKEY )  
group by T801.PERIODKEY)  
select distinct 0 as c1,  
    D1.c2 as c2,  
    D1.c1 as c3  
from  
    SAWITH0 D1  
order by c2
```

Practices for Lesson 12: Using Repository Variables

Chapter 12

Overview of Practices for Lesson 12

Lesson Overview

In these practices, you will create and use dynamic repository variables with initialization blocks.

Practice 12-1: Creating Dynamic Repository Variables

Goal

To create and use dynamic repository variables with initialization blocks

Scenario

Dynamic repository variables are useful for defining the content of logical table sources. ABC has two sources for information about orders: one source contains recent orders and the other source contains historical data. You must describe the content of these sources on the Content tab of the Logical Table Source dialog box. Without using dynamic repository variables, you would have to describe the content of the source containing recent data with a static expression such as: `SupplierSales."Fact-Sales"."Order Date" > '20081231'`, as you did in the previous practice. This content statement becomes invalid as new data is added to the recent source and older data is moved to the historical source. To accurately reflect any new data, you would have to modify the fragmentation content description manually. You can set up dynamic repository values to do this automatically. To use a dynamic repository variable, you must also set up an initialization block to refresh the variable on a continuing basis.

Outcome

A new initialization block called OrderSplit is created with a variable named EndHistoricalData. In the Business Model and Mapping layer, logical tables' sources are modified for the Fact-Sales logical table.

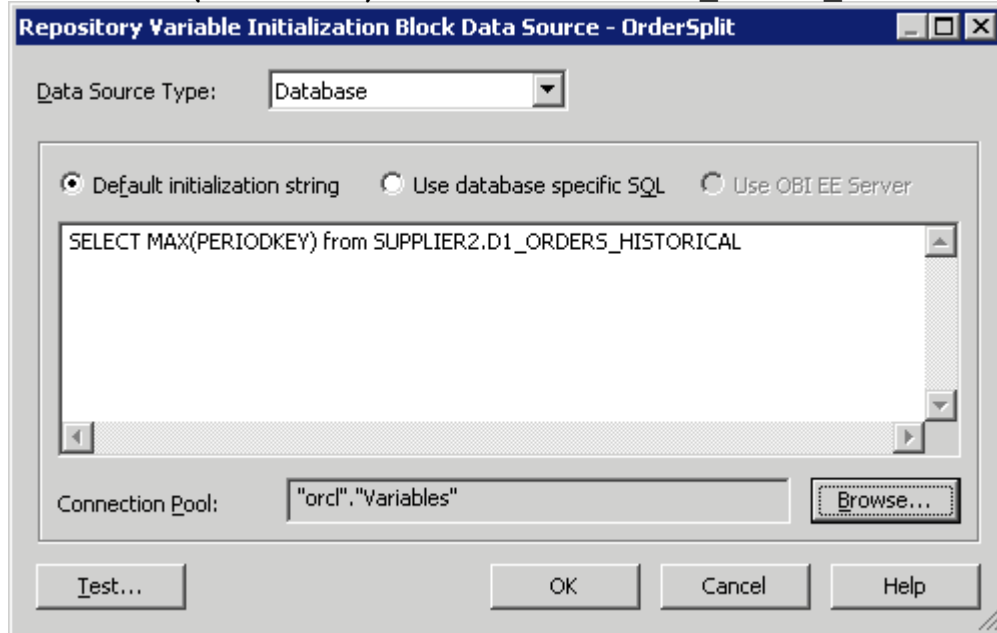
Time

25 minutes

Tasks

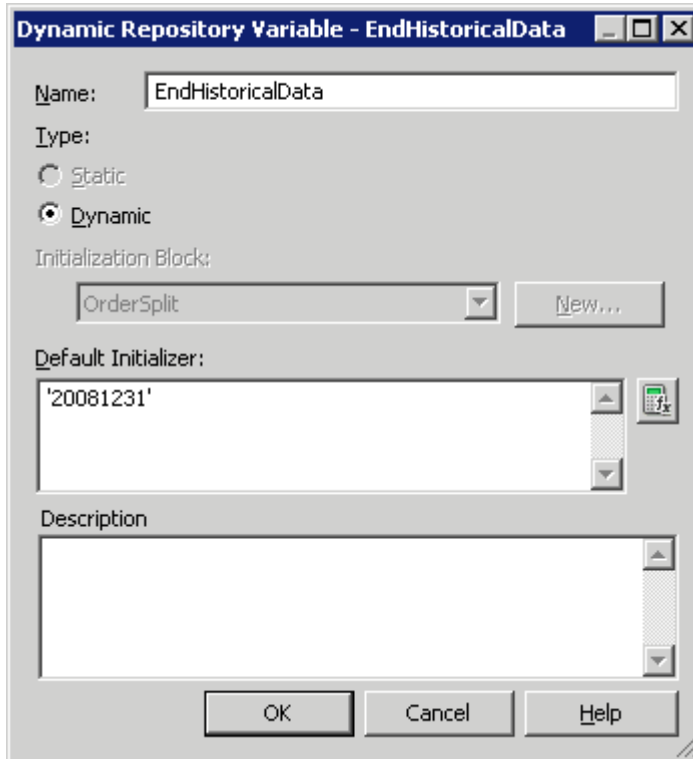
1. In this step, you create a dedicated connection pool for the variable initialization block. It is recommended that you create a dedicated connection pool for initialization blocks. This connection pool should not be used for queries.
 - a. In the Physical layer, right-click the **orcl** database object and select **New Object > Connection Pool** to open the Connection Pool dialog box.
 - b. On the General tab, name the connection pool **Variables**.
 - c. Enter **orcl** as the data source name.
 - d. Enter **supplier2** as the user name and password.
 - e. Click **OK** to close the Connection Pool dialog box.
 - f. Enter **supplier2** to confirm the password and click **OK**.
2. In this step, you create an initialization block. Initialization blocks contain SQL statements that initialize variables. The variables are repository variables and their values persist until the initialization block resets them the next time it runs. Repository initialization blocks execute whenever Oracle BI Server starts up, and thereafter according to any schedule that is applied.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. Select **Manage > Variables**.
 - c. Select **Action > New > Repository> Initialization Block**.
 - d. Name the initialization block **OrderSplit**.

- e. Leave the schedule set to the default.
- f. Click **Edit Data Source**.
- g. Click the **Browse** button.
- h. Double-click the **Variables** connection pool object to select it. The connection pool is added to the Repository Variable Init Block Data Source dialog box.
- i. Enter the following SQL in the Default Initialization String field to capture the most recent period key (for example 20081231) from the D1_ORDERS_HISTORICAL table:
SELECT MAX(PERIODKEY) from SUPPLIER2.D1_ORDERS_HISTORICAL



- j. Click **OK** to close the Repository Variable Initialization Block Data Source dialog box.
3. Create the variable.
- k. Click **Edit Data Target**.
 - l. Click the **New** button.
 - m. In the Name field, enter **EndHistoricalData**.

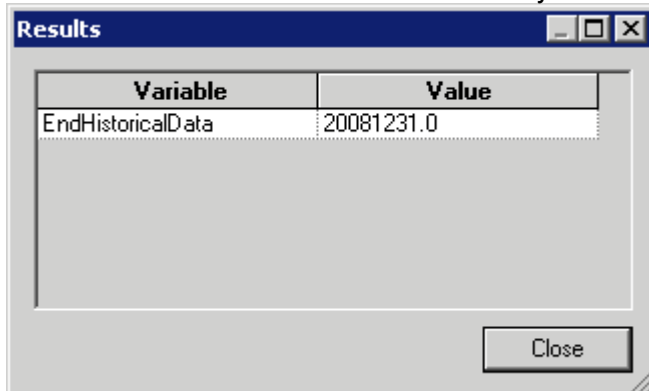
- n. Enter **'20081231'** in the Default Initializer field.



The dialog box is titled "Dynamic Repository Variable - EndHistoricalData". It contains the following fields and controls:

- Name:** A text field containing "EndHistoricalData".
- Type:** Two radio buttons: "Static" (unselected) and "Dynamic" (selected).
- Initialization Block:** A dropdown menu showing "OrderSplit" and a "New..." button.
- Default Initializer:** A text field containing "'20081231'" with a small icon to its right.
- Description:** A large empty text area.
- Buttons:** "OK", "Cancel", and "Help" at the bottom.

- o. Click **OK** to close the Dynamic Repository Variable dialog box.
- p. Click **OK** to close the Repository Variable Initialization Block Variable Target dialog box.
- q. Click **Test** to test the variable and check your results:



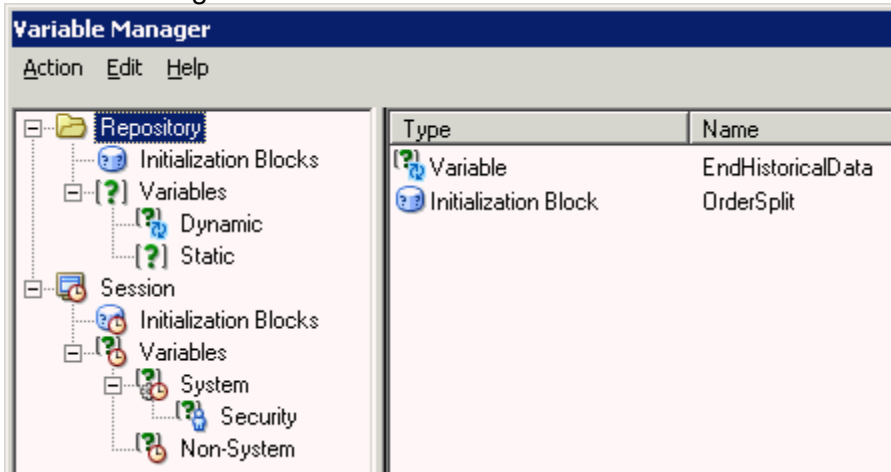
The dialog box is titled "Results". It contains a table with the following data:

Variable	Value
EndHistoricalData	20081231.0

There is a "Close" button at the bottom right of the dialog box.

- r. Click **Close**.
- s. Click **OK** to close the Repository Variable Initialization Block dialog box. The OrderSplit initialization block and EndHistoricalData dynamic repository variable are visible in the

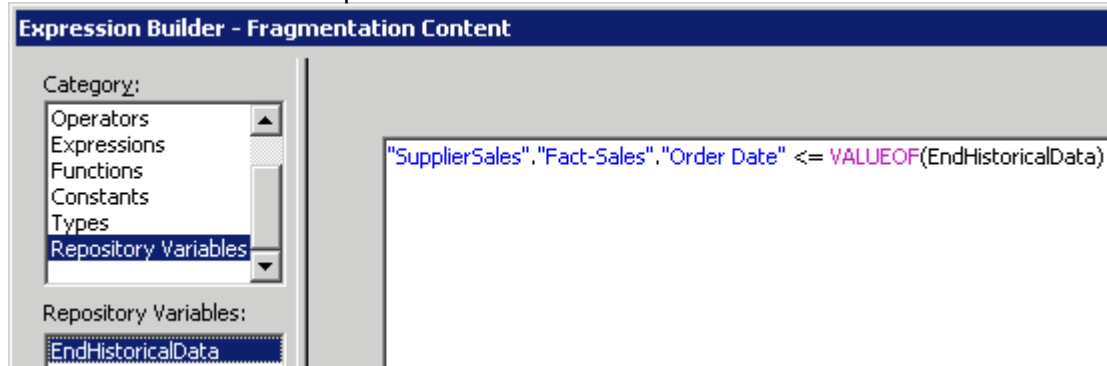
Variable Manager.



- t. Select **Action > Close** to close the Variable Manager.
4. Use the variable to dynamically determine the fragmentation content of the **Facts-Historical** logical table source.
 - a. In the **SupplierSales** business model, expand **Fact-Sales > Sources**.
 - b. Double-click **Facts-Historical**.
 - c. Click the **Content** tab.
 - d. Open the Expression Builder by clicking the **Edit Expression** button next to the Fragmentation content field.
 - e. Delete **'20081231'** from the expression after the <= operator.

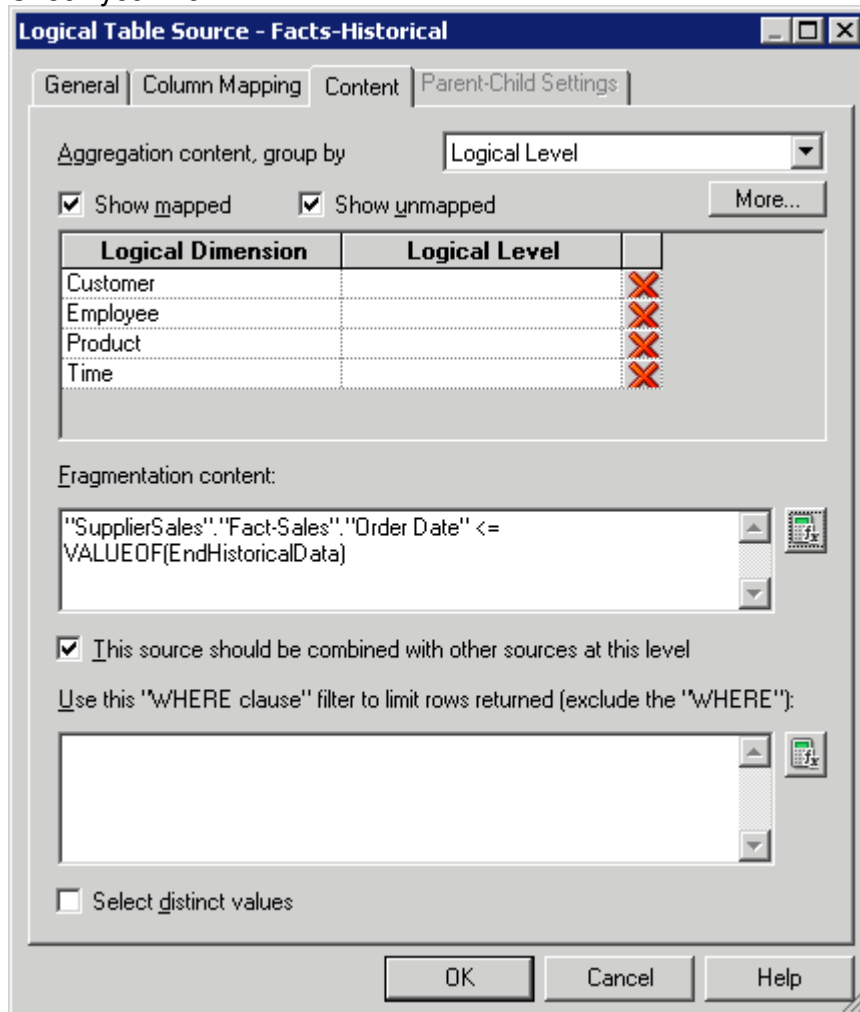
"SupplierSales"."Fact-Sales"."Order Date" <=

- f. In the left pane, select **Repository Variables** and double-click the **EndHistoricalData** variable to add it to the expression.



- g. Click **OK** to close Expression Builder.
- h. Ensure that the **This source should be combined with other sources at this level** check box is selected.

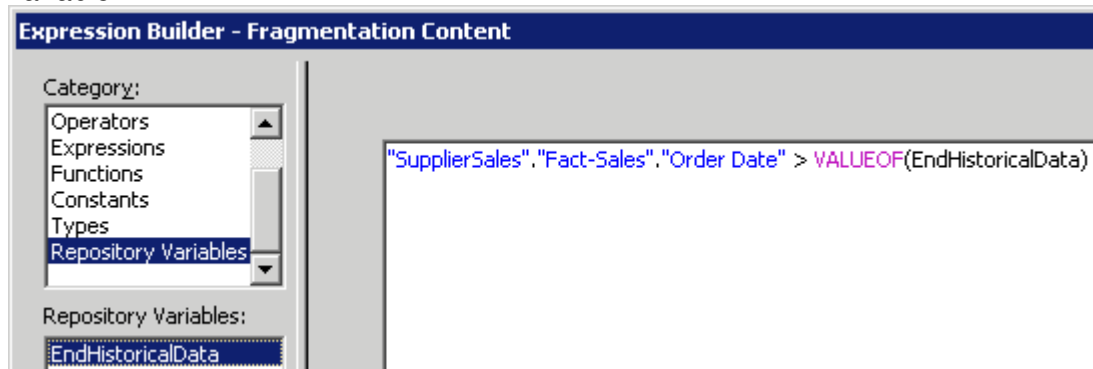
- i. Check your work.



The dialog box 'Logical Table Source - Facts-Historical' has four tabs: General, Column Mapping, Content, and Parent-Child Settings. The 'Content' tab is active. It features a dropdown for 'Aggregation content, group by' set to 'Logical Level'. Below this are two checked checkboxes: 'Show mapped' and 'Show unmapped', with a 'More...' button to the right. A table lists logical dimensions: Customer, Employee, Product, and Time, each with a red 'X' in the 'Logical Level' column. The 'Fragmentation content' section contains a text box with the expression: "SupplierSales"."Fact-Sales"."Order Date" <= VALUEOF(EndHistoricalData). Below this is a checked checkbox 'This source should be combined with other sources at this level' and a text box for a 'WHERE' clause filter. At the bottom is an unchecked checkbox 'Select distinct values' and three buttons: OK, Cancel, and Help.

Logical Dimension	Logical Level
Customer	X
Employee	X
Product	X
Time	X

- j. Click **OK** to close the Logical Table Source dialog box.
5. Use the variable to dynamically determine the content in the **Facts-Recent** logical table source.
 - a. Double-click **Facts-Recent**.
 - b. Click the **Content** tab.
 - c. Open the Expression Builder.
 - d. Modify the existing expression by replacing '20081231' with the **EndHistoricalData** variable:



The 'Expression Builder - Fragmentation Content' dialog box shows a 'Category' list on the left with 'Repository Variables' selected. Below this, 'EndHistoricalData' is listed under 'Repository Variables'. The main text area on the right displays the expression: "SupplierSales"."Fact-Sales"."Order Date" > VALUEOF(EndHistoricalData).

- e. Click **OK** to close Expression Builder.
- f. Ensure that the **This source should be combined with other sources at this level** check box is selected.
- g. Check your work.

Logical Table Source - Facts-Recent

General | Column Mapping | **Content** | Parent-Child Settings

Aggregation content, group by: Logical Level

☒ Show mapped ☒ Show unmapped More...

Logical Dimension	Logical Level	
Customer		X
Employee		X
Product		X
Time		X

Fragmentation content:

"SupplierSales"."Fact-Sales"."Order Date" >
VALUEOF(EndHistoricalData)

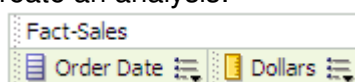
☒ This source should be combined with other sources at this level

Use this "WHERE" clause filter to limit rows returned (exclude the "WHERE"):

☐ Select distinct values

OK Cancel Help

- h. Click **OK** to close the Logical Table Source dialog box.
 - i. Save the repository.
 - j. Check consistency. Fix errors or warning messages before you proceed.
 - k. Close the repository.
 - l. Leave the Administration Tool open.
6. Test your work.
- a. Use Fusion Middleware Control to upload the ABC repository and restart Oracle BI components. If you need help, refer to steps from earlier practices.
 - b. Return to Analysis Editor.
 - c. Create an analysis:



- d. Click **Results**.

Order Date	Dollars
20080102	\$26,036
20080103	\$12,210
20080105	\$140,140
20080106	\$85,079
20080107	\$399,278
20080108	\$151,171
20080109	\$107,925
20080110	\$110,789

Notice that Order Date records are returned for both before and after 20081231 because no filters are applied. Click the **Display Maximum** button at the bottom of the table to confirm.

- e. Leave Analysis Editor open.
- f. Inspect the query log. Confirm that the query references both partition tables. Both partition tables are used because you did not apply a filter in the query. Notice that a separate `SELECT` statement is sent to each table and Oracle BI Server performs a `UNION ALL` to combine the results.

```
----- Sending query to database named orcl (id: <<2641>>),
connection pool named SUPPLIER CP: [[
WITH
SAWITH0 AS ((select T928.PERIODKEY as c2,
                  T928.DOLLARS as c3
from
    D1_ORDERS_RECENT T928 /* Fact_D1_ORDERS_RECENT */
union all
select T919.PERIODKEY as c2,
      T919.DOLLARS as c3
from
    D1_ORDERS_HISTORICAL T919 /* Fact_D1_ORDERS_HISTORICAL */
)),
SAWITH1 AS (select sum(D3.c3) as c1,
              D3.c2 as c2
from
    SAWITH0 D3
group by D3.c2)
select distinct 0 as c1,
              D2.c2 as c2,
              D2.c1 as c3
from
    SAWITH1 D2
order by c2
```

- g. Return to Analysis Editor and add a filter where **Order Date** is less than or equal to **20081231**.

New Filter

Column: Order Date

Operator: is less than or equal to

Value: 20081231

☐ Protect Filter

☐ Convert this filter to SQL

Buttons: Add More Options, Clear All, Help, OK, Cancel

- h. Run the analysis again.
- i. Verify that your results display only Order Dates that are less than or equal to 20081231.

Order Date	Dollars
20080102	\$26,036
20080103	\$12,210
20080105	\$140,140
20080106	\$85,079
20080107	\$399,278
20080108	\$151,171
20080109	\$107,925
20080110	\$110,789

- j. Leave Analysis Editor open.
- k. Check the query log and verify that the query accessed only one partition, **Fact_D1_ORDERS_HISTORICAL**.

```

----- Sending query to database named orc1 (id:
<<2630>>), connection pool named SUPPLIER CP: [[
WITH
SAWITH0 AS (select sum(T919.DOLLARS) as c1,
                T919.PERIODKEY as c2
from
    D1_ORDERS_HISTORICAL T919 /* Fact_D1_ORDERS_HISTORICAL */
where ( T919.PERIODKEY <= 20081231 )
group by T919.PERIODKEY)
select distinct 0 as c1,
                D1.c2 as c2,
                D1.c1 as c3
from
    SAWITH0 D1
order by c2 NULLS FIRST

```

- l. Return to the analysis and edit the filter to show records where **Order Date** is greater than 20081231.
- m. Run the analysis.

- n. Verify that your result only displays records where Order Date is greater than 20081231.

Order Date	Dollars
20090102	\$143,693
20090104	\$324,001
20090105	\$172,597
20090106	\$112,132
20090107	\$170,791
20090108	\$171,649
20090109	\$119,321
20090111	\$302,795
20090112	\$136,086

- o. Check the query log and confirm that the query being issued accesses the correct partition, Fact_D1_ORDERS_RECENT.

```
WITH
SAWITHO AS (select sum(T801.DOLLARS) as c1,
                  T801.PERIODKEY as c2
from
    D1_ORDERS_RECENT T801 /* Fact_D1_ORDERS_RECENT */
where ( 20081231 < T801.PERIODKEY )
group by T801.PERIODKEY)
select distinct 0 as c1,
               D1.c2 as c2,
               D1.c1 as c3
from
    SAWITHO D1
order by c2
```

- p. Sign out of Analysis Editor.

Practice 12-2: Using Dynamic Repository Variables as Filters

Goal

To create and use dynamic repository variables as filters

Scenario

Rather than creating hard-coded column filters like Year = 2009 in analyses, ABC wants to use variables that always return the current year, current month, and current day. You create these dynamic repository variables and then use them in column filters in Oracle BI analyses.

Outcome

A new initialization block, Current Periods, and three new dynamic repository variables (CurrentYear, CurrentMonth, and CurrentDay) are created.

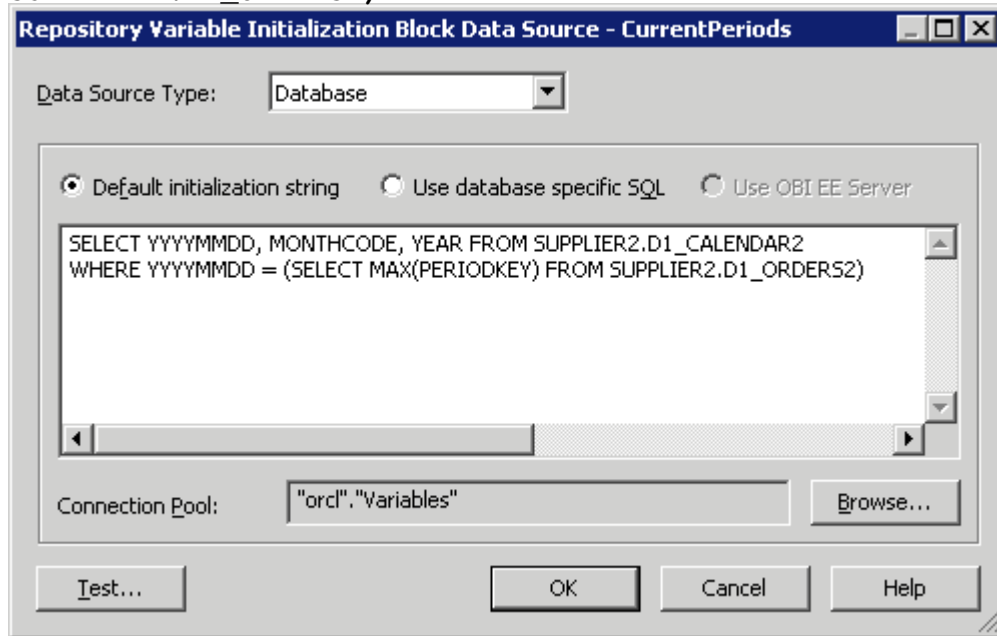
Time

15 minutes

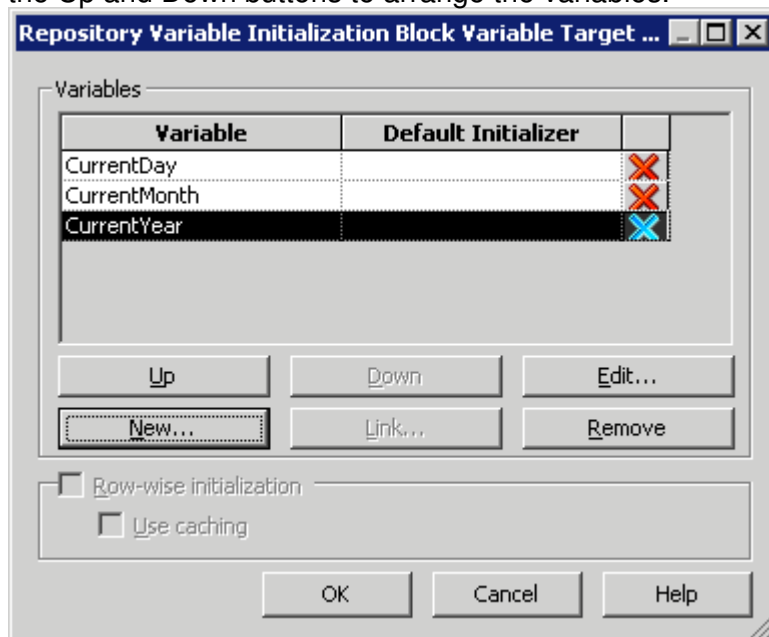
Tasks

1. In this step, you create a new initialization block. This initialization block will initialize three variables named CurrentYear, CurrentMonth, and CurrentDay. The variables will get their values from the initialization block SQL according to the schedule you set. In this example, the system determines the value of the current day by finding the maximum value of the period key (YYYYMMDD) in the D1_ORDERS2 table and then determining the month code and year that correspond to that value in the D1_CALENDAR2 table.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. Select **Manage > Variables**.
 - c. Select **Action > New > Repository > Initialization Block**.
 - d. Name the initialization block **CurrentPeriods**.
 - e. Click **Edit Data Source**.
 - f. Click the **Browse** button.
 - g. Double-click the **Variables** connection pool object to select it.
 - h. Enter the following SQL in the block to determine the value of the current day by finding the maximum value of the period key (YYYYMMDD):
**SELECT YYYYMMDD, MONTHCODE, YEAR FROM SUPPLIER2.D1_CALENDAR2
WHERE YYYYMMDD = (SELECT MAX(PERIODKEY) FROM**

SUPPLIER2.D1_ORDERS2)

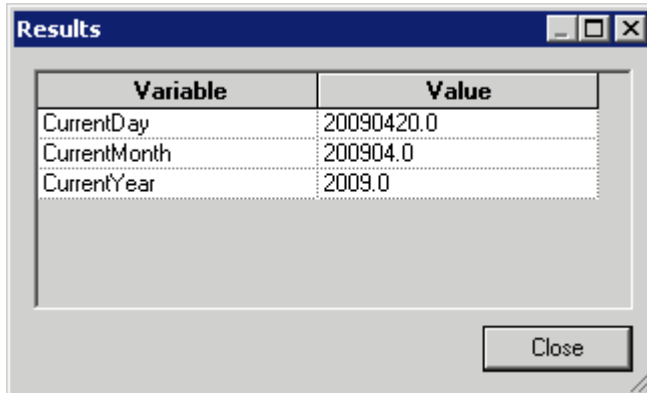


- i. Click **OK** to close the Repository Variable Initialization Block Data Source dialog box.
2. Create the variables.
 - a. Click the **Edit Data Target**.
 - b. Create three new variables: **CurrentDay**, **CurrentMonth**, and **CurrentYear**. The order is important. The value returned from the first column in the initialization block SQL, YYYYMMDD, is assigned to the CurrentDay variable. The value of the second column, MONTHCODE, is assigned to CurrentMonth (the second variable), and the value of the third column, YEAR, is assigned to CurrentYear (the third variable). If necessary, use the Up and Down buttons to arrange the variables.



- c. Click **OK** to close the Repository Variable Initialization Block Variable Target dialog box.

- d. Leave the default refresh interval set to every hour. This means that the variables will be reinitialized every hour.
- e. Click the **Test** button and check the results.



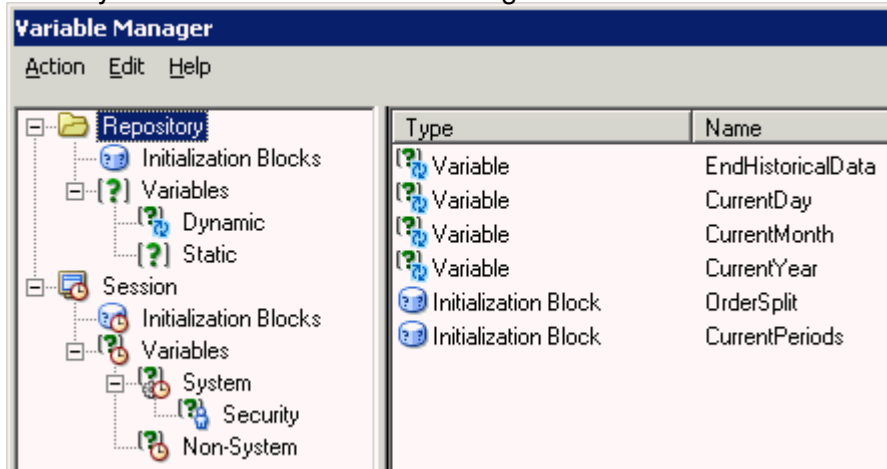
The Results window displays a table with two columns: Variable and Value. The data is as follows:

Variable	Value
CurrentDay	20090420.0
CurrentMonth	200904.0
CurrentYear	2009.0

A Close button is located at the bottom right of the window.

In this example, the results are determined by the data in the SUPPLIER2 database used for this course, which holds data only through April 2009.

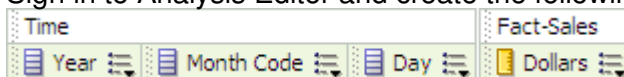
- f. Click **Close** to close the Results window.
- g. Click **OK** to close the Repository Variable Initialization Block dialog box.
- h. Check your work in the Variable Manager:



The Variable Manager window shows a tree view on the left and a list of variables on the right.

Type	Name
Variable	EndHistoricalData
Variable	CurrentDay
Variable	CurrentMonth
Variable	CurrentYear
Initialization Block	OrderSplit
Initialization Block	CurrentPeriods

- i. Close the Variable Manager.
 - j. Save the repository.
 - k. Check consistency. Fix any errors or warnings before proceeding.
 - l. Close the repository.
 - m. Leave the Administration Tool open.
3. Test your work.
 - a. Use Fusion Middleware Control to upload the ABC repository and restart Oracle BI components. If you need help, refer to steps from earlier practices.
 - b. Sign in to Analysis Editor and create the following analysis:



The Analysis Editor shows a table with columns: Time, Fact-Sales, Year, Month Code, Day, and Dollars. The columns are arranged in two rows.

Time	Fact-Sales
Year	Month Code
Day	Dollars

- c. Select **Filter** for the Year column. The New Filter dialog box opens.
- d. Select **Add More Options > Repository Variable**.

- e. In the Repository Variable field, enter **CurrentYear**.

New Filter

Column: Year

Operator: is equal to / is in

Value:

Repository Variable: CurrentYear

☐ Protect Filter

☐ Convert this filter to SQL

- f. Click **OK**.
- g. Repeat to add the **CurrentMonth** and **CurrentDay** repository variables as filters for the **Month Code** and **Day** columns, respectively.

Year is equal to / is in @{CurrentYear}
AND Month Code is equal to / is in @{CurrentMonth}
AND Day is equal to / is in @{CurrentDay}

- h. Click **Results** and ensure that only data for the current year, current month, and current day (2009 based on this data set) is returned.

Year	Month Code	Day	Dollars
2,009	200904	20090420	\$123,539.93

- i. Check the query log using the **Administration** link. In the SQL Request section, notice that the logical request filters for the variables. The following screenshot shows only a partial view of the log file:

```
##### LL
----- SQL Request:
SET VARIABLE QUERY_SRC_CD='Report';SELECT s_0, s_1, s_2, s_3,
s_4 FROM (SELECT
  0 s_0,
  "suppliersales"."Time"."Day" s_1,
  "suppliersales"."Time"."Month Code" s_2,
  "suppliersales"."Time"."Year" s_3,
  "suppliersales"."Fact-sales"."dollars" s_4
FROM "suppliersales"
WHERE
  (("Time"."Year" = VALUEOF("CurrentYear")) AND ("Time"."Month
Code" = VALUEOF("CurrentMonth")) AND ("Time"."Day" =
VALUEOF("CurrentDay")))
) djm ORDER BY 1, 4 ASC NULLS FIRST, 3 ASC NULLS FIRST, 2 ASC
NULLS FIRST
```

- j. Sign out of Analysis Editor.

Practices for Lesson 13: Modeling Time Series Data

Chapter 13

Overview of Practices for Lesson 13

Lesson Overview

In these practices, you will create time series calculation measures using Oracle BI time series functions.

Practice 13-1: Creating Time Series Comparison Measures

Goal

To create time series calculation measures using Oracle BI time series functions

Scenario

You use the Oracle BI time series functions to create new measures to enable users to compare current dollar performance to previous time periods. You then add the new measures to the presentation catalog and test the measures using analyses.

Time series functions include AGO, TODATE, and PERIODROLLING. These functions let you use Expression Builder to call a logical function to perform time series calculations instead of aliasing physical tables and modeling logically. The time series functions calculate AGO, TODATE, and PERIODROLLING functions based on the calendar tables in your data warehouse, not on standard SQL date manipulation functions.

Time

25 minutes

Background

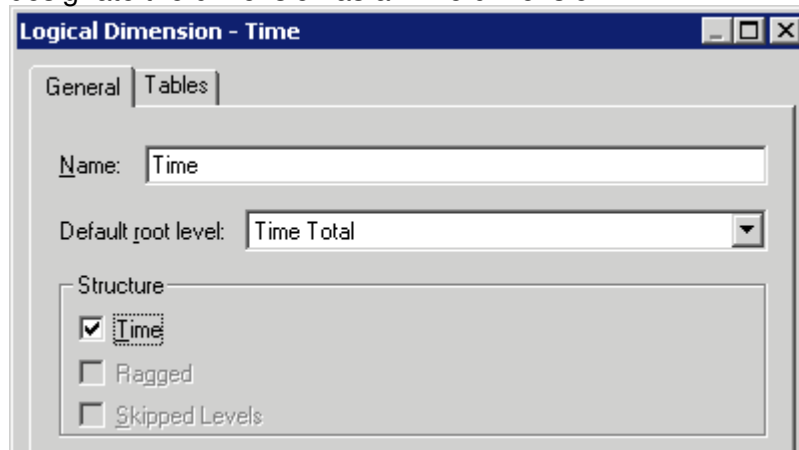
The ability to compare business performance with previous time periods is fundamental to understanding a business. Yet, as Ralph Kimball states, SQL was not designed to make straightforward comparisons over time:

“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

Tasks

1. In the SupplierSales business model, set the Time logical dimension as a time dimension.
 - a. Return to the Administration Tool, which should still be open, and open the **ABC** repository in offline mode with repository password **welcome1**.
 - b. In the SupplierSales business model, double-click the **Time** logical dimension.
 - c. On the General tab, select the **Time** check box. Time series functions operate on time-oriented dimensions. To use these functions on a particular dimension, you must

designate the dimension as a Time dimension.



- d. Click **OK** to close the dialog box.
2. Remove the logical key created for the Dim_MONTHS_AGG logical table source. This prevents consistency check errors related to hierarchy levels.
 - a. Double-click **Dim-Time** to open the Logical Table dialog box.
 - b. Click the **Keys** tab.
 - c. Select **Dim_MONTHS_AGG_Key**. This is a new logical key that was created when you created the Dim_MONTHS_AGG logical table source.



- d. Click **Delete** (the red X) to delete the key. This will prevent consistency check errors when you map a key column to a hierarchy level that is higher than the level with the detail key.
- e. Click **Yes** to confirm the deletion.
- f. Click **OK** to close the Logical Table dialog box.
3. Add MonthCode to the Month level in the Time logical dimension.
 - a. Expand the **Dim-Time** logical table.
 - b. Expand the **Time** logical dimension.
 - c. Drag **Month Code** from the **Dim-Time** logical table to the **Month** level in the **Time** logical dimension.
 - d. Right-click the **Month Code** level column and select **New Logical Level Key** to open the Logical Level Key dialog box.

- e. Deselect **Use for display** and click **OK**.

Logical Level Key - Month Code

Name:

☐ Qualified names

Columns:

- ☐ Month
- ☒ Month Code

☐ Use for display

Description:

4. Identify level keys as chronological keys. It is best practice to designate a chronological key for every level of a time dimension hierarchy.
 - a. Double-click the **Month** level to open the Logical Level dialog box.
 - b. Click the **Keys** tab.
 - c. Select the **Chronological Key** check box for **Month Code**.
 - d. Set **Month Code** as the primary key.

Logical Level - Month

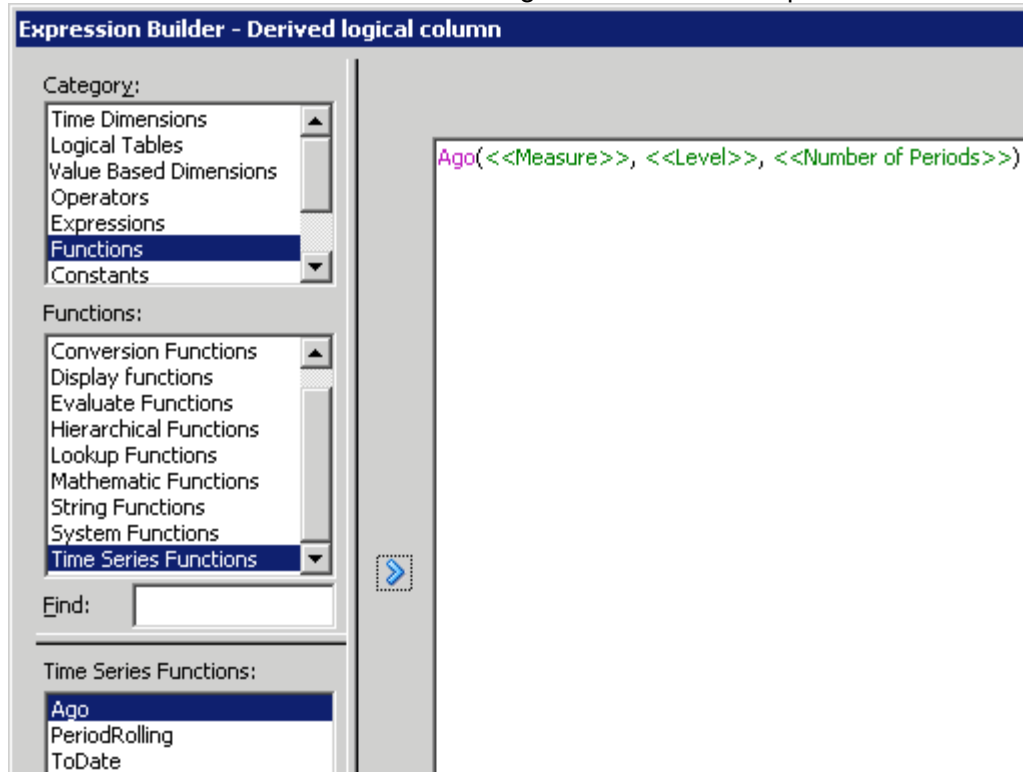
General | **Keys** | Preferred Drill Path

Primary key:

Key Name	Columns	Description	Use for Display	Chronological Key
Month	Month		<input checked="" type="checkbox"/>	<input type="checkbox"/>
Month Code	Month Code		<input type="checkbox"/>	<input checked="" type="checkbox"/>

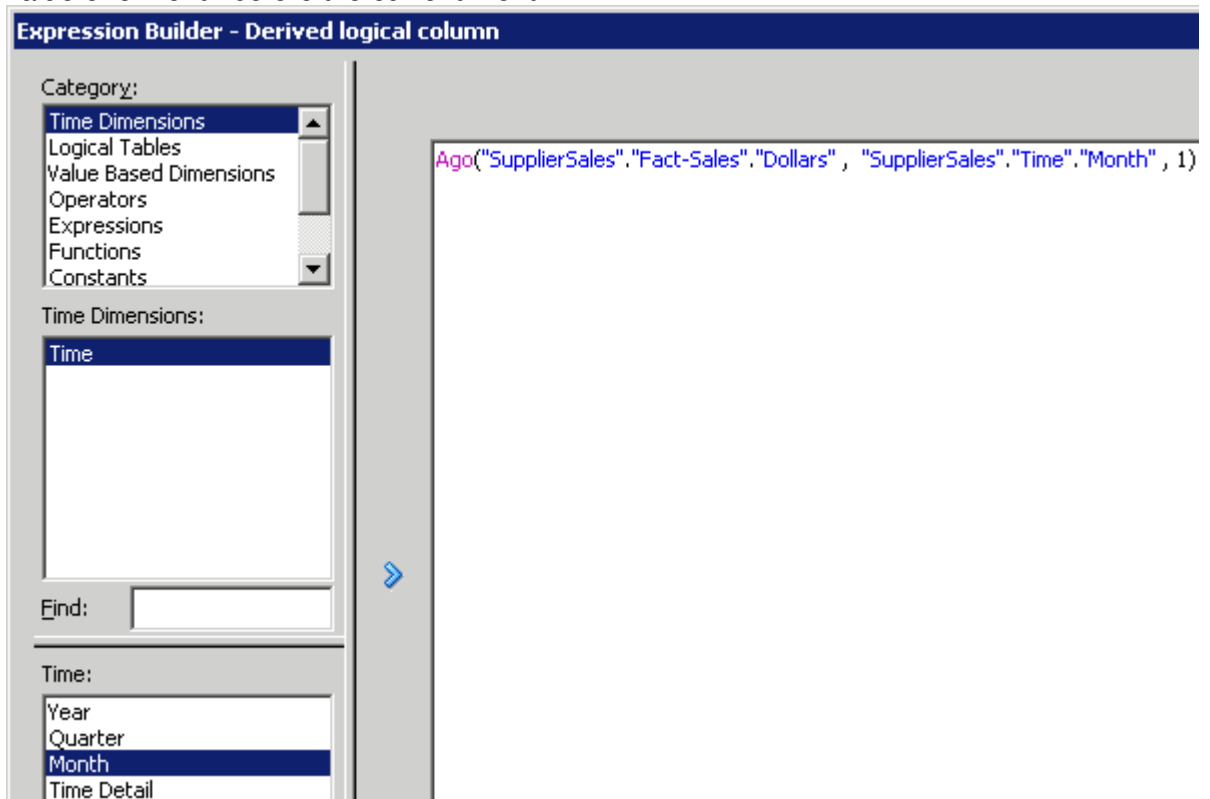
- e. Click **OK** to close the Logical Level dialog box.
 - f. Repeat and set chronological keys for the remaining levels:
Time Detail: Day
Quarter: Quarter
Year: Year
5. Create a measure that calculates dollars for the previous month by using the AGO function.
 - a. Right-click the **Fact-Sales** logical table and select **New Object > Logical Column**.
 - b. On the General tab, name the column **Month Ago Dollars**.
 - c. On the **Column Source** tab, select **Derived from existing columns using an expression**.

- d. Click the **Edit Expression** button to open the Expression Builder.
- e. Select **Functions > Time Series Functions > Ago**.
- f. Click **Insert selected item** to add the Ago function to the Expression Builder.



- g. Click **<<Measure>>** in the expression.
- h. Select **Logical Tables > Fact-Sales** and then double-click **Dollars** to add it to the expression.
- i. Click **<<Level>>** in the expression.
- j. Select **Time Dimensions > Time** and then double-click **Month** to add it to the expression.

- k. Click **<<Number of Periods>>** and enter **1**. The Ago function calculates the Dollars value one month before the current month.



- l. Click **OK** to close the Expression Builder.
- m. Check your work.

☒ Derived from existing columns using an expression
`Ago('SupplierSales', 'Fact-Sales', 'Dollars', 'SupplierSales', 'Time', 'Month', 1)`
- n. Click **OK** to close the Logical Column dialog box.
- o. Drag the **Month Ago Dollars** logical column to the **Fact-Sales** presentation table.
6. Create a measure that calculates the difference in dollars between the current month and the previous month.
 - a. Right-click the **Fact-Sales** logical table and select **New Object > Logical Column**.
 - b. On the General tab, name the new logical column **Change Month Ago Dollars**.
 - c. On the Column Source tab, select **Derived from existing columns using an expression**.
 - d. Open the Expression Builder.
 - e. Select **Logical Tables > Fact-Sales** and then double-click **Dollars** to add it to the expression.
 - f. Insert a **minus sign**.
 - g. Select **Logical Tables > Fact-Sales** and then double-click **Month Ago Dollars**.
 - h. Click **OK** to close the Expression Builder.

- i. Check your work.

Logical Column - Change Month Ago Dollars

General | Column Source | Aggregation | Levels

Data

Type: Length: ☒ Nullable

Derives from:

Column Source Type

☐ Derived from physical mappings

☒ Show all logical sources

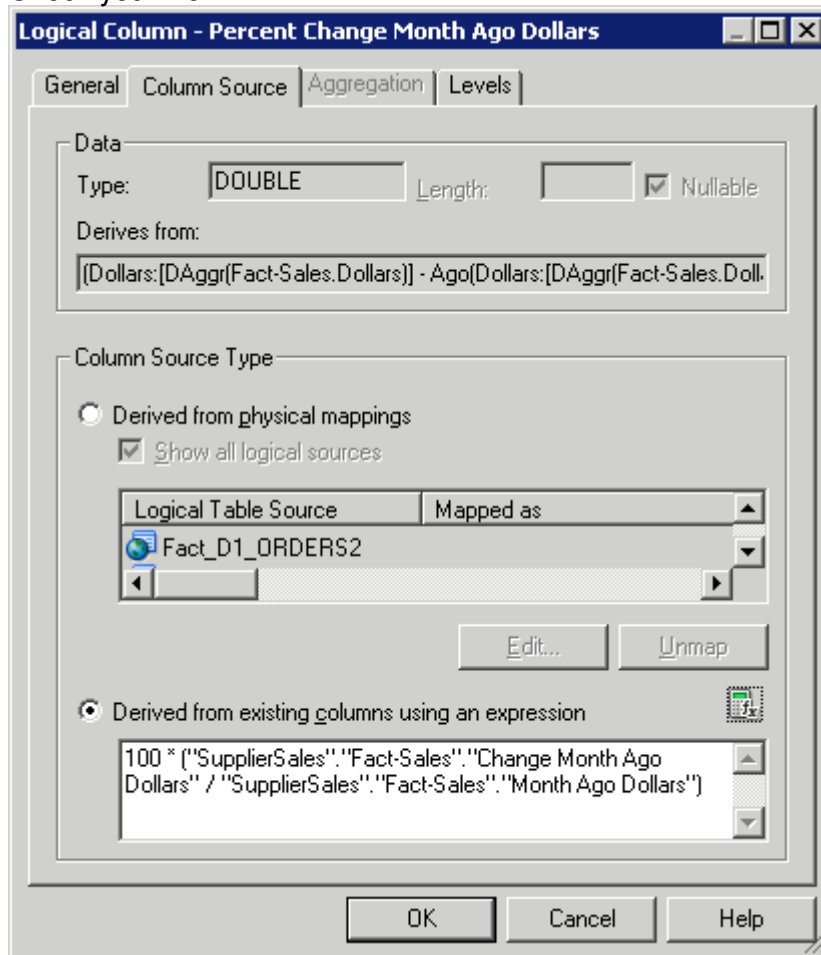
Logical Table Source | Mapped as

☒ Derived from existing columns using an expression

- j. Click **OK** to close the Logical Column dialog box.
 - k. Drag the **Change Month Ago Dollars** logical column to the Fact-Sales presentation table.
7. Create a measure that calculates the percentage change in dollars between the current month and the previous month.
 - a. Right-click **Fact-Sales** and select **New Object > Logical Column**.
 - b. Name the new logical column **Percent Change Month Ago Dollars**.
 - c. On the Column Source tab, select **Derived from existing columns using an expression**.
 - d. Open the Expression Builder and create the following expression:

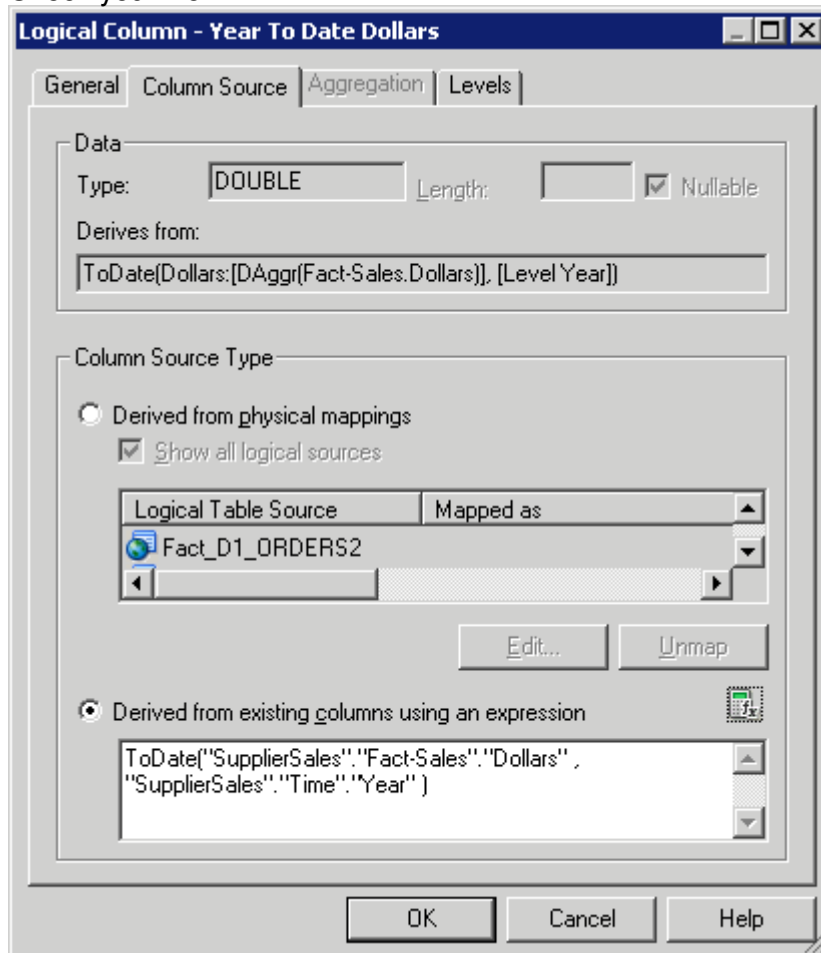

```
100 * ("SupplierSales"."Fact-Sales"."Change Month Ago Dollars" /
"SupplierSales"."Fact-Sales"."Month Ago Dollars")
```
 - e. Click **OK** to close the Expression Builder.

- f. Check your work.



- g. Click **OK** to close the Logical Column dialog box.
- h. Drag the **Percent Change Month Ago Dollars** logical column to the Fact-Sales presentation table.
8. Create a measure that calculates a running sum of dollars over the past year on a monthly basis using the TODATE function.
- Right-click **Fact-Sales** and select **New Object > Logical Column**.
 - Name the new logical column **Year To Date Dollars**.
 - On the Column Source tab, select **Derived from existing columns using an expression**.
 - Open the Expression Builder.
 - Select **Functions > Time Series Functions** and double-click **ToDate** to insert the expression.
 - Click **<<Measure>>**.
 - Select **Logical Tables > Fact-Sales** and then double-click **Dollars** to add it to the expression.
 - Click **<<Level>>**.
 - Select **Time Dimensions > Time** and then double-click **Year** to add it to the expression.
 - Click **OK** to close the Expression Builder.

- k. Check your work.



- l. Click **OK** to close the Logical Column dialog box.
 - m. Drag the **Year To Date Dollars** logical column to the Fact-Sales presentation table.
 - k. Save the repository.
 - l. Check consistency. Fix errors or warnings, if any, before you proceed.
 - m. Close the repository.
 - n. Leave the Administration Tool open.
11. Create analyses to check your work for the AGO and TODATE functions.
 - a. Use Fusion Middleware Control to upload the ABC repository and restart Oracle BI components. If you need help, refer to steps from earlier practices.
 - b. Sign in to Analysis Editor as **weblogic/welcome1**.
 - c. Create the following analysis and filter:

Time	Fact-Sales				
Month	Dollars	Month Ago Dollars	Change Month Ago Dollars	Percent Change Month Ago Dollars	Year To Date Dollars

Year is equal to / is in 2008

Note: Change the Dollars* columns data format to currency and the Percentage column data format to percentage.

- d. Click **Results**. Your results should look similar to the screenshot:

Month	Dollars	Month Ago Dollars	Change Month Ago Dollars	Percent Change Month Ago Dollars	Year To Date Dollars
January	\$3,595,669				\$3,595,669
February	\$3,945,187	\$3,595,669	\$349,518	9.72%	\$7,540,856
March	\$3,975,774	\$3,945,187	\$30,586	0.78%	\$11,516,630
April	\$3,907,292	\$3,975,774	-\$68,482	-1.72%	\$15,423,922
May	\$4,061,558	\$3,907,292	\$154,266	3.95%	\$19,485,480
June	\$3,994,531	\$4,061,558	-\$67,027	-1.65%	\$23,480,010
July	\$4,062,212	\$3,994,531	\$67,681	1.69%	\$27,542,222
August	\$4,242,611	\$4,062,212	\$180,399	4.44%	\$31,784,833
September	\$3,810,263	\$4,242,611	-\$432,348	-10.19%	\$35,595,096
October	\$4,596,372	\$3,810,263	\$786,109	20.63%	\$40,191,468
November	\$3,655,169	\$4,596,372	-\$941,203	-20.48%	\$43,846,637
December	\$3,997,616	\$3,655,169	\$342,448	9.37%	\$47,844,253

- e. Note that **Month** is sorted automatically based on the MonthCode sort-order column that you set in the repository.

Sign out of Oracle BI.

Practices for Chapter 14 : Security

Chapter 14

Overview of Practices for Chapter 14

Lesson Overview

In these practices, you will learn how to implement Oracle BI security.

Lab 14-1: Exploring Default Security Settings

Goal

To explore the default security settings for Oracle Business Intelligence

Scenario

During installation, three Oracle Business Intelligence security controls are preconfigured with initial values to form the default security model. The security controls include:

An embedded directory server functioning as an *identity store* designed to hold all user and group definitions, required to control authentication

A file-based *policy store* designed to hold the application role and permission grant mappings to users and groups, required to control authorization

A file-based *credential store* designed to hold all user and system credentials, required to control authentication or authorization

Before you implement data access security in the Oracle BI repository, you explore these default security settings.

Time

10 minutes

Tasks

1. Log in to the WebLogic Server Administration Console. This console is used to manage users and groups for the embedded LDAP server that serves as an out-of-the-box identity store.
 - a. Open a new tab on the browser and enter the following URL:
http://localhost:7001/console.
 - b. Log in as **weblogic** with password **welcome1**. During installation, you are prompted for a username and password to use as an Oracle BI Administrator. In this training environment, the Administrator user created during installation is **weblogic**. This is an arbitrary choice and there is nothing special about the name. This user has administrative privileges across the Oracle Business Intelligence (OBI) stack. This includes the OBI repository, the OBI presentation catalog, BI Publisher, RTD, Essbase, the identity store in WebLogic, and the Policy Store in Fusion Middleware.
2. Explore default settings for providers in the WebLogic security realm.
 - a. On the left side of the console, under Domain Structure, notice there is a single WebLogic domain named `bifoundation_domain` into which all the OBI applications are deployed.
 - b. Click **Security Realms**.
 - c. Notice that there is a single default security realm named `myrealm`. The OBI installer installs a single domain with a single security realm in it. A security realm is a container for the mechanisms that are used to protect WebLogic resources. This includes users, groups, security roles, security policies, and security providers. Whereas multiple security realms can be defined for the OBI domain, only one can be active, meaning designated as the default realm, at any given time.
 - d. Click **myrealm** to view the default settings.
 - e. Click the **Providers** tab.

- f. Notice that there is a default **WebLogic Authentication Provider**. An authentication provider establishes the identity of users and system processes, transmits identity information, and serves as a repository for identity information from which components can retrieve it. Oracle Business Intelligence is configured to use the directory server embedded in Oracle WebLogic Server as the default security provider. Alternate security providers can be used if desired and managed in the Oracle WebLogic Administration Console, but the WebLogic authentication provider is used by default.
 - g. Notice that there is a default **WebLogic Identity Assertion Provider**. WebLogic Identity Assertion Provider is used primarily for Single Sign On and is not covered in this training.
3. Explore default settings for users.
- a. Click the **Users and Groups** tab.
 - b. Click the **Users** subtab. The default identity store is pre-seeded with user names specific to Oracle Business Intelligence. These default user names are provided as a convenience so that you can begin using the Oracle Business Intelligence software immediately after installation but you are not required to maintain the default names in your deployment. The two default users are an administrative user and BISystemUser.
 - c. Notice the **weblogic** user. This is the administrative user created during the installation process. A single administrative user is shared by Oracle Business Intelligence and Oracle WebLogic Server. As stated earlier, this username is created during installation, can be any desired name, and therefore does not need to be “Administrator.” The password is likewise provided during installation. In the default security configuration, an administrative user is a member of the BIAdministrators group and has all rights granted to the Oracle Business Intelligence Administrator user in earlier releases, with the exception of impersonation. An administrative user cannot impersonate other users. An administrative user is also a member of the Oracle WebLogic Server default Administrators group, which enables this user to perform all its administration tasks, including the ability to manage Oracle WebLogic Server’s embedded directory server and policy store.
 - d. Notice the **BISystemUser** user. Oracle Business Intelligence system components establish a connection to each other as BISystemUser instead of as the Administrator user, the latter being the practice in earlier releases. Using a trusted system account, such as BISystemUser, to secure communication between Oracle BI components enables you to change the password of your deployment’s system administrator account without affecting communication between these components. The name of this user is the default, and it can be changed or a different user can be created for the purpose of inter-process communication. This is a highly privileged user whose credentials should be protected from non-administrative users.
 - e. Notice other users such as **Administrator_d** and **Administrator_f**. These users are provided as part of the training environment and are not default Oracle BI users (Ignore this point).
4. Explore default settings for groups.
- a. Click the **Groups** subtab. Groups are logical ordered sets of users. Creating groups of users who have similar system resource access needs enables easier security management. Managing a group is more efficient than managing a large number of users individually. Oracle recommends that you organize your users into groups for easier maintenance. As you will see later in this practice, groups are then mapped to application roles in order to grant rights.
 - b. Notice the three default groups specific to Oracle BI: **BIAdministrators**, **BIAuthors**, and **BIConsumers**. These default groups are provided as a convenience so you can

begin using the Oracle Business Intelligence software immediately after installation, but you are not required to maintain the default names in your deployment. Members of the BIAdministrators group have permissions equivalent to those of the Administrator user of earlier releases. Members of the BIAuthors group have the permissions necessary to create content for others to consume. Members of the BIConsumers group have the permissions necessary to consume content created by others. Groups are nested in a hierarchy. Members of the BIAdministrators group are by default members of both other groups. Members of BIAuthors are members of BIConsumers.

- c. Click the **Users** subtab again.
 - d. Click the **weblogic** user.
 - e. Click the **Groups** tab.
 - f. Notice that the **weblogic** user is a member of the two administrator groups: **Administrators** and **BIAdministrators**. Administrators is the WebLogic administrators group, which gives rights to administer WebLogic and FMW enterprise manager. Membership in both administration groups gives this user a single unified administration account for the entire product stack.
5. Explore the default settings for application roles. An application role defines a set of permissions granted to a user or group.
- a. Return to the browser tab where Fusion Middleware Control is open. If it is not open, enter **http://localhost:7001/em** and log in as **weblogic/welcome1**.
 - b. In the left pane, expand **Business Intelligence**.
 - c. Right-click **coreapplication** and select **Security > Application Roles** to navigate to the Application Roles page.
 - d. By default, the **obi** application stripe is selected and the default application roles are displayed.

Application Roles

Application roles are the roles used by security aware applications that are specific to the application. These roles are seeded by applications in single global policy store when the applications are registered. These are also application roles that are created in the context of end users accessing the application.

☒ To manage users and groups in the WebLogic Domain, use the [Oracle WebLogic Server Security Provider](#).

Policy Store Provider

Search

Enter search keyword for role name to query roles defined by this application. Use application stripe to search if application uses a stripe that is different from application name.

Select Application Stripe to Search ☒

Role Name

Role Name	Members	Description
BISystem	BISystemUser	
BIAdministrator	BIAdministrators	
BIAuthor	BIAuthors, BIAdministrator	
BIConsumer	BIConsumers, BIAuthor, authenticated-role	

BIAdministrator: Grants administrative permissions necessary to configure and manage the Oracle Business Intelligence installation. Any member of the BIAdministrator *group* is explicitly granted this role and implicitly granted the BIAuthor and BIConsumer roles

BIAuthor: Grants permissions necessary to create and edit content for others to consume. Any member of the BIAuthor *group* is explicitly granted this role and implicitly

granted the BIConsumer role.

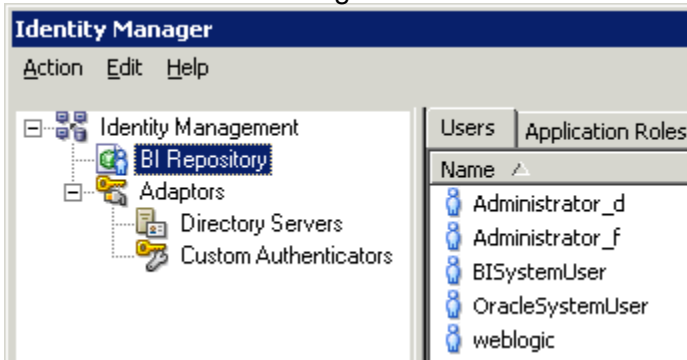
BIConsumer: Grants permissions necessary to consume content created by others. Any member of the BIAuthor *group* is explicitly granted this role.

BISystem: Grants the permissions necessary to impersonate other users. This role is required by Oracle Business Intelligence system components for inter-component communication.

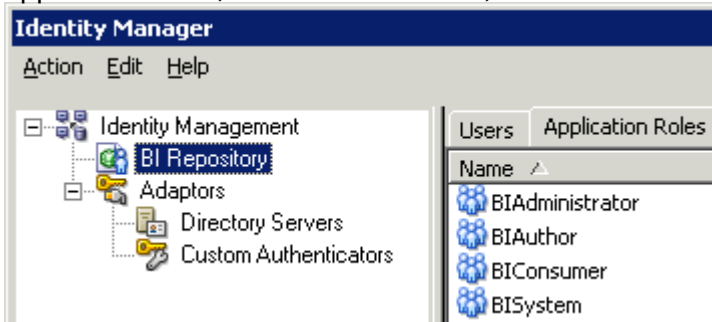
The default application roles are mapped to default groups in the default WebLogic LDAP. The groups are listed in the Members column. If you moved to a different LDAP server, rather than the default WebLogic LDAP server, you could map these roles to groups in the new LDAP server. Application roles are in the policy store, whereas groups are in the identity store.

6. Explore default settings for application policies.
 - a. In the left pane, right-click **coreapplication** and select **Security > Application Policies**.
 - b. By default, the **obi** application stripe is selected and the default application policies are displayed. The default file-based policy store is pre-seeded with the Oracle BI-specific permissions. All Oracle Business Intelligence permissions are provided and you cannot create additional permissions. These permissions are granted by the default application roles in the default security configuration. The default application role hierarchy and permission grants can be changed as needed. Also notice that these permissions are not the same as those used to define access to BI objects (metadata, dashboards, reports, etc.). Policy store permissions are only used to define what BI functionality the assigned roles can access. Notice, for example, that the BIAdministrator role has been granted the permission to manage repositories.
7. Explore default security settings in the repository.
 - a. Return to the Administration Tool and open the ABC repository in online mode with **welcome1** as the repository and user password.
 - b. Because weblogic is a member of the BIAdministrators group, which is a member of the BIAdministrator Role with the permissions assigned to that role, weblogic can log in to the Administration Tool.
 - c. Select **Manage > Identity** to open Identity Manager.
 - d. In the left pane, expand **Identity Management** and select **BI Repository**.
 - e. Click the **Users** tab. Initially, only the weblogic administrative user is visible.
 - f. Select **Action > Set Online User Filter**.
 - g. Enter an **asterisk** in the field and click **OK** to fetch users.
 - h. Notice that you can see the same set of users as those listed in the WebLogic Server Administration Console. The key point is that users are no longer in the repository as in previous OBI product releases. They are in the WebLogic LDAP, or whatever identity store your system is configured with. You must add a new user in the identity store, not in the repository, as you will see later in this set of practices. Identity Manager should

look similar to the following screenshot:



- i. Click the **Application Roles** tab to view all application roles in the policy store. As you will see later in this set of practices, you can now use the application roles to set access control permissions for repository objects. The recommended practice is to use application roles, not individual users, to set access control permissions.



- j. Select **Action > Close** to close Identity Manager without making any changes.
- k. Close the repository.
- l. Leave the Administration Tool open.

Practice 14-2: Creating Users and Groups

Goal

To create users and groups in the WebLogic identity store

Scenario

Groups are logical ordered sets of users. Managing a group is more efficient than managing a large number of users individually. Oracle recommends that you first organize all Oracle Business Intelligence users into groups that make sense for your organization's goals and map application roles to the groups in order to convey system privileges. The default identity store provided for managing users and groups is Oracle WebLogic Server's embedded directory server. You use WebLogic Administration Server Console to create users and groups.

Time

10 minutes

Tasks

1. Create new groups.
 - a. Return to the WebLogic Server Administration Console, which should still be open in a browser tab. If not, enter **http://localhost:7001/console** and log in as **weblogic/welcome1**.
 - b. Click **Security Realms**.
 - c. Click **myrealm**.
 - d. Click **Users and Groups**.
 - e. Click the **Groups** subtab.
 - f. Click the **New** button and create the following three groups:

Name	Description	Provider
SalesManagersGroup	Sales Managers Group	DefaultAuthenticator
SalesSupervisorsGroup	Sales Supervisors Group	DefaultAuthenticator
SalesAssociatesGroup	Sales Associates Group	DefaultAuthenticator

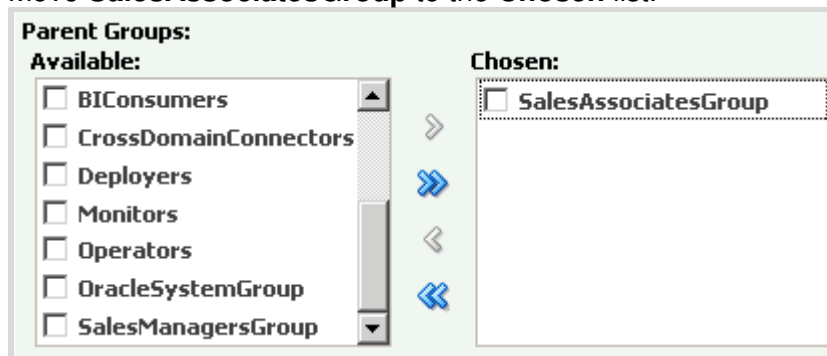
- g. Click **Next** at the bottom of the **Groups** table and verify that the new groups are visible.

Groups

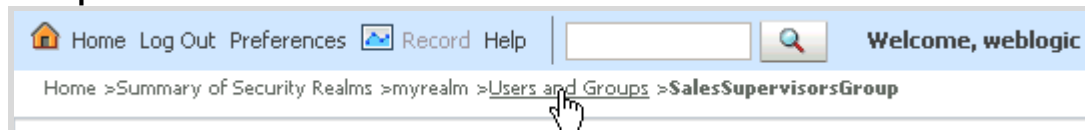
<input type="button" value="New"/>	<input type="button" value="Delete"/>	Showing 11 to 14 of 14 Previous Next		
<input type="checkbox"/>	Name	Description	Provider	
<input type="checkbox"/>	OracleSystemGroup	Oracle application software system group.	DefaultAuthenticator	
<input type="checkbox"/>	SalesAssociatesGroup	Sales Associates Group	DefaultAuthenticator	
<input type="checkbox"/>	SalesManagersGroup	Sales Managers Group	DefaultAuthenticator	
<input type="checkbox"/>	SalesSupervisorsGroup	Sales Supervisors Group	DefaultAuthenticator	
<input type="button" value="New"/>	<input type="button" value="Delete"/>	Showing 11 to 14 of 14 Previous Next		

2. Create a group hierarchy.
-

- a. Click **SalesSupervisorsGroup**.
- b. Click the **Membership** tab.
- c. In the Available list, select **SalesAssociatesGroup**.
- d. Move **SalesAssociatesGroup** to the **Chosen** list.



- e. Click **Save**. This means that any privileges or permissions assigned to the Sales Associates group will be inherited by the Sales Supervisors group.
- f. Use the browser Back button to return to the Groups page or click the **Users and Groups** link:



- g. Click **SalesManagersGroup**.
 - h. Click the **Membership** tab.
 - i. In the Available list, select **SalesAssociatesGroup**.
 - j. Move **SalesAssociatesGroup** to the **Chosen** list.
 - k. Click **Save**. This means that any privileges or permissions assigned to the Sales Associates group will be inherited by the Sales Managers group.
3. Create new users.
 - a. Click the **Users** tab.
 - b. Click the **New** button and create the following users and passwords:
- | Name | Description | Provider | Password |
|-------|-------------|----------------------|-----------|
| JCRUZ | Jose Cruz | DefaultAuthenticator | JoseCruz1 |
| AZIFF | Alan Ziff | DefaultAuthenticator | AlanZiff1 |
4. Assign users to groups.
 - a. Click **AZIFF** in the Users table.
 - b. Click the **Groups** tab.

- c. Move **BIAuthors** and **SalesAssociatesGroup** from the Available list to the Chosen list.

Settings for AZIFF

General Passwords Attributes **Groups**

Save

Use this page to configure group membership for this user.

Parent Groups:

Available:

- ☐ AdminChannelUsers
- ☐ Administrators
- ☐ AppTesters
- ☐ BIAdministrators
- ☐ BIConsumers
- ☐ CrossDomainConnectors
- ☐ Deployers

Chosen:

- ☐ BIAuthors
- ☐ SalesAssociatesGroup

- d. Click **Save**.
- e. Repeat to add JCRUZ to **BIAuthors**, **SalesSupervisorsGroup**, and **SalesManagersGroup**.

Settings for JCRUZ

General Passwords Attributes **Groups**

Save

Use this page to configure group membership for this user.

Parent Groups:

Available:

- ☐ BIConsumers
- ☐ CrossDomainConnectors
- ☐ Deployers
- ☐ Monitors
- ☐ Operators
- ☐ OracleSystemGroup
- ☐ SalesAssociatesGroup

Chosen:

- ☐ BIAuthors
- ☐ SalesManagersGroup
- ☐ SalesSupervisorsGroup

Practice 14-3: Creating Application Roles(Pending)

Goal

To create application roles in the policy store

Scenario

An application role conveys its permission grants to the users, groups, and application roles mapped to that role. Being mapped to an application role establishes membership in the role. Binding the permission grants to the application role streamlines the process of granting system privileges. Once the application role and permission grant definitions are established, you control system rights by managing membership in each role. Oracle recommends that you map groups to application roles and not individual users. Once mapped, all members of the group are granted the same rights. Controlling membership in a group reduces the complexity of tracking access rights for multiple individual users. You use Fusion Middleware Control to create application roles.

Time

20 minutes


Tasks

1. Create new application roles.
 - a. Return to Fusion Middleware Control, which should still be open in a browser tab. If not, enter **http://localhost:7001/em** and log in as **weblogic** with password **welcome1**.
 - b. In the left pane, expand **Business Intelligence > coreapplication**.
 - c. Right-click **coreapplication** and select **Security > Application Roles**.
 - d. By default, the **obi** application stripe is selected and the default application roles are displayed.
 - e. Click **Create**.
 - f. In the Role Name field, enter **SalesManagersRole**.
 - g. In the Display Name field, enter **Sales Managers Role**.
 - h. Click **Add Group** to open the Add Group dialog box.
 - i. Click the **blue arrow** next to the Group Name field to see a list of available groups from the identity store.
 - j. In the Available Groups list, select **SalesManagersGroup** and move it to the Selected Groups list.

Add Group

Specify criteria to search and select the enterprise groups that you want to grant permissions to.

Search

Group Name 

Select groups

Available Groups

- ☐ CrossDomainConnectors
- ☐ BIAuthors
- ☐ AppTesters
- ☐ AdminChannelUsers
- ☐ BIAdministrators
- ☐ OracleSystemGroup
- ☐ Operators
- ☐ SalesSupervisorsGroup
- ☐ SalesAssociatesGroup
- ☐ Deployers

Move

Move All

Remove

Remove All





Selected Groups

- ☐ SalesManagersGroup

OK Cancel

This means that any user who is a member of the selected group is assigned to this application role and receives any privileges or permissions assigned to the application role. It is possible to add individual users to a role, but best practice is to add groups, not individual users, to roles. In this example, Jose Cruz is a member of the Sales Managers group and is therefore assigned to the SalesManagersRole application role.

- k. Click **OK**.
- l. Click **OK** again to return to the Applications Role list.
- m. Verify that **SalesManagersRole** appears in the list with **SalesManagersGroup** listed as a member.

 Create...  Create Like...  Edit...  Delete...	
Role Name	Members
BISystem	BISystemUser
BIAdministrator	BIAdministrators
BIAuthor	BIAuthors, BIAdministrator
BIConsumer	BIConsumers, BIAuthor, authenticated-role
SalesManagersRole	SalesManagersGroup

- n. Repeat the steps to create the additional application roles and corresponding members listed in the following table. Please notice that you are adding a group and two application roles as members of the SalesAssociatesRole application role.

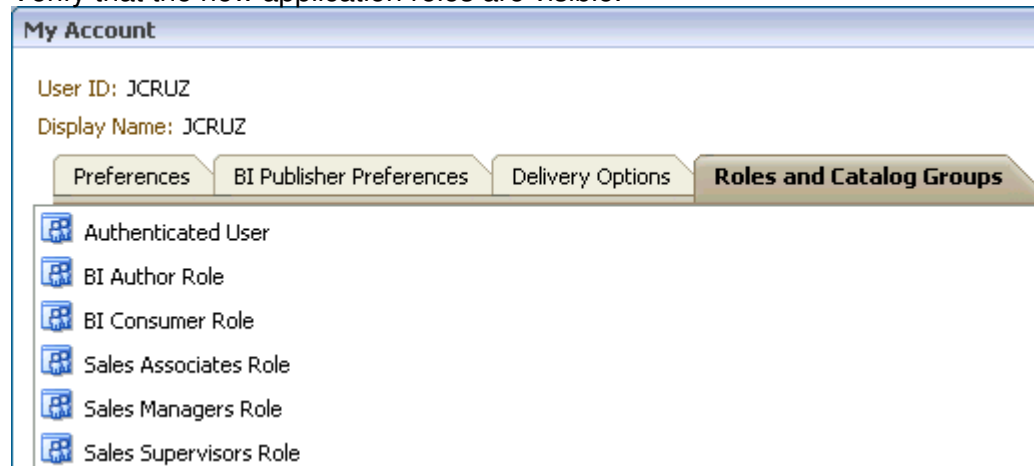
Application Role	Members
SalesSupervisorsRole	SalesSupervisorsGroup
SalesAssociatesRole	SalesAssociatesGroup SalesSupervisorsRole SalesManagersRole

- o. Check your work. Your application roles should look similar to the following screenshot:

Create... Create Like... Edit... Delete...	
Role Name	Members
BISystem	BISystemUser
BIAdministrator	BIAdministrators
BIAuthor	BIAuthors, BIAdministrator
BIConsumer	BIConsumers, BIAuthor, authenticated-role
SalesManagersRole	SalesManagersGroup
SalesSupervisorsRole	SalesSupervisorsGroup
SalesAssociatesRole	SalesAssociatesGroup, SalesManagersRole, SalesSupervisorsRole

As with the groups that you created in the WebLogic identity store, you now have an application role hierarchy in the policy store. This means that any privileges or permissions assigned to the SalesAssociatesRole application role will be inherited by the SalesSupervisorsRole role and the SalesManagersRole role. What is the difference between users and groups created in the identity store in the WebLogic LDAP Server and application roles created in the policy store? In the WebLogic LDAP server you have users and groups. An application role is a logical role that can be used within the application to secure content in a way that is independent of any particular LDAP server and the users and groups within that LDAP server. Security rules are built using application roles. If the underlying LDAP environment changes, the security rules persist. In a different LDAP environment, where group or user names might be different, you could remap the application roles to different groups or users and the BI security structure, built with application roles, would not be affected.

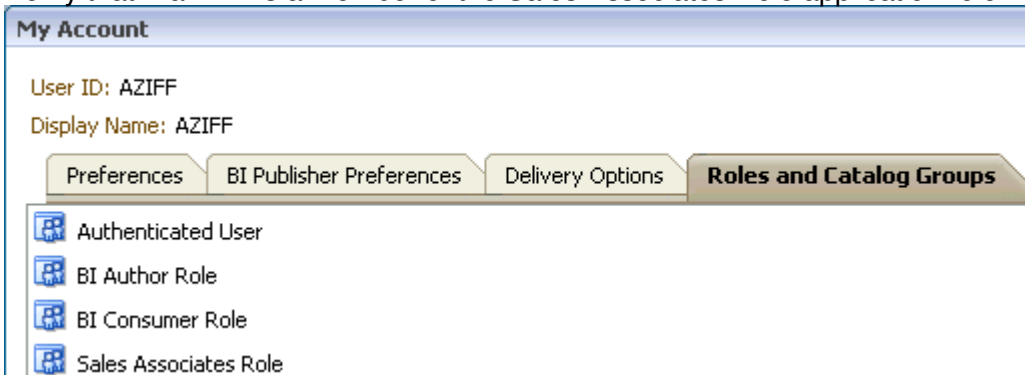
2. Verify that the new users and application roles are now visible in Oracle BI.
 - a. Sign in to Oracle BI as **JCRUZ** with password **JoseCruz1**.
 - b. In the upper-right corner, select **JCRUZ > My Account**.
 - c. Click the **Roles and Catalog Groups** tab.
 - d. Verify that the new application roles are visible.



Jose Cruz is a member of the Sales Managers group, which is a member of the Sales Managers Role application role. Jose Cruz is also a member of the Sales Supervisors group, which is a member of the Sales Supervisors Role application role. Because both of these roles are members of Sales Associate Role, he is also a member of that role. By default, all BI users are also members of the default group, Authenticated Users, and the default application role, BIConsumer. Recall that you also added Jose Cruz to the BI Author Role application role.

- e. Click **OK** to close the My Account dialog box.

- f. Sign out and sign back in as **AZIFF** with password **AlanZiff1**.
- g. Select **AZIFF > My Account**.
- h. Click **Roles and Catalog Groups**.
- i. Verify that Alan Ziff is a member of the Sales Associates Role application role.



Alan Ziff is a member of the Sales Associates group, which is a member of the Sales Associates Role application role. Because the Sales Associates Role application role is the highest role in the hierarchy, he is not a member of the other two application roles in the hierarchy.

- j. Click **OK** to close the My Account dialog box.
- k. Sign out of Oracle BI. Application roles serve a variety of purposes in both development and production environments. In a development environment, developers can be granted one or more of the roles. One approach is to build roles that will eventually be used in production, and then map developers to those roles for administering, building, and testing the development environment. As you will see in the next practice, you also use the logical application roles to secure access to repository objects and data. Therefore, application roles can be used to control access to both objects and functionality in the product. The value of using application roles comes from the fact that you can move the system that you have built between environments without having to rewire all the security. For example, you would not have to change security settings in your presentation catalog or repository. You can just remap your application roles to the target environment.

Practice 14-4: Setting Up Object Permissions

Goal

To set up object permissions in the repository

Scenario

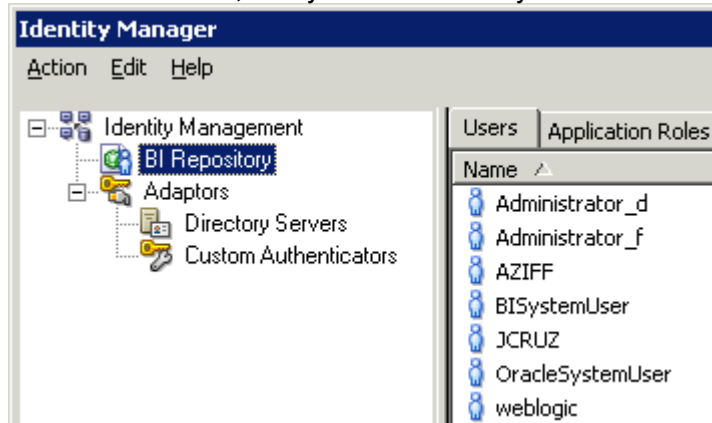
You can set up object permissions in your repository to control access to Presentation layer and Business Model and Mapping layer objects. You set object permissions using the Administration Tool. There are two approaches to setting object permissions: you can set permissions for particular users or application roles in the Identity Manager, or you can set permissions for individual objects in the Presentation layer. In this practice you use both approaches. Setting up object permissions for individual users or application roles is useful when you want to define permissions for a large set of objects at one time. Although it is possible to set up object permissions for individual users, the recommended practice is to set up object permissions for particular application roles rather than for individual users.

Time

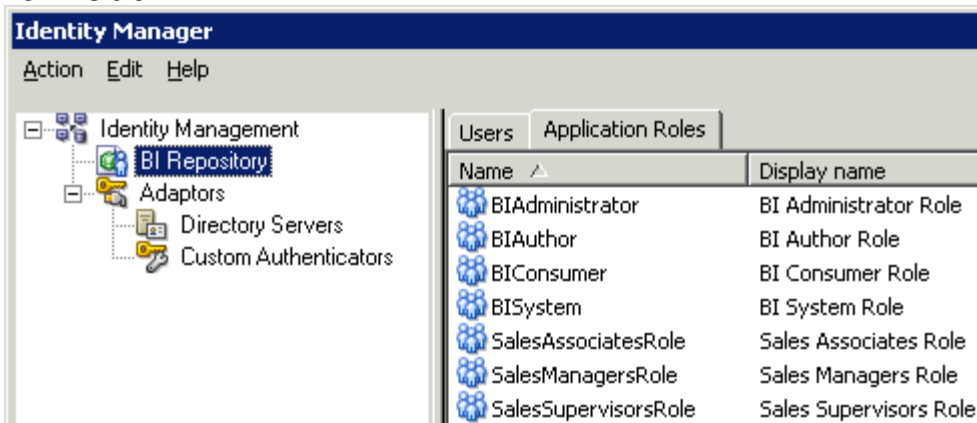
30 minutes

Tasks

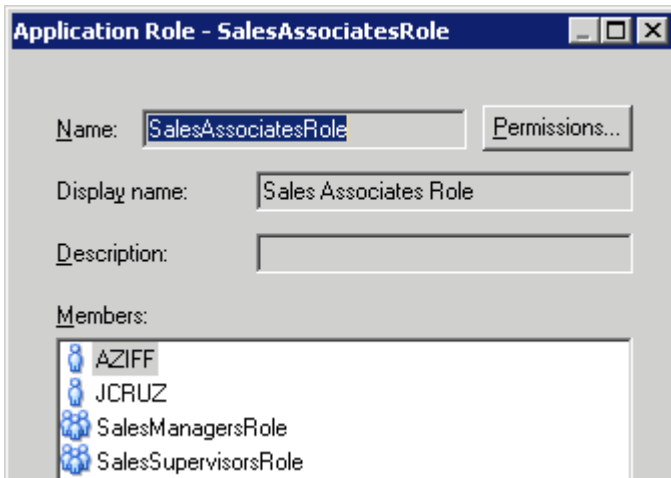
1. Verify that users and application roles are now visible in the repository.
 - a. Return to Fusion Middleware Control and restart Oracle BI components.
 - b. Open the repository in online mode, using **welcome1** as the password for both the repository and user.
 - c. Select **Manage > Identity**.
 - d. Select **Identity Management > BI Repository** in the left pane.
 - e. Select **Action > Set Online User Filter**.
 - f. Enter an **asterisk** in the field and click **OK**.
 - g. On the **Users** tab, verify that the users you created in the identity store are now visible.



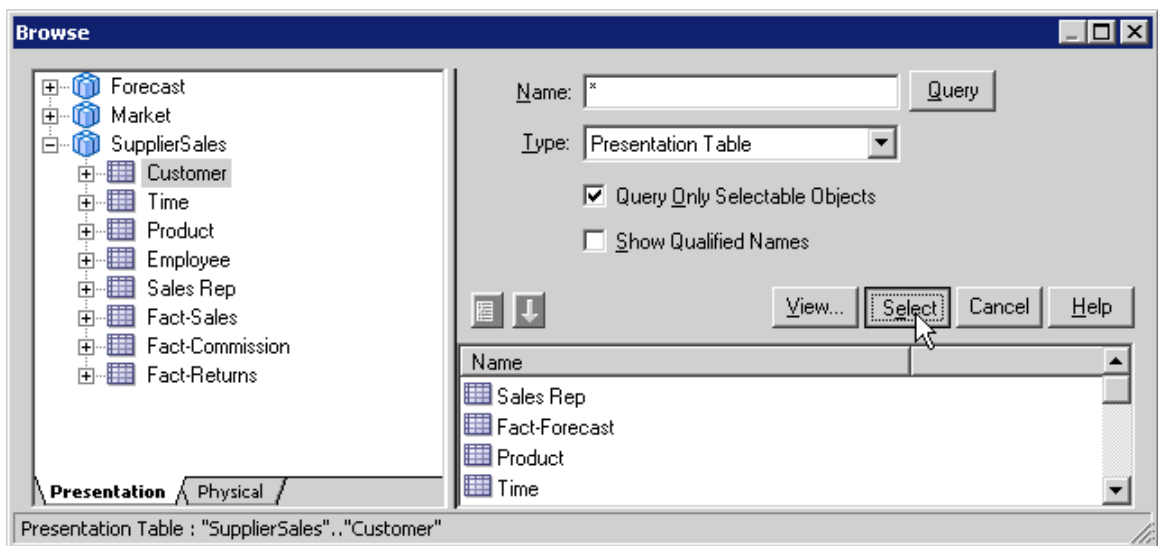
- h. Click the **Application Roles** tab and verify that the application roles you created are now visible.



2. Set permissions for a repository object. It is strongly recommended that you perform data access security tasks in the Administration Tool in online mode. If you must apply data access security in offline mode, be aware that users and application roles do not appear in the Administration Tool in offline mode unless you have first modified them in the Administration Tool in online mode.
 - a. Double-click the **SalesAssociatesRole** application role.
 - b. Click **Yes** to check it out.
 - c. Notice that the application roles and users assigned to this role are visible. Users who are members of these roles are therefore indirect members of this role as well.



- d. Click **Permissions** to open the User/Application Role Permissions dialog box.
- e. Click the **Object Permissions** tab.
- f. Click the **Name** field to open the Browse dialog box.
- g. In the left pane, verify that **Presentation** is selected at the bottom of the left pane.
- h. Select **SupplierSales** in the left pane.
- i. Select the **Customer** presentation table in the right pane, and then click **Select**. You can also double-click the object to select it.



The Customer presentation table is added to the Name field in the User/Application Role Permissions dialog box.

- j. Click **No Access**.

Type	Name	Read	Read/Write	No Access
	"SupplierSales".. "Customer"	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
Click here to add an object & set permission		<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

- k. Click **OK** to close the User/Application Role Permissions dialog box.
- l. Click **Check Out**.
- m. Click **OK** in the Application Role dialog box.
- n. Repeat the steps and set the permissions for the **Customer** presentation table to **No Access** for the **BIConsumer** role.
- o. Close **Identity Manager**.
- p. In the Presentation layer, expand **SupplierSales**.
- q. Right-click the **Customer** presentation table and select **Permission Report**. Your report should look similar to the screenshot:

Permission Report					
Type	Name		Description		
Presentation Table	"SupplierSales".. "Customer"		Customer Data		
User/Application Role	Read	Read/Write	No Access	Description	
Everyone	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>		
SalesAssociatesRole	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>		
BIConsumer	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>		

- r. Click **Cancel**.
- s. Check in the changes.
- t. Check consistency. Fix any errors or warnings before proceeding.
- u. Save the repository.
- v. Close the repository.
- w. Click **OK** when you see the "In order for your online changes to take effect..." message.
- x. Leave the Administration Tool open.

3. Check your work in Analysis Editor.
 - a. Sign in to Oracle BI as **AZIFF** with password **AlanZiff1**.
 - b. Select **AZIFF > My Account**.
 - c. Click **Roles** and **Catalog Groups**.

- d. **Alan Ziff** is a member of the following application roles:

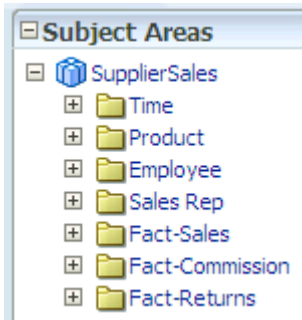
BI Author Role

BI Consumer Role

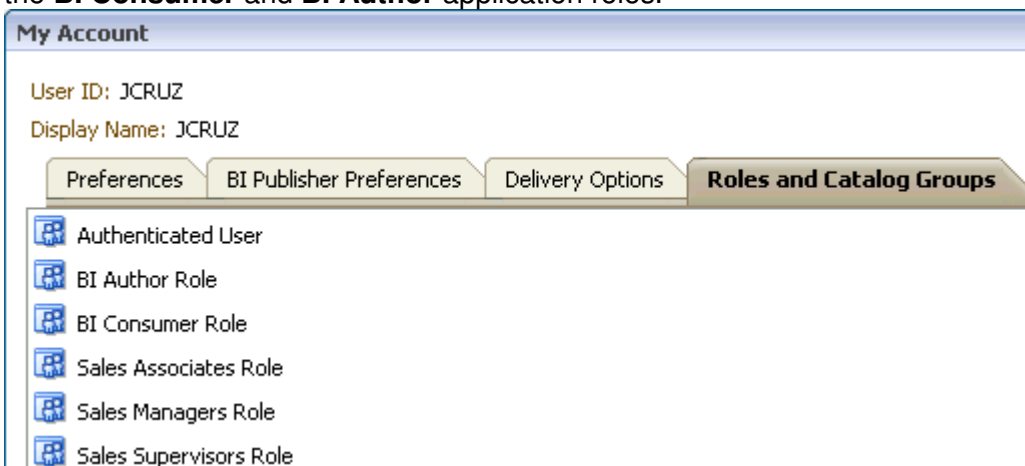
Sales Associates Role

Recall that you restricted access to the Customer presentation table for the Sales Associates and BI Consumer application roles. Given that, do you expect that Alan Ziff will have access to the Customer presentation table? Continue with the next step to verify your answer.

- e. Click **OK** to close the My Account dialog box.
 - f. Click **New > Analysis**.
 - g. Select the **SupplierSales** subject area.
 - h. Notice that the **Customer** table is not visible.



- i. Sign out of Oracle BI and log back in as **JCRUZ** with password **JoseCruz1**.
 - j. Select **JCRUZ > My Account**.
 - k. Click **Roles** and **Catalog Groups**.
 - l. **Jose Cruz** is a member of the **Sales Associates Role** application role as well as the **Sales Managers Role** and **Sales Supervisors Role** application roles, in addition to the **BI Consumer** and **BI Author** application roles.



Recall that you restricted access to the Customer presentation table for the **Sales Associates Role** application role, but not for **Sales Managers Role** or **Sales Supervisors Role**. Given that, will Jose Cruz have access to the Customer presentation table? Continue with the next step to verify your answer.

- m. Click **OK** to close the My Account dialog box.
 - n. Select **New > Analysis**.
 - o. Select the **SupplierSales** subject area.
 - p. Notice that the **Customer** table is not visible. Explanation: Jose Cruz is a member of the **Sales Managers Role** and **Sales Supervisors Role** application roles, which are members of the **Sales Associates Role** application role. When you restricted access for **Sales Associates Role**, the permissions were inherited by the other two roles. In the next set of steps you give Sales Supervisors Role explicit access to the Customer table.
 - q. Sign out of Oracle BI.
4. Set permissions to give members of Sales Supervisors Role explicit access to the Customer presentation table.
 - a. Return to the Administration Tool and open the **ABC** repository in online mode.
 - b. Open **Identity Manager**.
 - c. Click the **Application Roles** tab.
 - d. Double-click **SalesSupervisorsRole** to open the Application Role dialog box.
 - e. Click **Yes** to check out the object.
 - f. Click **Permissions**.
 - g. Click the **Object Permissions** tab.
 - h. Click the **Name** field to open the Browse dialog box.
 - i. Select **SupplierSales** in the left pane.
 - j. Double-click the **Customer** presentation table to add it to the User/Application Role Permissions dialog box.
 - k. Verify that the permission is set to **Read** (this is the default).

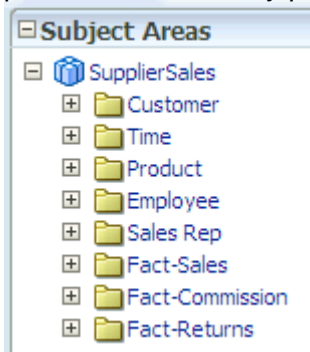
Type	Name	Read	Read/Write	No Access
	"SupplierSales".. "Customer"	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
	Click here to add an object & set permission	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

- l. Click **OK** to close the User/Application Role Permissions dialog box.
- m. Click **Check Out**.
- n. Click **OK** to close the Application Role dialog box.
- o. Close **Identity Manager**.
- p. In the Presentation layer, expand **SupplierSales**.
- q. Right-click **Customer** and select **Permission Report**. Verify that **SalesSupervisorsRole** has explicit Read access for the Customer table. Your report should look similar to the following screenshot:

Permission Report					
Type	Name	Description			
Presentation Table	"SupplierSales".. "Customer"	Customer Data			
User/Application Role	Read	Read/Write	No Access	Description	
Everyone	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>		
SalesSupervisorsRole	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>		
SalesAssociatesRole	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>		
BIConsumer	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>		

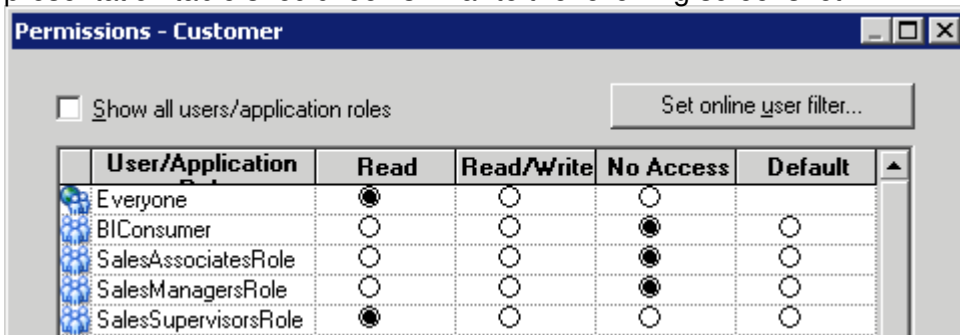
- r. Click **Cancel**.
- s. Check in changes.

- t. Check consistency. Fix any errors or warnings before proceeding.
- u. Save the repository.
- v. Close the repository.
- w. Click **OK**.
- x. Leave the Administration Tool open.
5. Check your work in Analysis Editor.
 - a. Sign in to Oracle BI as **weblogic** with password **welcome1**.
 - b. Click **New > Analysis**.
 - c. Select the **SupplierSales** subject area.
 - d. Select **Reload Server Metadata**. Only users with administrative privileges can reload server metadata.
 - e. Sign out and then sign back in as **JCRUZ** with password **JoseCruz1**.
 - f. Click **New > Analysis**.
 - g. Select the **SupplierSales** subject area.
 - h. Verify that the **Customer** table is visible. In this example, Jose Cruz is a member of both Sales Associates Role and Sales Supervisors Role. You restricted access to the Customer presentation table for Sales Associates Role, but gave explicit Read access for Sales Supervisors Role. Permissions granted explicitly to an application role take precedence over any permissions granted through other application roles.

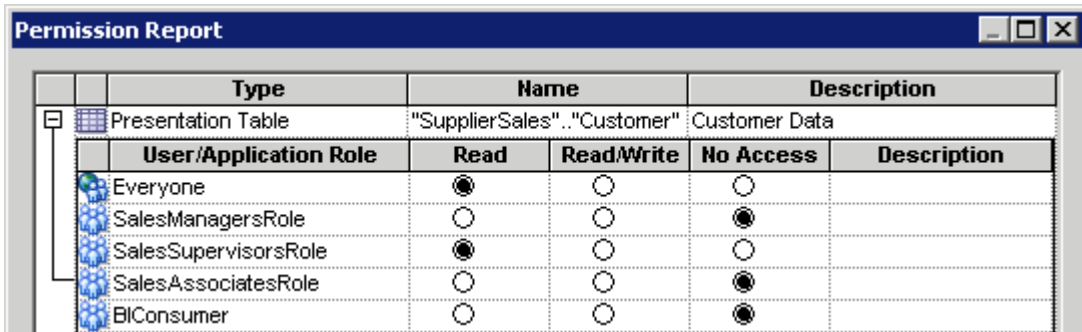


- i. Sign out of Oracle BI.
6. Set conflicting permissions for application roles at the same level in a hierarchy.
 - a. Return to the Administration Tool and open the **ABC** repository in online mode.
 - b. Double-click the **Customer** presentation table to open the Presentation Table dialog box.
 - c. Click **Check Out**.
 - d. On the **General** tab, click **Permissions** to open the Permissions dialog box.
 - e. Select **Show all users/application roles**.
 - f. Select **No Access** for the **SalesManagersRole** application role.

- g. Deselect **Show all users/application roles**. Permissions for the Customer presentation table should look similar to the following screenshot:



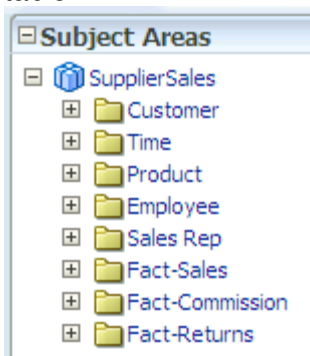
- h. Click **OK** to close the Permissions dialog box.
 i. Click **Yes** to check out the object.
 j. Click **OK** to close the Presentation Table dialog box.
 k. Right-click **Customer** and select **Permission Report**. Verify that **SalesManagersRole** has No Access for the Customer table. Your report should look similar to the following screenshot:



Jose Cruz now has conflicting permissions across two application roles. **Sales Supervisors Role** gives him read access to the Customer table. **Sales Managers Role** gives him no access to the Customer table. Given that, will Jose Cruz have access to the Customer presentation table? Continue with the next steps to verify your answer.

- l. Click **Cancel**.
 m. Check in changes.
 n. Check consistency. Fix any errors or warnings before proceeding.
 o. Save the repository.
 p. Close the repository.
 q. Click **OK**.
 r. Leave the Administration Tool open.
7. Check you work in Analysis Editor.
- Sign in to Oracle BI as **weblogic** with password **welcome1** and reload server metadata for the **SupplierSales** subject area.
 - Sign out and then sign back in as **JCRUZ** with password **JoseCruz1**.
 - Click **Analysis**.
 - Select the **SupplierSales** subject area.
 - Verify that the **Customer** table is still visible for Jose Cruz. If multiple application roles act on a user or application role at the same level, but with conflicting security

attributes, the user or application role is granted the least restrictive security attribute. In this case, the least restrictive security attribute is read access for the Customer table.



- f. Sign out of Oracle BI.

Practice 14-5: Setting Row-Level Security (Data Filters)

Goal

To set row-level security (data filters) in the repository

Scenario

Data filters provide a way to enforce row-level security rules in the repository. Data filters are set up in the repository by using the Administration Tool and are applied for a particular user or application role.

Data filters can be set for objects in both the Business Model and Mapping layer and the Presentation layer. Applying a filter on a logical object will affect all Presentation layer objects that use the object. If you set a filter on a Presentation layer object, it is applied in addition to any filters that might be set on the underlying logical objects. It is a best practice to set up data filters for particular application roles rather than for individual users.

In this practice you set a filter on the Customer presentation table so that customer data is visible only for those records where Jose Cruz or his direct reports are the sales representatives.

Time

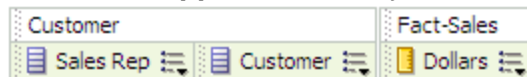
15 minutes

Tasks

1. Set a data filter for the Customer presentation table.
 - a. Return to the Administration Tool and open the **ABC** repository in online mode.
 - b. Open **Identity Manager**.
 - c. Click the **Application Roles** tab.
 - d. Double-click **SalesSupervisorsRole** to open the Application Role dialog box.
 - e. Click **Check Out**.
 - f. Click **Permissions** to open the User/Application Role Permissions dialog box.
 - g. Click the **Data Filters** tab.
 - h. Click the **Name** field to open the Browse dialog box.
 - i. Select the **SupplierSales > Customer** presentation table to return to the User/Application Role Permissions dialog box.
 - j. Click the **Data Filter** field.
 - k. Click the **Edit Expression icon** to open the Expression Builder.
 - l. Build the following expression. This sets a filter on the Customer presentation table so that customer data is visible only for those records where Jose Cruz or his direct reports are the sales representatives:
"SupplierSales"."Dim-Customer"."Sales Rep" = 'JOSE CRUZ' OR
"SupplierSales"."Dim-Customer"."Sales Rep" = 'ALAN ZIFF' OR
"SupplierSales"."Dim-Customer"."Sales Rep" = 'BETTY NEWER'
 - m. Click **OK** to close Expression Builder to return to the User/Application Role Permissions dialog box. The expression is added to the Data Filter field. The following screenshot shows only a partial view.

Type	Layer	Name	Status	Data Filter
	Presentation	"SupplierSales"."Customer"	Enabled	"SupplierSales"."Dim-Customer"."Sales Rep" =
		Click here to add an object an		

- n. Click **OK** to close the User/Application Role Permissions dialog box.
 - o. Click **OK** to close the Application Role dialog box.
 - p. Close **Identity Manager**.
 - q. Check in changes.
 - r. Save the repository.
 - s. Close the repository.
 - t. Click **OK**.
 - u. Leave the Administration Tool open.
2. Check your work in Analysis Editor.
 - a. Sign in to Oracle BI as **JCRUZ** with password **JoseCruz1**.
 - b. Select the **SupplierSales** subject area and create the following analysis:



- c. Click **Results**.
- d. Verify that customer data is visible only for those records where Jose Cruz or his direct reports are the sales representatives.

Sales Rep	Customer	Dollars
ALAN ZIFF	Chang's Mongolian Grill	\$331,217
	Globus Office	\$212,182
	Half-Shell Restaurant	\$880,096
	Times On Bay	\$104,436
BETTY NEWER	Arloi Dee	\$569,507
	Dentico's Italian Villa	\$1,728
	Felix	\$1,022,360
	Johnny's	\$3,153,640
	Satterwhite Restaurant & Ctrng	\$9,533
JOSE CRUZ	Aibonitos Restaurant	\$185,964
	Baseline Cafe	\$447,187
	Mayflower Cuisinier	\$399,553
	Peter's Pub	\$14,939

- e. Sign out of Oracle BI.

Practice 14-6: Setting Query Limits and Timing Restrictions

Goal

To manage the query environment by setting query limits in the repository

Scenario

You can manage the query environment by setting query limits (governors) in the repository for users or application roles. You may want to prevent queries from consuming too many resources by limiting how long a query can run and how many rows a query can retrieve. You also may want to regulate when individual users can query databases to prevent users from querying when system resources are tied up with batch reporting, table updates, or other production tasks. In this practice, you set the maximum rows of any query to five rows, the maximum time of any query to 1 minute, and restrict access to a database on Sunday from 12:00 AM to 7:00 AM.

It is a best practice to set query limits for particular application roles rather than for individual users.


Time

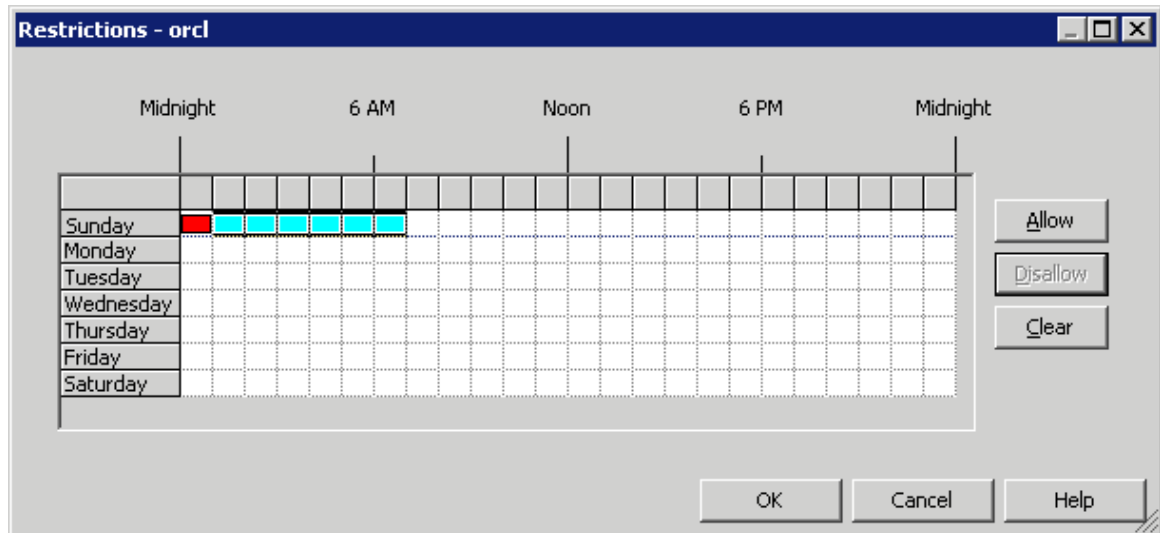
20 minutes

Tasks

1. Disallow queries that may consume too many system resources by setting query limits for the SalesSupervisorsRole application role.
 - a. Return to Fusion Middleware Control and restart Oracle BI components.
 - b. Return to the Administration Tool and open the **ABC** repository in online mode.
 - c. Open **Identity Manager**.
 - d. Click the **Application Roles** tab.
 - e. Double-click **SalesSupervisorsRole** to open the Application Role dialog box.
 - f. Click **Check Out**.
 - g. Click **Permissions**.
 - h. Click the **Query Limits** tab.
 - i. Locate the **orcl** database and change its **Max Rows** value to **5**. This specifies the maximum number of rows that each query can retrieve from the orcl database for members of the **SalesSupervisorsRole** application role.
 - j. In the Status Max Rows column, select **Enable** from the drop-down list.

Type	Database	Restrict	Status Max Rows	Max Rows
	xls_quota		Ignore	100000
	orcl		Enable	5
	Essbase		Ignore	100000

2. Restrict the time period for which users can access specified repository resources.
 - a. Click the ellipsis button  in the **Restrict** column of the **orcl** database.
 - b. Highlight the blocks from Sunday at midnight to 7 AM.
Hint: Click the first box and drag to the last block.
 - c. Click the **Disallow** button.

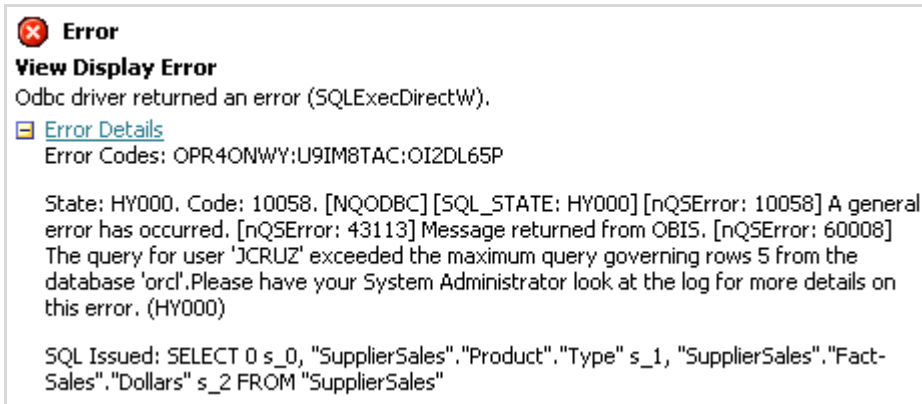


If a time period is not highlighted, the access rights remain unchanged. If access is allowed or disallowed explicitly to one or more groups, the user is granted the least restrictive access for the time periods that are defined.

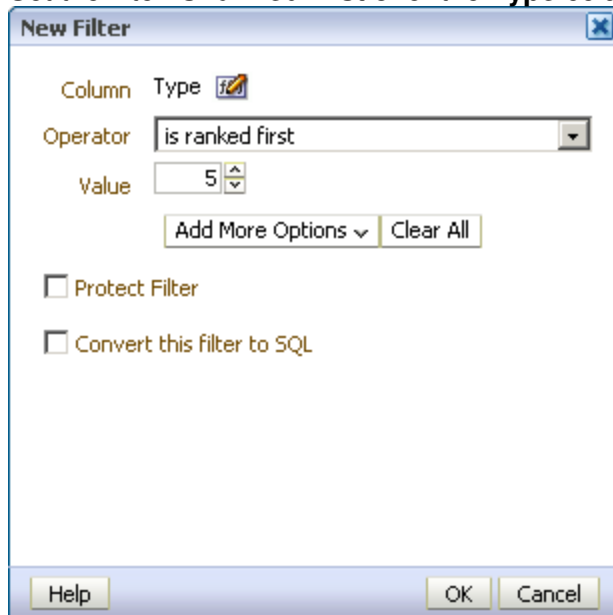
- d. Click **OK** to close the Restrictions dialog box.
 - e. Click **OK** to close the User/Application Role Permissions dialog box.
 - f. Click **OK** to close the Application Role dialog box.
 - g. Close **Identity Manager**.
 - h. Check in changes
 - i. Save the repository.
 - j. Close the repository.
 - k. Click **OK**.
 - l. Leave the Administration Tool open.
3. Check your work in Analysis Editor.
 - a. Sign in to Oracle BI as **JCRUZ** with password **JoseCruz1**.
 - b. Select the **SupplierSales** subject area and create the following analysis:

Product	Fact-Sales
Type	Dollars
 - c. Click **Results**.

- d. Expand **Error Details** to view message. The error message states that the maximum row limit is exceeded.



- e. Click the **Criteria** tab.
- f. Set the filter **is ranked first 5** for the **Type** column and click **Results**.



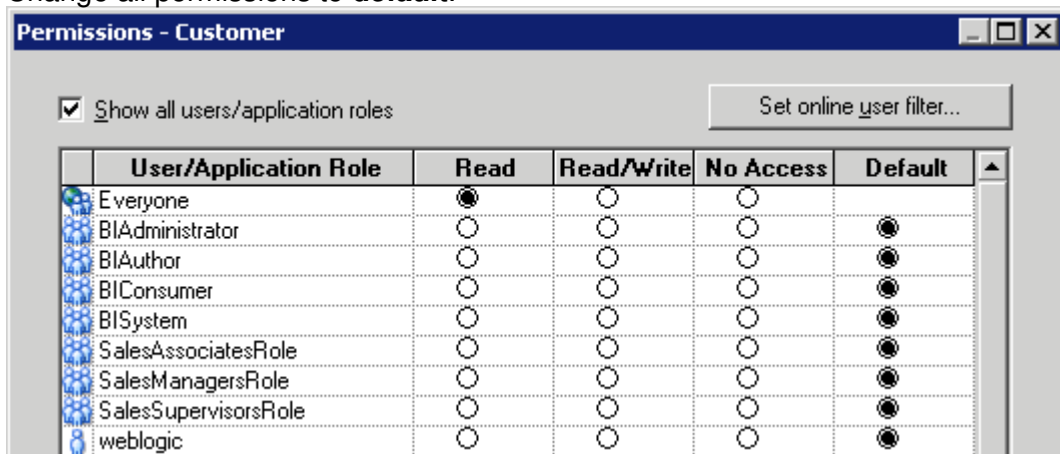
- g. Verify that results are now returned.

Type	Dollars
Baking	\$4,925,521
Beef	\$4,916,016
Beverage	\$4,398,107
Bread	\$1,578,743
Cereal	\$1,309,071

Because of the filter, the query returned only five rows, which does not exceed the maximum query limit.

- h. Sign out of Oracle BI.
4. Deactivate the restrictions you set.
- Return to Fusion Middleware Control and restart Oracle BI components.
 - Return to the Administration Tool and open **ABC** repository in online mode.
 - Open **Identity Manager**.
 - Click the **Application Roles** tab.

- e. Double-click the **SalesSupervisorsRole** application role.
- f. Click **Check Out**.
- g. Click the **Permissions** button.
- h. Click the **Query Limits** tab.
- i. In the **orcl** row, set **Max Rows** to **100000** and select **Ignore** for **Status Max Rows**.
- j. Click **Restrict** and clear the restriction.
- k. Click **OK**.
- l. Close all dialog boxes and Identity Manager.
- m. Expand **SupplierSales** subject area in the Presentation layer.
- n. Double-click the **Customer** presentation table.
- o. Click **Check Out**.
- p. Click **Permissions**.
- q. Select **Show all users/application roles**.
- r. Change all permissions to **default**.



User/Application Role	Read	Read/Write	No Access	Default
Everyone	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	
BIAdministrator	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
BIAuthor	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
BIConsumer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
BISystem	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
SalesAssociatesRole	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
SalesManagersRole	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
SalesSupervisorsRole	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
weblogic	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

- s. Click **OK**.
- t. Click **Check Out**.
- u. Click **OK** to close the Presentation Table dialog box.
- v. Check in the changes.
- w. Check consistency.
- x. Save the repository.
- y. Close the repository.
- z. Click **OK**.
- aa. Leave the Administration Tool open.

Practices for Lesson 15: Cache Management

Chapter 15

Overview of Practices for Lesson 15

Lesson Overview

In these practices, you will enable query caching and inspect cache entries by using Cache Manager in the Administration Tool.

Practice 15-1: Enabling Query Caching

Goal

To enable query caching and inspect cache entries using Cache Manager

Scenario

You use Fusion Middleware Control to enable query caching and then use Cache Manager in the Administration Tool to inspect cache entries and analyze cache hits and non-hits.

Time

25 minutes

Tasks

1. Set logging levels for users to allow you to track queries in query logs.
 - a. Open the **ABC** repository in online mode with username **weblogic** and password **welcome1**.
 - b. Select **Manage > Identity** to open Identity Manager.
 - c. In the left pane, select **Identity Management > BI Repository**.
 - d. Select **Action > Set Online User Filter**.
 - e. Enter an asterisk in the field and click **OK** to retrieve users.
 - f. In the right pane, double-click **JCRUZ** to open the User properties dialog box.
 - g. Set the logging level to **2** and click **OK**.
 - h. Click **Check Out**.
 - i. Repeat to set the logging level to 2 for **AZIFF**.
 - j. Close Identity Manager.
 - k. Check in the changes.
 - l. Save the repository.
 - m. Close the repository.
 - n. Click **OK**.
2. Enable query caching.
 - a. Return to Fusion Middleware Control and log in as **weblogic** with password **welcome1**.
 - b. In the left pane, expand **Business Intelligence**.
 - c. Click **coreapplication**.
 - d. Click **Capacity Management**.
 - e. Click the **Performance** tab.
 - f. Click **Lock and Edit Configuration**.

- g. Select **Cache Enabled**.

Overview **Capacity Management** Diagnostics Security Deployment

Metrics Availability Scalability **Performance**

Performance Options

Use this page to tune the performance of this BI Instance.

Enable BI Server Cache

Enabling the server cache can greatly improve performance by enabling users who share data visibility to retrieve row sets from queries that have already been run at the cost of the possibility of seeing stale data.

☒ Cache enabled

Maximum cache entry size

Maximum cache entries

- h. Click **Apply** to activate the changes.
- i. Click **Activate Changes**.
- j. Restart **Oracle BI components** to apply recent changes.
3. Create and run an analysis.
- a. Sign in to Oracle BI as **JCRUZ** with password **JoseCruz1**.
- b. Create the following analysis in the SupplierSales subject area:

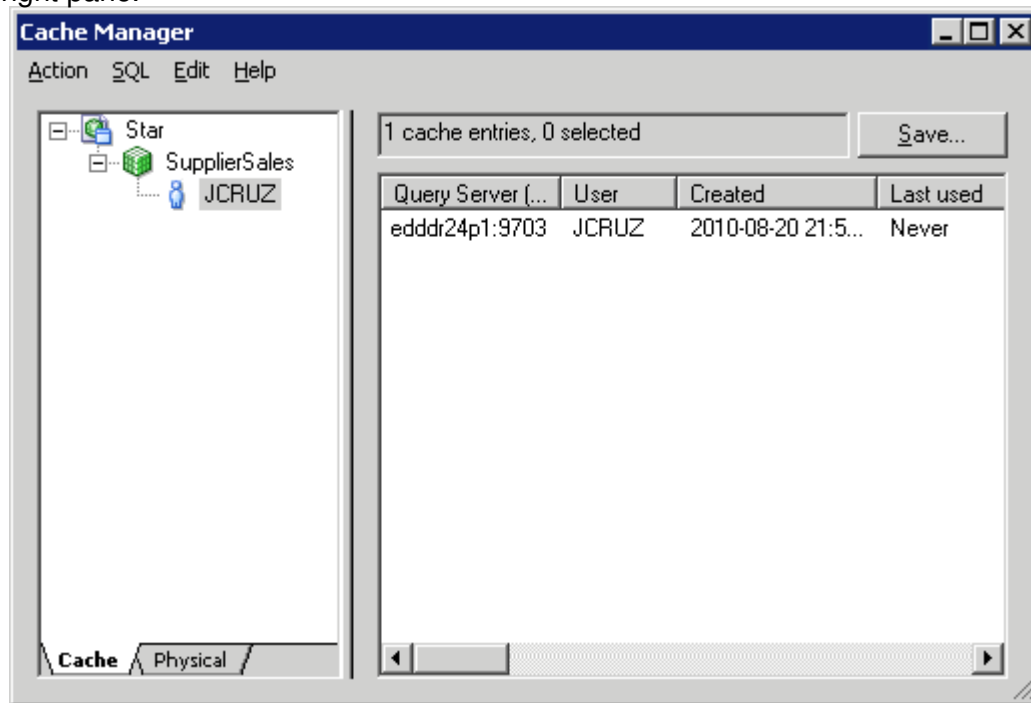
Product
Type Generic Price

- c. Click **Results**.

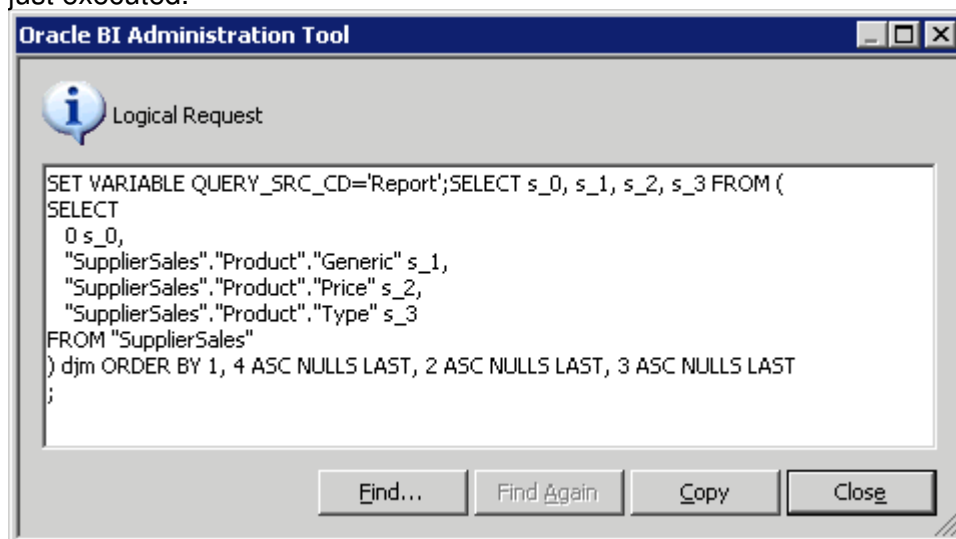
Type	Generic	Price
Baking	Beef Bouillon	\$17.23
	Biscuit Mix	\$22.37
	Breading Mix	\$22.24
	Brown Sugar	\$38.01
	Egg Substitute	\$16.84
	Iodized Salt	\$8.00
	Pancake Mix	\$18.60
	Powdered Sugar	\$15.39
	Stuffing Mix	\$16.66
	Vanilla Extract	\$10.60
	White Flour	\$9.18
Beef	Breakfast Sausage Links	\$30.13
	Canned Beef	\$26.89
	Chorizo Sausage	\$23.98
	Deli-Style Pastrami	\$38.44
	Frozen Sausage	\$19.82
	Frozen Steaks	\$75.97
	Frozen Veal Cutlets	\$15.33
	Hamburger Patties	\$144.58
	Lean Ground Beef Patties	\$19.63
	Wieners	\$16.39
Beverage	Coconut Cream Drink Mix	\$79.10
	Orange Juice	\$16.05

- d. Sign out of Oracle BI.

4. Open the Cache Manager and verify that the query is listed as a cache entry.
 - a. Open the **ABC** repository in online mode and select **Manage > Cache** to open Cache Manager.
 - b. In the left pane, expand **Star > SupplierSales**. By selecting the appropriate leaf of the tree in the left pane, you can limit the cache entries that appear in the right pane. The Cache Manager allows you to view cache entries by repository, subject area, and user.
 - c. Click **JCRUZ** in the left pane. All cache entries associated with this user appear in the right pane.

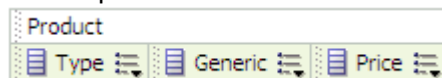


- d. Select the cache entry (there should only be one).
- e. Select **SQL > Show**.
- f. By inspecting the SQL you can verify that this is the cache entry for the query that you just executed.

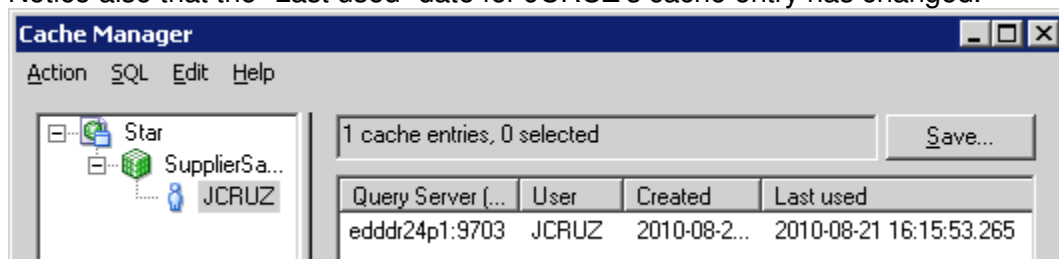


- g. Click **Close** to close the SQL.

- h. With this entry still selected, scroll to the right to view the columns. Notice that values for the **Created** and **Last used** columns are not the same. The Created column shows the date and time that the cache entry was created. Write down this date and time. The Last used column shows that the cache entry has never been used. This indicates that the results for this query were returned directly from the database and not derived from an existing query.
 - i. Leave the Cache Manager open.
 - j. Leave the ABC repository open in online mode.
5. Inspect the cache file and ensure that the results of the query were stored as a file in the cache directory. The modified time of the file should coincide with the time that you originally created the request (the time you recorded above).
 - a. Navigate to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplicatio**
n_obis1\cache. The file in this directory is the cache file that resulted from your query.
 - b. Verify that the modified time of this file is the same as the time that you created the request.
 6. Create and run the same analysis logged in as a different user.
 - a. Sign in to Oracle BI as **AZIFF** with password **AlanZiff1**.
 - b. Create the same analysis in the SupplierSales subject area that you created earlier and inspect the results.



- c. Click **Results**.
7. In response to AZIFF's request, determine whether a new cache entry was made in the Cache Manager and filed in the Cache directory. If a new entry was made in the Cache Manager and filed in the directory, then an existing cache was not used to satisfy his request.
 - a. Return to the **ABC** repository open in online mode
 - b. Select **Action > Refresh** in the **Cache Manager**.
 - c. Notice that no cache entry is listed with AZIFF as the user.
 - d. Notice also that the "Last used" date for JCRUZ's cache entry has changed.



AZIFF's query was fulfilled by the cache entry that resulted from JCRUZ's earlier query. Because AZIFF's request was identical to the cache, the server used the cache (cache hit) instead of processing against the database. The last used date is updated to reflect this.

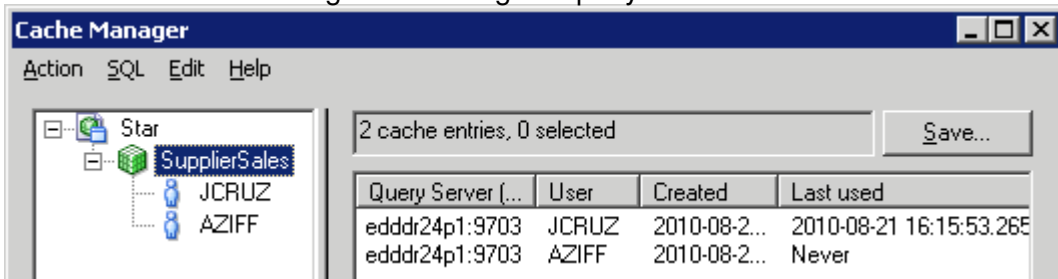
- e. Leave Cache Manager open.
- f. Leave the repository open in online mode.
- g. Navigate to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplicatio**
n_obis1\cache.

- h. Notice that no new cache file is created as a result of AZIFF's request. This is because AZIFF's request was satisfied by an existing cache entry.
8. Create and run a new analysis to illustrate a non-cache hit using a dimension-only query. In all of the steps that follow, be sure to create a *new* analysis when instructed to ensure correct cache results.
 - a. Return to Analysis Editor where you should still be logged in as **AZIFF** with password **AlanZiff1**.
 - b. Create and run the following new analysis in the SupplierSales subject area:



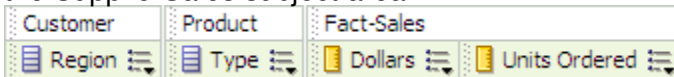
Notice that this is a dimension-only query, meaning that no fact or measure is included in the query. Notice also that this query contains a subset of the columns in the previous query.

- c. Leave Oracle BI open.
9. Determine whether a new cache entry was made in the Cache Manager.
 - a. Return to the **ABC** repository open in online mode.
 - b. Select **Action > Refresh** in the **Cache Manager**.
 - c. Select **Star > SupplierSales** in the left pane.
 - d. Notice the new cache entry listed for AZIFF. The presence of a new entry indicates that there was no cache hit against the original query.



Explanation: If a query is dimension only, meaning that no fact or measure is included in the query, only an exact match of the projection columns of the cached query will hit the cache. This behavior prevents false positives when there are multiple logical sources for a dimension table. In this example, only two columns, Type and Generic, matched the columns of the cached query, which included three columns: Type, Generic, and Price.

- e. Leave Cache Manager open.
- f. Leave the ABC repository open in online mode.
10. Create and run a new analysis.
 - a. Return to Analysis Editor and create and run the following new analysis as AZIFF in the SupplierSales subject area:



- b. Refresh the Cache Manager and verify that a new cache entry is created for this analysis, indicating that there was no cache hit.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	Never

- c. Return to Analysis Editor and create and run the following new analysis. Notice that this analysis contains all the columns from the previous query, plus one additional table and column, Time and Year.

Time	Customer	Product	Fact-Sales
Year	Region	Type	Dollars
			Units Ordered

- d. Return to Cache Manager and select **Action > Refresh**. Notice that a new cache entry is listed for AZIFF, indicating there was no cache hit against the previous query.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	Never
AZIFF	2010-03-11 01:12:27.921	Never

Explanation: The set of logical tables must match. To qualify as a cache hit, all incoming queries must have the same set of logical tables as the cache entry. In this example the Time logical table did not exist in the cached query.

- e. Return to Analysis Editor and create and run the following new analysis in the SupplierSales subject area:

Customer	Product	Fact-Sales
Region	Type	Dollars per Units Ordered

- f. Refresh Cache Manager and notice that the “Last used” column is updated for AZIFF’s second query, indicating a cache hit.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	2010-03-11 01:16:01.171
AZIFF	2010-03-11 01:12:27.921	Never

Explanation: Columns in the SELECT list can be composed of expressions on the columns of the cached queries. The Oracle BI Server can calculate expressions on cached results to answer the new query, but all the columns have to be in the cached result. In this example, Dollars per Units Ordered (dollars/units ordered) can be computed from Dollars and Units Ordered in the cached query that was hit.

- g. Return to Analysis Editor and create and run the following new analysis and filter in the SupplierSales subject area:

Customer	Product	Fact-Sales
Region	Type	Dollars
		Units Ordered

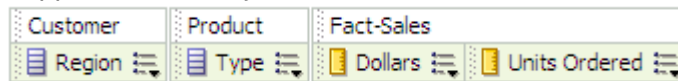
Region is equal to / is in East

- h. Refresh **Cache Manager** and watch the Last used column for the second AZIFF query. The column is updated, indicating a cache hit.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	2010-03-11 01:25:27.250
AZIFF	2010-03-11 01:12:27.921	Never

Explanation: For the query to qualify as a cache hit, the WHERE clause constraints must be either equivalent to the cached results, or a subset of the cached results. In this example, a query requesting fewer elements of an IN list cached query qualify for a cache hit. The WHERE clause in the cached query is `WHERE Region in ('East', 'West', 'Central')`. The WHERE clause in the query that hit the cache is a subset of the cached results: `WHERE Region in ('East')`.

- i. Return to Analysis Editor and create and run the following new analysis and filter in the Supplier Sales subject area where Dollars is less than or equal to \$2 million:



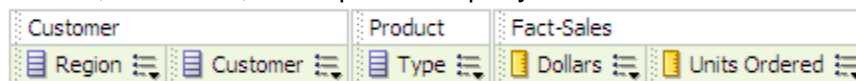
Y Dollars is less than or equal to 2000000

- j. Refresh Cache Manager and watch the “Last used” column for the second AZIFF query. The column is updated, indicating a cache hit.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	2010-03-11 01:44:45.140
AZIFF	2010-03-11 01:12:27.921	Never

Explanation: Again, the WHERE clause constraints are a subset of the cached results. In this example, the WHERE clause contains a logical subset of a literal comparison.

- k. Return to Analysis Editor and create and run the following analysis, which adds a new column, Customer, to the previous query:



- l. Refresh Cache Manager. Notice that a new cache entry is created.

User	Created	Last used
JCRUZ	2010-03-11 00:55:15.406	2010-03-11 01:00:03.312
AZIFF	2010-03-11 01:02:57.187	Never
AZIFF	2010-03-11 01:08:59.015	2010-03-11 01:44:45.140
AZIFF	2010-03-11 01:12:27.921	Never
AZIFF	2010-03-11 01:57:58.078	Never

Explanation: All of the columns in the SELECT list of a new query have to exist in the cached query in order to qualify for a cache hit, or they must be able to be calculated from the columns in the query. In this example, the Customer column did not exist in the cached query.

- m. Sign out of Oracle BI.
- n. Leave Cache Manager and the repository open for the next practice.
- o. Navigate to **D:\bi\instances\instance1\bifoundation\OracleBIServerComponent\coreapplicatio**
n_obis1\cache and verify there are now five files, one for each cache entry.

Practice 15-2: Modifying Cache Parameters

Goal

To use Fusion Middleware Control and the NQSConfig.ini file to modify cache parameters

Scenario

You use the Cache Manager to inspect the cache parameters. Then you modify the number of rows per cache, as well as the number of cache entries allowed. In addition to modifying cache parameters, you make some tables noncacheable.

Outcome

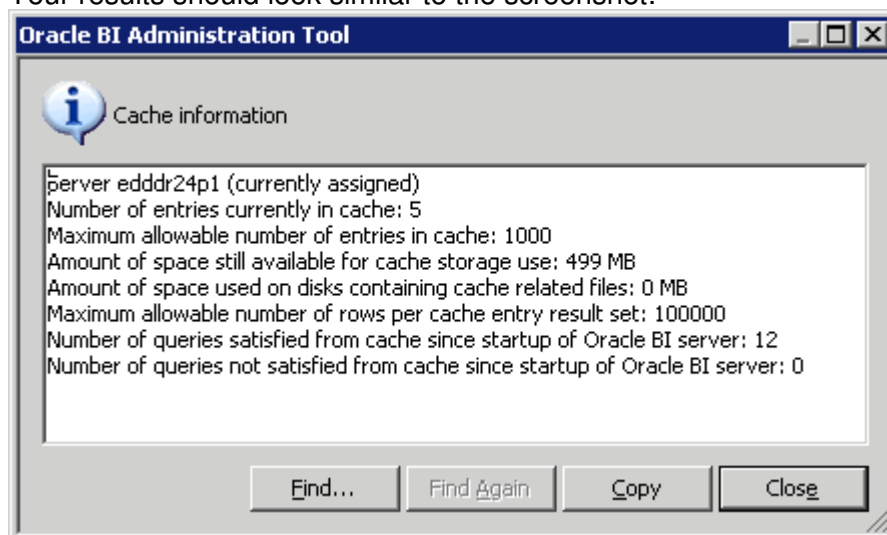
A decrease in the number of rows per cache, as well as a decrease in the number of cache entries allowed

Time

20 minutes

Tasks

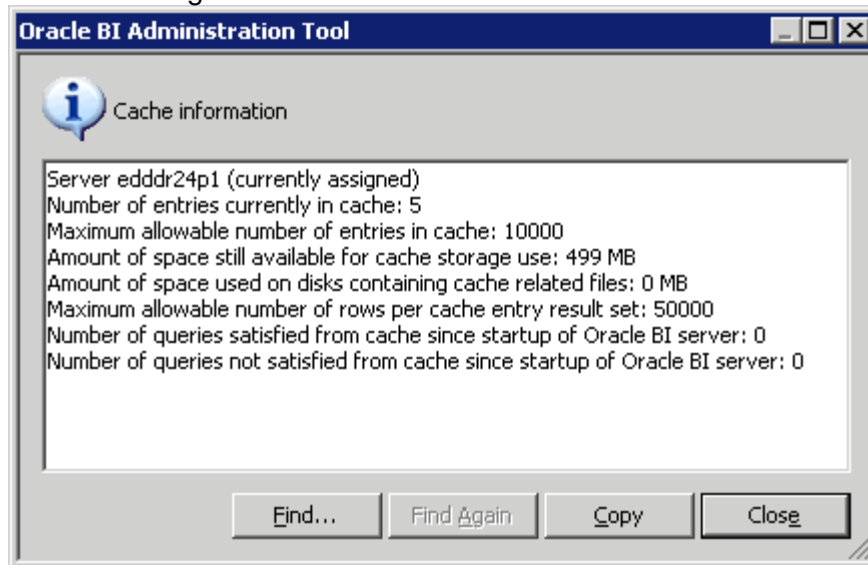
1. Display global cache information in the Cache Manager.
 - a. In the Cache Manager, select **Action > Show Info** to display global cache information. Your results should look similar to the screenshot:



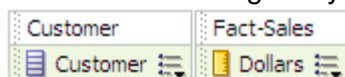
- b. **Number of entries currently in cache** displays the current number of entries in your global cache. These entries may relate to multiple repositories.
 - c. **Maximum allowable number of entries in cache** displays the maximum number of entries that can be in the cache.
 - d. **Amount of space still available for cache storage use** displays the amount of space, in megabytes, still available for cache storage.
 - e. **Amount of space used on disks containing cache related files** displays the total amount of space, in megabytes, used on the disk containing cache-related files (not just space used for the cache-related files). In this case, the amount of used space is less than 1 MB.

- f. **Maximum allowable number of rows per cache entry result set** displays the maximum number of rows allowed for each cache entry's result set. This number is set in the MAX_ROWS_PER_CACHE_ENTRY parameter in the NQSConfig.INI file.
 - g. **Number of queries satisfied from cache since startup of Oracle BI Server** displays the number of cache hits since the last time the Oracle BI Server was started.
 - h. **Number of queries not satisfied from cache since startup of Oracle BI Server** displays the number of cache misses, since the last time the Oracle BI Server was started.
 - i. Click **Close**.
 - j. Close the Cache Manager.
 - k. Close the repository.
2. Use Fusion Middleware Control to set query cache parameters. You can use Fusion Middleware Control to set the maximum number of cache entries in the query cache, as well as the maximum size for a single cache entry.
 - a. Return to Fusion Middleware Control, which should still be open. If it is not open, enter the **http://localhost:7001/em** and log in to Fusion Middleware Control as **weblogic** with password **welcome1**.
 - b. In the left pane, expand **Business Intelligence**.
 - c. Click **coreapplication**.
 - d. Click **Capacity Management**.
 - e. Click the **Performance** tab.
 - f. Click **Lock and Edit Configuration**.
 - g. Change **Maximum cache entries** from 1000 to **10000**.
 - h. **Apply** and **activate** your changes.
 - i. Leave Fusion Middleware Control open, but do not restart to apply recent changes at this time.
 3. Use NQSConfig.ini to view and manually edit additional query cache parameters.
 - a. Navigate to **D:\bi\instances\instance1\config\OracleBIServerComponent\coreapplication_obis1**.
 - b. Before making changes to NQSConfig.ini, make a copy of the file and paste it in the same directory.
 - c. Open **NQSConfig.ini**.
 - d. Navigate to the **CACHE** section.
 - e. Make the following modifications:
 - f. Modify the MAX_ROWS_PER_CACHE_ENTRY parameter as follows:
MAX_ROWS_PER_CACHE_ENTRY = 50000
 This parameter controls the maximum number of rows for any cache entry. Limiting the number of rows is a useful way to avoid using up the cache space with runaway queries that return large numbers of rows. If the number of rows a query returns is greater than the value specified in the MAX_ROWS_PER_CACHE_ENTRY parameter, the query is not cached.
 - g. Notice the DATA_STORAGE_PATH parameter. This parameter specifies one or more directories for query cache storage, and the maximum size for each storage directory. These directories are used to store the cached query results and are accessed when a cache hit occurs.

- h. Notice that the **MAX_CACHE_ENTRIES** parameter is changed to 10000. Changes made to cache configuration in Fusion Middleware Control are written to this file.
 - i. Save and close **NQSConfig.INI**.
 - j. Return to Fusion Middleware Control and restart Oracle BI components.
4. Validate your changes in the Administration Tool.
- a. Return to the Administration Tool and open the **ABC** repository in online mode.
 - b. Select **Manage > Cache**.
 - c. Select **Action > Show Info** and confirm that changes are applied. Maximum allowable number of entries in cache should now equal **10000**. Maximum allowable number of rows per cache entry result should now equal **50000**. Your results should look similar to the following screenshot:

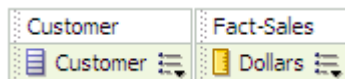


- d. Click **Close**.
 - e. Close the Cache Manager.
5. Make some tables noncacheable, that is, if a request is made against them, there are no cache entries.
- a. In the Physical layer, double-click the **orcl.SUPPLIER2.D1_CUSTOMER2** table.
 - b. Click **Check Out**.
 - c. Click the **General** tab.
 - d. Deselect the **Cacheable** check box.
 - e. Click **OK**.
 - f. Repeat this process for the **D1_ORDERS2** table.
 - g. Check in the changes and save the repository. You do not need to check consistency.
 - h. Leave the repository open.
6. Test your work.
- a. Log in to Oracle BI as **JCRUZ** with password **JoseCruz1**.
 - b. Create the following analysis in the SupplierSales subject area:

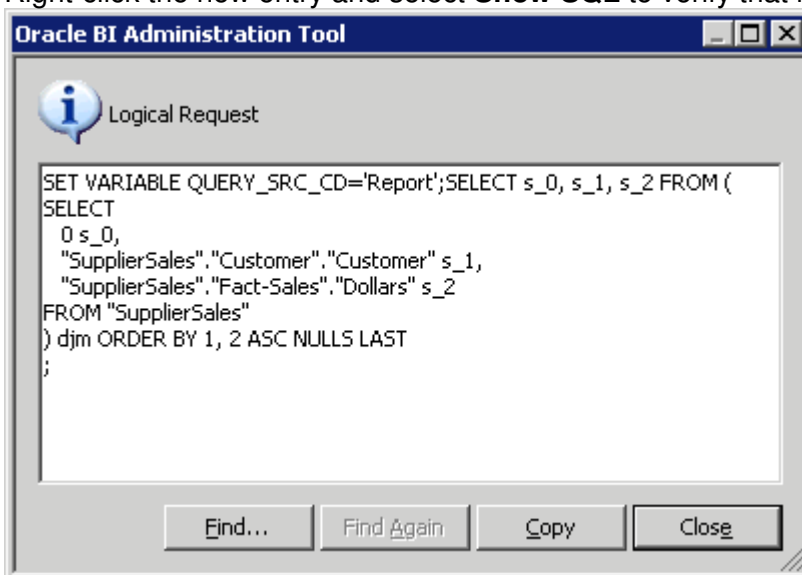


- c. Click **Results**.

- d. Leave Analysis Editor open.
- e. Return to the repository open in online mode in the Administration Tool and open the Cache Manager.
- f. Verify that you do not see a cache entry for the analysis you just executed.
- g. Close the Cache Manager.
7. Make the tables cacheable.
 - a. In the Physical layer, double-click the **D1_CUSTOMER2** table.
 - b. Click **Check Out**.
 - c. Select the **General** tab.
 - d. Select the **Cacheable** check box.
 - e. Click **OK**.
 - f. Repeat this process for the **D1_ORDERS2** table.
 - g. Check in the changes and save the repository. You do not need to check for consistency.
 - h. Leave the repository open.
8. Test your work.
 - a. Return to Analysis Editor.
 - b. Sign out and sign back in as **weblogic** with password **welcome1**.
 - c. Click **Administration**.
 - d. Under Maintenance and Troubleshooting click **Reload Files and Metadata**.
 - e. Create and run a new analysis using the same columns from the SupplierSales subject area:

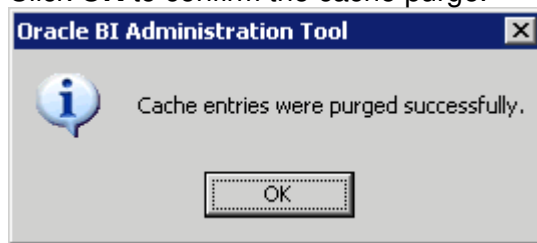


- f. Return to the repository open in online mode in the Administration Tool and open the Cache Manager.
- g. Verify that there is a new cache entry for the analysis executed by weblogic.
- h. Right-click the new entry and select **Show SQL** to verify that it is the expected query:



- i. Click **Close**.

9. Alter how the Cache Manager displays information.
 - a. Select **Edit > Options**.
 - b. Notice that you can deselect columns and use the **Up** and **Down** buttons to change the column order.
 - c. Click **Cancel**.
10. Purge cache entries.
 - a. Right-click the cache entry for **JCRUZ** and select **Purge**, or select **Edit > Purge**. In this pane, it is possible to purge a single cache entry, multiple entries, or all entries.
 - b. Click **OK** to confirm the cache purge.



- c. Click the **Physical** tab at the bottom of the left pane.
- d. In the left pane, expand **orcl > Supplier2**.
- e. Select the **D1_PRODUCTS** table. A message appears on the right stating that all associated cache entries will be purged for this table. In this pane, it is possible to delete cache entries for a single table, multiple tables, or the entire schema.

All associated cache entries will be purged for the following tables on server edddr24p1:9703:
"orcl"."SUPPLIER2"."D1_PRODUCTS"

- f. Select **Edit > Purge**.
- g. Click **OK** to confirm the cache purge.
- h. Click the **Cache** tab at the bottom of the left pane.
- i. Verify that the **AZIFF** cache entries are deleted.
- j. Close the Cache Manager.
- k. Leave the repository open in online mode.

Practice 15-3: Seeding the Cache

Goal

To create an analysis and an agent to seed the Oracle BI Server cache

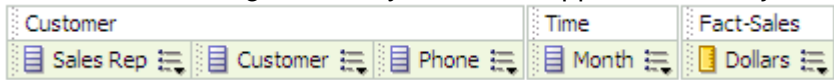
Scenario

You have identified requests that are used frequently by sales representatives. To improve performance, you want to seed the cache with this data. In this practice, you create and save a query to populate the cache, and then create an agent to seed the cache. During this process, you use a programmatic ODBC call to purge the cache.

Time

15 minutes

Tasks

1. Create a query to seed the cache.
 - a. Return to Analysis Editor where you should still be signed in as **weblogic** with password **welcome1**.
 - b. Create the following new analysis in the SupplierSales subject area:
 - c. Save the request as **Cache Seed** in **My Folders**.
2. Use an ODBC procedure to purge the cache before seeding the cache. The Oracle BI Server provides ODBC-extension functions for the Oracle BI Administrator to use for purging cache entries. Some of these functions are particularly useful for embedding in an Extract, Transform, and Load (ETL) task. For example, after a nightly ETL is performed, the entire Oracle BI Server cache can be purged. If only the fact table was modified, only cache related to that table can be purged. In some cases, you may need to purge the cache entries associated with a specific database. **Note:** Only Oracle BI Administrators have the right to purge the cache. Therefore, scripts that call these ODBC-extension functions must run under an Oracle BI Administrator login ID. Also, this is not a required step for seeding the cache. It is provided here for training purposes only.
 - a. Click **Administration** on the top of the screen.
 - b. Under Maintenance and Troubleshooting, click **Issue SQL** to issue SQL directly to the Oracle BI Server.

- c. Enter the following command:

Call SAPurgeAllCache ()

Issue SQL

Enter a SQL statement to issue directly against Oracle BI Server. This page is for testing Or
entered here into an Oracle BI Request.

Call SAPurgeAllCache()

Issue SQL Logging Level **Default** ☒ Use Oracle BI Presentation Services Cache

- d. Click **Issue SQL**.

- e. Verify that the operation succeeded.

RESULT_CODE	RESULT_MESSAGE
integer	varchar
1	[59118] Operation SAPurgeAllCache succeeded!

- f. Click **Home** to navigate to the Oracle BIEE Home page.

- g. Return to the ABC repository, which should still be open in online mode.

- h. Open the **Cache Manager** and verify that there are no cache entries.

- i. Leave the Cache Manager open.

3. Create and schedule an Agent to seed the cache with the saved query. It is common to schedule an Agent to run immediately after a daily load to reseed the cache. For example, the data warehouse is loaded at midnight and the cache is purged during or after the load, then the cache is re-seeded by an Agent. In this example, you run the Agent immediately.

- a. Return to the **Oracle BIEE Home** page.

- b. Under Actionable Intelligence, click **Agent**.

- c. Click the **Schedule** tab.

- d. Set Frequency to **Once**.

General **Schedule** Condition Delivery Content Recipients Destinations Actions

When do you want the Agent to be scheduled to run?

Enabled ☒

Frequency **Once**

Start 08/22/2010 12:12:00 PM Default

Re-run Agent Every ☐ 1 Minutes

Until 11:59:00 PM Default

- e. Click the **Delivery Content** tab.

- f. Enter **Cache Seed** in the Subject field.

- g. Verify that Content is set to **Analysis**.

- h. Click **Browse**.

- i. Select the **Cache Seed** request under My Folders and click **OK**. The request is added to the Agent.

Specify the content to deliver with the Agent

Subject: Cache Seed

Content: Analysis [Browse... Customize... Clear]

/My Folders/Cache Seed

Format: (Device default)

Delivery: ☒ Deliver results directly ☐ Deliver as attachment

- j. Click the **Destinations** tab.
- k. Select the **Oracle BI Server Cache** check box.
- l. Deselect all other check boxes.

Specify where this Agent will be delivered.

User Destinations: ☐ Home Page and Dashboard ☐ Devices

☒ Active Delivery Profile ☒ Specific Devices (will override a user's Active Delivery Profile)

☐ Email ☐ Pager ☐ Digital Phone ☐ Handheld Device

System Services: ☒ Oracle BI Server Cache (For seeding cache)

- m. Save the Agent in My Folders as **Cache Seed Agent**. When you save the Agent it is run immediately by the Oracle BI Scheduler.
4. Check your work.
- a. Return to the Administration Tool. The ABC repository should still be open in online mode.
 - b. In the Cache Manager, select **Action > Refresh**.
 - c. Ensure that there is an entry in the cache for the weblogic user. Right-click the entry and select **Show SQL** to verify that it is the expected query. You have successfully seeded the cache using an Agent. The only difference between cache seeding agents and other agents is that they clear the previous cache automatically and do not appear on the dashboard as Alerts. Notice that cache seeding agents only purge exact match queries, so stale data may still exist. Your caching strategy should always include cache purging, because agent queries do not address ad hoc queries or drills.
 - d. Click **Close** to close the SQL window.
 - e. Close **Cache Manager**.
 - f. Close the repository.
 - g. Click **OK**.
 - h. Leave the Administration Tool open.

