

SUN SEEBEYOND

eBAM™ STUDIO USER'S GUIDE

Release 5.1.3



Copyright © 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved. Sun Microsystems, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at <http://www.sun.com/patents> and one or more additional patents or pending patent applications in the U.S. and in other countries. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties. Sun, Sun Microsystems, the Sun logo, Java, Sun Java Composite Application Platform Suite, SeeBeyond, eGate, eInsight, eVision, eTL, eXchange, eView, eIndex, eBAM, eWay, and JMS are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon architecture developed by Sun Microsystems, Inc. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd. This product is covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

Copyright © 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés. Sun Microsystems, Inc. détient les droits de propriété intellectuelle relatifs à la technologie incorporée dans le produit qui est décrit dans ce document. En particulier, et ce sans limitation, ces droits de propriété intellectuelle peuvent inclure un ou plus des brevets américains listés à l'adresse <http://www.sun.com/patents> et un ou les brevets supplémentaires ou les applications de brevet en attente aux Etats - Unis et dans les autres pays. L'utilisation est soumise aux termes de la Licence. Cette distribution peut comprendre des composants développés par des tierces parties. Sun, Sun Microsystems, le logo Sun, Java, Sun Java Composite Application Platform Suite, Sun, SeeBeyond, eGate, eInsight, eVision, eTL, eXchange, eView, eIndex, eBAM et eWay sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd. Ce produit est couvert à la législation américaine en matière de contrôle des exportations et peut être soumis à la réglementation en vigueur dans d'autres pays dans le domaine des exportations et importations. Les utilisations, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers les pays sous embargo américain, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exhaustive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

Part Number: 820-0944-10

Version 20070417100756

Contents

List of Figures	8
-----------------	---

List of Tables	12
----------------	----

Chapter 1

Introduction	13
--------------	----

About eBAM Studio	13
-------------------	----

What's New in This Release	14
----------------------------	----

eBAM Version 5.1.3 Enhancements	14
---------------------------------	----

Additional Platform Support	14
-----------------------------	----

Architectural and Runtime Enhancements	14
--	----

Documentation Enhancements	14
----------------------------	----

eBAM Version 5.1.2 Enhancements	15
---------------------------------	----

Documentation Enhancements	15
----------------------------	----

eBAM Version 5.1.1 Enhancements	15
---------------------------------	----

Architectural and Runtime Enhancements	15
--	----

eBAM Version 5.1.0 Enhancements	15
---------------------------------	----

Architectural and Runtime Enhancements	15
--	----

General and GUI Enhancements	16
------------------------------	----

Documentation Enhancements	16
----------------------------	----

About This Document	17
---------------------	----

What's in This Document	17
-------------------------	----

Scope	17
-------	----

Intended Audience	18
-------------------	----

Text Conventions	18
------------------	----

Screenshots	18
-------------	----

Related Documents	18
-------------------	----

Sun Microsystems, Inc. Web Site	19
---------------------------------	----

Documentation Feedback	19
------------------------	----

Chapter 2

Overview of eBAM Studio	20
-------------------------	----

eBAM and Business Activity Monitoring	20
---------------------------------------	----

eBAM Application Requirements	20
-------------------------------	----

Summary of eBAM Components	21
eBAM and Java CAPS	22
eBAM Data Flow	23

Chapter 3

Installing eBAM Studio	24
System Requirements	24
Prerequisite Java CAPS Products	24
Related Products	25
Supported Operating Systems	25
Steps for Installing eBAM	25
Uploading eBAM to the Repository	25
Updating Enterprise Designer with eBAM	29

Chapter 4

Using eBAM Studio	31
eBAM Applications	31
Components of an eBAM Application	32
Setting Up Data Definitions	33
Creating and Validating Data Definitions	33
Modifying Data Definitions	39
Importing Sample Data	39
Add, Update, and Delete Data From the eBAM Data Store	43
The ADD Web Service	44
The UPSERT Web Service	45
The DELETE Web Service	47
Database Considerations	49
Using the Default Flatfile Database	49
Flatfile Database Considerations	49
Using eBAM with an Oracle Database	50
Oracle Database Considerations	50

Chapter 5

Using eBAM Queries	55
About eBAM Queries	55
Setting Up the Application's Queries	56
Example of an eBAM Query Using a Runtime Argument	60
Using the Condition Editor	62

Chapter 6

Using eBAM Alerts 65

eBAM Alerts	65
What You Need to Know About Alerts	66
Example of an Alert	66
Queries: Conditions and Output Data	66
Things to Keep in Mind	67
A Negative Example	67
Setting Up the Application's Alert Conditions	68

Chapter 7

Using eBAM Charts 70

eBAM Charts	70
About Charts	70
Setting Up the Application's Charts	71

Chapter 8

Sample and Tutorial 72

Overview	72
Sample	73
Tutorial	73
Installing and Running the eBAM Sample	74
Installing the Sample Files	74
Importing the Sample Project	75
Hands-on Steps for Designing an eBAM Application	77
Starting Up	78
Creating and Validating OTDs	79
Creating an eBAM Application and Data Definitions	81
Procedures	82
Creating eBAM Sample Queries	85
eBAM Query qryS_CountDealerReports	86
qryG_BelowMinimumSales	87
qryG_ExtractSalesByCarName	89
eBAM Query qryC_SumSalesByDealer	92
eBAM Query qryC_SumSalesByCarID	93
eBAM Query qryCj_SumSalesByMakerAndCarID	94
eBAM Query qryCj_SumSalesByCarnameAndYear	95
Creating eBAM Sample Alerts	97
Creating eBAM Sample Charts	99
eBAM Chart chPie_SalesByDealer	99
eBAM Chart chBar_SalesByCarId	102
eBAM Chart chBar_SalesByCarnameAndYear	104
eBAM Chart chBar_SalesByCarnameAndYear_stacked	105

Creating Navigation Links	106
Creating and Validating Business Processes	107
Business Processes bpAddCarDefs and bpAddReports	107
Business Process bpQryExtractSalesByCarName	111
Business Process bpDelTables	112
Business Process bpAlertBelowMinimum	113
Creating and Configuring Connectivity Maps	115
Starting the Domain to Run the Sample	119
Building, Deploying, and Monitoring a Project	120
Monitoring Key Performance Indicators with Live Data	123
Adding New Data	123
Sending Alerts and Notifications	125
Using a Dynamic Query	125
Deleting Existing Data	126

Chapter 9

Using eBAM Charts in Portals	127
About Portals and Portlets	127
About Java CAPS Application Files	127
About Sun Java Portal Server 6 and 7	128
Using Sun Java System Portal Server	129
Reference Documentation	130
Before Installing Portal Server	131
Necessary Patches	131
Portal Server Version Compatibility	131
Release Notes and Additional Information	132
Running eBAM Studio Applications with Portal Server 6 2005Q4	132
Before You Start	132
System Requirements	132
Supported Operating Systems	133
Portal Server 6 2005Q4 on Windows: Procedural Overview	133
Running eBAM Studio Applications with Portal Server 7	155
Before You Start	155
System Requirements	156
Supported Operating Systems	156
Installation	156
Setting Up an IFrame Channel	156
Setting Up a JSR 168 Portlet Channel	161
Running eBAM Studio Applications with Portal Server 7.1	163
Before You Start	163
System Requirements	164
Supported Operating Systems	164
Installation	164
Setting Up an IFrame Channel	164
Setting Up a JSR 168 Portlet Channel	169

Appendix A**Chart Types 172****Chart Types 172**

Appendix B**Persistence and Monitoring 178****Overview 178****Setting Up the Database Schema for eInsight Engine 179**

Exporting and Extracting Database Scripts for eInsight 179

Running Database Scripts for eInsight 179

Configuring the eInsight Engine for Runtime 180

Configuring Specific BPs to Use Persistence 182

Turning Monitoring On or Off for Specific BPs 183

Monitoring 183

About the Adobe SFV plug-in 184

Using Enterprise Manager to Monitor BP Activity 184

Using the Monitor Tool Palettes 185

BP Monitoring Tools 185

Tools for Controlling Display Modes 185

Controlling the Display of Instance Data 185

Logging 186Setting Logging Levels 186

Appendix C**SQL Reserved Words 187****SQL Reserved Words 187****Index 194**

List of Figures

Figure 1	eBAM Studio and Java CAPS	22
Figure 2	eBAM's Layers for Task Differentiation	23
Figure 3	Java CAPS Products Installed	26
Figure 4	Java CAPS Product List by Category	27
Figure 5	Selecting Files to Install	28
Figure 6	Documentation Page After Selecting eBAM Studio	29
Figure 7	Creating a New eBAM Application	34
Figure 8	eBAM Application Wizard - Enter the eBAM Application Name	34
Figure 9	eBAM Application - Enter the Data Definition Name	35
Figure 10	eBAM Application - Define the Data Definition Elements	35
Figure 11	eBAM Application - Specify Data Retention Intervals	36
Figure 12	eBAM Application - Create Data Definition	37
Figure 13	eBAM Component Editor Showing Data Definition	38
Figure 14	Show Sample Data Icon on eBAM Editor Tool Palette	40
Figure 15	Portion of the Sample Data Window	40
Figure 16	Show Sample: Preview of Field Layout	42
Figure 17	Sample Data Displayed in eBAM Component Editor	43
Figure 18	Three Data Definitions	43
Figure 19	Example of Dragging an Add Web Service	44
Figure 20	OTD with Nine Fields Connected to an Add Service	44
Figure 21	Add Service Output	45
Figure 22	Example of Attaching Upsert to an OTD Unmarshal	45
Figure 23	Example of an Unmapped OTD	45
Figure 24	Example of a Mapped Field	46
Figure 25	Example With All Fields Mapped	46
Figure 26	Mapped Example with WHERE Clause	47
Figure 27	Example of Dragging a Delete Web Service	47
Figure 28	Example of a WHERE Clause	48
Figure 29	Example Where the Text Becomes the Entire WHERE Clause	48
Figure 30	Example Where Text Becomes a Variable Value	48
Figure 31	Creating a new Oracle External	51
Figure 32	Configuring an Oracle External	52

Figure 33	eBAM Application and Oracle External on a Connectivity Map	53
Figure 34	Connecting an eBAM Application's "oracle" Service to an Oracle External	53
Figure 35	Choosing the Outbound Oracle eWay Connection Type	53
Figure 36	Component Editor, Showing Newly Created (Unconfigured) Query	57
Figure 37	Adding Input Runtime Arguments to eBAM Queries	58
Figure 38	Configuring a Condition	58
Figure 39	Configuring a Data Definition Condition Using Runtime Arguments	59
Figure 40	SQL String in Properties Dialog Box	59
Figure 41	Configuring the Dataset	59
Figure 42	Example of a Well-Configured eBAM Query	61
Figure 43	Implementing an eBAM Query in a Business Process	62
Figure 44	Operator Palette for SQL Operations	64
Figure 45	Charts Displayed for Sample	73
Figure 46	Sample Project prjCars, Showing Components in Project Tree	76
Figure 47	Seven-Field Sample OTD (OTD7)	80
Figure 48	Validation of OTD5 Using Data From inputReports5_original.~in	81
Figure 49	eBAM Component Editor Showing New Data Definition ddnCarDefs	82
Figure 50	Showing Sample Data: Preview of Data Definition Field Layout	84
Figure 51	Sample Data Displayed in Sample Data Pane	85
Figure 52	Summary Query to Count Dealer Reports	87
Figure 53	Setting a Condition on the totalSales Field	88
Figure 54	General Query to Check the Condition totalSales < 1000.00	89
Figure 55	Creating a Runtime Argument	90
Figure 56	Setting a Dynamic Argument Condition on the totalSales Field	90
Figure 57	Inner Join	91
Figure 58	Summary Query to Extract Sales for carName	92
Figure 59	Category Query to Report Summed totalSales by dealerName	93
Figure 60	Category Query to Report Summed totalSales by carModelCode	94
Figure 61	Category Query to Report Sales by carMaker and then carCode	95
Figure 62	Edit Join Condition	96
Figure 63	Category Query to Report Sales by carMaker and then carCode	97
Figure 64	General Query with a Condition	98
Figure 65	Alert for Notifying When Total Sales Drop Below \$1,000.00	99
Figure 66	Previewing a Category-Based Pie Chart Using the Committed Sample Data	100
Figure 67	Category Chart chPie_SalesByDealer with Modified Chart Properties	101
Figure 68	Category Chart chPie_SalesByDealer Redrawn as a Bar Chart	102
Figure 69	Previewing a Simple Bar Chart Using the Committed Sample Data	103
Figure 70	Category Chart chBar, Modified to a Trafficlight Array	104

Figure 71	Previewing a Category-Series Bar Chart Using All Committed Sample Data	105
Figure 72	Previewing a Category-Series Bar Chart Using All Committed Sample Data	106
Figure 73	Business Processes bpAddCarDefs and bpAddReports	107
Figure 74	Mapping OTD7.unmarshal.Output to ddnCarDefs.add.Input	109
Figure 75	Mapping OTD5.unmarshal.Output to ddnReports.add.Input	110
Figure 76	Business Process bpQryExtractSalesByCarName	111
Figure 77	Mapping OTD3.marshal.Output to FileClient.write.Input	112
Figure 78	Business Process bpDelTables	112
Figure 79	Business Process bpAlertBelowMinimum	113
Figure 80	Mapping OTD5.unmarshal.Output to ddnReports.add.Input	115
Figure 81	Connectivity Map for Sample	116
Figure 82	Connecting Both Services of bpDelTables to bamCars	118
Figure 83	All Connections Completed Between bamCars and BPs	119
Figure 84	Deployment Profile for Sample, Before Component Mapping	120
Figure 85	Deployment Profile for Sample, After Component Mapping	121
Figure 86	Using Enterprise Manager to Deploy and Enable the Application	122
Figure 87	Pie Chart Showing Car Sales Breakdown by Dealer	124
Figure 88	Bar Chart Showing Sales Totals by Car Code	124
Figure 89	Customize Installation User Interface	134
Figure 90	Admin Console Page	136
Figure 91	Admin Console Page With Enterprise Applications Frame	137
Figure 92	Deploy Enterprise Application Frame: Initial	137
Figure 93	Deploy Enterprise Application Frame: After .ear File Selection	138
Figure 94	New Enterprise Applications List	139
Figure 95	Access Manager Console Page	140
Figure 96	Access Manager Console With User General Properties	141
Figure 97	Available Services for New User	142
Figure 98	Available Services for New User List: Selected	142
Figure 99	Information for New User List	143
Figure 100	Initial Window: Portal Server Services (Identity Management)	144
Figure 101	Access Manager Page With Portal Desktop	145
Figure 102	Access Manager Page: Portal Desktop - Channels	145
Figure 103	List of Existing Channels	146
Figure 104	New Channel Frame With IFrame Provider	147
Figure 105	Edit Properties Frame for New Channel	148
Figure 106	List of Container Channels	150
Figure 107	Initial MyFrontPageTabPanelContainer Frame	151
Figure 108	Channel Management Section: Initial	152

Figure 109	Channel Management Section: First Add	153
Figure 110	Channel Management Section: Second Add	154
Figure 111	Portal Server 6 Portal Desktop Welcome Page	155
Figure 112	Portal Server 7 Console Page	157
Figure 113	Manage Containers and Channels : portal1 Page: Initial	158
Figure 114	Manage Containers and Channels : portal1 Page: With New Channel	159
Figure 115	Channel Properties Pane	160
Figure 116	Portal Server 7 Portal Desktop Welcome Page	161
Figure 117	Portal Server 7.1 Console Page	165
Figure 118	Manage Containers and Channels: portal1 Page: Initial	166
Figure 119	Manage Containers and Channels: portal1 Page: With New Channel	167
Figure 120	Channel Properties Pane	168
Figure 121	Portal Server 7.1 Portal Desktop Welcome Page	169
Figure 122	eInsight Engine Configuration Properties	181
Figure 123	Setting Persistence for a Business Process	182
Figure 124	Setting Database Script Properties for BPs	183
Figure 125	Monitor View	184

List of Tables

Table 1	Text Conventions	18
Table 2	Locations of SAR Files Used in This Example	28
Table 3	eBAM Application Components and Their Icons and Web Services	32
Table 4	Metadata for Creating a Data Definition	33
Table 5	Ways of Modifying Data Definitions	39
Table 6	Specifying Formatting Type and Encoding for Imported Sample Data	40
Table 7	Parsing Information for Imported Sample Delimited Data	41
Table 8	Parsing Information for Imported Sample Fixed-Width Data	41
Table 9	Tools Provided in the Graphical Canvas of the Condition Editor	62
Table 10	Tools Provided in the Tool Palette of the Query-Editing Canvas	63
Table 11	Alert Properties for eMail	69
Table 12	Comparing the Sample to the Tutorial	73
Table 13	OTDs Included With the eBAM Sample	79
Table 14	Metadata (Field Names and Data Types) for ddnCarDefs	81
Table 15	Metadata (Field Names and Data Types) for ddnReports	81
Table 16	Choices for Imported Sample Data for Sample Data Definitions	83
Table 17	Queries Included With the eBAM Sample	85
Table 18	Charts Included with the eBAM Sample	99
Table 19	Business Processes Included with the eBAM Sample	107
Table 20	Procedures for Generating and Deploying Applications to Sun Java Portal Server	128
Table 21	Properties Common to All Chart Types	173
Table 22	Properties Common to Pie and Bar Charts	173
Table 23	Properties Specific to Pie Charts	174
Table 24	Properties Specific to Bar Charts	174
Table 25	Properties Specific to Meters	174
Table 26	Properties Specific to Trafficlights	176
Table 27	Display Modes	185
Table 28	Monitor: Display Instance Data	185
Table 29	SQL Reserved Words	187

Introduction

This chapter provides a brief overview of Sun SeeBeyond eBAM™ Studio (eBAM) and general information about this guide.

What's in This Chapter

- [“About eBAM Studio” on page 13](#)
- [“What's New in This Release” on page 14](#)
- [“About This Document” on page 17](#)
- [“Related Documents” on page 18](#)
- [“Sun Microsystems, Inc. Web Site” on page 19](#)
- [“Documentation Feedback” on page 19](#)

1.1 About eBAM Studio

Business Activity Monitoring (BAM) involves the collection, aggregation, and presentation of business activity data according to specified Key Performance Indicators (KPIs). KPIs provide a context for business processes by turning raw data into useful information, allowing the business analyst to focus on monitoring and analyzing measurable operations and processes across the enterprise.

eBAM Studio provides tools for generating custom cross-application graphical dashboards for defining and monitoring KPIs that summarize the aggregated business data collected through the eInsight™ Business Process Manager (eInsight) or eGate™ Integrator (eGate) application layers. Data can be collected over time and transformed into meaningful, rich visual presentations.

eBAM renders real-time and historical data in familiar visual formats, such as pie and bar charts, for display in graphical dashboards. These recognizable, easy-to-read contexts enable you to quickly translate information into action—for example, if you are a business analyst, a custom dashboard of performance data can help you to identify business trends in real time.

eBAM allows you to transform collected data into a visual context for accessibility over the web. By leveraging the web, key business information can be made accessible in a visual context across the enterprise to provide visibility, reporting, and analysis.

1.2 What's New in This Release

Beginning with the 5.1.0 release and continuing through the 5.1.1 release eBAM Studio has received a number of new features and enhancements. Stable after the 5.1.1 release, the 5.1.3 release includes additional platform support, a run time enhancement, and improved documentation.

1.2.1 eBAM Version 5.1.3 Enhancements

There are no architectural or GUI enhancements with the 5.1.3 release of eBAM Studio. However, there is new platform support, a run time enhancement, and the documentation (*eBAM™ Studio User's Guide* and online HTML Help) has been expanded and improved.

Additional Platform Support

With the release of 5.1.3 eBAM Studio supports the following platforms:

- Red Hat Enterprise Linux AS 4 (AMD Opteron)
- SUSE Linux Enterprise Server 10 (AMD Opteron)
- Japanese Sun Solaris 8, 9, and 10 with required patches (SPARC)
- Japanese HP-UX 11i (11.11) on PA-RISC, and 11i v2.0 (11.23) on Itanium with required patches and parameter changes
- Japanese IBM AIX 5L, versions 5.2 and 5.3 with required maintenance level patches
- Japanese Microsoft Windows 2000 SP3 and SP4, Windows XP SP2, and Windows Server 2003 R2

Architectural and Runtime Enhancements

New features related to architecture and run time include:

- eBAM now supports Sun Java System Application Server 8.2 Enterprise Edition.
- eBAM now supports Sun Java Portal Server 7.1.

Documentation Enhancements

Improvements to the documentation include:

- The Sample and Tutorial chapter in the User's Guide has undergone a rewrite to make it more user friendly.
- The Sample and Tutorial topics in the HTML Help have undergone a rewrite to make them more user friendly.

1.2.2 eBAM Version 5.1.2 Enhancements

There are no architectural, run time, or GUI enhancements with the 5.1.2 release of eBAM Studio. However, the documentation (*eBAM™ Studio User's Guide* and online HTML Help) has been expanded and improved.

Documentation Enhancements

Improvements to the documentation include:

- Online HTML Help usability has been improved making the Help easier to use and navigate.
- User's Guide has been fine-tuned, and includes additional information on:
 - ♦ Default flatfile database
 - ♦ Oracle database
 - ♦ Web services **add**, **upsert**, and **delete**
 - ♦ Alerts

1.2.3 eBAM Version 5.1.1 Enhancements

The 5.1.1 release of eBAM Studio included an architectural and run time enhancement. There were no new GUI or documentation enhancements.

Architectural and Runtime Enhancements

New features related to architecture and run time include:

- As of Release 5.1.1, eBAM now supports BEA WebLogic Server 9.1.

1.2.4 eBAM Version 5.1.0 Enhancements

The 5.1.0 release of eBAM Studio included architectural, run time, and GUI enhancements. It also included online HTML Help.

Architectural and Runtime Enhancements

New features related to architecture and run time include:

- eBAM now supports Oracle 9i version 9.2 and Oracle 10g (in addition to the prepackaged data storage functionality): Each eBAM application instance on a Connectivity Map has an invoked service named **oracle** that allows you to connect it to an Oracle eWay.

For information on the Oracle eWay, see the *Sun SeeBeyond eWay Adapter for Oracle™ User's Guide*.

- eBAM now supports Sun Java System Application Server (AS) 8.2 Enterprise Edition.

- In conjunction with AS 8.2, Sun Java System Portal Server can be used to view eBAM charts, either in iFrames (using Portal Server 6 or Portal Server 7) or in JSR 168 portlets (using Portal Server 7 and 7.1 only).

For more information, see the corresponding chapter in the *Sun SeeBeyond eBAM™ Studio User's Guide* and the Sun documentation for Portal Server.

- eBAM now fully supports data persistence via the eInsight runtime recoverability database when the eInsight Engine is properly configured. (This requires Oracle, Sybase, SQL Server, or DB2.)

For information on viewing/modifying database scripts and configuring the eInsight Engine, see the *Sun SeeBeyond eInsight™ Business Process Manager User's Guide*.

General and GUI Enhancements

New features related to the GUI include:

- Workflow and usability improvements make the process of creating and configuring queries, charts, and alerts smoother and more intuitive.
- You can now edit data definitions after initial creation; in other words, you can now rename data definitions, change their data retention properties, add or delete columns to them, and modify column names or types.
- You can now use two or more data definitions in a single eBAM application.
- You can use a new chart type—stacked traffic-lights—for queries that define categories and series.
- You can add a navigation link to any chart, allowing you to navigate from one chart to another when browsing them on the web.
- eBAM provides a new **delete** web service (for data definitions without data retention), and the **execute** web service (for queries) can now be dragged into any eInsight business process or eVision page flow.
- Version Control is fully supported for eBAM applications (at the application level, but not for individual components within an application).
- Copy-and-paste of eBAM applications (but not components) is fully supported.
- Command-line CodeGen is fully supported.

Documentation Enhancements

New features related to documentation include:

- New online HTML Help.

1.3 About This Document

The section provides a chapter overview, purpose and scope of the user's guide, intended audience, the writing conventions used in this document, and a screenshot disclaimer.

1.3.1 What's in This Document

This document is organized topically as follows:

- **Chapter 1 "Introduction" on page 13** describes the purpose of this User's Guide, including writing conventions and a list of related documents.
- **Chapter 2 "Overview of eBAM Studio" on page 20** provides an overview of the general structure, architecture, and operation of eBAM Studio and its place within the Sun Java Composite Application Platform Suite.
- **Chapter 3 "Installing eBAM Studio" on page 24** explains how to install eBAM Studio.
- **Chapter 4 "Using eBAM Studio" on page 31** describes generally how to exercise the GUI and explains setting up eBAM data definitions and using the **add**, **upsert**, and **delete** web services.
- **Chapter 5 "Using eBAM Queries" on page 55** explains setting up eBAM queries and using the **execute** web service.
- **Chapter 6 "Using eBAM Alerts" on page 65** explains setting up eBAM alerts and using the **notify** web service.
- **Chapter 7 "Using eBAM Charts" on page 70** explains setting up eBAM charts.
- **Chapter 8 "Sample and Tutorial" on page 72** provides both a quick-start guide to getting a sample eBAM application up and running, and also an extensive hands-on tutorial that guides you through the process of creating and configuring all the components used in the sample.
- **Chapter 9 "Using eBAM Charts in Portals" on page 127** provides background and instructions for displaying eBAM web applications using Sun Java System Portal Server.
- **Appendix A "Chart Types" on page 172** lists the types of charts that eBAM provides along with property information.
- **Appendix B "Persistence and Monitoring" on page 178** explains how to use the eInsight engine to collect and persist data from your business processes.
- **Appendix C "SQL Reserved Words" on page 187** lists reserved SQL terms that must not be used as field names in eBAM applications.

1.3.2 Scope

This User's Guide provides general information about the features and operation of eBAM Studio. For information on any other product in the suite, see the product's

user's guide. For information on system management, see the *Sun SeeBeyond eGate Integrator System Administration Guide*.

1.3.3 Intended Audience

This User's Guide is intended for experienced computer users who are involved in developing and maintaining composite applications using eBAM Studio. Primarily, these are individuals who will be using Enterprise Designer to build projects—often in conjunction with other modules in the Sun Java Composite Application Platform Suite. This guide also provides a basic overview of eBAM Studio for those attempting to gain a general understanding of how the product works.

This guide assumes that the reader is an experienced computer user, familiar with Windows-style GUI operations, and also has an in-depth understanding of each operating system on which eBAM Studio will be installed.

1.3.4 Text Conventions

The following conventions are observed throughout this document.

Table 1 Text Conventions

Text Convention	Used For	Examples
Bold	Names of buttons, files, icons, parameters, variables, methods, menus, and objects	<ul style="list-style-type: none">Click OK.On the File menu, click Exit.Select the eGate.sar file.
Monospaced	Command line arguments, code samples; variables are shown in bold italic	<code>java -jar filename.jar</code>
Blue bold	Hypertext links within document	See Text Conventions on page 18
<u>Blue underlined</u>	Hypertext links for Web addresses (URLs) or email addresses	http://www.sun.com

1.3.5 Screenshots

Depending on what products you have installed, and how they are configured, the screenshots in this document may differ from what you see on your system.

1.4 Related Documents

The following documents provide additional information about eBAM Studio as explained in this guide:

- *Java Composite Application Platform Suite Installation Guide*
- *Sun SeeBeyond eGate Integrator System Administration Guide*
- *Sun SeeBeyond eInsight Business Process Manager User's Guide*

For information on a specific add-on product, such as the eWay Intelligent Adapter for Oracle, see the User's Guide for that product. A complete list of Sun documentation is included in the *Java Composite Application Platform Suite Primer*.

The documentation for the Sun Java Composite Application Platform Suite is distributed as a collection of online documents, which can be accessed through the Enterprise Manager (see the *Java Composite Application Platform Suite Installation Guide*). These documents are in Adobe Acrobat format, which requires that Acrobat Reader be installed on your computer. Acrobat Reader can be from Adobe Systems as a free download from the following URL:

<http://www.adobe.com>

1.5 Sun Microsystems, Inc. Web Site

The Sun Microsystems web site is your best source for up-to-the-minute product news and technical support information. The site's URL is:

<http://www.sun.com>

1.6 Documentation Feedback

We appreciate your feedback. Please send any comments or suggestions regarding this document to:

CAPS_docsfeedback@sun.com

Overview of eBAM Studio

eBAM Studio allows you to create business activity monitoring applications that intercept the flow of data through other components of Java Composite Application Platform Suite (Java CAPS) to produce visual presentations of data analysis based on KPIs and alerts.

This chapter provides overviews of eBAM's architecture and overall approach to business activity monitoring.

What's in This Chapter

- [“eBAM and Business Activity Monitoring” on page 20](#)
- [“Summary of eBAM Components” on page 21](#)
- [“eBAM and Java CAPS” on page 22](#)

2.1 eBAM and Business Activity Monitoring

The eBAM web interface allows business analysts to transform data that has been collected over time into meaningful, rich visual presentations by providing different views of performance data that enable the identification of time-critical business trends. Leveraging Java CAPS and the web, eBAM provides visibility to key business information, recasts it in a graphical format, and makes it available across the enterprise.

eBAM Application Requirements

An eBAM application requires access to runtime resources, from which it gathers raw data and processes it (via user-configured aggregation and presentation) to render it into a meaningful format. An eBAM application requires the following:

- Access to runtime data from suite processes.
- Data definitions (the facts that form the basis for the calculation of a KPI value).
- The dashboard (visual) representation of the KPI.
- The events that are triggered when the value of the KPI crosses a preset threshold.
- Graphical dashboard presentations—visual presentation of data using bar charts, pie charts, and meters (dial indicators).

- User-defined timeframes, such as frequency and starting/ending dates and times.

eBAM ties into raw data via JMS, eGate, and web services to perform real-time analysis on data received from external systems, business processes, or composite applications.

2.2 Summary of eBAM Components

eBAM takes a simple approach to business activity monitoring. Each eBAM application provides a mechanism for sampling the data traffic and storing results in a way that allows standing queries to update and communicate their findings.

The components of an eBAM application consist of:

- **Data definitions**

Data definitions provide the raw ingredients used by all the other components in an eBAM application.

A data definition specifies the following:

- ♦ An identifying label and a datatype for each of the data elements collected.
- ♦ The length of time to retain data before flagging it for removal and the frequency at which expired data is removed when data retention is specified.

- **Queries**

Queries provide the recipes that combine and manipulate the data elements, and are based on SQL SELECT statements. Each *condition* of a query is a WHERE clause in the SELECT statement.

The eBAM Query Editor provides:

- ♦ SQL-compliant visual tools for filtering and manipulating the data.
- ♦ The ability to join two or more data definitions.
- ♦ The ability to alter its filters according to input supplied dynamically at run time.

- **Charts**

Charts are products that an eBAM application produces. Each chart is based a particular query:

- ♦ Charts provide interactive visual representations of KPIs, and they update along with the data stream, serving up a dashboard of results for the business analyst or executive, accessible via a web browser.

Each eBAM KPI queries and filters the results and communicates them to the GUI. The GUI provides the data view that best suits the user, such as an overall sense of business performance, or specific data trends or anomalies.

▪ Alerts

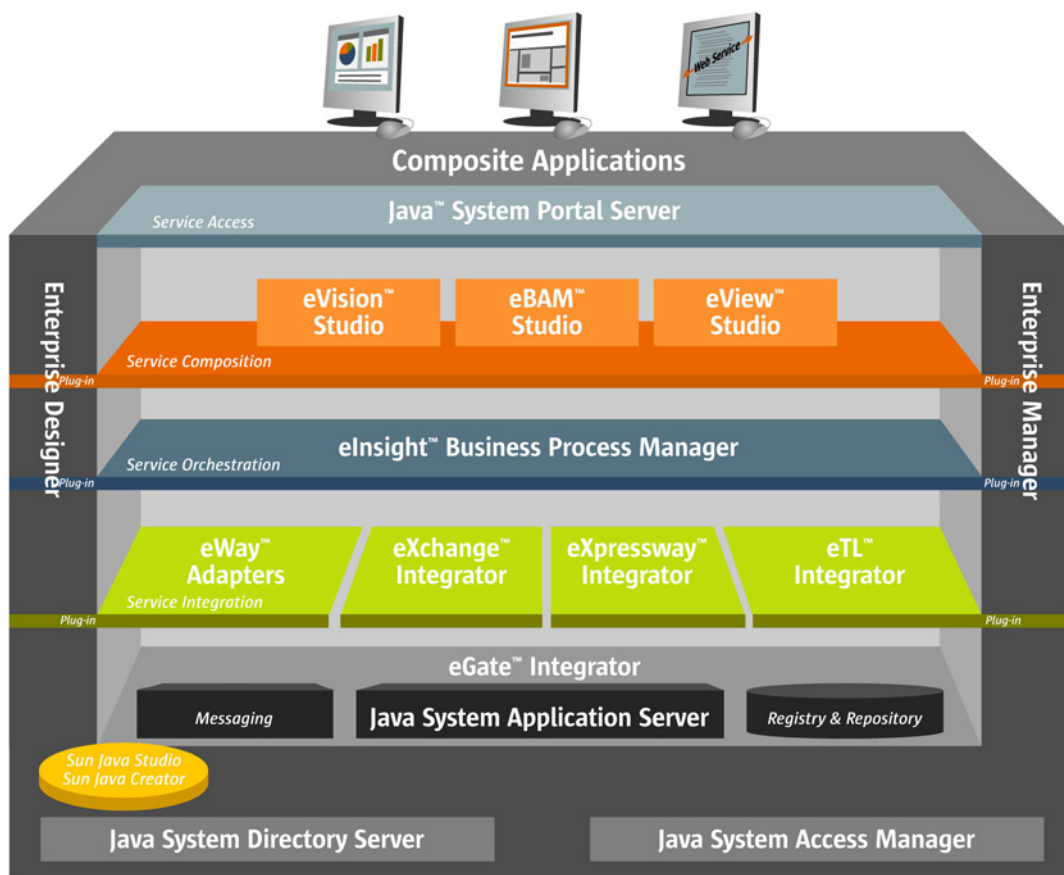
Alerts are also products that an eBAM application produces. Each alert is based a particular query:

- ♦ A simple email alert sends a specified message to a specified recipient whenever the results of the query meet a specified condition.
- ♦ The *notify* web service of an alert can be used to trigger a business process to communicate data or a KPI when a query's results meet a specified condition.

2.3 eBAM and Java CAPS

eBAM's integration with Java CAPS is shown in Figure 1.

Figure 1 eBAM Studio and Java CAPS



eBAM leverages other products and features in Java CAPS, including:

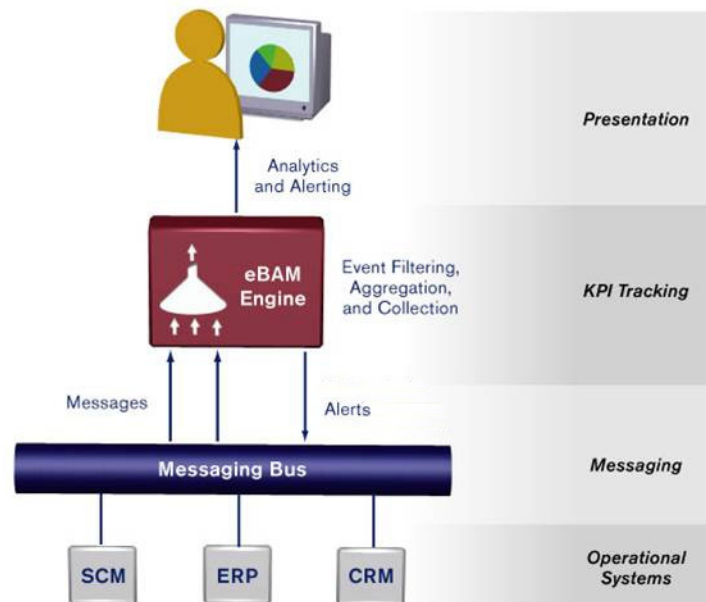
- The Enterprise Designer GUI is leveraged for the design of the data collection, graphical display, and notification processes.
- The eInsight Business Process Modeler is used for alert handling. **add**, **upsert**, and **delete**.
- Execute runtime resources are provided by the Integration Server/Application Server.
- Web applications created by eBAM Studio can be viewed either in a standalone browser or as an IFrame or portlet in the Sun Java System Portal Server.

eBAM Data Flow

To allow for task differentiation, scalability, tunability, and maintainability/robustness, the eBAM data flow consists of a stack of separable layers (see Figure 2).

- The *presentation layer* provides a front-end GUI that allows business analysts to see only as much or as little as they decide to see, using the indicators they find meaningful, presented in the fashion they find most congenial.
- The *tracking layer* collects, aggregates, and filters data, passing periodic updates of the information-of-interest upwards to the presentation layer.
- The *messaging layer* communicates with external systems, mediated by eInsight's Business Processes.

Figure 2 eBAM's Layers for Task Differentiation



Installing eBAM Studio

The eBAM Studio installation is similar to the basic Java CAPS installation. However, before proceeding with the installation of eBAM, ensure that you are aware of all of the prerequisites it requires and have first installed the major components of Java CAPS as discussed in the *Java Composite Application Platform Suite Installation Guide* (CAPS_Install_Guide.pdf).

What's in This Chapter

- “System Requirements” on page 24
- “Steps for Installing eBAM” on page 25

3.1 System Requirements

This section lists prerequisite products and related products. It also informs you which document contains the most up-to-date information on system requirements for each supported platform.

3.1.1 Prerequisite Java CAPS Products

The prerequisite Java CAPS products for installing eBAM 5.1.3 consist of the following:

- eGate Integrator 5.1.3 (**eGate.sar**)
- eInsight Business Process Manager 5.1.3 (**eInsight.sar**)
- eBAM Studio 5.1.3 (**eBAM.sar**)

If you want to use Oracle as your database for holding eBAM data (recommended), then the following Java CAPS product is also required:

- **OracleWay.sar**

If you want to run the sample project and follow the steps provided in [Chapter 8](#), then the following Java CAPS products are also required:

- **eBAMDocs.sar**
- **FileeWay.sar**

eGate.sar must be uploaded *before* uploading **eBAM.sar**; **eInsight.sar** and other SAR files can be uploaded along with **eBAM.sar**.

Complete instructions for uploading the SAR files and downloading the documentation and sample files are provided in [“Steps for Installing eBAM” on page 25](#).

3.1.2 Related Products

Other products work in conjunction with eBAM but are not part of Java CAPS:

- Microsoft Internet Explorer—For viewing eBAM charts via a web browser.
- Oracle 9.2 or Oracle 10—Provides faster data processing and more reliable data access than the file-based database system supplied with eBAM; see [“Database Considerations” on page 49](#).
- Sun Java System Application Server 8.2 Enterprise Edition—Provides an alternative to the SeeBeyond Integration Server.
- Sun Java System Portal Server 6.0, 7.0, or 7.1 (requires Sun Java AS)—Provides capabilities for viewing eBAM charts from within a portal.
- Database for data persistence / recoverability. If you use the eInsight Engine to persist data in the eInsight runtime recoverability database, see [“Persistence and Monitoring” on page 178](#). Refer to the *Sun SeeBeyond eInsight Business Process Manager User’s Guide* for information on database and platform support.

3.1.3 Supported Operating Systems

The **eBAM_Readme.txt** file on the product media contains the most up-to-date information on system requirements for each supported platform.

3.2 Steps for Installing eBAM

The steps for installing eBAM are the same as for other Java CAPS products. General product installation steps are provided in the *Java Composite Application Platform Suite Installation Guide* on the product media and accessible via the Suite Installer Documentation tab.

3.2.1 Uploading eBAM to the Repository

Before you begin

A Repository server must be running on the computer where you upload the eBAM product files, and **eGate.sar** must have already been uploaded to this Repository.

To upload eBAM product files to the Repository

- 1 On a Windows computer, start a web browser and point it at the computer and port where the Repository server is running:

`http://<hostname>:<port>`

where

- ♦ *<hostname>* is the name of the computer running the Repository server.
- ♦ *<port>* is the starting port number assigned when the Repository was installed.

For example, the URL you enter might look like either of the following:

```
http://localhost:12000
http://serv1234.company.com:32000
```

- 2 When the Java CAPS Suite Installer becomes active enter your username and password on the **Java CAPS Login** window, and then click **Login**.

The Suite Installer defaults to the **Administration** tab and displays a list of the products you have already installed (see Figure 3).

Figure 3 Java CAPS Products Installed

Click to install additional products.			
Java Composite Application Platform Suite Products Installed			
Product Name	Currently Installed	Installed by User	Date/Time of Installation
eGate	5.1.2	Administrator	Monday, August 21, 2006 8:30:30 AM PDT
Enterprise_Manager_SVGPlugin-win32	5.1.2	Administrator	Monday, August 21, 2006 8:42:52 AM PDT
eGateDocs	5.1.2	Administrator	Monday, August 21, 2006 8:46:20 AM PDT
logicalhost-win32	5.1.2	Administrator	Monday, August 21, 2006 8:46:31 AM PDT
Enterprise_Manager-Win32	5.1.2	Administrator	Monday, August 21, 2006 8:47:40 AM PDT

- 3 Select **Click to install additional products**. The Java CAPS List by Category appears (see Figure 4).

Figure 4 Java CAPS Product List by Category

Welcome to the Java Composite Application Platform Suite Installer.

To update the product list, please browse for the new file (Product_List.sar).

Product List:

(Product_List.sar)

Select Java Composite Application Platform Suite Products to Install

	Product Name	Install Version	Currently Installed	Installed by User	Date/Time of Installation
<input type="checkbox"/>	+ Core Product				
<input type="checkbox"/>	+ Enterprise Manager				
<input type="checkbox"/>	+ Logical Host				
<input type="checkbox"/>	+ Web Service				
<input type="checkbox"/>	+ eWay				
<input type="checkbox"/>	+ OTD				
<input type="checkbox"/>	+ eGate API Kit				
<input type="checkbox"/>	+ JMS Grid				
<input type="checkbox"/>	+ Documentation				

- 4 Open the appropriate product categories by clicking the “+” and select the check boxes FOR all of the following components/documentation that you have not previously uploaded:

Under Core Product:

- ♦ **eBAM**
- ♦ **eInsight**

Under eWay:

- ♦ **OracleWay** (if you use Oracle for faster data processing and reliability)
- ♦ **FileeWay** (if you plan to run the sample Project)

Under Documentation:

- ♦ **eBAMDocs** (if you plan to run the sample Project)

Note: *eBAMDocs.sar* contains documentation and sample files. It is recommended that you upload the documentation for any components you upload (such as *eInsightDocs*). The File eWay is used in the sample implementation to read data from the files and write output.

After making your selections, click **Next**. The Selecting Files to Install window appears (see Figure 5).

Figure 5 Selecting Files to Install

- 5 Click the **Browse** to locate that appropriate SAR file (in this example: **eBAM.sar**), and select it. See Table 2 for the locations of the SAR files.

Table 2 Locations of SAR Files Used in This Example

SAR File	ISO Image	DVD Part Number
eBAM.sar	java-caps-5_1_2-products-cd2.iso	Part No. 708-0192-10
eInsight.sar	java-caps-5_1_2-products-cd2.iso	Part No. 708-0192-10
OracleWay.sar	java-caps-5_1_2-products-cd3.iso	Part No. 708-0192-10
FileeWay.sar	java-caps-5_1_2-products-cd3.iso	Part No. 708-0192-10
eBAMDos.sar	java-caps-5_1_2-products-cd2.iso	Part No. 708-0192-10
eInsightDos.sar	java-caps-5_1_2-products-cd2.iso	Part No. 708-0192-10
FileeWayDocs.sar	java-caps-5_1_2-products-cd3.iso	Part No. 708-0192-10

Repeat this step for all of your selections, clicking **Next** after each selection. The components install.

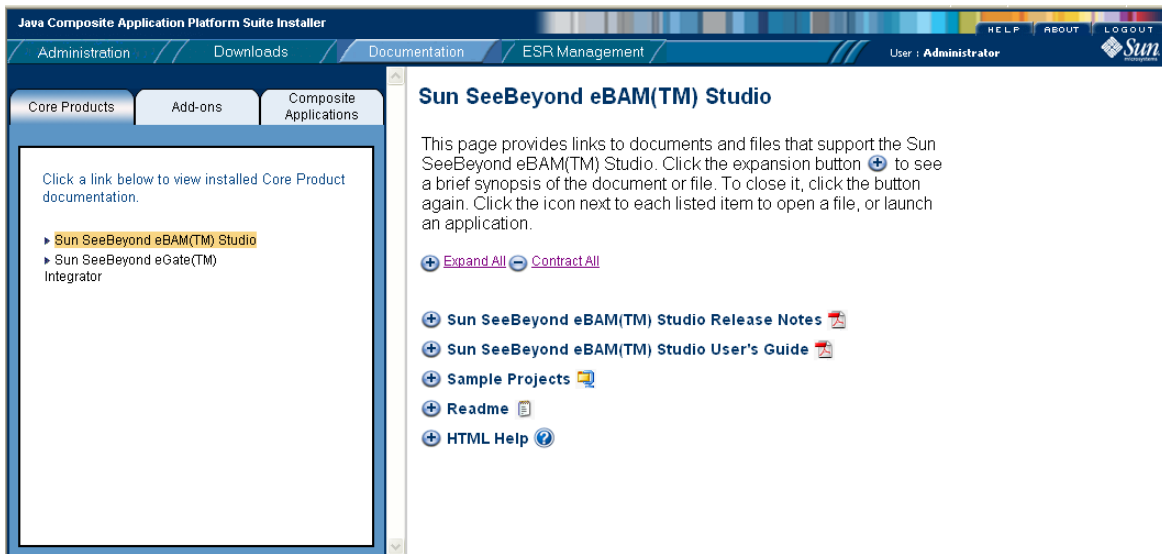
- 6 When the installation completes (“Installation finished” appears), select the **Downloads** tab and then the **Administration** tab to see a listing of what you have installed.

Your Repository can now serve the files in **eBAM.sar** to any Enterprise Designer that connects to it and uses the Update Center (see Figure 3, which has been appended with the additional installation data).
- 7 To view the documentation, click the **Documentation** tab. The Documentation page appears.
- 8 In the left pane, click the **Core Products** tab; and then click the product for which you want to view the documentation (for example: **Sun SeeBeyond eBAM Studio**).

Note: Click the **Add-ons** tab to see the eWay documentation.

The right pane is populated with all the PDF files, Readme text files, HTML files, Javadocs, and sample Projects that are associated with the selected product, and are ready to download (see Figure 6).

Figure 6 Documentation Page After Selecting eBAM Studio



- 9 To download documentation and sample files, click the icons to the right of the corresponding text. For instructions on running the sample, see [“Sample and Tutorial” on page 72](#).

3.2.2 Updating Enterprise Designer with eBAM

You must use Enterprise Designer’s Update Center to complete the installation of eBAM. This process downloads the modules, verifies the digital signatures, and completes the installation.

Before you begin

- Enterprise Designer has already been downloaded and installed.
- A Repository server is running on the computer where you uploaded the eBAM product files.

To refresh an existing installation of Enterprise Designer

- 1 Start Enterprise Designer.
- 2 On the **Tools** menu, click **Update Center**. The Update Center Wizard appears.
- 3 When the Select Update Category wizard appears, click **Next**. The Select Updates to Install wizard appears, displaying a list of components ready for updating.
- 4 On the Select Updates to Install wizard, click **Add All** (the button with a doubled chevron pointing to the right) to move all the modules into the **Include in Install** list.
- 5 Click **Next** and, when the License Agreement appears, click **Accept**.
- 6 When the progress bars indicate the download has completed, click **Next**.
- 7 Review the certificates and installed modules listed on the next wizard, and then click **Finish**.

- 8 When prompted to restart Enterprise Designer, click **OK**.

When Enterprise Designer restarts, the installation of eBAM Studio is complete, and you can use all the provided eBAM tools.

Using eBAM Studio

Before beginning to use eBAM and creating an application you should have a clear idea exactly what data you want to capture visually. Understanding the features of eBAM and how to use them is key for creating useful eBAM applications. The step-by-step instructions assume you are already familiar with GUIs such as the OTD Editor and eInsight Business Process Editor, and focus on the concepts and design-time GUI operations that are specific to eBAM.

What's in This Chapter

- [eBAM Applications](#) on page 31
- [Setting Up Data Definitions](#) on page 33
- [Importing Sample Data](#) on page 39
- [Add, Update, and Delete Data From the eBAM Data Store](#) on page 43
- [Database Considerations](#) on page 49

4.1 eBAM Applications

When you create an eBAM application the following considerations are important to keep in mind:

- *External data flow*—You must identify a source of data, decide how to bring it into the eBAM application (reading records from a file, accessing a topic, and so forth), and parse it using the **unmarshal** operation of the OTD that models your data. This is standard eGate methodology, and therefore not described in detail here.
- *Data definition*—For the parsed data, you must determine which data elements to pass to the eBAM application, and create names for the data elements of interest. The eBAM component you use to frame the data in this way is a data definition. Data within the eBAM application is controlled by the **add**, **upsert**, and **delete** activities; these web services are generated from the data definition.
- *Queries*—A query determines whether and how a data item or set is interesting. You use queries to set up one or more conditions that must be met before data is eligible for further processing by eBAM, and to calculate results (metrics, or KPIs). A *general* query typically sets the threshold for triggering an alert, and cannot be used for charts. A *summary* query returns data that can be used to construct simple charts, such as meters, pie charts, or one-dimensional trafficlights, and a specific

kind of bar chart. A *category* query returns data that can be used in trafficlighs, bar charts, and pie charts, and the data is automatically grouped by category.

- *What to communicate*—For alerts, typically the only item to communicate is simply that the triggering condition was met, perhaps also providing the interesting result. For charts, you usually calculate a metric such as a key performance indicator (KPI), using a query to identify one or more fields and operations to create the metric you want to communicate.
- *How to communicate*—Each alert or chart is based on the result of a particular query.

Alerts provide two distinct options: You can send an automatic email message, and/or you can place an alert's **notify** activity into a business process (BP) or page flow.

For charts, you select a type (pie, bar, et cetera) and specify its properties (update frequency, size, color, font, spacing, and so forth). Charts are displayed in a special web application called the Charts Viewer, and can be viewed using a web browser or, if you deploy the application to a Sun Java System Application Server, using iFrames (with Portal Server 6 or 7) or portlets (with Portal Server 7 only).

Components of an eBAM Application

Each eBAM application consists of one or more data definitions, one or more queries against the data definition(s), and one or more alerts or charts; each alert or chart is based on exactly one query. Table 3 shows the icons and web services for the components.

Table 3 eBAM Application Components and Their Icons and Web Services







Icon	Component and Associated Web Service(s)	For more information...
	<p>Data definitions are the infrastructure of the application. In them, you specify which data fields are of interest to you, and how to label and organize the data. In essence, the data definition specifies the column names and data types in a database table.</p> <p>The add web service is provided by every data definition. Data definitions that bypass data retention provide two other web services: upsert and delete.</p>	See “Setting Up Data Definitions” on page 33 .
 	<p>Queries are stored procedures run against accumulated data.</p> <p>The execute web service provided by the query can be used as an activity in a business process (often using runtime input).</p>	See “About eBAM Queries” on page 55 and “Setting Up the Application’s Queries” on page 56 .
 	<p>Alerts are triggered whenever a condition is met by one of the data items, such as exceeding a threshold. You can use an alert condition to trigger an automatic email.</p> <p>The notify web service provided by the alert can be used in a business process or page flow, as the first activity that triggers subsequent processing.</p>	See “Setting Up the Application’s Alert Conditions” on page 68 .

Table 3 eBAM Application Components and Their Icons and Web Services

Icon	Component and Associated Web Service(s)	For more information...
	Charts provide real-time feedback on the current data set according to conditions you specify. Charts do not provide web services.	See “About Charts” on page 70 and “Chart Types” on page 172 .

4.2 Setting Up Data Definitions

Before you begin: You must have a clear idea of the data you want to gather before you create the application, keeping in mind that the data definition specifies the column names and data types in a database table. You should know the answers to the following questions:

- Should the data be allowed to expire? If so, how many hours, days, months, or years should a dataset be retained, and how often should data cleanup occur?
- Altogether, how many data elements should be defined, and of what data type? Each data element must be of type **char**, **float**, **integer**, **timestamp**, or **varchar**.
- For each data element, what is the best descriptor? Each data element in an application must have a unique name; this is the name used when constructing conditions and charts, but is not necessarily the label that appears on the chart itself. The name you choose for a data element appears as the column heading for those data elements when you preview sample data.

In other words, be prepared with information whose form is similar to that of Table 4:

Table 4 Metadata for Creating a Data Definition

	Name	Data Type
1	OrderNum	integer
2	DateOfOrder	timestamp
3	CustName1	varchar
(...)	(...)	(...)
n	TotalDollarsThisOrder	float

4.2.1 Creating and Validating Data Definitions

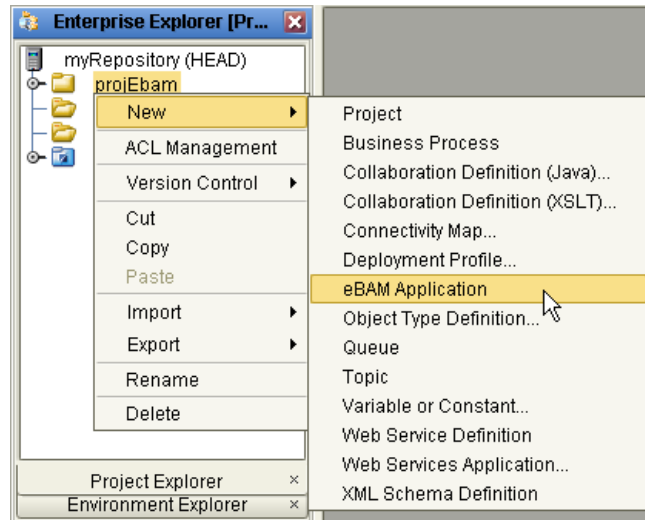
Data definitions can be created in two ways:

- As you create an eBAM application, you must also create its first data definition (see [“To create a new eBAM application and configure its first data definition” on page 34](#)).
- After you have created an eBAM application, you can add more data definitions (see [“To create and configure a new data definition in an existing application” on page 38](#)).

To create a new eBAM application and configure its first data definition

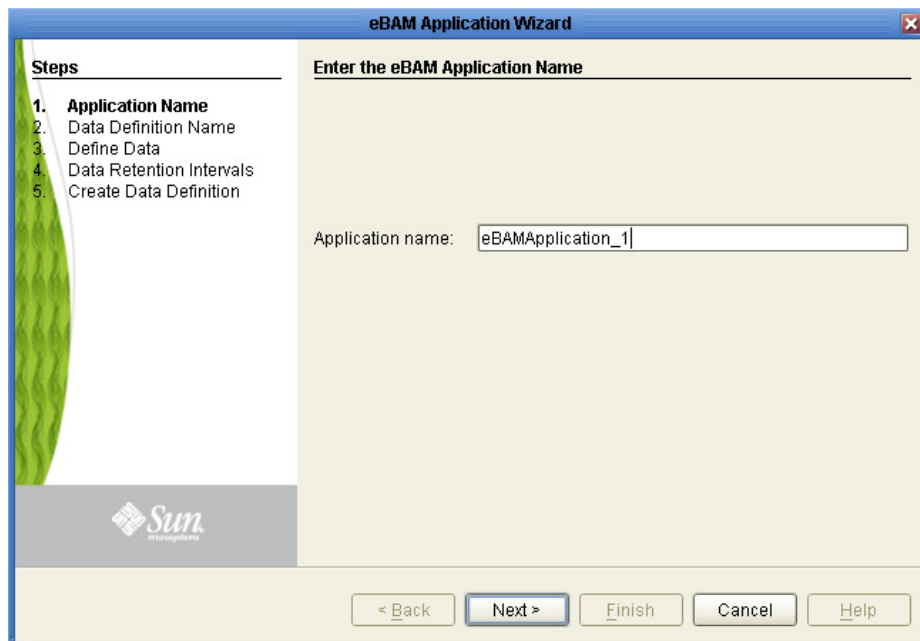
- 1 In Enterprise Designer, in the project tree (left pane), right-click the Project and, on the popup context menu, point at **New** and then click **eBAM Application** (see Figure 7).

Figure 7 Creating a New eBAM Application



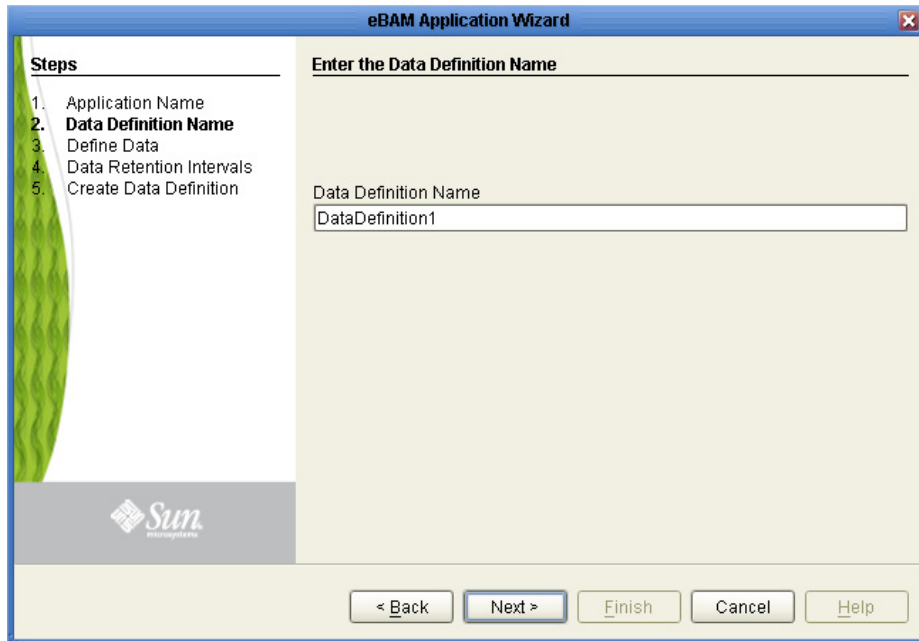
The eBAM Application Wizard opens (see Figure 8).

Figure 8 eBAM Application Wizard - Enter the eBAM Application Name



- 2 Keep the default name in the Application name field (eBAMApplication_1) or type a new name and then click **Next**. The eBAM Application - Enter the Data Definition Name wizard appears (see Figure 9).

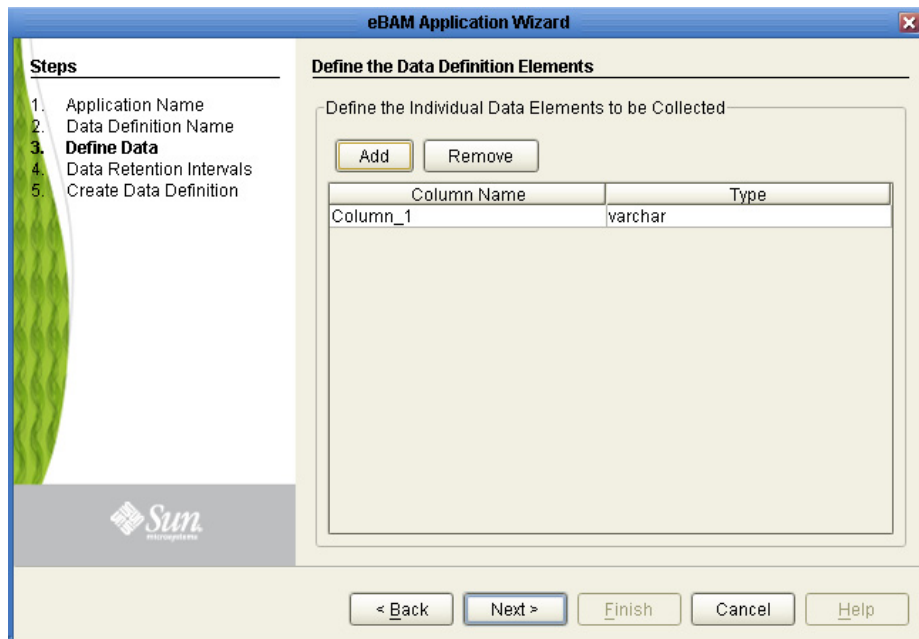
Figure 9 eBAM Application - Enter the Data Definition Name



The screenshot shows the 'eBAM Application Wizard' window. On the left, a 'Steps' list shows five steps: 1. Application Name, 2. **Data Definition Name**, 3. Define Data, 4. Data Retention Intervals, and 5. Create Data Definition. Step 2 is highlighted. The main area is titled 'Enter the Data Definition Name' and contains a text field labeled 'Data Definition Name' with the value 'DataDefinition1'. At the bottom, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted. The Sun Microsystems logo is visible in the bottom left corner.

- 3 In the Data Definition Name field, either keep the default name (DataDefinition1) or type a new name and then click **Next**. The eBAM Application - Define the Data Definition Elements wizard appears (see Figure 10).

Figure 10 eBAM Application - Define the Data Definition Elements



The screenshot shows the 'eBAM Application Wizard' window. On the left, the 'Steps' list shows five steps: 1. Application Name, 2. Data Definition Name, 3. **Define Data**, 4. Data Retention Intervals, and 5. Create Data Definition. Step 3 is highlighted. The main area is titled 'Define the Data Definition Elements' and contains a section 'Define the Individual Data Elements to be Collected'. Below this section are 'Add' and 'Remove' buttons. A table with two columns, 'Column Name' and 'Type', is shown. The first row contains 'Column_1' and 'varchar'. At the bottom, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted. The Sun Microsystems logo is visible in the bottom left corner.

Column Name	Type
Column_1	varchar

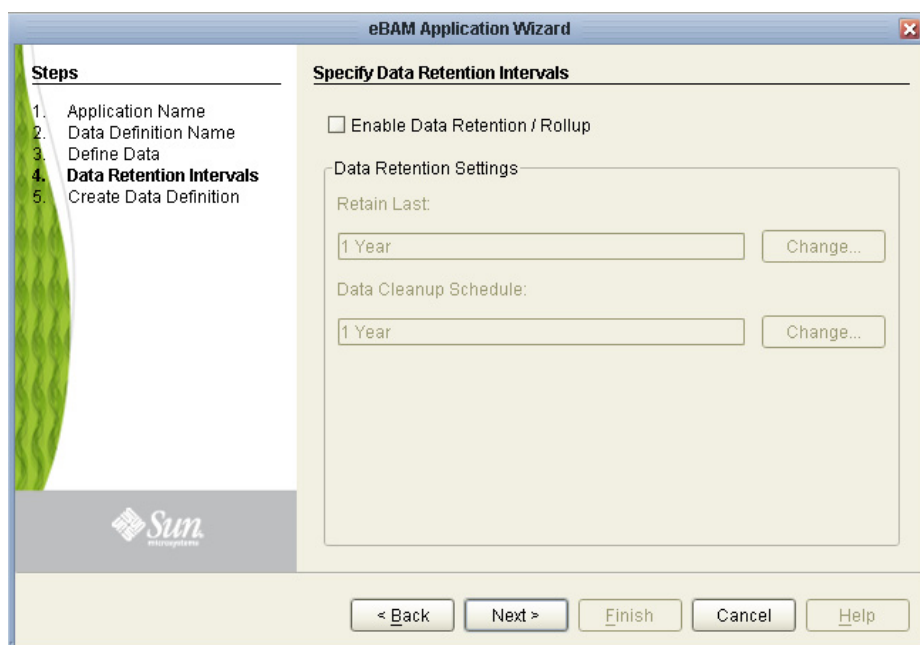
- 4 Before defining the data definition elements click **Add** several times, to create a number of columns. Next supply column names (by typing directly into the columns) and data types for the data elements (by opening the drop-down box and

making selections). Each column name must start with a letter followed by zero or more letters, numbers, or underscores.

Note: *If the order of the elements is important to you, double-check frequently to ensure you have not omitted an element—clicking **Add** appends each new item to the end. (However, order is unimportant to the queries.)*

As needed, click **Add** to append additional data elements or click **Remove** to remove unneeded ones. When you have specified the column name and data type of each data element your application gathers, click **Next**. The eBAM Application - Specify Data Retention Intervals wizard appears (see Figure 11).

Figure 11 eBAM Application - Specify Data Retention Intervals



- 5 In the eBAM Application - Specify Data Retention Intervals wizard you bypass or specify Data Retention. To bypass, simply leave the check box cleared. If, instead, you select the “Enable Data Retention/Rollup” check box, you must specify two settings:

- ♦ **Retain Last**—How long data is allowed to age before it expires.
- ♦ **Data Cleanup Schedule**—How often to remove expired data from the database. These values cannot be greater than the **Retain Last** values.

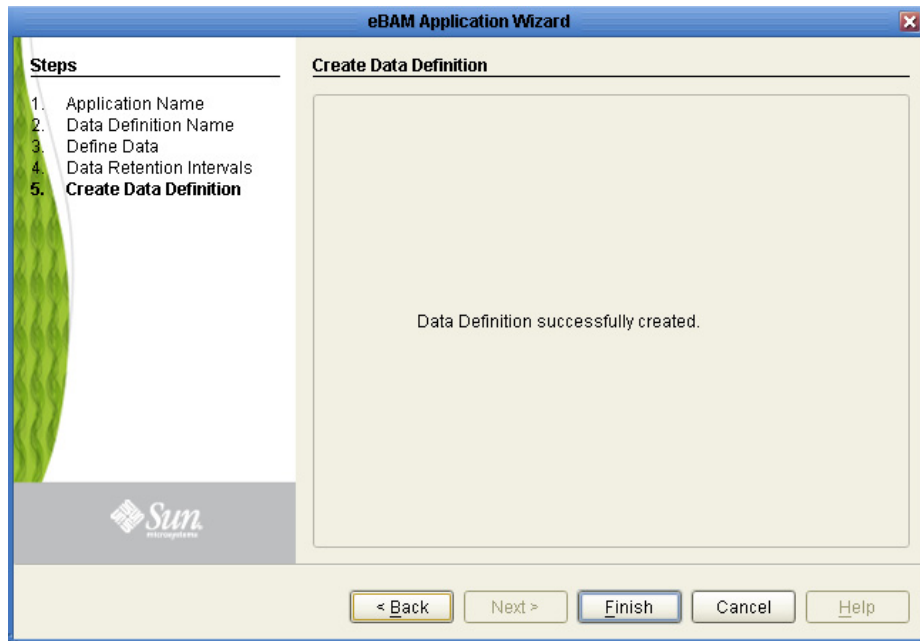
For example, to keep data for three months, set **Retain Last** to **3 Months**; to remove expired data every week, set **Data Cleanup Schedule** to **7 Days**.

Note: *If data retention is enabled for a data definition, it has only one web service: **add**. If it is bypassed, the data definition has three web services: **add**, **upsert**, and **delete**. Once data retention has been specified for a particular data definition, it cannot be un-specified: data retention parameters settings can be modified, but not disabled.*

When you have specified the settings, or have decided to bypass them, click **Next**. The eBAM Application - Create Data Definition wizard appears, and informs you: “Data Definition successfully created” (see Figure 12).

Even though the procedure informed you that the data definition has been created, the eBAM Application - Create Data Definition wizard allows you to go back and review your choices. As needed, click **Back** and **Next** to return to a step and make changes.

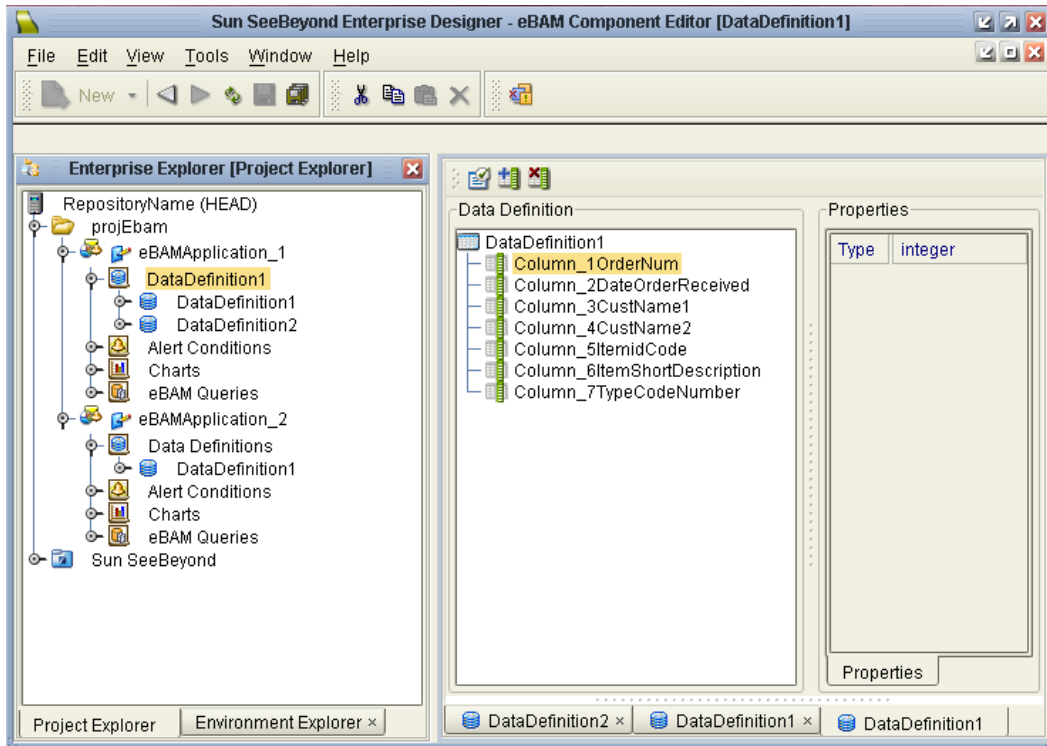
Figure 12 eBAM Application - Create Data Definition



- 6 Click **Finish** when you are satisfied with the data definition.

In the left pane, the Project Explorer tree displays a new data definition under the eBAM application. On the right side (the canvas), the eBAM Component Editor opens to display the data definition.

Figure 13 eBAM Component Editor Showing Data Definition



Important: Any time you complete a new data definition, it is good practice to use the *Show Sample Data* tool to double-check the data definition against sample data. See [“Importing Sample Data” on page 39](#).

To create and configure a new data definition in an existing application

This procedure is almost exactly the same as the one you followed when you created an eBAM application and its first data definition.



- 1 In Enterprise Designer, in the project tree (left pane), open an existing application, right-click its data definitions subfolder, and, on the pop-up context menu, click **New Data Definition**. The eBAM Application Wizard appears.
- 2 On the eBAM Application - Enter the Data Definition Name wizard, either keep the default name (DataDefinition2) or type a new name and then click **Next**. The eBAM Application - Define the Data Definition Elements wizard appears.
- 3 The rest of the procedure is the same as the preceding one. If you need additional information see [“To create a new eBAM application and configure its first data definition” on page 34](#).

Important: Any time you complete a new data definition, it is good practice to use the *Show Sample Data* tool to double-check the data definition against sample data. See [“Importing Sample Data” on page 39](#).

4.2.2 Modifying Data Definitions

Before modifying an existing data definition, refer to Table 5 to gain an understanding of what can be changed.

Table 5 Ways of Modifying Data Definitions

To change this...	... do this:	Notes and Caveats
Name of the data definition itself	In the Project Explorer tree, right-click the data definition name. Alternatively, in the Component Editor, click the data definition name, pause, and then click it again.	Data definition name changes are propagated automatically to any queries that use the data definition.
Data retention properties	In the Component Editor, click the data definition name and, in the Properties pane, change settings for Data Retention Time and/or Data Cleanup Interval.	Changes to data retention have no effect on any other properties of the data definition. If data retention is specified for a particular data definition, it cannot be removed.
Number of columns	In the Component Editor, use the buttons on the tool palette to add or remove columns:	Do not add or remove columns if the data definition is already in use by any queries: the downstream impact can be severe, and very difficult to diagnose and remedy.
	 Add Column	
	 Remove Column	
Column name	In the Component Editor, click the column name, pause, and then click it again.	Do not rename columns if the data definition is used by any queries.
Column datatype	In the Component Editor, click the column name and, in the Properties pane, change the setting for Type.	Do not modify a column's data type if the data definition is used by any queries.

4.2.3 Importing Sample Data

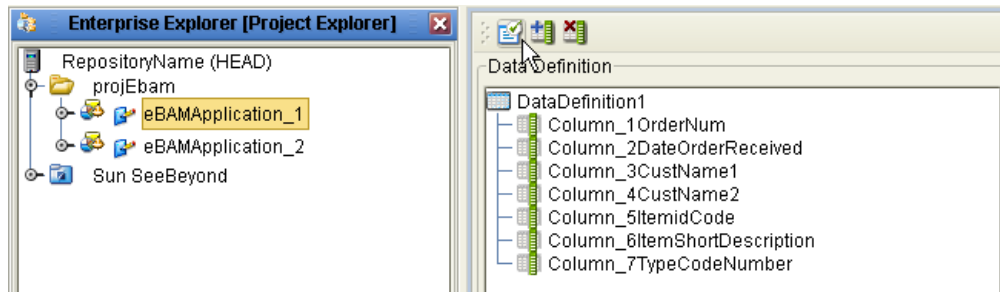
Sample data is useful when creating and testing your eBAM applications. Use imported sample data to validate data definitions, queries, charts that are created off of the queries, and alerts. Validation allows you to quickly see if the output is as you expected or if it is askew and needs reworking.

To validate the metadata definition

- 1 In the eBAM Editor, on the tool palette, click the **Show Sample Data** icon (see Figure 14 for the icon location on the eBAM Editor).

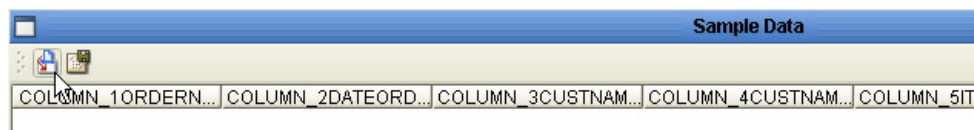
Note: *The eBAM Editor is the tool you use to create and edit your eBAM applications. The actions available at any given time vary depending upon what you have selected in eBAM.*

Figure 14 Show Sample Data Icon on eBAM Editor Tool Palette



The Sample Data pane appears below the main eBAM Editor.

Figure 15 Portion of the Sample Data Window



- 2 On the Sample Data tool palette, click the **Import** icon (see Figure 15). The Select File for Import wizard appears.
- 3 In the File Import wizard's **Select File** step, browse to the location of the sample data file and select it. The act of bringing the data file into eBAM creates a flatfile table that is located inside your Enterprise Designer directory (for example: **C:\JavaCAPS513\edesigner\usrdir\eBAMflatfiles ...**).
- 4 In the formatting step, make the appropriate choices (see Table 6), and then click **Next**.

Table 6 Specifying Formatting Type and Encoding for Imported Sample Data

Item	Choices	Notes
Encoding scheme	ASCII (ISO646-US)	ASCII — 8-bit encoding, roman characters.
File format	Delimited Fixed-width	Delimited files specify escape characters to distinguish fields in a record and one record from the next; fixed-width files specify a preset length for each record.

- 5 In the parsing step, make the appropriate choices for delimited data (see Table 7) or fixed-width data (see Table 8), and then click **Next**.

Table 7 Parsing Information for Imported Sample Delimited Data

Item	Value	Notes
Default SQL Type	float integer char timestamp varchar	float — Number with decimal point, such as: -99.99 integer — Number without decimal point, such as: -16777216 char — Single printable character, such as: A timestamp — Date and time in <i>ccyy-mm-dd hh:mm:ss</i> format, such as: 2006-12-31 23:59:59 <ul style="list-style-type: none"> Note that Sun has expanded what the timestamp accepts such when you use <i>eInsight's</i> “get current time” option, which uses a slightly different format (2006-12-31T23:59:59.99z); the extra characters are stripped out and the format is converted to the default. varchar — Text string, such as: This is a varchar example. <ul style="list-style-type: none"> Note that varchar is the default, works in most cases, even for all other data types, and should always work for you.
Record Delimiter	{newline (lf)} {cr} {cr-lf}	newline — each new line starts a new record. cr — each carriage-return starts a new record. cr-lf — each carriage-return+linefeed starts a new record.
	You can also type in a character or control character; for example, to specify TAB, type in: \t	
Field Delimiter	{comma} {tab} {pipe}	{comma} — each instance of , starts a new field. {tab} — each tab character starts a new record. {pipe} — each instance of starts a new field.
Text Qualifier	none " '	You can indicate whether text is distinguished by quote marks; only double quotes or single quotes are supported.
First line contains field names?	False True	You can indicate whether the first record of the data consists of labels for the fields. When this is set to true , the first line is used to name the fields, instead of being imported as data.

Table 8 Parsing Information for Imported Sample Fixed-Width Data

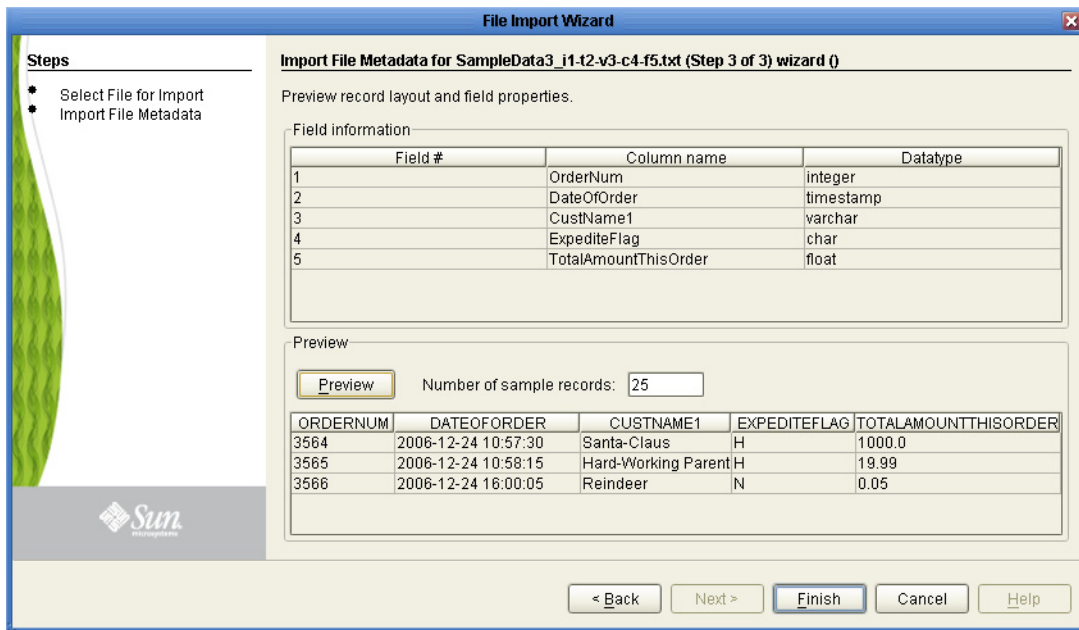
Item	Value	Notes
Default SQL Type	float integer char timestamp varchar	float — A number with a decimal point, such as: -99.99 integer — A whole number, such as: -16777216 char — A single printable character, such as: A timestamp — A date and time in <i>ccyy-mm-dd hh:mm:ss</i> format, such as: 2006-12-31 23:59:59 <ul style="list-style-type: none"> Note that Sun has expanded what the timestamp accepts such when you use <i>eInsight's</i> “get current time” option, which uses a slightly different format (2006-12-31T23:59:59.99z); the extra characters are stripped out and the format is converted to the default. varchar — Text string, such as: This is a typical text string. <ul style="list-style-type: none"> Note that varchar is the default, works in most cases, even for all other data types, and should always work for you.

Table 8 Parsing Information for Imported Sample Fixed-Width Data (Continued)

Item	Value	Notes
Record Length	0	To override the default, type in a nonzero number.
HeaderBytesOffset	0	To override the default, type in a nonzero number.

- 6 In the next step, verify the record layout and field properties in the Field information pane, enter the number of sample records to read and display in the preview pane, and then click **Preview**. See Figure 16, or [Figure 50 on page 84](#).

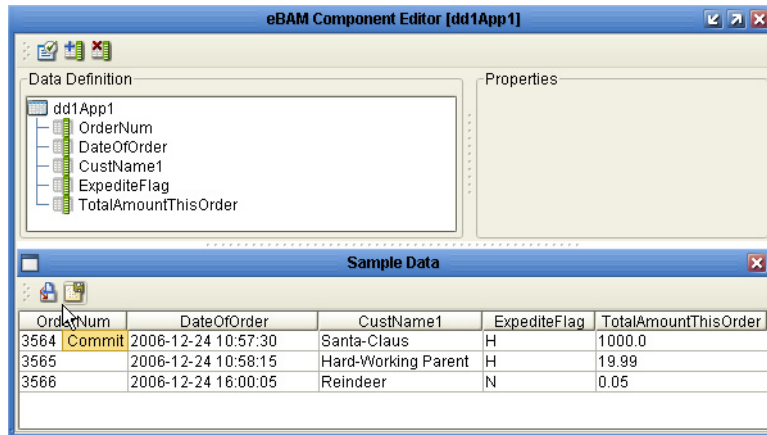
Figure 16 Show Sample: Preview of Field Layout



If the sample data is valid (that is, in accord with the layout, encoding, formatting, and properties you specified), the Preview pane displays a table whose rows are the first few records, with each column headed by the data element name.

- 7 As needed, you can click **Back** and **Next** to return to a step and make changes to layout or formatting, or specify a different input file. When satisfied, click **Finish**. The entire data set is parsed and displayed in the Sample Data pane. For examples, see [Figure 51 on page 85](#) and Figure 17.

Figure 17 Sample Data Displayed in eBAM Component Editor



- 8 When ready, click **Commit** to create your sample.

This allows the sample data to be used in a query, but only for design time; the “remembered” data has no effect whatsoever on run time.

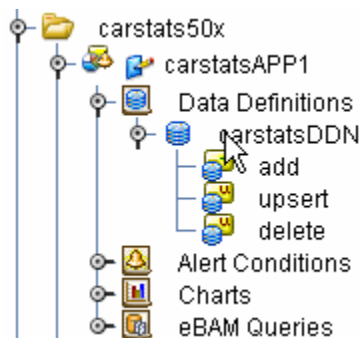
Note: You can edit-in-place any of the contents of the sample data cells.

4.2.4 Add, Update, and Delete Data From the eBAM Data Store

Once an eBAM application is created, it makes available a web service named **add** that can be used in an eInsight business process. For an example, see [Figure 73 on page 107](#) and [“Adding New Data” on page 123](#).

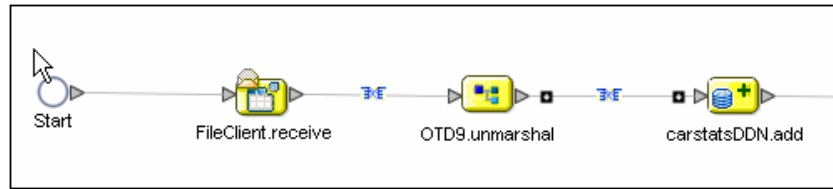
If you use data retention, only the **add** web service is available. However, if you do not have data retention, three web services become available. In addition to **add**, the two other web services are **upsert** and **delete**. In our example (see [Figure 18](#)) the eBAM application is **carstatsAPP1** and the DDN is **carstatsDDN**. The three web services are under **carstatsDDN**.

Figure 18 Three Data Definitions



You use these web services in an eInsight business process by dragging them onto the canvas (see [Figure 19](#)).

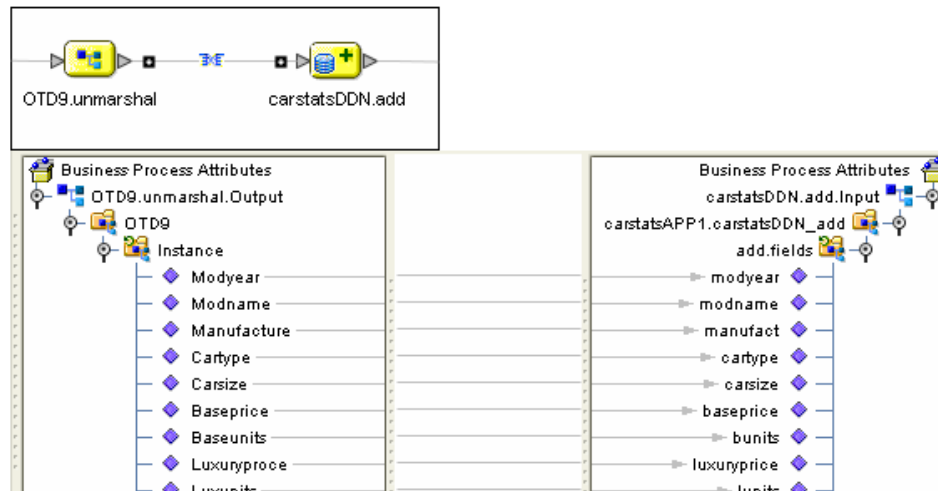
Figure 19 Example of Dragging an Add Web Service



The ADD Web Service

The **add** web service has inputs for every data column in the DDN. In the following illustration (see Figure 20) we see the OTD with nine fields. It is connected to the **add** service via a business rule.

Figure 20 OTD with Nine Fields Connected to an Add Service



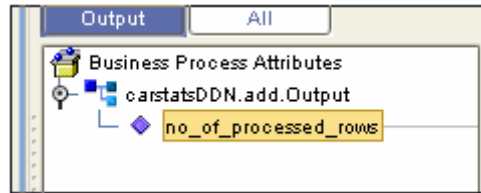
In the eInsight Business Rules Designer, when you open all subnodes beneath the `<appname>_add.Input` node, the fields under the **add.fields** element display the data definition fields to which data can be mapped. Mapping data into these fields during run time causes eBAM to add additional records to the dataset used for its charts, queries, and alerts.

In the example, the User Defined OTD had a repeating node element, which contained nine fields. The nine fields were mapped to the **add.fields** node of the **carstatsAPP1.carstatsDDN_add** node of the **carstatsDDN.add.Input**.

The **add** service also generates output (see Figure 21). Every time the **add** service is executed, it returns an integer value of the number of rows that were added to the table. There is no requirement to use this output; it is there as a convenience if desired.

Note: *upsert* and *delete* also have *no_of_processed_rows* available as an output.

Figure 21 Add Service Output



The UPSERT Web Service

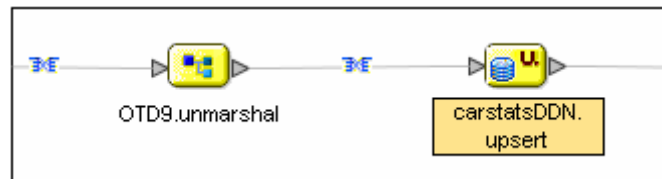
Use the **upsert** service to modify existing records, or to add new records.

The **upsert** web service for an eBAM data definition has two properties. The first property is a group of all fields in the data definition, and the second property, the WHERE clause, is used to define the condition which determines whether an existing record will be updated or a new record inserted. The condition is defined using standard SQL notations.

Note: *The WHERE clause must follow SQL syntax conventions.*

To modify an existing record, drag the **upsert** onto the canvas, connect to the OTD unmarshal, and add a business rule (see Figure 22).

Figure 22 Example of Attaching Upsert to an OTD Unmarshal



At this point you have not mapped anything (see Figure 23).

Figure 23 Example of an Unmapped OTD



Begin mapping the fields. You can map every one of them, or you can choose to map some of them. The fields you map determine which columns are updated in the record (see Figure 24 and Figure 25).

Figure 24 Example of a Mapped Field

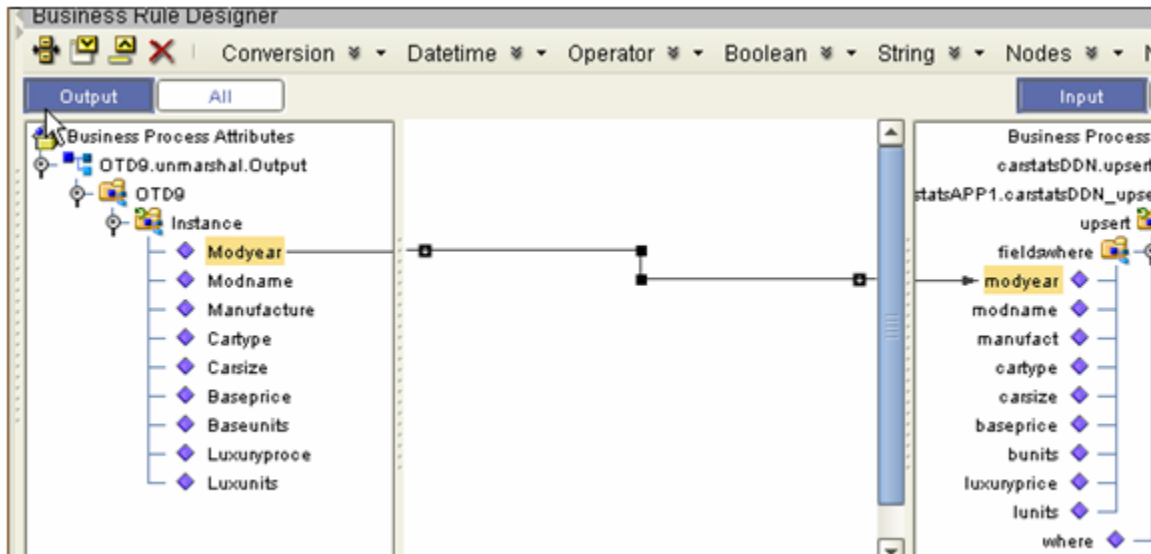
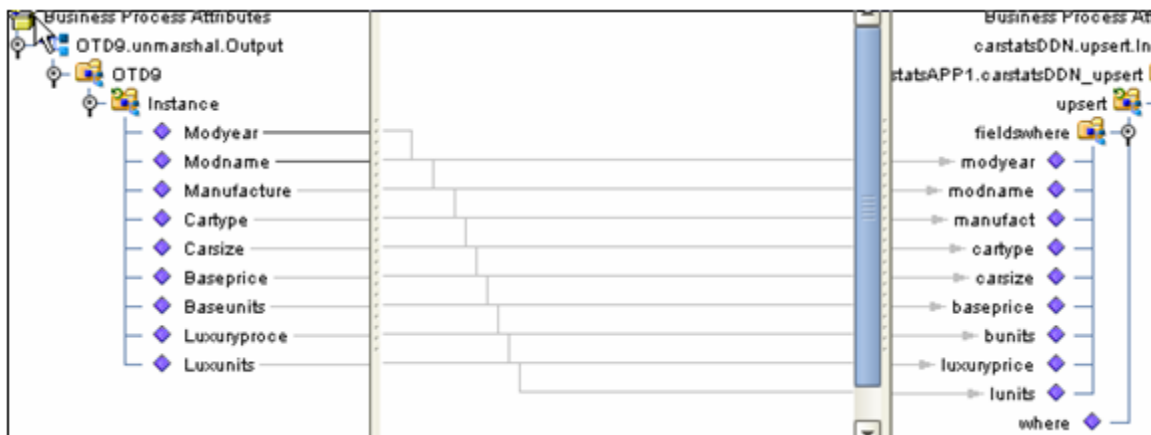
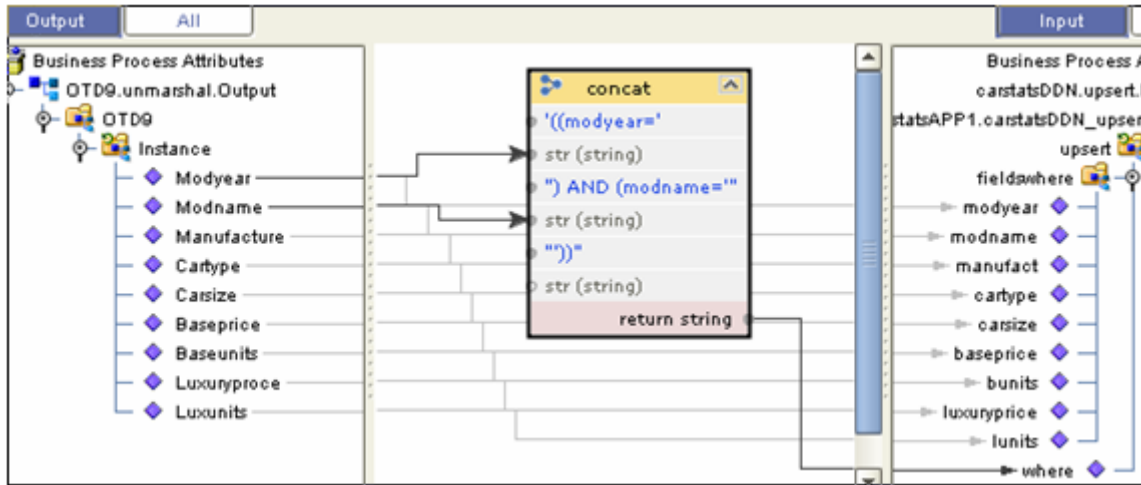


Figure 25 Example With All Fields Mapped



After all the fields have been mapped, you are ready to add a WHERE clause; in this example: **((modyear=<ModYear>) AND (modname=<Modname>))**. We determined the column names 'modyear' and 'modname' from examining the original DDN. The variable values came from each record as it appeared in the unmarshalled data (see Figure 26).

Figure 26 Mapped Example with WHERE Clause



The condition is executed on a record-by-record basis from the unmarshalled output of the OTD. If a matching record, as defined by the condition in the WHERE clause, is found in the existing eBAM table, an update occurs. If there is no match for the WHERE clause, then a new record is added, or inserted, into the eBAM database.

If a WHERE clause is not used, the **upsert** web service acts exactly like the **add** service.

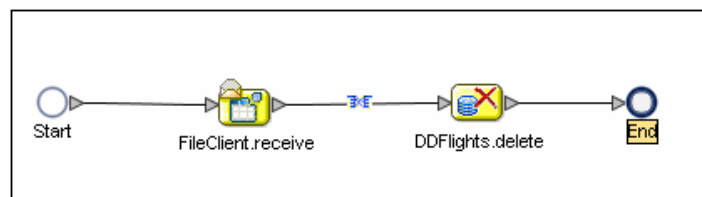
Note: As with *add* and *delete*, *upsert* also has *no_of_processed_rows* available as an output.

The DELETE Web Service

The **delete** web service works in a similar way to how the **upsert** service works. A WHERE clause is specified via the business rules mapper. Any rows that meet the WHERE clause condition are deleted from the eBAM database table.

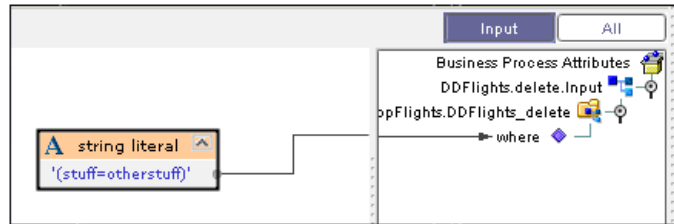
To begin the process of deleting rows from an eBAM database table, drag the delete service onto the canvas and attach it (see Figure 27).

Figure 27 Example of Dragging a Delete Web Service



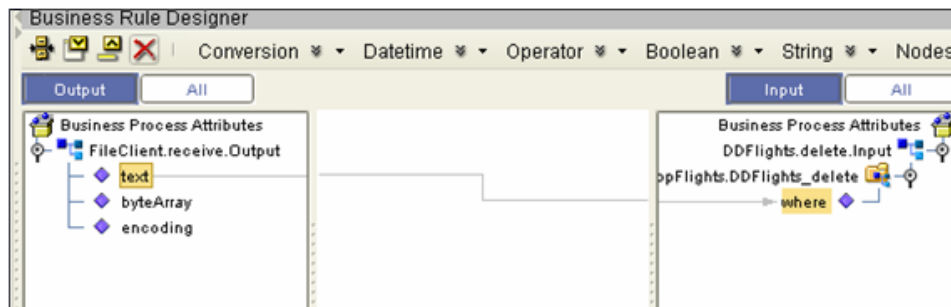
In our first example, when the file triggers the business process, a WHERE clause is triggered that is the same every time (see Figure 28).

Figure 28 Example of a WHERE Clause



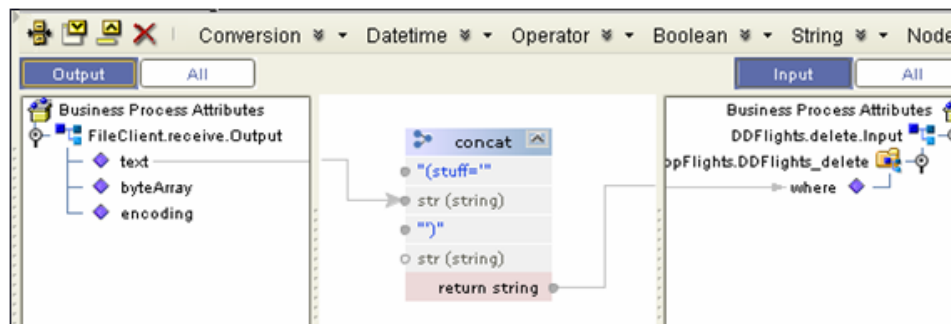
In the second example, the text contained in the file becomes the entire WHERE clause (see Figure 29).

Figure 29 Example Where the Text Becomes the Entire WHERE Clause



In the third example, the text in the input file acts as a variable value, a runtime argument if you will, and is combined into another string to form the WHERE clause (see Figure 30).

Figure 30 Example Where Text Becomes a Variable Value



If a **delete** web service is invoked without using a WHERE clause, all the rows in the table are deleted. The table remains, but without any records in it.

The **delete** web service deletes individual records, but it does not delete the table definition, and it cannot drop a table.

Note: As with *add* and *upsert*, *delete* also has *no_of_processed_rows* available as an output.

4.3 Database Considerations

Although you can use eBAM without a formal database (see [“Using the Default Flatfile Database” on page 49](#)), best practice is to use eBAM in conjunction with an Oracle database, as discussed in [“Using eBAM with an Oracle Database” on page 50](#). Regardless of whether you use a “virtual database” flatfile or a actual Oracle database, setup and configuration steps are minimal, and you do not need to run any database scripts.

Note: *The database considerations discussed in this section are unrelated to the eInsight database used for persistence; for information on the eInsight database and the eInsight engine, see [“Persistence and Monitoring” on page 178](#).*

4.3.1 Using the Default Flatfile Database

You can deploy eBAM applications using a “virtual database” wherein data is persisted via a flatfile. This means you do not need to take any special setup or configuration steps except to supply a valid writable directory name as a value for “Application Workspace Directory” in the properties of each application server where you deploy the application.

In this case, each eBAM application uses a “virtual database” that is specific to the Project name and deployment name. Therefore, if you activate the same application using two or more deployment profiles, or if you copy and paste an eBAM application to a different project, each application deployment instance is unaware of any others, and data is not shared from one to another. Similarly, if you rename a Project, activation of this “new” Project creates a new workspace that is unaware of any data gathered under the previous name.

Note: *This is the same database technology used in design time.*

Flatfile Database Considerations

- Releasing the Database Lock

Occasionally after an unplanned integration server shut down, or rarely after redeploy of the Project, the flatfile database becomes inaccessible. When this happens an error in the log file indicates that you need to delete the **lockfile.txt** file, which is located in the

`/<Application Workspace Directory>/<projectname_deploymentname>` directory.

- Deleting the eBAM data Repository

To completely clean out your eBAM data Repository, delete the entire `/<Application Workspace Directory>/<projectname_deploymentname>` directory, and then restart your Logical Host.

4.3.2 Using eBAM with an Oracle Database

To use eBAM in conjunction with an Oracle database, you must have installed the Oracle eWay, you must be able to access the Oracle database with appropriate user privileges, you must have one or more well-configured Oracle externals in your environment, and you must connect your eBAM application instances to Oracle externals in the Connectivity Map.

This section provides a brief summary of the default configuration used by the Oracle eWay; for full information, see the *Sun SeeBeyond eWay™ Adapter for Oracle User's Guide*.

Oracle Database Considerations

eBAM applications that use an Oracle database, as opposed to the “virtual database” flatfile (see 4.3.1 “[Using the Default Flatfile Database](#)” on page 49) have the following characteristics:

- At run time, a database connection is established if and only if the database is running and the configuration parameters of the Oracle external are validated (in other words: servername, databasename, user ID, and password).
- Multiple eBAM applications, in the same or different Java CAPS Projects, can share the same Oracle database.
- The Oracle database user should have the ability to create and drop tables.
- eBAM data in the Oracle database is stored according to database user information. Therefore, renaming the Project or deployment profile has no effect on data known to the application, and different instances of the same eBAM application—such as a single application deployed via multiple deployment profiles to the same environment, or an application copied into another Project but deployed to the same environment, or even an application imported to another Repository but deployed in such a way as to reference the same database—share the same data, just so long as they point at the same database via the same user ID and password. Records written by one such application instance are visible and accessible to all such instances.
- Tables are created automatically as needed, and accessed during run time according to the parameters of the application's data definition(s).

There are three types of tables created by eBAM:

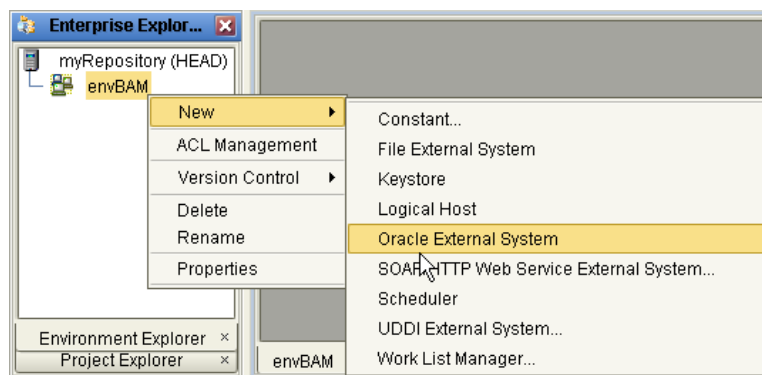
- ♦ *Alert*: A table that contains timestamp information on when alerts have been sent.
 - ♦ *EBAMINT11_11*: A table that contains information eBAM uses for table creation.
 - ♦ *Data tables*: A separate table is created for each data definition in the eBAM application.
- Each data definition is assigned a unique internal name by eBAM, and this internal name is used as the table name during run time. To determine this name, open a query that is using that DDN and select the **Shows SQL** option. Thus, if you ever need to learn the name of the table (such as for troubleshooting purposes), you can read it from the SQL of the query.

- If you are using Oracle 10g, tables are actually *dropped*, instead of being deleted or purged from the database. Purging or archival of stale data must be managed by your Oracle database administrator. (Standard practice is to use this feature as a recycling bin: The tablespace is allowed to grow until it reaches the autogrow limit, and then the oldest version of the table is purged.)
- Under certain circumstances eBAM will drop tables when a data definition is modified, but it does not drop tables when you undeploy a Project. When eBAM is no longer in use, an Oracle database administrator (or your company's equivalent) must manually delete any tables that are no longer needed.
- The computer which runs the Application Server does not need to have an Oracle client installed.

To create and configure a new Oracle external in the Environment

- 1 In Enterprise Designer, Environment Explorer tab, right-click the environment and, on the pop-up context menu, point at New, and click Oracle External System. See Figure 31.

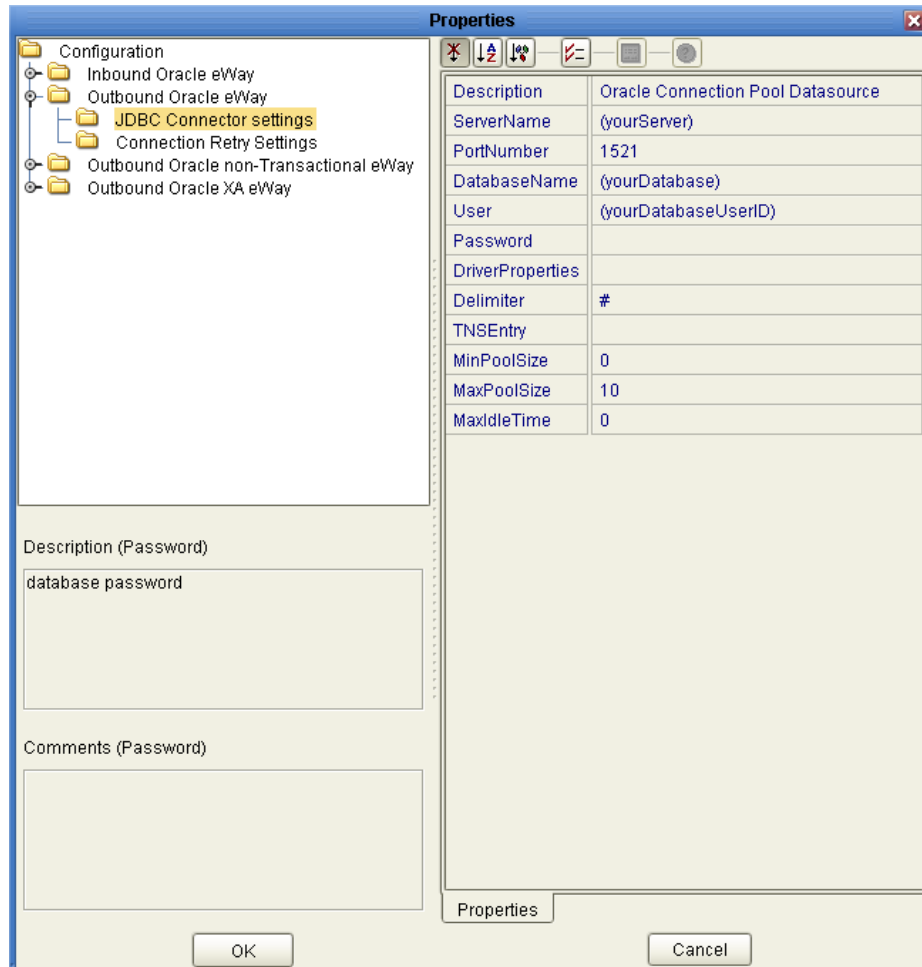
Figure 31 Creating a new Oracle External



- 2 Provide a name and click **OK**.
- 3 In the Environment tree, right-click the newly created Oracle external and, on the pop-up context menu, click **Properties**.
- 4 In Outbound (under "JDBC Connector settings"), configure the properties appropriately. At a minimum, you need to supply valid settings for **ServerName**, **DatabaseName**, **User**, and **Password** (see Figure 32). The Properties dialog box appears.

Note: For complete information on the configurable properties of the Oracle external, refer to the Sun SeeBeyond eWay™ Adapter for Oracle User's Guide.

Figure 32 Configuring an Oracle External

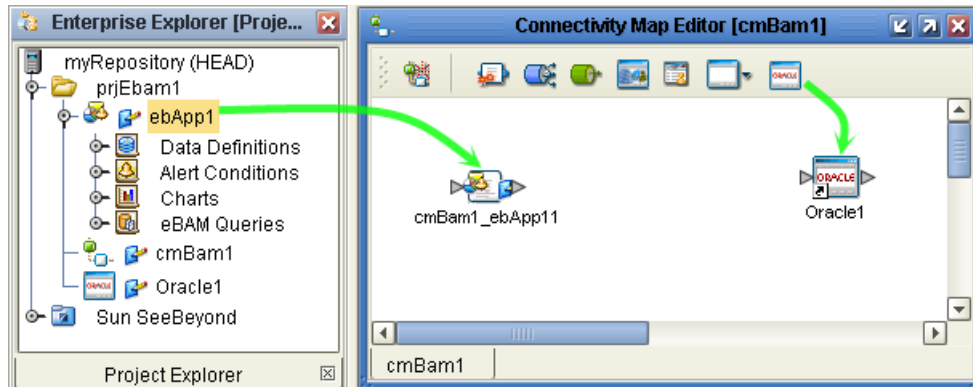


- 5 When finished setting the properties, click **OK**.

To connect an eBAM application to an Oracle external on the Connectivity Map

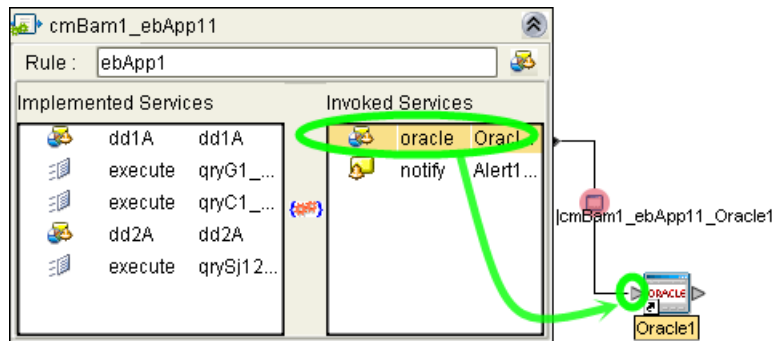
- 1 In Enterprise Designer, if you have not already done so, drag an instance of the eBAM application from the Project Explorer tree onto the Connectivity Map, and drag an instance of the Oracle external from the tool palette down onto the Connectivity Map. See Figure 33.

Figure 33 eBAM Application and Oracle External on a Connectivity Map



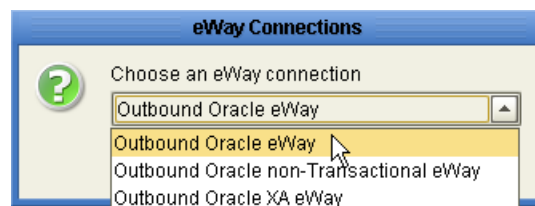
- 2 Double-click the eBAM application instance and drag its Oracle service (on the right side, under Invoked Services) to the inbound arrow of the Oracle external. The connection is created, but the Oracle eWay appears as a red dot, since it has not yet been configured. See Figure 34.

Figure 34 Connecting an eBAM Application's "oracle" Service to an Oracle External



- 3 Double-click the red dot ("cmBam1_ebApp1_Oracle1" in the example in Figure 34). The eWay Connections drop down list that appears (see Figure 35).

Figure 35 Choosing the Outbound Oracle eWay Connection Type



- 4 Select **Outbound Oracle eWay** and then click **OK**. This creates the Outbound Oracle eWay.

Note: For complete information on the configurable properties of the Oracle eWay, see the *Sun SeeBeyond eWay™ Adapter for Oracle User's Guide*.

- 5 In the Properties dialog box, unless you want to modify the default configuration parameters, simply click **OK**. (The default configuration is the simplest one.)

Using eBAM Queries

eBAM applications use queries. Before setting up and using queries in eBAM you should understand how they function. The step-by-step instructions for creating, configuring, and using queries in eBAM applications assume you are already familiar with Enterprise Designer and such eBAM concepts as data definitions.

What's in This Chapter

- [“About eBAM Queries” on page 55](#)
- [“Setting Up the Application's Queries” on page 56](#)
- [“Using the Condition Editor” on page 62](#)

5.1 About eBAM Queries

Queries are used in three ways by eBAM:

- As part of a Chart definition
- As part of an Alert definition
- As a standalone web service, embedded in an eInsight business process

In SQL terms, an eBAM query is a SELECT statement, and each condition is a WHERE clause. When the Query wizard prompts you to select the data definition(s), it uses them to create the SELECT statements; when you use the Condition Builder, the Query Editor adds a WHERE clause to the SQL statement.

Queries come in three forms: *summary*, *category*, and *general*:

- Summary queries use only the first line of the returned data, and are used for simple charts: meters, pie charts, and simple trafficlights and bar charts.
- Category queries provide automatic data grouping into categories, and are used by trafficlight arrays, pie charts, and bar charts.
- General queries create SELECT statements without any restrictions on returned data (unlike the summary query) and without restrictions on format (unlike the category query), and they allow you to use runtime arguments as variables in their WHERE clauses. However, general queries cannot be used for charts.

5.2 Setting Up the Application's Queries

Each eBAM query consists of the following:

- A condition type, a name, and one or more data definitions. These are set when you run the wizard that creates the query. See [“To create a new query”](#) below.
- Optionally, one or more runtime arguments; see [“To add or modify runtime inputs to a configured query” on page 57](#) and [“Using the Condition Editor” on page 62](#). (In SQL terms, a “runtime argument” is a variable in a WHERE clause.)
- One or more conditions, using fields in the data definition; see [“To use runtime arguments in the Condition Editor” on page 58](#) and [“Using the Condition Editor” on page 62](#). (In SQL terms, a “condition” is a WHERE clause.)
- One or more fields in the dataset, with appropriate mappings to them from operators and data definition fields; see [“To configure the dataset, mapping operators and data definition fields to its fields” on page 59](#). (The term “dataset” refers to the output columns of the SQL SELECT statement.)
- Optional settings for the query's characteristics as a whole. These consist of Group-By (aggregation) mappings and Order-By (sort) mappings; see [“To configure Group-By and Order-By characteristics of the entire query” on page 60](#).

To create a new query

- 1 In the project tree, under the eBAM application, right-click **eBAM Queries** and, on the popup context menu, click: **New eBAM Query**
- 2 In step 1 of the Query wizard, specify condition type (general, summary, or category) and a name.

After a query is created, you can change its name up until the time it is used as the basis for a chart or alert, but you cannot change its condition type.

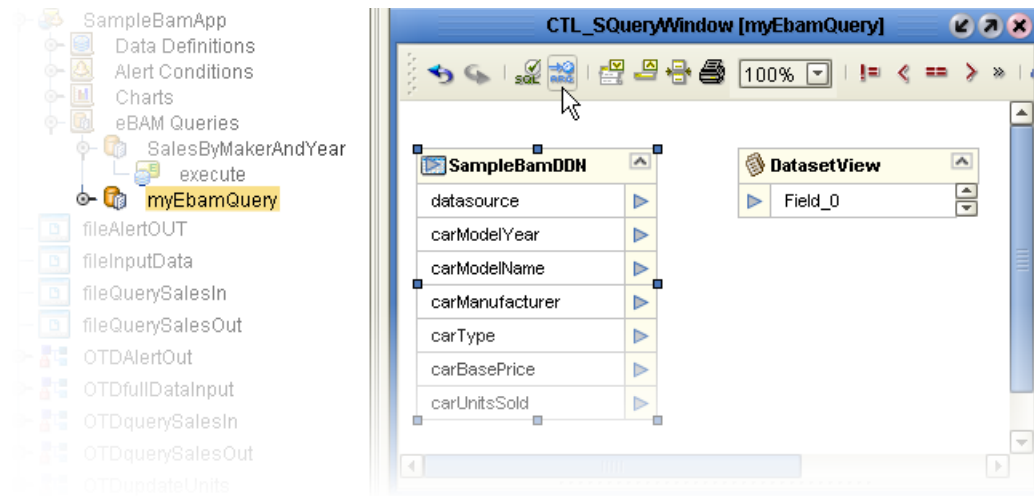
- 3 In step 2 of the wizard, select one or more check boxes for the data definition.

After a query is created, you cannot add more data definitions to it.

- 4 Click **Finish**.

The project tree displays the new object under eBAM Queries, and the Component Editor opens a query-editing canvas that displays the data definition(s) on the left side of the canvas and an empty dataset view on the right side. See Figure 36.

Figure 36 Component Editor, Showing Newly Created (Unconfigured) Query



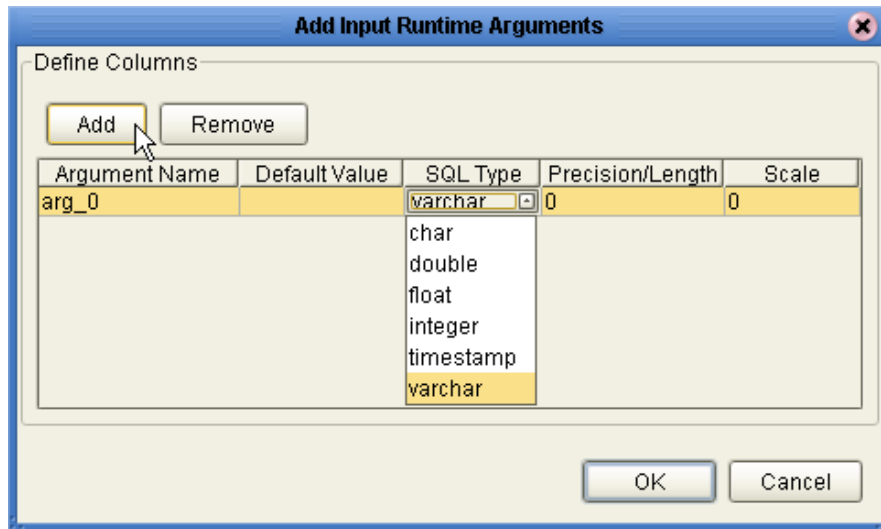
Note: The tool palette for the query-editing canvas is mostly the same as for the Condition Editor. [Table 10 on page 63](#) explains the graphical controls, and [Figure 44 on page 64](#) lists the SQL operators. In Figure 36, the cursor points at the additional tool in the palette, **Add/Edit Runtime Inputs**; this tool is explained in detail in the following procedure.

To add or modify runtime inputs to a configured query

The Add/Edit Runtime Inputs tool allows the eBAM application to accept input at run time. In SQL terms, this tool allows you to make variables accessible to your WHERE clauses. Conditions are met or unmet -depending on the data values at run time.

- 1 In the Component Editor, with the query-editing canvas active (see [Figure 36 on page 57](#)), on the toolbar palette, click the **Add/Edit Runtime Inputs** tool.
- 2 In the Add Input Runtime Arguments dialog box (see Figure 37), click **Add** as many times as needed to create additional runtime arguments.

Figure 37 Adding Input Runtime Arguments to eBAM Queries



- 3 Double-click each argument name and modify it appropriately.

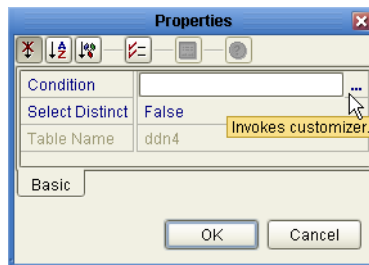
Note: Do not use field names that match SQL reserved words, irrespective of uppercase, lowercase, or mixed case. See [“SQL Reserved Words” on page 187](#).

- 4 As needed, change each argument's SQL type from varchar to one of the following: char, double, float, integer, or timestamp.
- 5 As needed, set default values for arguments.
- 6 When finished, click **OK**.

To use runtime arguments in the Condition Editor

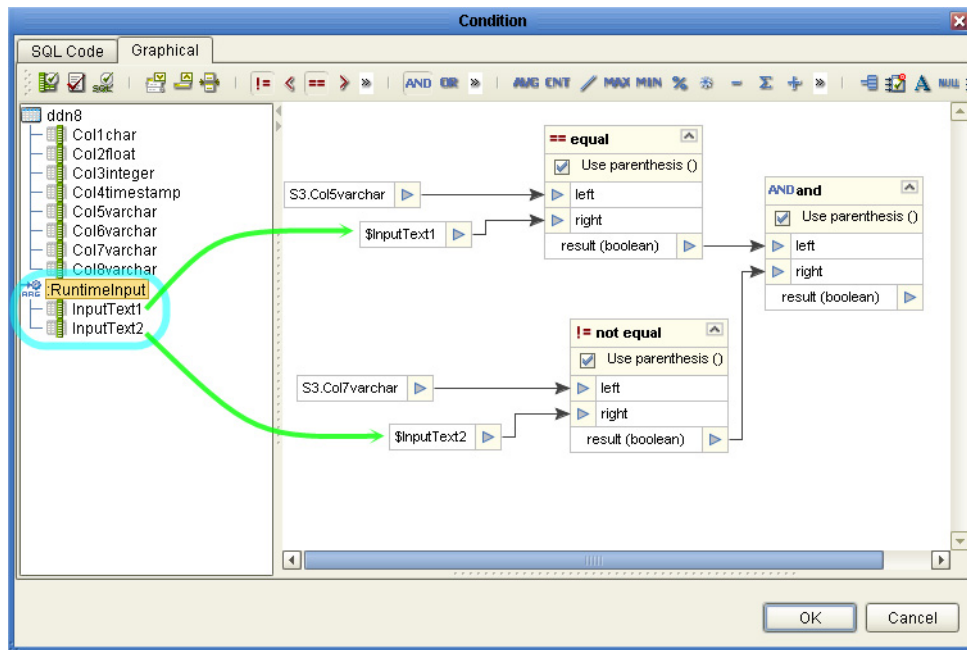
- 1 In the Component Editor, with the query-editing canvas active, right-click a data definition and click menu item **Properties**. In the Properties dialog, click **Condition**, and then click the ellipsis [...] to open the Condition editor. See Figure 38.

Figure 38 Configuring a Condition



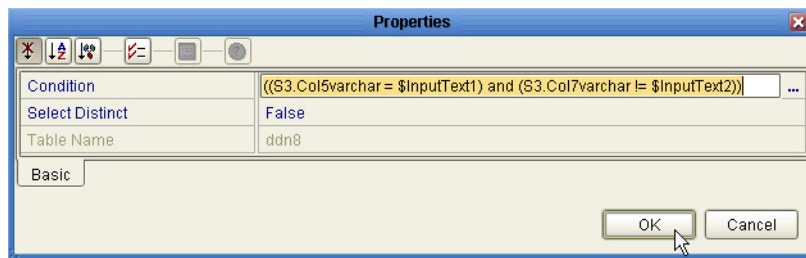
- 2 For general conditions, see [“Using the Condition Editor” on page 62](#). If you defined input runtime arguments (see previous procedure), their fields are also available to you, on the left side of the Condition editor. For a query to make use of the runtime input, one or more of the conditions must involve a runtime argument. See Figure 39.

Figure 39 Configuring a Data Definition Condition Using Runtime Arguments



- 3 When finished, click OK. Then, inspect the resulting SQL string displayed in the Properties dialog box (see Figure 40). When you are satisfied with it, click **OK**.

Figure 40 SQL String in Properties Dialog Box

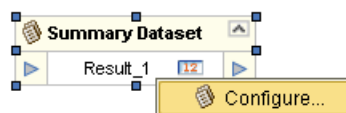


The condition is created, and for each column in the data definition involved in the condition, a small red “filter” icon appears to the right of the column name.

To configure the dataset, mapping operators and data definition fields to its fields

- 1 In the Component Editor’s query-editing canvas, right-click the dataset object and, on the pop-up context menu, click: **Configure** (see Figure 41).

Figure 41 Configuring the Dataset



- 2 In the **Configure Dataset View** dialog, click **Add** as many times as needed to accommodate all the columns you will need. (For General queries, you can only

add columns; for Summary queries, you can only add columns of type Value; and for Category queries, you can only add columns of type Series.)

- 3 Double-click each field, edit the name, and press ENTER. When finished, click **OK**.

Note: *Do not use field names that match SQL reserved words, irrespective of uppercase, lowercase, or mixed case. See “SQL Reserved Words” on page 187.*

- 4 In the query-editing canvas, construct what you want to appear in the dataset view:
 - ♦ Drag operators as needed from the operator palette (see [Figure 44 on page 64](#)) into the canvas.
 - ♦ Using as your input fields from data definitions (and from runtime arguments, if present), modify and/or combine them via operators as needed, with all output ultimately going into the dataset view.

For examples of this, see [Figure 42 on page 61](#), or see [Figure 59 on page 93](#).

Note: *Data type conversions are done automatically where needed; for timestamps converted to varchar, permit truncation to occur.*

To configure Group-By and Order-By characteristics of the entire query

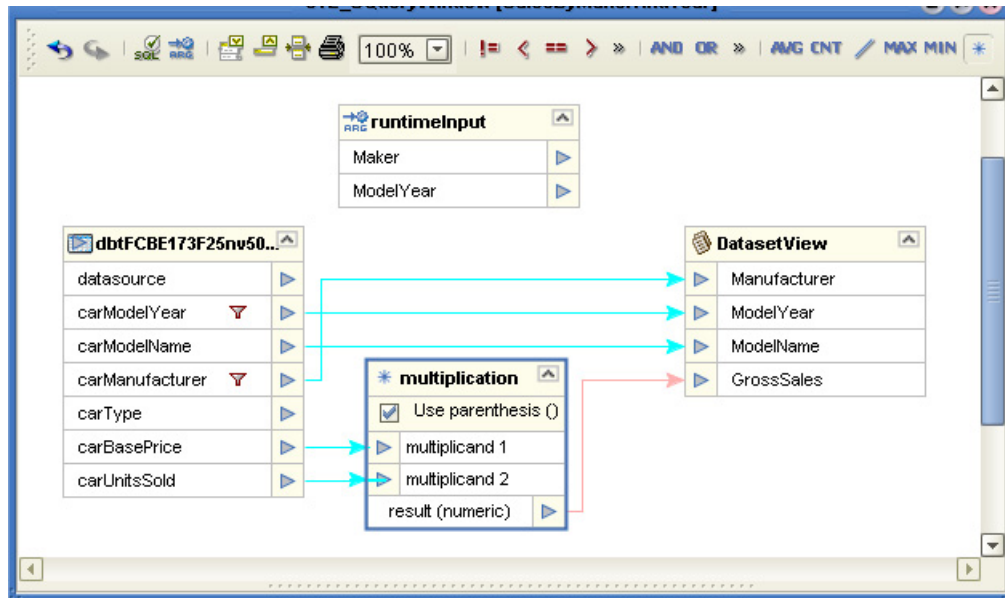
Note: *Summary queries cannot have Group-By or Order-By characteristics. Category queries are always grouped first by the Category column and then by each Series column, and therefore cannot have additional Group-By characteristics.*

- 1 In the Component Editor's query-editing canvas, as needed, drag in a Group-By and an Order-By from the tool palette.
- 2 If you want to establish subordinate groupings or orderings (as opposed to a single grouping/ordering or multiple independent groupings/orderings), right-click the Group-By or Order-By and click **Add** as many times as needed.
- 3 In the query-editing canvas, construct what you want to appear in the DatasetView:
 - ♦ Drag operators as needed from the operator palette (see [Figure 44 on page 64](#)) into the canvas.
 - ♦ Using as your input fields from data definitions (and from runtime arguments, if present), modify and/or combine them via operators as needed, and map the output into an unused entry in a Group-By or Order-By. (The same output can be used in multiple Group-By and/or Order-By entries.)
- 4 If you want to reverse-sort any Order-By, right-click it and choose **Switch Order Direction**.

Example of an eBAM Query Using a Runtime Argument

The sample implementation discussed in [Chapter 8](#) uses a simple stored query with a runtime input argument. Another example is shown in [Figure 42](#).

Figure 42 Example of a Well-Configured eBAM Query



This query takes a seven-field data definition and sets up a filtering condition using two of those fields (carManufacturer and carModelYear), using the runtime input arguments (Maker and ModelYear). The filtering condition is: *If carManufacturer matches the runtime input value supplied for Maker and, at the same time, carModelYear matches runtime input value supplied for ModelYear, then perform the mapping.*

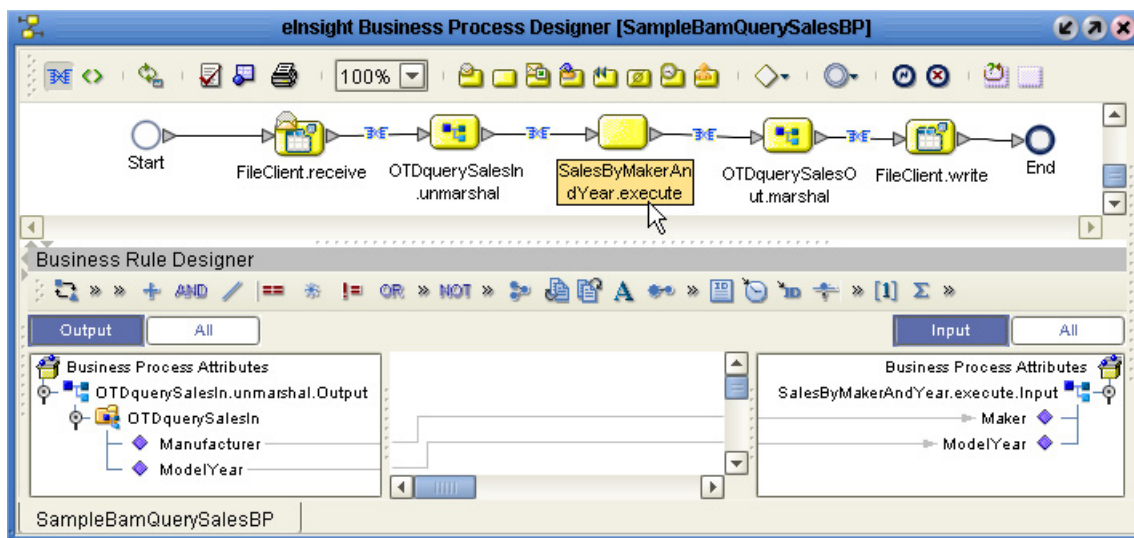
When the query is executed, its output will have four fields: The first three will simply echo the make, model, and year of the data matching the conditions, and the fourth will be a computed KPI (“GrossSales”) that reports the product of BasePrice and UnitsSold.

- The input file (TestQueryIn.*) provided with the sample looks like this:
Ford, 2003
- The output file (TestQueryOut.*), which shows per-model gross sales figures, might look like this when the input arguments are “Ford,2003”:

```
Ford,2003,Crown Victoria,27260750.0
Ford,2003,Escape,12891190.0
Ford,2003,Excursion,43284780.0
Ford,2003,Expedition,28638000.0
```

Runtime arguments used in a query communicate by implementing the query’s **execute** in a business process, as illustrated in Figure 43, and also demonstrated in the sample by business process **bpQryExtractSalesByCarNames**.

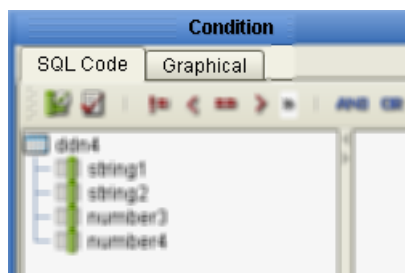
Figure 43 Implementing an eBAM Query in a Business Process



5.3 Using the Condition Editor

The Condition editor constructs filters that constitute conditions on the data definition. Although you can type in (or use CTRL+V to paste) native SQL in text form, the editor offers features that help SQL-adept users avoid making mistakes, while allowing other users with less SQL knowledge to construct statements by purely graphical means.

In the Condition editor, you can switch at will both between the two user modes (SQL Code and Graphical).








- The **SQL Code** canvas allows you to enter native SQL commands and/or to drag elements or operators from the trees on the left.
- The **Graphical** canvas allows you to create conditions simply by dragging elements and operators from the trees on the left. Its tool palette provides several tools, listed in Table 9:

Table 9 Tools Provided in the Graphical Canvas of the Condition Editor

	Show Table Tree allows you to display or hide the left pane, where available runtime arguments and data definition columns are displayed.
--	---

Table 9 Tools Provided in the Graphical Canvas of the Condition Editor (Continued)

	Validate checks the syntactical validity of the graph and, in the Output pane, reports success or any errors found, highlighting all operators that are unsatisfied.
	Show SQL for Condition tries to validate the graph and, if successful, displays the corresponding SQL statement for the entire condition.
	Expand All Graph Icons provides more detail, by showing all input to (and output from) all fields in all operators on the canvas.
	Collapse All Graph Icons provides more screen space by minimizing all operators, showing only the connections to and from them.
	Autolayout All Graph Icons disentangles crossed connections and overlaps, and creates left-to-right flow of input to output.

When a data definition column or runtime argument has a condition placed on it, the query-editing canvas displays a small red funnel (filter) to the right of its name.

Optionally, you may want to click **Show SQL** occasionally to see SQL statements you are creating graphically, or use other graphical design tools listed in Table 10.

Table 10 Tools Provided in the Tool Palette of the Query-Editing Canvas




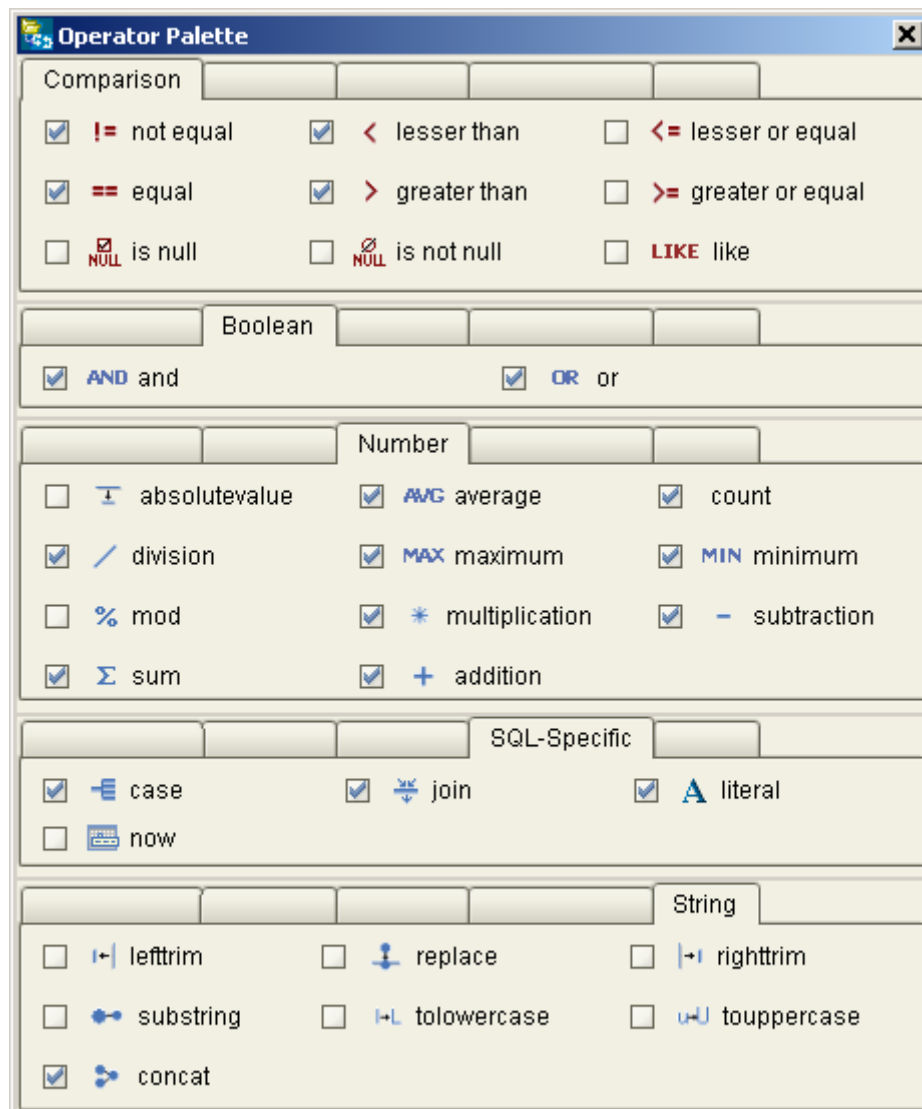
	Undo and Redo allow you travel backward and forward through the sequence of modifications you have made to the graphical canvas.
	
	Print Graph allows you to print the graph, using various scaling options.

Figure 44 Operator Palette for SQL Operations



Using eBAM Alerts

eBAM applications use alerts to deliver a notification when a particular condition is met by a query, and background information is provided to help you better understand the alerts and their conditions. Step-by-step instructions lead you through the processes for creating, configuring, and using alerts in eBAM applications. These procedures assume you are already familiar with Enterprise Designer and such eBAM concepts as data definitions and queries.

What's in This Chapter

- [“eBAM Alerts” on page 65](#)
- [“Setting Up the Application’s Alert Conditions” on page 68](#)

6.1 eBAM Alerts

Alerts provide an easy way for eBAM to automatically transmit data on a periodic basis. You can transmit any data you want.

- For example, you might transmit a predetermined text string along with a computed value and the current date and time: “As of *date*, total sales are *value*” (this is an example of a query that does not rely on a condition).
- More typically, you might transmit data only when a certain condition has been satisfied or a threshold has been reached: “The following dealers are currently underperforming defined as below 75% of quota: *(followed by a list)*.”

Each eBAM alert is based on an eBAM query (for complete information on queries, see [“Using eBAM Queries” on page 55](#)). The purpose of an alert is to provide periodic notification, often only when a particular condition is met by the query. This notification can take the form of an email, or it can trigger an eInsight business process and provide data to it via the notify web service. Alert conditions have the following characteristics:

- *(required)* A name and an associated query.
- *(required)* Values for timing properties: Notification Interval and Resend Frequency.
- *(required only for email alerts)* Values for email properties, such as mailserver name.
- *(depending on the associated query)* At least one key field that is valuated in the dataset of the query.

6.1.1 What You Need to Know About Alerts

The two foundations of an alert are its query and its properties. The query determines the data to be transmitted. The properties determine how often the alert notification is sent, and in what form.

Alert notifications are sent out repeatedly, and continue to be sent whenever the eBAM data store contains data satisfying the alert criteria. The eBAM database contains a separate table that keeps a record of all alert notifications that have been sent.

Constructing an alert requires three considerations:

- **Notification Interval** - This determines how often the alert query is executed.
- **Resend Frequency** - This determines how often to re-send an alert notification that was previously sent.
- **Query to be used** - This determines the data to be sent, as well as any conditions used to filter the data.

Example of an Alert

Here is a simple alert showing the three considerations:

- *Notification Interval* - 5 minutes
- *Resend Frequency* - 30 minutes
- *Query1* - "SELECT column1, column2 WHERE column3 < 45"

Every 5 minutes (the Notification Interval), *Query1* is executed against the current data in the eBAM table, and data satisfying the query criteria is placed into a temporary queue. In this example, if the value in column3 is less than 45, then the values in column1 and column2 are placed into the temporary queue.

In the temporary queue, a comparison is made, using the key values as unique identifiers, to determine whether the new data matches an alert notification that was previously generated and sent:

- If a match is *not* found against a previous alert, then a new alert notification is sent out without delay.
- If a match *is* found, then Resend Frequency comes into play: The time of the most recent alert notification is subtracted from the current time to determine whether it should be resent. In this example, if more than 30 minutes have elapsed, the alert notification is sent out again. Otherwise, after the next Notification Interval has passed, the query is executed again. If the same record appears, it is again compared to the last sent value to determine if it should be resent. This is repeated until 30 minutes have elapsed since the previous one was sent out, at which point the alert is resent, and the last sent value is reset to the current time.

6.1.2 Queries: Conditions and Output Data

All eBAM queries are SQL Select statements. The general form of a Select statement is:

```
SELECT columnA[, columnB[, ...]] FROM tableX [WHERE (condition statement)]
```

When you use the Query Editor, you specify the column or columns and eBAM automatically provides the FROM clause. To create a WHERE clause (if you want one), you must use the Condition Editor to create the condition statement.

Note:

- The WHERE clause, if present, filters the candidate data.
- The SELECT clause specifies the actual output data.

6.1.3 Things to Keep in Mind

The following guidelines can be helpful when you construct an alert:

- An eBAM application can contain many alerts, or only one, or none at all.
- Each alert is based upon exactly one eBAM query. An eBAM query can be used by many alerts, or only one, or none.
- Although any query type can be used, General and Category are the only useful types of query for alerts.
- Alert notifications can only send data selected from the columns in a single query. It is not possible to concatenate queries; in other words, you cannot use the results of a WHERE clause to start a subsequent and separate SELECT statement.
- The query used by an alert need not contain a condition. If the query does contain a condition (in other words, if the query's SQL statement contains a WHERE clause), then the data transmitted is from the columns of the SQL statement.
- The WHERE clause is *not* a trigger; its purpose is to filter down the candidates of data to be transmitted. Triggering of query execution (and the operation of the alert notification) is controlled by the Notification Interval.
- After an alert has been created, the only way to completely disable it is to stop or undeploy the eBAM application, or delete the alert completely and redeploy. Even with an extremely large Resend Value, such as one year, the query is still executed at least once upon startup of the eBAM application. Thus, the query is executed whenever the application is redeployed to a running Domain, or upon restarting a Domain where the application is deployed.

A Negative Example

This is an example of a plausible alert that you *cannot* create using eBAM:

- CarModel is a variable, and several different models might conceivably exceed \$10,000,000 in total sales.
- The condition you want to test is this: For a particular model, CarModelX, does the sum of all sales across all CarDealers exceed \$10,000,000?
- The data you want to transmit is this: A list of all CarDealers that have recorded sales of CarModelX.

The reason this alert cannot be created is that it would require two separate queries to be executed: One query to return a list of all car models whose total sales exceeded \$10,000,000; another query to determine, from the results of the first, which dealers had sold a particular model. This is impermissible, because it would require one query to be embedded inside the WHERE clause of the main SELECT statement. Although such nested SELECT statements are permitted in some SQL implementations, such as Oracle's, they are not supported by eBAM.

6.2 Setting Up the Application's Alert Conditions

eBAM allows you to set up tests whereby an alert notification is sent (and repeated with a specified frequency) whenever a condition is met. The condition can be as simple as a particular data item exceeding a threshold, or it might be a complex comparison between ratios of many aggregates of data items.

For each alert condition you create or configure, consider the following:

- *What to communicate:* One or more fields in the query's dataset that are valued (that is, that have mappings to them from operators and data definition fields; see ["To configure the dataset, mapping operators and data definition fields to its fields" on page 59](#)).
- *How to communicate:* Settings for the alert as a whole, such as resend frequency (see below) or email properties (see ["To set up email notification that an alert condition has been met" on page 69](#)).

To create and configure an alert condition

- 1 In the project tree, under the eBAM application, right-click **Alert Conditions** and, on the popup context menu, click: **New Alert Condition**

The eBAM Component Editor creates an alert based on the last-edited query.

- 2 Choose the correct query from the drop-down list. The Edit button to the right of the list allows you to enter the Query Editor.
- 3 If appropriate, select or clear check boxes constituting the key(s) for the alert.

The key, or a set of keys, is used to prevent duplication. When a record is found that matches the condition, the alert is *not* sent if any of its specified keys has a match in a record that was already staged. In other words: If a duplicate key is found, then the alert remains unsent until the alert's Resend Frequency is exceeded.

- 4 In the Properties pane, Alert tab, set Notification Interval and Resend Frequency to appropriate values:
 - ♦ Notification Interval specifies how often to run the process.
 - ♦ Resend Frequency specifies how often to re-issue a previously sent alert.
- 5 If you do *not* want e-mail sent, you are done. If, instead, you do want to set up automatic e-mail notification whenever an alert condition has been met, continue with the following procedure.

To set up email notification that an alert condition has been met

- 1 In the eBAM Component Editor, in the Properties pane, click the **Email** tab.
- 2 Supply values appropriate for your site, yourself, and this alert (see Table 11).

Table 11 Alert Properties for eMail

Name	Default Value	Comment
Send Email	False	To arrange for e-mail messages to be sent, change this to True . The value False causes all other properties to be ignored.
SMTP Server Host	(blank)	The hostname or IP address of the SMTP server at your site. (SMTP = simple mail transfer protocol). Hostname is case-insensitive; domain is optional. Thus, all the following examples are valid: <ul style="list-style-type: none"> ▪ mySmtpServer ▪ mysmtptserver ▪ mysmtptserver.mydomain.com ▪ 10.18.133.200
SMTP Server Port	25	The port number that this machine uses for email.
Username	(blank)	The login ID of a user who is authorized to send email on this SMTP server. Required only if the SMTP server requires authentication. If provided, eBAM will try to authenticate.
Password	*****	The password for this user on this SMTP server. All text entered in this field is masked as a row of asterisks (*****).
Send From	(blank)	The e-mail address of the message originator; can be blank. Case-insensitive, but preserves the case as entered. For internal email, either of the following forms is valid: <ul style="list-style-type: none"> ▪ myname ▪ myname@mydomain.com For external email, a domain must be specified; in other words: <ul style="list-style-type: none"> ▪ myname@mydomain.com
Send To	(blank)	The email address of the message recipient(s). If sending to more than one recipient, separate e-mail addresses by commas. Case-insensitive, but preserves the case as entered.
Message	(blank)	The text of the message to be sent. For easier viewing and editing, click the ellipsis [...] to the far right of this field.

- 3 When you have finished configuring the alert properties, audition your email message's appearance by clicking the Preview tab's **Refresh alert message** button.

Using eBAM Charts

eBAM uses charts that provide real-time feedback. “Using eBAM Charts” provides information on these charts along with step-by-step instructions for creating, modifying, and previewing them. Because the instructions assume you are already familiar with eGate GUIs, they focus on the concepts and design-time GUI operations that are specific to eBAM.

What’s in This Chapter

- [“eBAM Charts” on page 70](#)
- [“Setting Up the Application’s Charts” on page 71](#)

7.1 eBAM Charts

eBAM charts provide real-time feedback on the current data set according to conditions you set up. Charts are based on queries; a query can be used by many charts, but each chart is based on exactly one query. (For complete information on queries, see [“Using eBAM Queries” on page 55](#).) The purpose of a chart is to display easy-to-grasp visual information about a body of data or its statistics and key performance indicators (KPIs). The datastream and facts-of-interest are determined by the query; the presentation is determined by the chart parameters.

About Charts

Charts are graphical representations of KPIs. An eBAM application can contain many charts, or only one, or none at all. All eBAM charts have the following characteristics:

- A name, an associated Summary or Category query, and a *chart type*: bar, pie, meter, or trafficlight.
- Properties set in the **Common** tab, such as alignment, font, and update frequency. For details, see Table 21.
- (*except for meters*) Properties set in the **Chart** tab, such as 3D appearance, chart orientation, and navigation controls. For details, see Table 22.
- Properties specific to the chart type. For details, see Table 23 (Pie), Table 24 (Bar), Table 25 (Meters), and Table 26 (Trafficlights).

7.2 Setting Up the Application's Charts

To create and configure a chart

- 1 In the project tree, under the eBAM application, right-click **Charts** and, on the popup context menu, click: **New Chart**.
The eBAM Component Editor creates a chart based on the last-edited query.
- 2 If appropriate, choose the correct query from the drop-down list. The Edit button to the right of the list allows you to edit the query.
- 3 For Chart Type, choose from the drop-down list: Pie, Bar, Meter (available only if the query is a Summary), or Light.
- 4 If you committed preview data when you validated the data definition(s), click the **Refresh Chart** button to audition the chart's appearance.
- 5 In the Properties tabs, adjust parameters appropriately. With the Preview tab selected, you can:
 - ♦ Use the **Refresh** button to update the preview any time you make a change to the chart properties.
 - ♦ Use the **Print** button to print a hardcopy of the previewed chart.

Sample and Tutorial

To help you understand eBAM Studio and its capabilities, Sun provides a tutorial which instructs you on installing and then running a sample Project and the files supplied with the product. The step-by-step instructions are followed by a hands-on explanation of how to re-create and modify the sample on your own. The instructions assume you have installed the File eWay, which the sample implementation uses for input and output of the sample data.

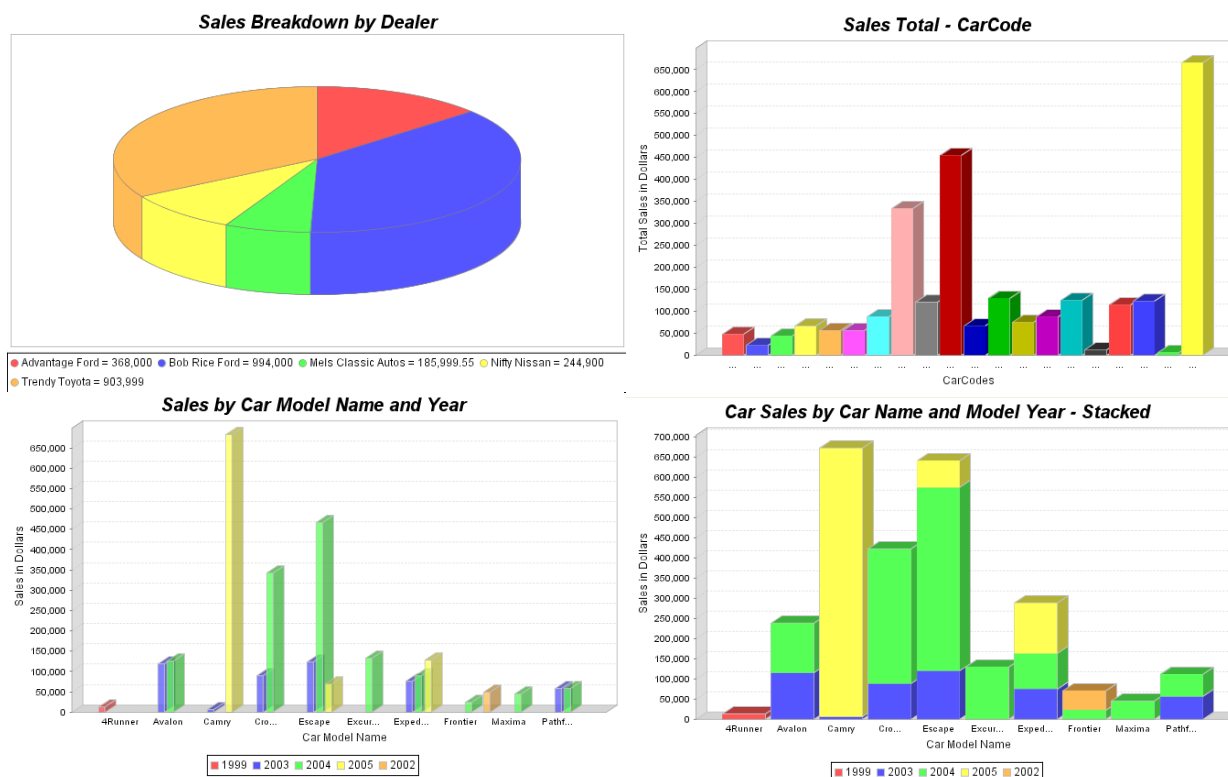
What's in This Chapter

- [“Installing and Running the eBAM Sample” on page 74](#)
- [“Hands-on Steps for Designing an eBAM Application” on page 77](#)
- [“Building, Deploying, and Monitoring a Project” on page 120](#)
- [“Monitoring Key Performance Indicators with Live Data” on page 123](#)

8.1 Overview

When you finish studying the tutorial and running the sample, you will have installed, deployed, run, and monitored the sample eBAM application supplied in **eBAMDocs.sar**. After feeding in sample data, the results include the following: One pie chart and three bar charts as shown in Figure 45, with the display updating every 30 seconds; an output alert file that is written out to a file whenever a particular threshold is reached; and an output results file generated by a dynamic query. The output results will vary according to the contents of a file you feed in.

Figure 45 Charts Displayed for Sample



eBAM Studio provides a complete sample implementation and a detailed hands-on tutorial.

Sample

The sample is included in the **eBAMDocs.sar** file, and it allows you to see the end results without having to go through all the design steps.

Tutorial

The tutorial provides a detailed hands-on guide to creating all the sample components, including ones that are not specific to eBAM. Many procedures are recapitulated using concrete examples.

Table 12 compares the goals and tasks of the two approaches.

Table 12 Comparing the Sample to the Tutorial

	Purpose	Tasks
Sample 8.2 on page 74	Provides the quickest route to seeing an eBAM application in action.	Install the sample; import the project; create the deployment profile and build/ deploy the project; view initial results; experiment with feeding modified data.

Table 12 Comparing the Sample to the Tutorial (Continued)

	Purpose	Tasks
Tutorial 8.3 on page 77	Provides complete steps for creating and configuring the working eBAM application provided in the sample.	Create the environment and project; add and configure all components that will be used (OTDs, eBAM components, business processes, and a Connectivity Map), build/deploy the project; view initial results; experiment with feeding modified data.

If you decide to run the sample and study the tutorial, do the sample first as this allows you to use it as reference material and backup when you go through the tutorial.

8.2 Installing and Running the eBAM Sample

This section contains procedures for installing and running the eBAM sample, omitting all design-time steps. It assumes your installation runs entirely on a Windows machine referred to as “localhost” that uses defaults for all ports and configuration parameters, that all usernames and passwords are “Administrator” and “STC,” and that you have a Sun SeeBeyond Integration Server that is already running a Domain called “domain1.” (For instructions on starting domain1, see [“Starting the Domain to Run the Sample” on page 119](#). For instructions on deployment to the Sun Java Application Server for viewing web applications on a Sun Java Portal Server, see [Chapter 9 “Using eBAM Charts in Portals” on page 127](#).)

Setup procedures

- [“Installing the Sample Files” on page 74](#)
- [“Importing the Sample Project” on page 75](#)

Activation procedures

- [“Building, Deploying, and Monitoring a Project” on page 120](#)

Runtime procedures

- [“Monitoring Key Performance Indicators with Live Data” on page 123](#)

8.2.1 Installing the Sample Files

These steps assume that you install the sample files to a temporary eBAM directory named `C:\temp\eBAM\` whose `Workspace\` subdirectory will be used by eBAM.

To install the sample files

Important: *Make sure your Repository is running and you have logged into Enterprise Designer before installing the sample.*

If you have already uploaded the documentation for eBAM, you can skip steps 1 through 4 and start with step 5.

- 1 In the **Administration** tab, select **Click to install additional products**.
- 2 Under **Select Java Composite Application Platform Suite Products to Install**, select the following if not uploaded previously:
 - ♦ Under eWay: **FileeWay**
 - ♦ Under Documentation: **eBAMDocs**

After you have finished making these selections, click **Next**.

Note: *eBAMDocs.sar* contains documentation and sample files. The File eWay is used in the sample implementation to read data from the files and write output.

- 3 When **Selecting Files to Install** appears, browse to the location of **eBAMDocs.sar**, select it, and click **Next**; repeat as needed for other product files.
- 4 When the installation completes, click the **Documentation** tab.
- 5 In the **Documentation** tab, click **Core Products**, and then click **Sun SeeBeyond eBAM Studio**.
- 6 In the right pane, click the icon to the right of **Sample Projects**.
- 7 Click **Open** and then extract the files to **C:\temp\eBAM** (making sure that you preserve the file paths).

The following directories and files are created:

```
C:\temp\eBAM\Sample\Projects\SampleBamProject.zip
C:\temp\eBAM\Sample\Data\In\input*_original.~in
C:\temp\eBAM\Sample\Data\In\TriggerDelete_original.~in
C:\temp\eBAM\Sample\Data\Out\outputAlert_comparison.dat
```

8.2.2 Importing the Sample Project

To import and validate the sample project

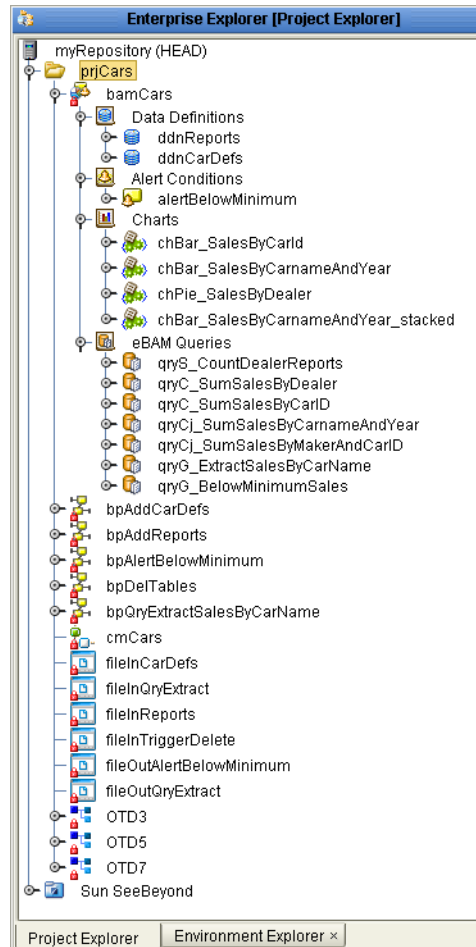
Important: *Make sure your Repository is running and you have logged into Enterprise Designer before installing the sample.*

- 1 In Project Explorer (in Enterprise Designer), right-click the Repository and, on the pop-up context menu, click **Import Project**, and then click **Yes** to continue.
- 2 In the Import Manager dialog box, browse to **C:\temp\eBAM\Sample\Projects**, select **SampleBamProject.zip**, and then click **Open**.
- 3 Back in the Import Manager dialog box, click **Import**.
 - ♦ If a warning is displayed about APIs missing from the target Repository, ignore it and click **Continue**.
- 4 When the import ends successfully, click **OK**, and then click **Close**.

The sample Project is now visible in the Project Explorer tree. See Figure 46.

Note: *If the Project does not import successfully, it is because you have not installed all the required components. Return to the Suite Installer and see what still needs to be installed.*

Figure 46 Sample Project prjCars, Showing Components in Project Tree



- 5 In Environment Explorer, right click **envBam**, select **Version Control** and then click **Check Out**. When the Version Control - Check Out dialog box appears, click **Check Out**.
- 6 Open **envBam**, right click **isEbam**, select **Version Control**, and click **Check Out**. When the Version Control - Check Out dialog box appears, click **Check Out**.
- 7 Right click **isEbam**, select **Properties**, and then enter the **Password** in the Properties dialog box.

Note: *If needed, adjust other the settings as well; for example, in a UNIX environment, edit the parameter settings "C:\temp\..." in the properties of extFileBam and isEbam.*

- 8 In Project Explorer, under prjCars, check out bamCars. Then, open each of its two data definitions and validate them by previewing and committing the sample data supplied; for instructions, see [procedure on page 83](#).

- 9 Open each of the four charts and preview them; see step 4 in [procedure on page 100](#).
- 10 Create a deployment profile named **dpCars** (referencing cmCars and envBam), automap the components, build the Project, and deploy the resulting EAR file (named **prjCarsdpCars.ear**); for instructions, see [procedure on page 120](#).
- 11 After the project is built and deployed, point your browser at the URL for monitoring the eBAM charts for this project:

```
http://localhost:18001/prjCarscmCarsbamCars1dpCars/
```

If your Integration Server is running on another computer, or if it does not use port 18001 for HTTP, then substitute the correct computer name and port in the URL.
- 12 See how the charts change as you feed in data; for instructions, see [procedure on page 122](#).

8.3 Hands-on Steps for Designing an eBAM Application

This section provides steps for:

- [“Starting Up” on page 78](#), where you set up the environment (named envBam) and the Project (named prjCars).
- [“Creating and Validating OTDs” on page 79](#), where you create three simple OTDs: OTD7, OTD5, and OTD3, containing seven, five, and three fields.
- [“Creating an eBAM Application and Data Definitions” on page 81](#), where you create the eBAM application, named bamCars, and two data definitions: ddnCarDefs (seven fields) and ddnReports (five fields).
- [“Creating eBAM Sample Queries” on page 85](#), where you set up the queries that are used to manipulate the datastream (for use in charts) and identify conditions where the datastream crosses a threshold (for use in alerts).
- [“Creating eBAM Sample Alerts” on page 97](#), where you set up an alert to issue a notification when a condition is met.
- [“Creating eBAM Sample Charts” on page 99](#), where you set up and preview charts to graphically display current data values and KPIs.
- [“Creating and Validating Business Processes” on page 107](#), where you create four business processes (bpAddCarDefs, bpAddReports, bpDelTables, and bpAlertBelowMinimum), link web services and OTD methods, and define the business rules between them.
- [“Creating and Configuring Connectivity Maps” on page 115](#), where you use the Enterprise Designer’s Connectivity Map Editor to connect instances of Project components and configure parameters for inbound and outbound File eWays.
- [“Starting the Domain to Run the Sample” on page 119](#), where you start the server and activate the Domain where your completed Project is deployed.

- **“Building, Deploying, and Monitoring a Project” on page 120**, where you use Enterprise Designer to create a deployment profile and build and deploy a Project.

After the application is running, use the same runtime procedures used for the sample:

- **“Monitoring Key Performance Indicators with Live Data” on page 123**, where you feed data to the Logical Host and use the Chart Viewer to monitor results.

8.3.1 Starting Up

Follow these procedures to get started with a blank canvas for your design work.

To re-create the sample environment

If you already installed the sample Project, in the Project Explorer rename it to **prjCarsOLD**. If you already installed the environment for the sample Project, rename it to **envBamOLD**.

These steps assume you use a default Integration Server running on default ports. If you use anything other than a Sun SeeBeyond Integration Server on ports 18000–18009, make adjustments in step 5 below or in the URL in step 11 in the [procedure on page 77](#).

- 1 In Enterprise Designer, click the **Environment Explorer** tab.
- 2 In the environment tree, right-click the Repository and, on the pop-up context menu, click **New Environment**.
 - ♦ Right-click the newly created environment and rename it to **envBam**.
- 3 Right-click **envBam** and, on the pop-up menu, point at **New** and then click **File External System**.
 - A Name the new external **extFileBam** and click **OK**.
 - B Right-click **extFileBam**, open its properties, and configure it as follows:
 - ♦ Choose **Inbound File eWay**, point at **Parameter Settings**, select **Directory**, and then enter **C:\temp\eBAM\Sample\Data\In**.
 - ♦ Choose **Outbound File eWay**, point at **Parameter Settings**, select **Directory**, and then enter **C:\temp\eBAM\Sample\Data\Out**.
- 4 Right-click **envBam** and, on the pop-up menu, point at **New** and then click **Logical Host**.
 - ♦ Right-click the newly created Logical Host and rename it to **lhEbam**.
- 5 Right-click **lhEbam**, point at **New** and then click **SeeBeyond Integration Server**.
 - A Right-click the newly created Integration Server and rename it to **isEbam**.
 - B Right-click **isEbam**, open its properties, and configure it as follows:
 - ♦ Application Workspace Directory: **C:\temp\eBAM\Workspace**.
 - ♦ Username and password: If not “Administrator” and “STC,” enter the values that have been created by your system administrator.

You have re-created the sample environment provided in the sample. It contains all servers that are needed, and all required server configuration steps have been taken. You can add or modify parameter settings to tailor them appropriately for your needs.

To re-create the sample eBAM project container

- 1 In Enterprise Designer, click the **Project Explorer** tab.
- 2 In the project tree, right-click the repository and, on the pop-up menu, click **New Project**. Rename the new project to: **prjCars**.

8.3.2 Creating and Validating OTDs

This procedure helps you get acquainted with the sample OTDs listed in Table 13 by taking you through the steps for creating and testing them.

Table 13 OTDs Included With the eBAM Sample

Name	Purpose	Sample Data to Test
OTD7	In the bpAddCarDefs BP, this OTD unmarshals data (seven fields per record) so that it can be added to the ddnCarDefs data definition.	...\Sample\Data\In\inputCarDefs*.txt Seven fields per record: car ID number; model name; manufacturer; car type; car size; model year; and base price.
OTD5	In the bpAddReports BP, this OTD unmarshals data (five fields per record) so that it can be added to the ddnReports data definition. Additionally, in the bpAlertBelowMinimum BP, this OTD marshals data so that it can be written to a file.	...\Sample\Data\In\inputReports*.txt. Five fields per record: dealername; car code letter; car ID number; saleprice; and time-of-sale.
OTD3	In the bpQryExtractSalesByCarName BP, this OTD marshals data (three fields per record) so that it can be written to a file.	Not applicable.

To create the sample eBAM OTDs

- 1 In the project tree, right-click prjCars and, on the pop-up menu, point at **New** and click **Object Type Definition**.

The New [OTD] Wizard appears.

- 2 In step 1 (Select Wizard Type), click **User-Defined OTD** and then click **Next**.
- 3 In step 2 (Enter OTD Name), type **OTD7** and click **Finish**.
- 4 In the OTD Editor, click **OTD7** (under Object Type Definition in the center pane)
- 5 In the Properties pane for OTD7, click **delim**.

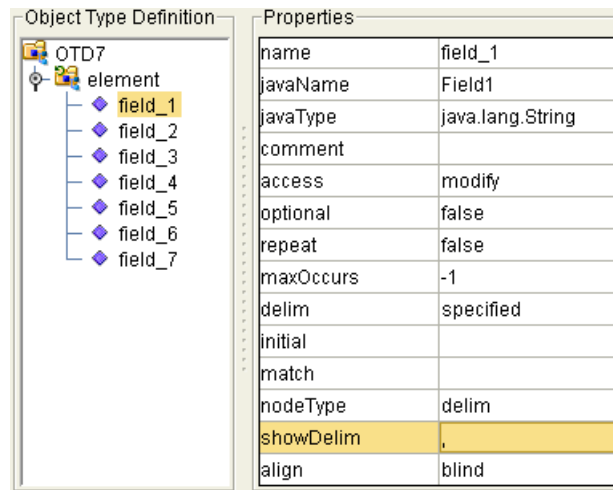
You are now ready to create two new levels of delimiters.

- ♦ Click the delim ellipsis (...) and add the following:
 - ♦ \r\n (return+linefeed) as a normal level-one delimiter
 - ♦ , (comma) as a normal level-two delimiter

- 6 Right-click **OTD7** (under Object Type Definition) and add an element under it.
- 7 Select the **element** you created and change its **repeat** property to **true** in the Properties pane.
- 8 Under **element**, create eight fields (through **field_7**).

Note: When finished, delete the first field (see Figure 47).

Figure 47 Seven-Field Sample OTD (OTD7)



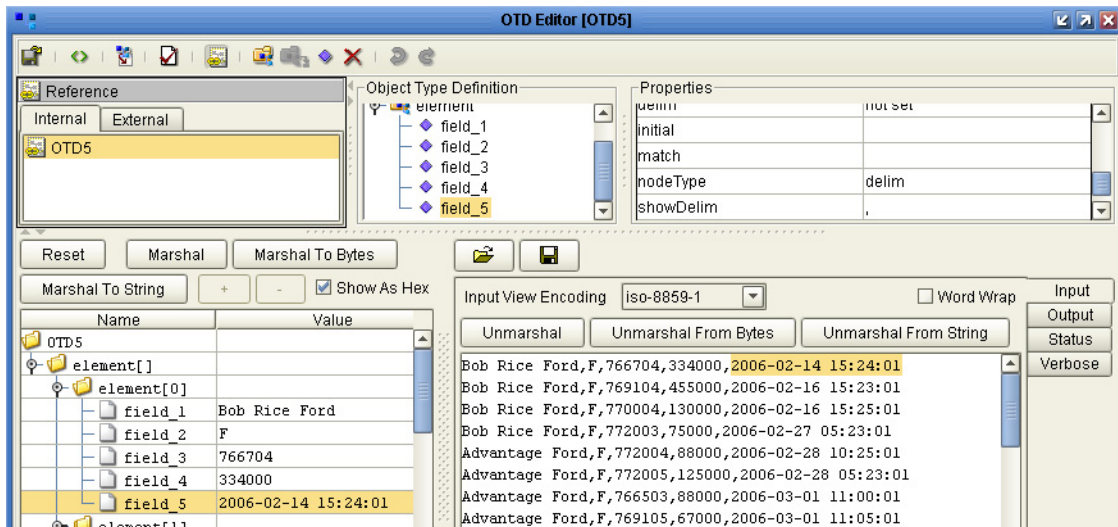
- 9 Repeat steps 1 through 8 to create a three-field OTD named **OTD3**, containing one \r\n-delimited repeating element as above, but with three comma-delimited fields.
- 10 Repeat steps 1 through 8 to create a five-field OTD named **OTD5**, containing one \r\n-delimited repeating element as above, but with five comma-delimited fields.

Note: The OTDs you created are listed at the bottom of the canvas. To move between them and make another OTD active on the canvas, click its name. To close an OTD, click the "x" next to its name.

To validate the sample OTDs

- 1 In the toolbar of the OTD Editor, click **Run Test** to open the OTD Tester pane.
- 2 In the toolbar of the OTD Tester, with the **Input** tab selected, click **Open a file**.
- 3 In the Open dialog box, browse to the location (C:\temp\eBAM\Sample\Data\In) of the file containing the sample data for this OTD (see Table 13) and open the file.
You are going to validate the OTDs from a comma-delimited file.
- 4 After successful unmarshaling, open the tree in the left pane of the OTD tester to see how the sample data was parsed. See Figure 48.

Figure 48 Validation of OTD5 Using Data From inputReports5_original.~in



8.3.3 Creating an eBAM Application and Data Definitions

The procedures in this section explain how to create and configure the data definitions used in the eBAM sample. The field names and data types used for the fields are shown in Table 14 and Table 15.

Table 14 Metadata (Field Names and Data Types) for ddnCarDefs

Name	Type	Meaning
carCode	integer	Six-digit ID, such as "886503" or "611104"
carName	varchar	Model name, such as "Excursion" or "Camry"
carMaker	varchar	Manufacturer, such as "Ford" or "Toyota"
carType	varchar	Type of car, such as "luxury", "SUV", "truck", or "car"
carSize	char	Single letter: "S", "M", or "L"
carModelYear	integer	Model year, such as "2004" or "2006"
carSugBasePrice	float	Base price, such as "36775.00" or "19045"
Note: Use the sample data file in C:\temp\eBAM\Sample\Data\In\inputCarDefs7* to validate ddnCarDefs.		

Table 15 Metadata (Field Names and Data Types) for ddnReports

Name	Type	Meaning
dealerName	varchar	Dealer name, such as "Springfield Ford"
dealerType	char	Code letter, such as "F" (for Ford) or "V" (for various)
carModelCode	integer	Six-digit ID, such as "766704" or "883704"
totalSales	float	Dollar figure for a car sale, such as: "14499.95"
reportDate	timestamp	Transaction date, such as "2006-04-29 13:05:01"

Table 15 Metadata (Field Names and Data Types) for ddnReports (Continued)

Name	Type	Meaning
Note: Use the sample data file C:\temp\eBAM\Sample\Data\In\inputReports5* to validate ddnReports.		

Procedures

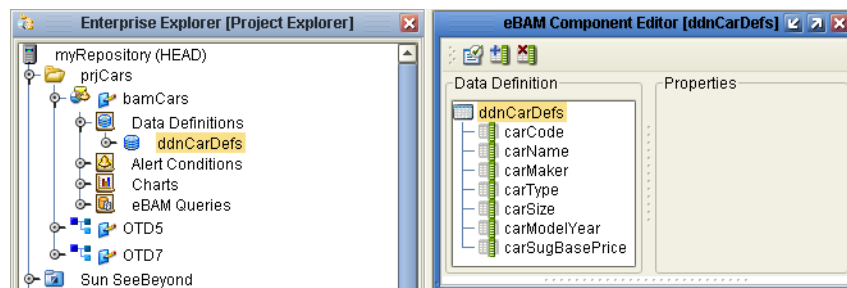
- “To create the sample eBAM application and data definitions” on page 82
- “To validate the data definition you created” on page 83

To create the sample eBAM application and data definitions

- 1 In the project tree, right-click prjCars and, on the pop-up menu, point at **New** and click **eBAM Application**. The eBAM Application Wizard appears.
- 2 For Application name, type **bamCars** and click **Next**.
- 3 For Data Definition Name, type **ddnCarDefs** and click **Next**.
- 4 The next step in the wizard prompts you to define the data definition elements. Click **Add** six times, and then supply column names and data types for the seven ddnCarDefs fields as shown in Table 14. When you have specified all column names and data types, click **Next**.
- 5 The next step in the wizard prompts you to specify or bypass Data Retention. Leave the check box cleared and click **Next**.
- 6 The final step of the wizard allows you to go back as needed to review your choices. As needed, click **Back** and **Next** to return to a step and make changes. Click **Finish** when you are satisfied with the data definition.

In the left pane, under prjCars, the project tree displays a new application named bamCars. On the right side (the canvas), the eBAM Component Editor opens to display the data definition; see Figure 49.

Figure 49 eBAM Component Editor Showing New Data Definition ddnCarDefs



- 7 In the project tree, under bamCars, right-click **Data Definitions** and then click pop-up menu item **New Data Definition**. Then repeat steps 3 through 6 above to create the five-field data definition **ddnReports**, using the field names and data types shown in Table 15.

Note: *Any time you complete a new data definition, it is good practice to use the **Show Sample Data** tool to double-check the data definition against sample data, preview it, and commit it for later use. Use the following procedure.*

To validate the data definition you created

This procedure is optional, but highly recommended. It starts with **ddnCardefs**.

- 1 In the eBAM Editor, on the tool palette, click: **Show Sample Data**.
The Sample Data pane appears below the main eBAM Editor.
- 2 On the Sample Data tool palette, click **Import**.
- 3 In the File Import wizard's **Select File** step, browse to the location of the sample data files (**C:\temp\eBAM\Sample\Data\In**) and select **inputCarDefs7*~in**.
- 4 In the following steps, keep the default choices shown in Table 16 and then click **Next**.

Table 16 Choices for Imported Sample Data for Sample Data Definitions

Item	Default
Encoding scheme	ASCII (ISO646-US)
File format	Delimited
Default SQL Type	varchar
Record Delimiter	{newline (lf)}
Field Delimiter	{comma}
Text Qualifier	none
First line contains field names?	False

- 5 In the next step, verify the record layout and field properties in the Field information pane, enter the number of sample records to read and display in the preview pane, and then click **Preview**.

If the sample data is valid (in accord with the layout, encoding, formatting, and properties you specified), the Preview pane displays a table whose rows are the first few records, with each column headed by the data element name. See Figure 50.

Figure 50 Showing Sample Data: Preview of Data Definition Field Layout

File Import Wizard

Import File Metadata for inputCarDefs.txt.-in (Step 3 of 3) wizard ()

Preview record layout and field properties.

Field information

Field #	Column name	Datatype
1	carCode	integer
2	carName	varchar
3	carMaker	varchar
4	carType	varchar
5	carSize	char
6	carModelYear	integer
7	carSugBasePrice	float

Preview

Number of sample records:

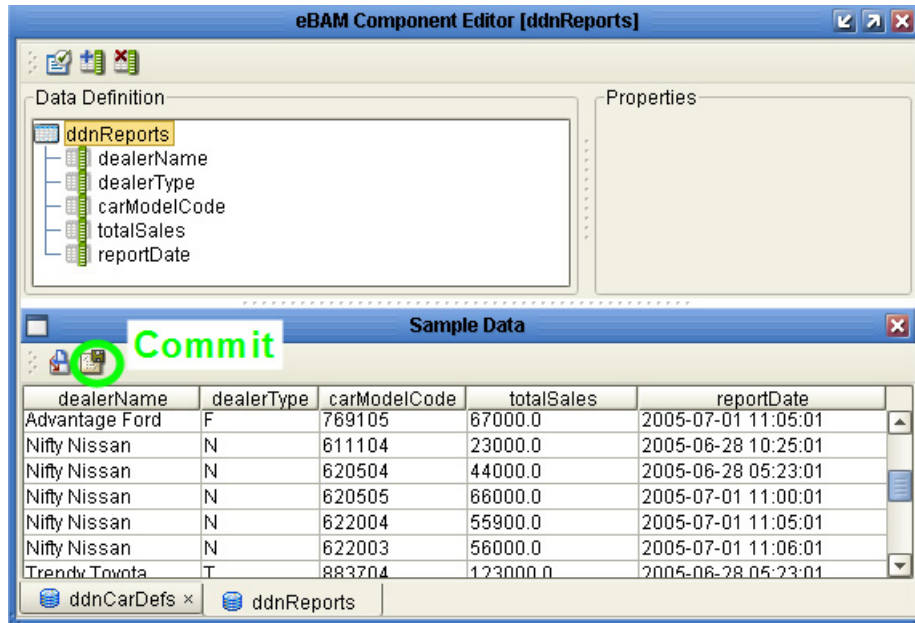
CARCODE	CARNAME	CARMAKER	CARTYPE	CARSIZE	CARMODELYEAR	CARSUGBASEPRICE
882207	Prius	Toyota	car	S	2004	26999.0
882206	Prius	Toyota	car	S	2003	24990.0
882205	Prius	Toyota	car	S	2002	22000.0
882204	4Runner	Toyota	SUV	M	2004	27170.0
882203	4Runner	Toyota	SUV	M	2003	27055.0

< Back Next > **Finish** Cancel Help

- 6 Click **Finish**.
- 7 In the toolbar of the Sample Data pane, click the **Commit** icon (Figure 51 shows an example).
- 8 Back in the project tree, under bamCars, double-click **ddnReports** and repeat steps 1 through 6 above to validate ddnReports using sample input file **inputReports5***.

The data set is parsed and displayed in the Sample Data pane. See Figure 51.

Figure 51 Sample Data Displayed in Sample Data Pane



9 Click Commit.

Note: When you finish a new data definition, it is good practice to commit sample data. This not only validates that you have set up the metadata correctly, it also makes the data available for other purposes, such as previewing the appearance of a chart.

After a data definition has been created, configured, and validated, it is available for use in one or more queries.

8.3.4 Creating eBAM Sample Queries

The procedures in this section explain how to create the queries in the eBAM sample.

Table 17 Queries Included With the eBAM Sample

Name	Type	Attributes	Target Data Definition(s)	Operation and Purpose
qryS_CountDealerReports	Summary		ddnReports	Counts the dealerName field.
qryG_BelowMinimumSales	General Filtered		ddnReports	Filters against the totalSales field only, watching for a value less than 1000.0. Used by alertBelowMinimum.

Table 17 Queries Included With the eBAM Sample (Continued)

Name	Type	Attributes	Target Data Definition(s)	Operation and Purpose
qryG_ExtractSalesByCarName	General	JoinView, filtered, with runtime input	ddnReports, ddnCarDefs	Reports three items: carName (filtered to find only values that match the runtime input), carModel, and a KPI (the calculated sum of totalSales). The web service execute is used in bpQryExtractSalesByCarName.
qryC_SumSalesByDealer	Category, but without Series		ddnReports	Category is dealerName; value is a KPI (calculated sum of totalSales).
qryC_SumSalesByCarID	Category, but without Series		ddnReports	Category is carModelCode; value is a KPI (the calculated sum of totalSales). Used by one chart, chBar_SalesByCarId.
qryCj_SumSalesByMakerAndCarID	Category, with Series	JoinView	ddnReports, ddnCarDefs	Category is carMaker; series is carCode; value is a KPI (the calculated sum of totalSales).
qryCj_SumSalesByCarname AndYear	Category, with Series	JoinView	ddnReports, ddnCarDefs	Category is carName; series is carModelYear; value is a KPI (the calculated sum of totalSales). Used by three charts: chBar_SalesByCarname AndYear, chPie_SalesByDealer, and chBar_SalesByCarname AndYear_stacked

eBAM Query qryS_CountDealerReports

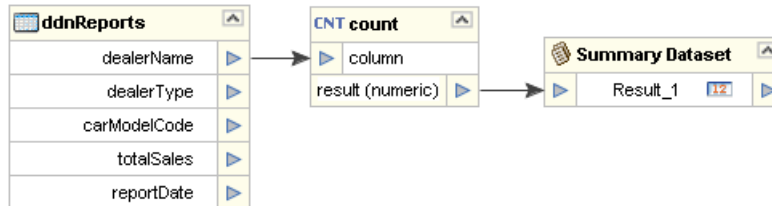
This summary query simply counts how many reports are currently in the database.

To create the qryS_CountDealerReports query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and then click **New eBAM Query**. The New eBAM Query Wizard appears.

- 2 For eBAM Query Condition Type, choose **Summary**; for eBAM Query Condition Name, enter **qryS_CountDealerReports**; and then click **Next**.
- 3 Select the checkbox for **ddnReports** and then click **Finish**.
- 4 From the Operator Palette (Number), drag the **count** operator onto the canvas.
- 5 Connect the output of **dealerName** to the **column** input of **count**, and then connect the count **result (numeric)** to **Result_1**. See Figure 52.

Figure 52 Summary Query to Count Dealer Reports



qryG_BelowMinimumSales

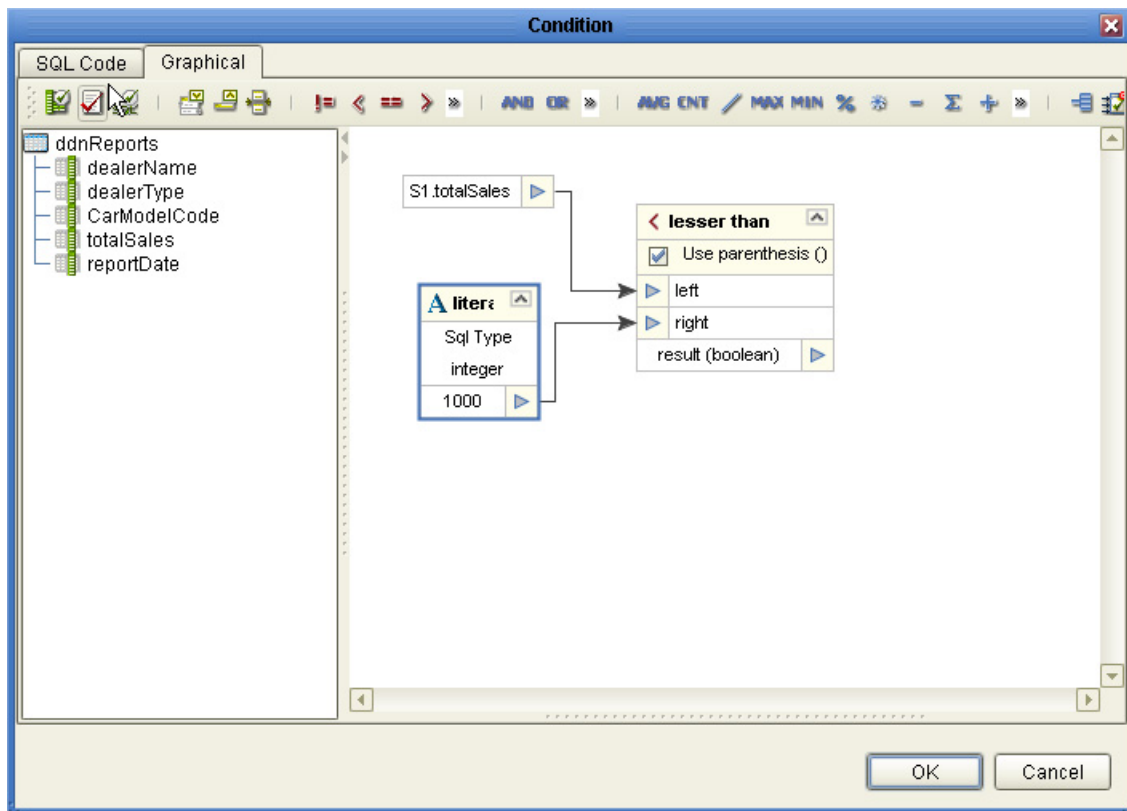
This general query has a filter against totalSales, with a threshold of totalSales < 1000.0.

To create the qryG_BelowMinimumSales query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and then click **New eBAM Query**. The New eBAM Query Wizard appears.
- 2 For eBAM Query Condition Type, choose **General**; for eBAM Query Condition Name, enter **qryG_BelowMinimumSales**; and then click **Next**.
- 3 Select the checkbox for **ddnReports** and then click **Finish**.
- 4 In the Component editor, right-click **ddnReports** and select **Properties**, and then, in the Properties dialog box, for **Condition**, click the ellipsis [...] to open the Condition editor.
- 5 In the Condition editor, using either SQL Code or the Graphical editor, construct this condition: **S1.totalSales < 1000**. See Figure 53.

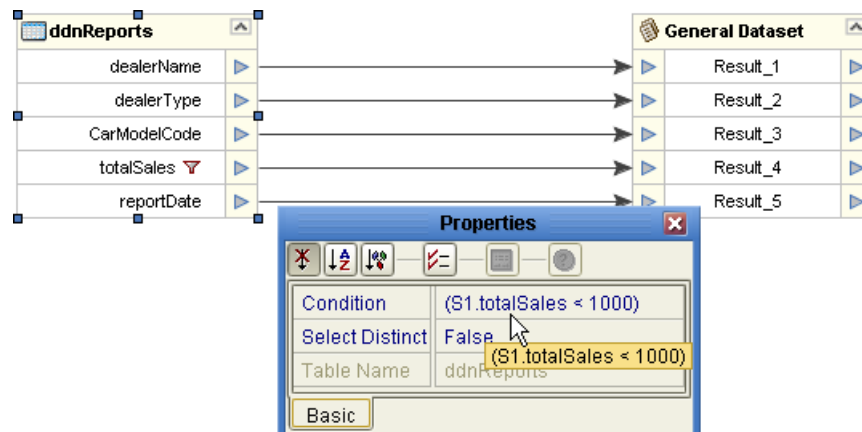
Note: *The Graphical editor is easier to use as you must enter code if you use the SQL Code editor.*

Figure 53 Setting a Condition on the totalSales Field



- 6 Validate the condition by clicking the **Validate Graph** icon on the menu bar. Then click **OK** twice, to exit the editor and the dialog box.
- 7 In the Component editor, right-click **General Dataset** and then click **Configure**.
- 8 In the Configure Dataset View dialog box, click **Add** four times. This creates Result_2 through Result_5.
 - A Click **OK** to exit the dialog box.
 - B Connect each field on the left with the corresponding column on the right. See Figure 54.

Figure 54 General Query to Check the Condition totalSales < 1000.00



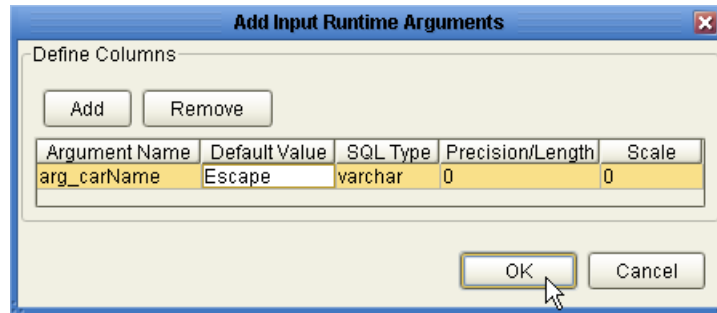
qryG_ExtractSalesByCarName

This general query creates an inner-join of the two data definitions, filters against carName to check for a match against a runtime input, and sums totalSales. It provides an example of a dynamic query that uses a runtime argument; this allows you to create different outputs that vary dynamically according to the value you provide at run time for the queried argument. In this sample, you will set up the Connectivity Map so that the output consists of files named **outputQryNameResults*.dat**.

To create the qryG_ExtractSalesByCarName query

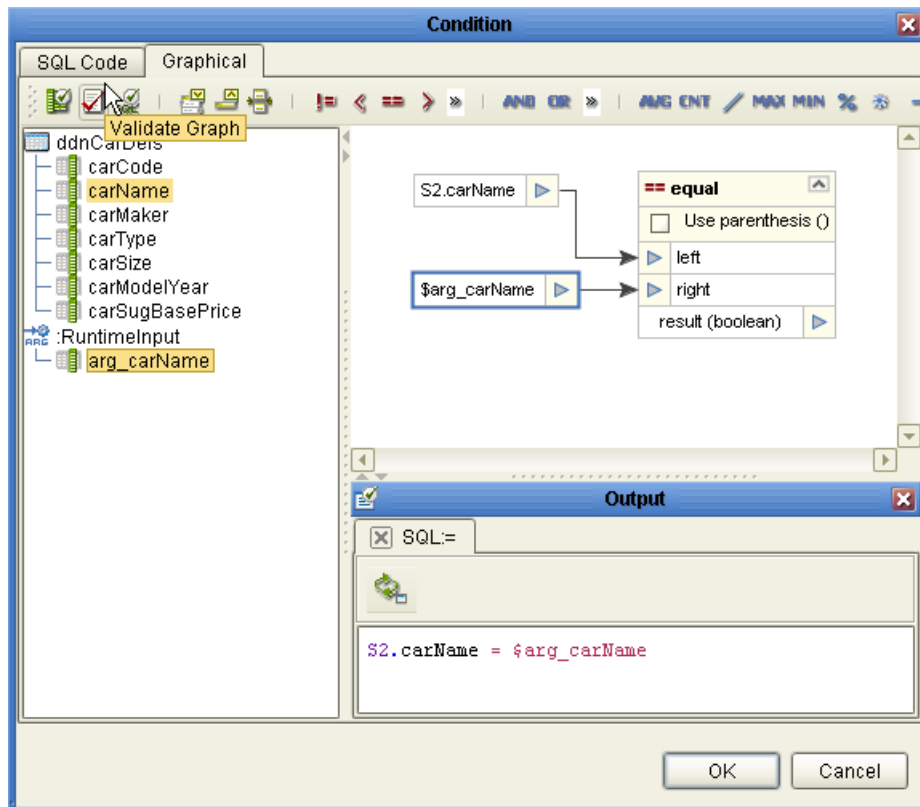
- 1 In the project tree: Open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and click **New eBAM Query**. The New eBAM Query Wizard appears.
- 2 For eBAM Query Condition Type, choose **General**; for eBAM Query Condition Name, enter **qryG_ExtractSalesByCarName**; and then click **Next**.
- 3 Select the checkboxes for **ddnReports** and **ddnCarDefs** and then click **Finish**.
- 4 On the toolbar, click the icon for **Add/Edit Runtime Inputs** icon.
- 5 In the Add/Edit Input Runtime Arguments dialog box:
 - A For Argument Name, change arg_0 to **arg_carName**.
 - B For Default Value, enter **Escape**.
 - C For the other columns, keep the defaults (see Figure 55).
 - D When finished, click **OK**.

Figure 55 Creating a Runtime Argument



- 6 In the Component Editor, right-click **ddnCarDefs** and select **Properties**.
- 7 In the Properties dialog box, for Condition click the ellipsis [...] to open the Condition editor.
- 8 In the Condition editor, using either SQL code or the graphical editor, construct this condition: **S2.carName = \$arg_carName** (see Figure 56).

Figure 56 Setting a Dynamic Argument Condition on the totalSales Field



- 9 Validate the condition by clicking the **Validate Graph** icon. To see the SQL code (see Figure 56) on the Output canvas, click the **Show SQL for Condition** icon.
- 10 To exit the editor and the dialog box click **OK** twice.
- 11 In the Component Editor, right-click **General Dataset** and then click **Configure**.

- 12 In the Configure Dataset View dialog box, click **Add** two times and rename the bulleted attributes as follows:

- ♦ Rename Result_1 to **Result_carName**.
- ♦ Rename Result_2 to **Result_carModelYear**.
- ♦ Rename Result_3 to **Result_SumOfSales**.

When finished click **OK**.

- 13 From the Operator Palette (Number), drag the **sum** operator onto the canvas.

- 14 From ddnCarDefs, connect:

- A The output of **carName** to the **Result_carName** input.
- B The output of **carModelYear** to the **Result_carModelYear** input.

- 15 From General Dataset, connect:

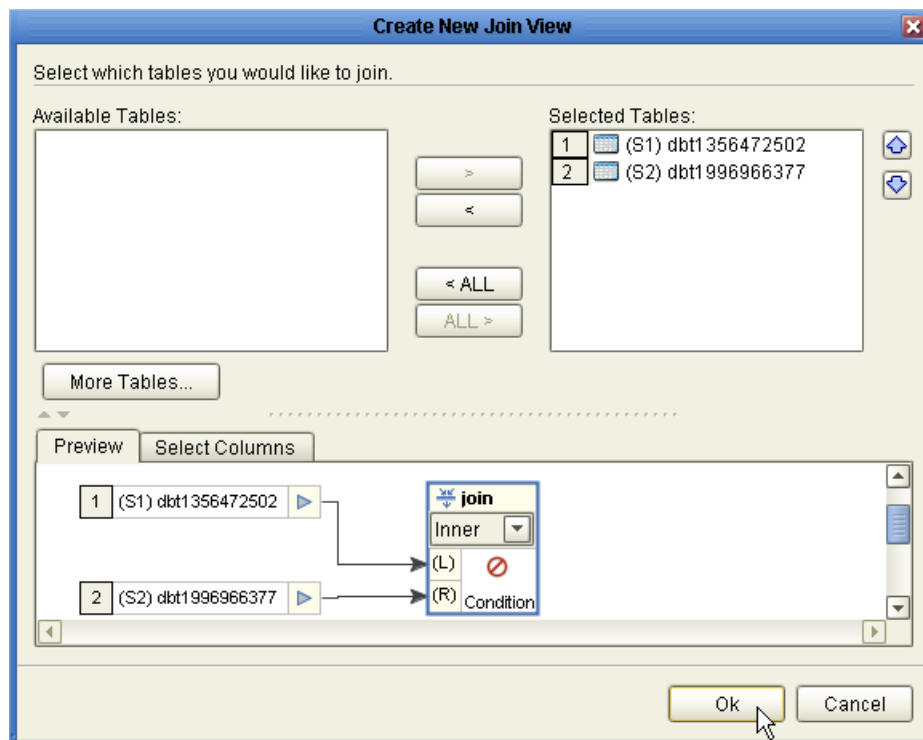
- A The output of **Result_carName** to **Group 1**.
- B The output of **Result_carMod** to **Group 2**.

- 16 From ddnReports, connect:

- A The output of **totalSales** to the **column** input of **sum**.
- B The sum **result (numeric)** to **Result_SumOfSales**.

- 17 In response to the question “create a join view?,” click **OK**. The Create New Join View dialog box appears (see Figure 57).

Figure 57 Inner Join




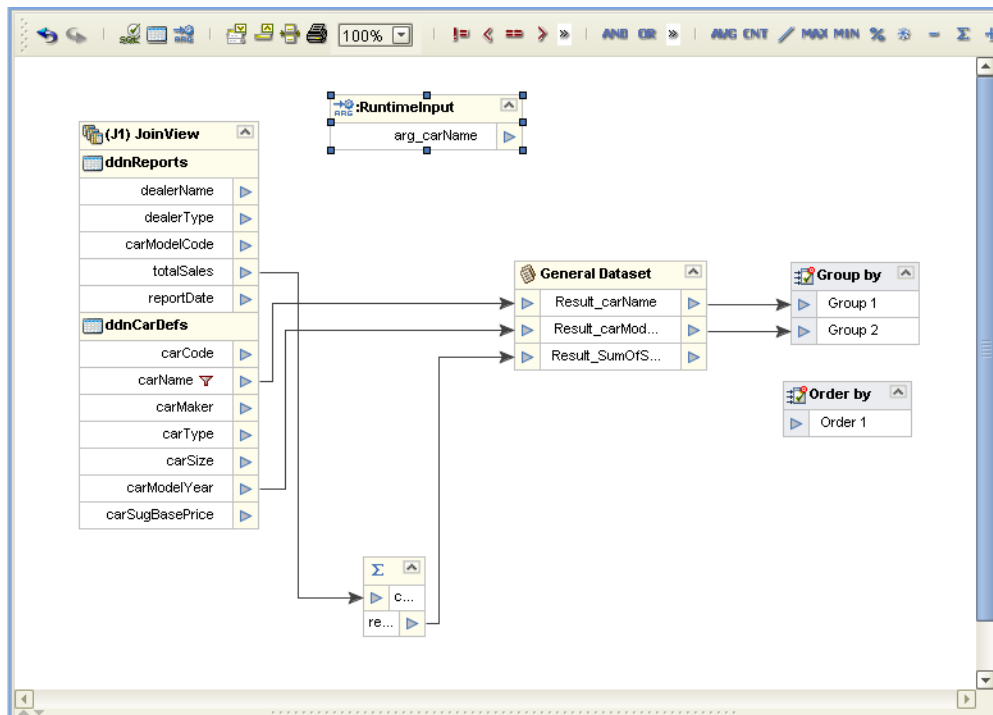
- 18 You need to set the condition of the join. In the **join** (at the bottom of the dialog box), click  (red circle with the slash through it). The Join Condition Editor appears (see Figure 62).
- 19 Select the **Graphical** tab, and perform the following:
 - A Drag **carModelCode** (from **ddnReports**) and **carCode** (from **ddnCarDefs**) onto the canvas.
 - B Drag the **=** (equal) icon from the tool bar onto the canvas.
 - C Connect the **S1.carModelCode** output to the **left** input.
 - D Connect the **S2.carCode** output to the **right** input.
 - E Click **OK** to set the condition. The Join Condition Editor closes and the Create New Join View dialog box reappears.
- 20 Keep the default join type (Inner) and click **OK** (see Figure 58).

Figure 58 Summary Query to Extract Sales for carName



eBAM Query qryC_SumSalesByDealer

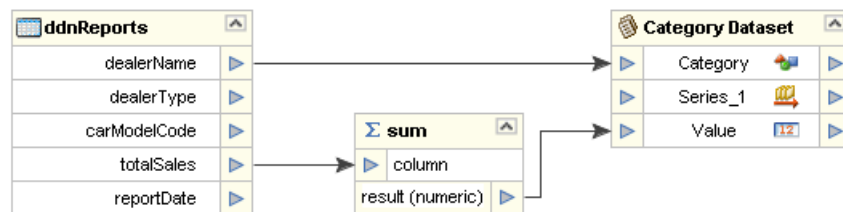
This category query outputs dealerName as the category, has no series, and outputs the value computed by summing totalSales.

To create the qryC_SumSalesByDealer query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and then click **New eBAM Query**. The New eBAM Query Wizard appears.

- 2 For eBAM Query Condition Type, choose **Category**; for eBAM Query Condition Name, enter **qryC_SumSalesByDealer**; then click **Next**.
- 3 Select the checkbox for **ddnReports** and then click **Finish**.
- 4 From the Operator Palette (Number), drag the **sum** operator onto the canvas.
- 5 Connect the following (see Figure 59):
 - A The output of **dealerName** to the **Category** input.
 - B The output of **totalSales** to the **column** input of **sum**.
 - C The sum **result (numeric)** to **Value**.

Figure 59 Category Query to Report Summed totalSales by dealerName



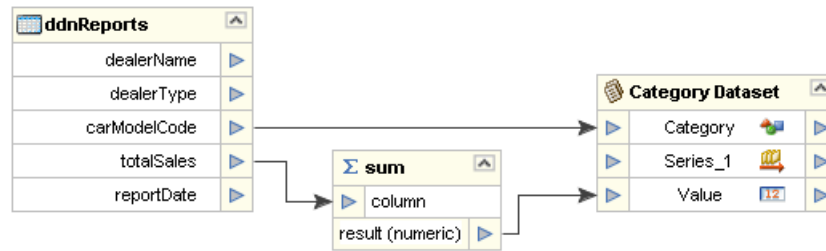
eBAM Query qryC_SumSalesByCarID

This category query outputs carModelCode as the category, has no series, and outputs the value computed by summing totalSales.

To create the qryC_SumSalesByCarID query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and click then **New eBAM Query**. The New eBAM Query Wizard appears.
- 2 For eBAM Query Condition Type, choose **Category**; for eBAM Query Condition Name, enter **qryC_SumSalesByCarID**; and then click **Next**.
- 3 Select the checkbox for **ddnReports** and then click **Finish**.
- 4 From the Operator Palette (Number), drag the **sum** operator onto the canvas.
- 5 Connect the following (See Figure 60).
 - A The output of **carModelCode** to the **Category** input.
 - B The output of **totalSales** to the **column** input of **sum**.
 - C The sum **result (numeric)** to **Value**.

Figure 60 Category Query to Report Summed totalSales by carModelCode



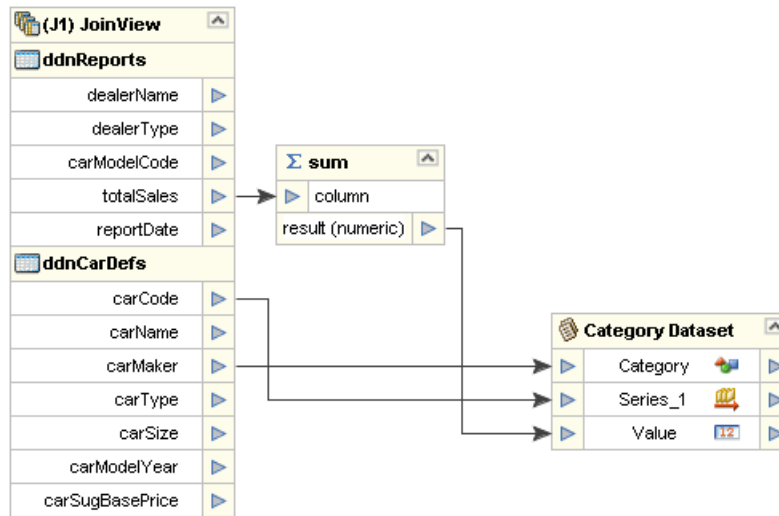
eBAM Query qryCj_SumSalesByMakerAndCarID

This category query creates an inner-join of the two data definitions, outputs carMaker as the category and carCode as the only series, and outputs the value computed by summing totalSales.

To create the qryCj_SumSalesByMakerAndCarID query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and then click **New eBAM Query**. The New eBAM Query Wizard appears.
- 2 For eBAM Query Condition Type, choose **Category**; for eBAM Query Condition Name, enter **qryCj_SumSalesByMakerAndCarID**; and then click **Next**.
- 3 Select the checkboxes for **ddnReports** and **ddnCarDefs**, and then click **Finish**.
- 4 From the Operator Palette (Number), drag the **sum** operator onto the canvas.
- 5 From ddnCarDefs, connect the following:
 - A The output of **carMaker** to the **Category** input.
 - B The output of **carCode** to the **Series_1** input.
- 6 From ddnReports, connect the following:
 - A The output of **totalSales** to the **column** input of **sum**.
 - B The sum **result (numeric)** to **Value**.
- 7 In response to the question “create a join view?,” click **OK**.
- 8 In the Create New Join View dialog box (see [Figure 57 on page 91](#)), keep the default join type (Inner) and click **OK** (see Figure 61).

Figure 61 Category Query to Report Sales by carMaker and then carCode



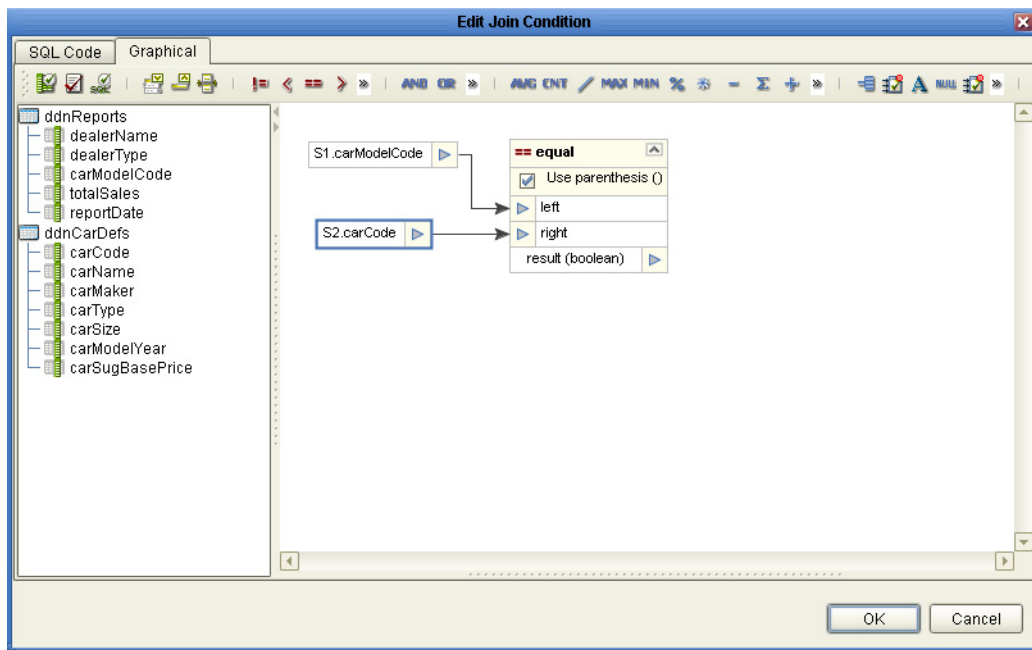
eBAM Query qryCj_SumSalesByCarnameAndYear

This category query creates an inner-join of the two data definitions, outputs *carName* as the category and *carModelYear* as the only series, and outputs the value computed by summing *totalSales*.

To create the **qryCj_SumSalesByCarnameAndYear** query

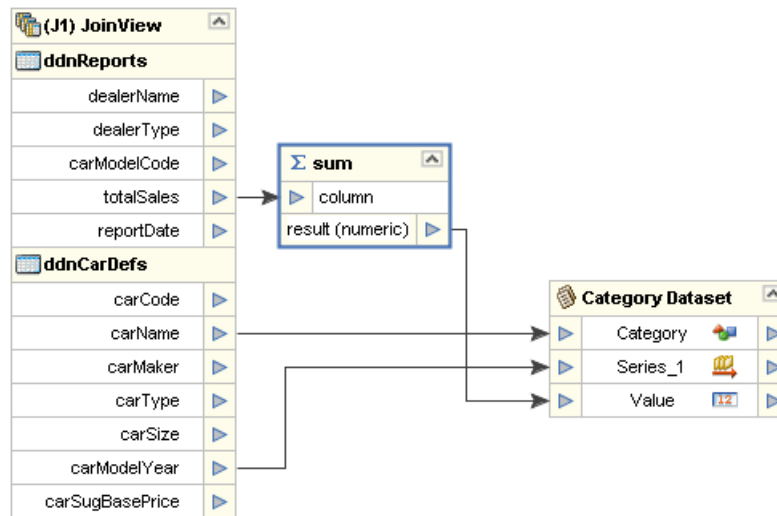
- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **eBAM Queries**, and then click **New eBAM Query**. The New eBAM Query Wizard appears.
- 2 For eBAM Query Condition Type, choose **Category**; for eBAM Query Condition Name, enter **qryCj_SumSalesByCarnameAndYear**; and then click **Next**.
- 3 Select the checkboxes for **ddnReports** and **ddnCarDefs**, and then click **Finish**.
- 4 From the Operator Palette (Number), drag the **sum** operator onto the canvas.
- 5 From **ddnCarDefs**, connect the following:
 - A The output of **carName** to the **Category** input.
 - B The output of **carModelYear** to the **Series_1** input.
- 6 From **ddnReports**, connect the following:
 - A The output of **totalSales** to the **column** input of **sum**.
 - B The sum **result (numeric)** to **Value**.
- 7 In response to the question "Create a join view?," click **OK**. The Create New Join View dialog box appears (see [Figure 57 on page 91](#)).
- 8 You need to set the condition of the join. In the **join** (at the bottom of the dialog box), click **Ø** (red circle with the slash through it). The Join Condition Editor appears (see [Figure 62](#)).

Figure 62 Edit Join Condition



- 9 Select the **Graphical** tab, and perform the following:
 - A Drag **carModelCode** (from **ddnReports**) and **carCode** (from **ddnCarDefs**) onto the canvas.
 - B Drag the = (equal) icon from the tool bar onto the canvas.
 - C Connect the **S1.carModelCode** output to the **left** input.
 - D Connect the **S2.carCode** output to the **right** input.
 - E Click **OK** to set the condition. The Join Condition Editor closes and the Create New Join View dialog box reappears.
- 10 Keep the default join type (Inner) and click **OK** (see Figure 63).

Figure 63 Category Query to Report Sales by carMaker and then carCode



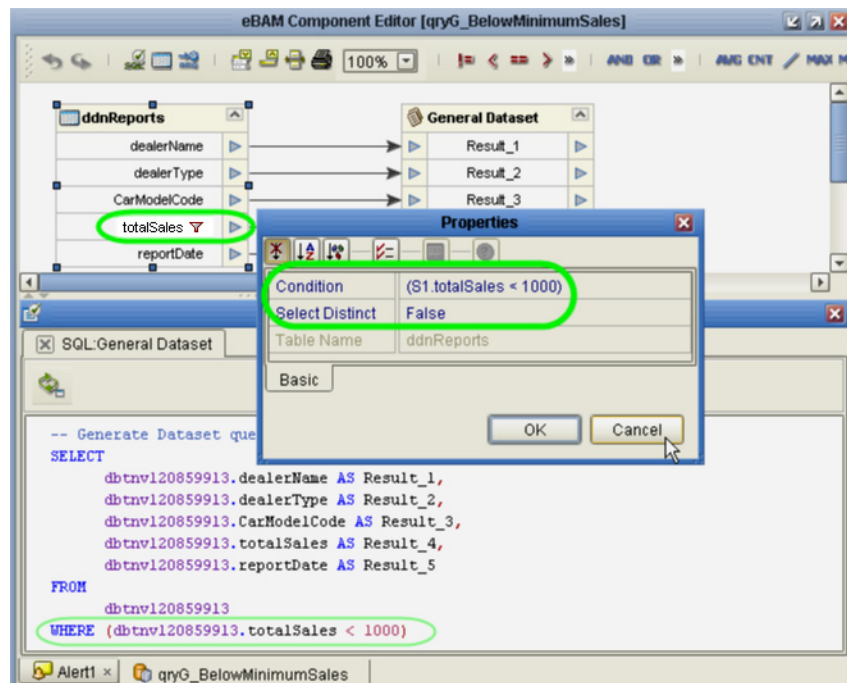
8.3.5 Creating eBAM Sample Alerts

The procedure in this section explains how to create the alert used in the sample

To create the alertBelowMinimum alert, based on general query

- 1 In the project tree, open **prjCars** and then **bamCars**, right-click **Alert Conditions**, and then click **New Alert**. When the alert is created, rename it to **alertBelowMinimum**.
- 2 In the Component Editor, for Stored Query, choose **qryG_BelowMinimumSales**.
- 3 (optional) If you want to see the query and ensure that it operates correctly, click **Edit**. You may have to refresh your memory to ensure that the query has been updated (see Figure 64). When satisfied that the query is set up properly move on to step #4 to continue with the procedure.
- 4 Click the **Alert** tab at the bottom of the canvas (in our example: **alertBelowMinimum**). The eBAM Component Editor (alert Below Minimum) dialog box appears (see Figure 65).
- 5 Ensure that the Stored Query (in the Component Editor) is still **qryG_BelowMinimumSales**; if it is not, select it from the drop down list.

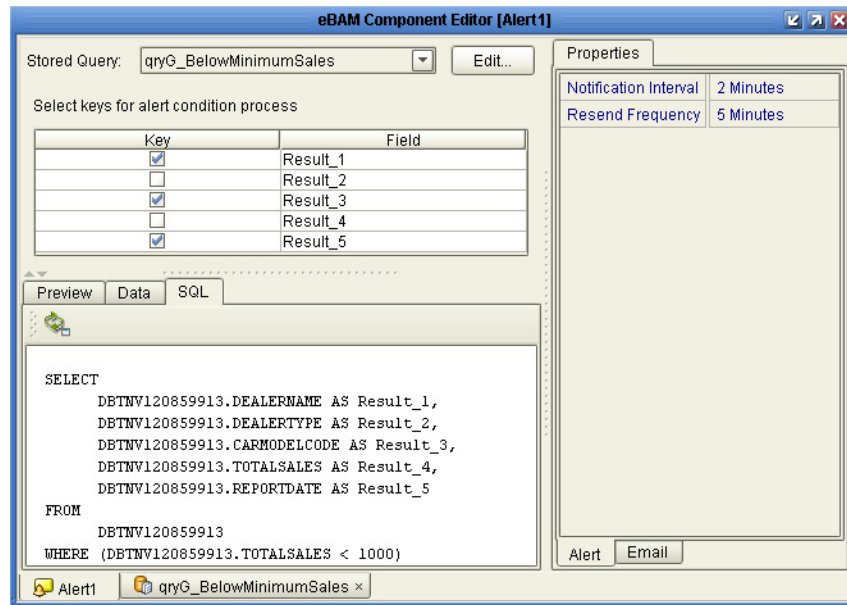
Figure 64 General Query with a Condition



- 6 Under “Select keys for alert condition process,” select the checkboxes for **Result_1**, **Result_3**, and **Result_5**.
- 7 Under the **Alert** tab (of the Properties pane), keep the default values for both **Notification Interval** and **Resend Frequency**.

As Figure 65 shows, the alert supplies three key results: DEALERNAME (Result_1), CARMODELCODE (Result_3), and REPORTDATE (Result_5). Whenever total sales drop below \$1,000, the condition is checked every 2 minutes and alerts are sent out every 5 minutes. In the Connectivity Map, files are set up and named **outputAlertBelowMinimum*.dat**.

Figure 65 Alert for Notifying When Total Sales Drop Below \$1,000.00



8.3.6 Creating eBAM Sample Charts

The procedures in this section explain how to create and configure the charts used in the eBAM sample.

Table 18 Charts Included with the eBAM Sample

Name	Type	Query Details	Notes
chPie_SalesByDealer	Pie	qryC_SumSalesByDealer: Category = carName; no Series	Navigation: chBar_SalesByCarId
chBar_SalesByCarId	Bar (simple)	qryC_SumSalesByCarID: Category = CarModelCode; no Series	Navigation: chBar_SalesByCarnameAndYear
chBar_SalesByCarnameAndYear	Bar (simple)	qryCj_SumSalesByCarnameAndYear: Category = carName; series = carModelYear	Navigation: chBar_SalesByCarnameAndYear_stacked
chBar_SalesByCarnameAndYear_stacked	Bar (stacked)	qryCj_SumSalesByCarnameAndYear: Category = carName; series = carModelYear	No navigation

eBAM Chart chPie_SalesByDealer

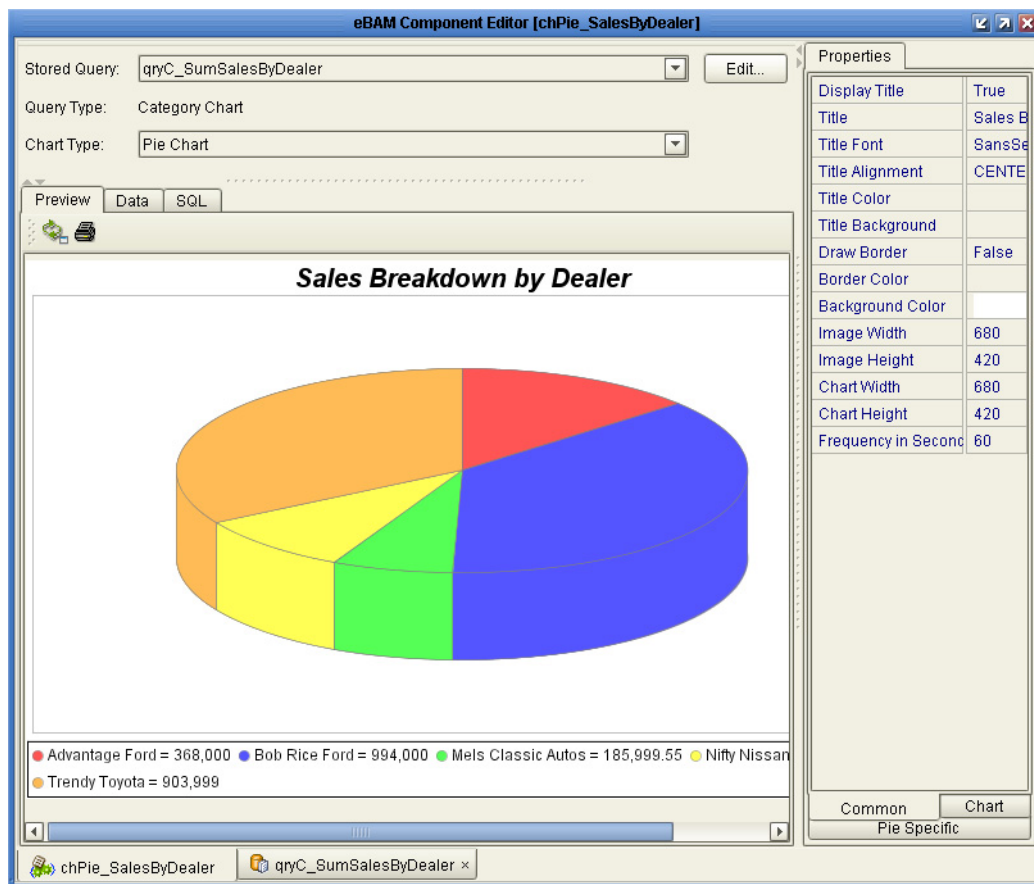
In this pie chart, each slice represents a different car dealer, and the size of the slice is proportional to that dealer's sales.

To create the `chPie_SalesByDealer` chart

- 1 In the project tree, **open prjCars** and then **bamCars**, right-click **Charts**, and click **New Chart**. Name the new chart `chPie_SalesByDealer`.
- 2 In the eBAM Component Editor:
 - A For Stored Query, choose `qryC_SumSalesByDealer`.
 - B For Chart Type, choose **Pie Chart**.
- 3 In the Properties pane:
 - A For Title (under **Common** tab), enter **Sales Breakdown by Dealer**.
 - B For Include Legend (under **Chart** tab), change to **True**.
- 4 Under the eBAM Component Editor **Preview** tab, click the **Refresh chart** icon.

If you committed data for `ddnReports` as mentioned in the [procedure on page 83](#), the Output pane displays the data reported by five dealerships. Each slice represents that dealer's contribution to the sum total of all sales reported. See Figure 66 and the upper left chart in [Figure 45 on page 73](#).

Figure 66 Previewing a Category-Based Pie Chart Using the Committed Sample Data



(optional) To experiment with the `chPie_SalesByDealer` chart

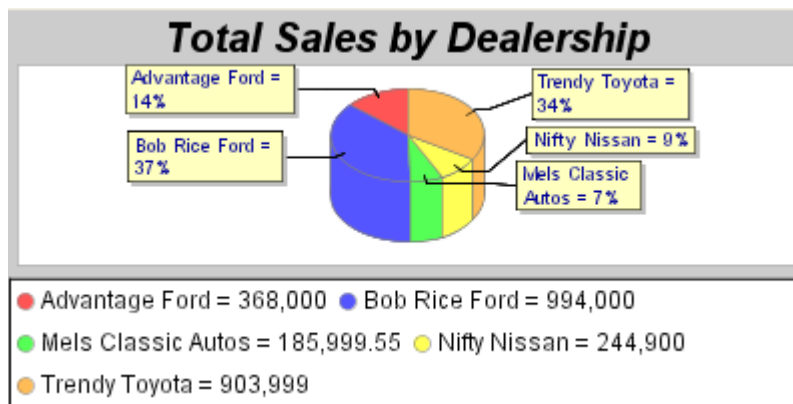
- 1 In the right pane of the eBAM Component Editor is a three-tabbed property sheet for the pie chart.

Note: *The tabs are at the bottom of the pane.*

Experiment with the chart by making changes to various properties, such as:

- ♦ In the **Common** tab, change the Title to **Total Sales by Dealership**
 - ♦ In the **Common** tab, change **Image Width** and **Image Height** to 500 and 200
 - ♦ In the **Chart** tab, change Depth Factor to **0.4** and **Circular** to **True**
 - ♦ In the **Pie Specific** tab, change the Show Section Labels to **True**
- 2 In the Preview pane, click the **Refresh chart** icon. Notice how the chart title and dimensions change: the pie has become a cylinder and the labels are called out in boxes.
 - 3 Repeat the previous two steps to explore how the various chart properties affect the pie chart display (see Figure 67 for an example).

Figure 67 Category Chart `chPie_SalesByDealer` with Modified Chart Properties



- 4 To view a bar chart of the same data, simply change the Chart Type to **Bar Chart** and click the **Refresh chart** icon (see Figure 68 for an example).

Figure 68 Category Chart chPie_SalesByDealer Redrawn as a Bar Chart



- 5 To swap back and forth between the visual representation and the raw numeric data, click the **Data** tab and the **Preview** tab.

Note: After selecting the **Data** tab, click the **Refresh SQL result data** icon to see the raw numeric data.

eBAM Chart chBar_SalesByCarId

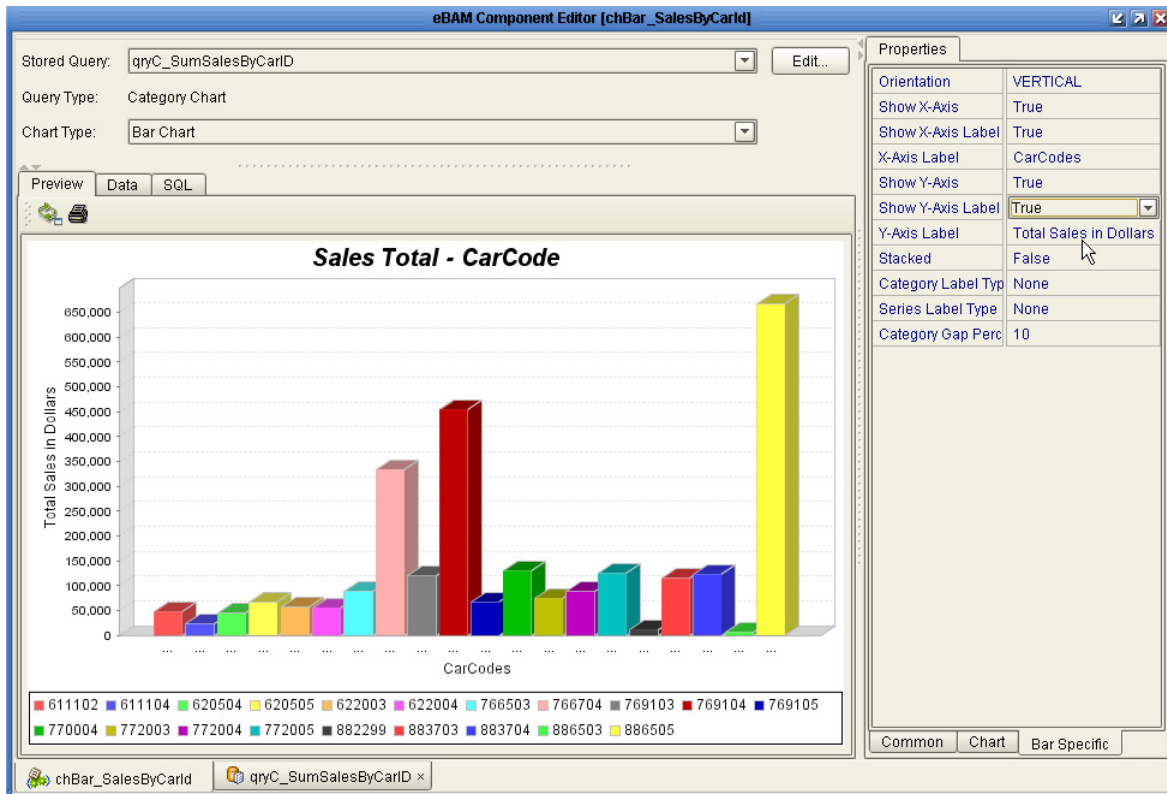
In this bar chart, each bar represents a different car ID, a six-digit code that combines make, model, and year. Total sales of each car ID are represented by the bar length.

To create the chBar_SalesByCarId chart

- 1 In the project tree: Open **prjCars** and then **bamCars**, right-click **Charts**, and then click **New Chart**. Name the new chart **chBar_SalesByCarId**.
- 2 In the eBAM Component Editor: For Stored Query, choose **qryC_SumSalesByCarID**; for Chart Type, choose **Bar Chart**.
- 3 In the Properties pane:
 - ♦ For Title (**Common** tab) enter **Sales Total - CarCode**.
 - ♦ For Include Legend (**Chart** tab) change to **True**.
 - ♦ For X-Axis Label (**Bar Specific** tab) change to **Car Codes**.
 - ♦ For Y-Axis Label (**Bar Specific** tab) change to **Total Sales in Dollars**.
- 4 In the eBAM Component Editor (**Preview** tab), click the **Refresh chart** icon.

If you committed data for ddnReports as mentioned in the [procedure on page 83](#), the Output pane displays the data reported by five dealerships. Each bar represents a different car make-model-year, so the highest-grossing units can be seen at a glance. See Figure 69 and the upper right chart in Figure 45.

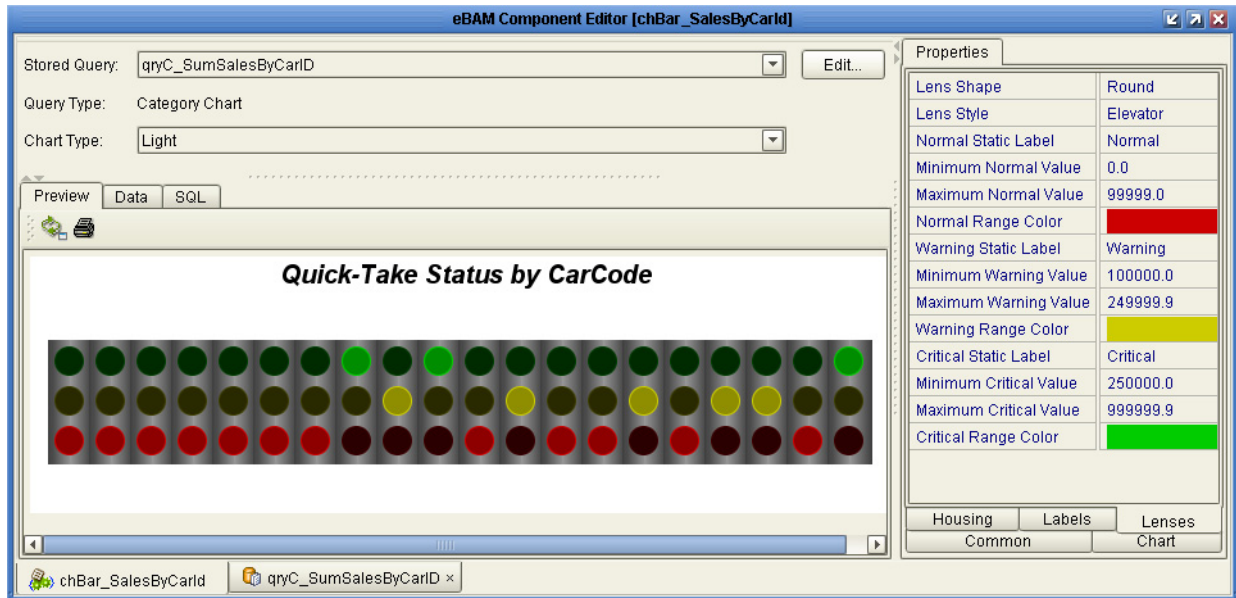
Figure 69 Previewing a Simple Bar Chart Using the Committed Sample Data



(optional) To experiment with the chBar_SalesByCarID chart

- 1 In the eBAM Component Editor, change the chart type from Bar Chart to **Light**, and make the following adjustments in the properties:
 - ♦ In the **Common** tab, change **Title** to: **Quick-Take Status by CarCode**.
In the **Common** tab, change **Image Height** and **Chart Height** to 200 and 220.
 - ♦ In the **Housing** tab, change **Linearize Data** to **True**.
In the **Housing** tab, change **Use Fixed Horizontal Display Limit** to **False**.
 - ♦ In the **Lenses** tab, change **Lens Style** to **Elevator**.
In the **Lenses** tab, change **Maximum Normal Value** to 99999.9.
In the **Lenses** tab, change **Minimum Warning Value** to 100000.0.
In the **Lenses** tab, change **Maximum Warning Value** to 249999.9.
In the **Lenses** tab, change **Minimum Critical Value** to 250000.0.
In the **Lenses** tab, change **Maximum Critical Value** to 999999.9.
- 2 In the Preview pane, click **Refresh chart**. Compare the new result in Figure 70 to the previous display in Figure 69.

Figure 70 Category Chart chBar, Modified to a Trafficlight Array



- 3 Repeat the previous two steps, to explore how the various chart properties affect the trafficlight array display.

eBAM Chart chBar_SalesByCarnameAndYear

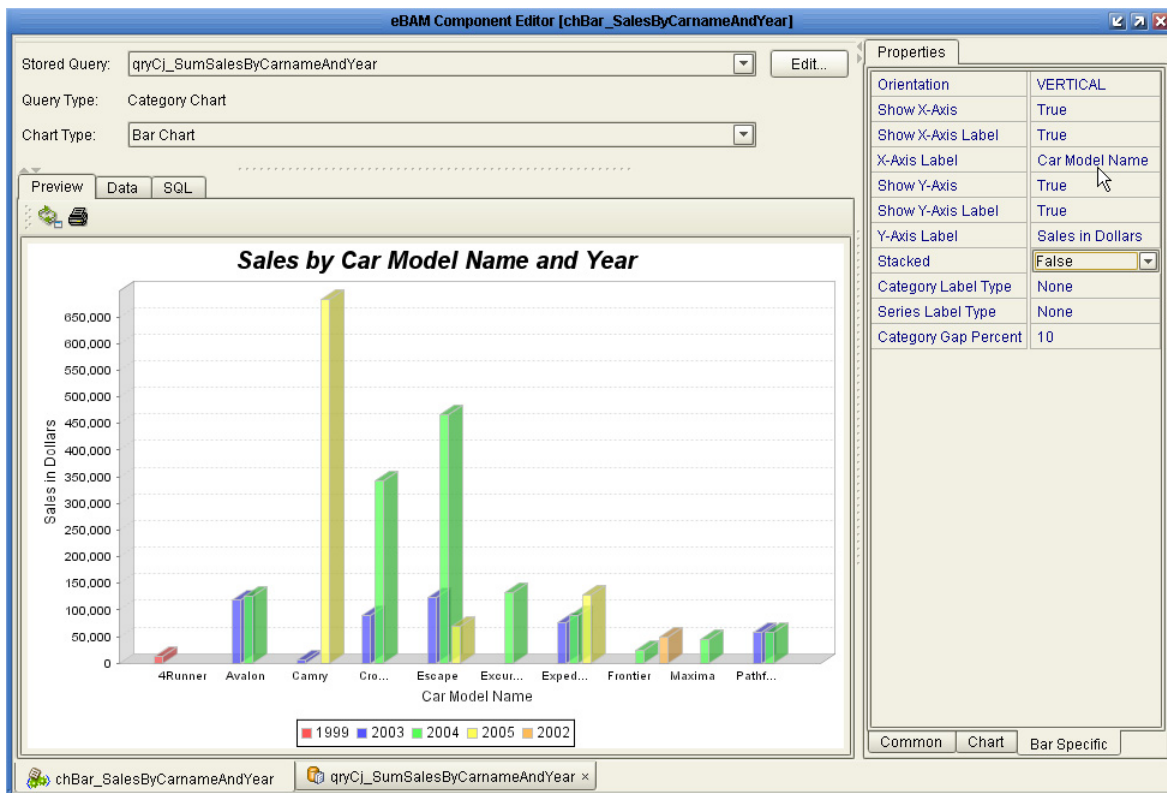
In this category/series bar chart, based on a join query, the bars are clustered by model, and each individual bar is a model year. Total sales are represented by the bar length.

To create the chBar_SalesByCarnameAndYear

- 1 In the project tree: Open **prjCars** and then **bamCars**, right-click **Charts**, and click **New Chart**. Name the new chart **chBar_SalesByCarnameAndYear**.
- 2 In the eBAM Component Editor: For Stored Query, choose **qryCj_SumSalesByCarnameAndYear**; for Chart Type, choose **Bar Chart**.
- 3 In the Properties pane:
 - ♦ For Title (**Common** tab), enter **Sales by Car Model Name and Year**.
 - ♦ For Include Legend (**Chart** tab), change to **True**.
 - ♦ For X-Axis Label (**Bar Specific** tab), change to **Car Model Name**.
 - ♦ For Y-Axis Label (**Bar Specific** tab), change to **Sales in Dollars**.
- 4 In the eBAM Component Editor (**Preview** tab), click the **Refresh chart** icon.

If you committed data for both **ddnReports** and **ddnCarDefs**, the Output pane displays the sales data (from **ddnReports**), parsed by model name and year (from **CarDefs**). This is the same data shown in the previous chart, without the need to know the car IDs. See Figure 71 and the lower left chart in Figure 45.

Figure 71 Previewing a Category-Series Bar Chart Using All Committed Sample Data



eBAM Chart chBar_SalesByCarnameAndYear_stacked

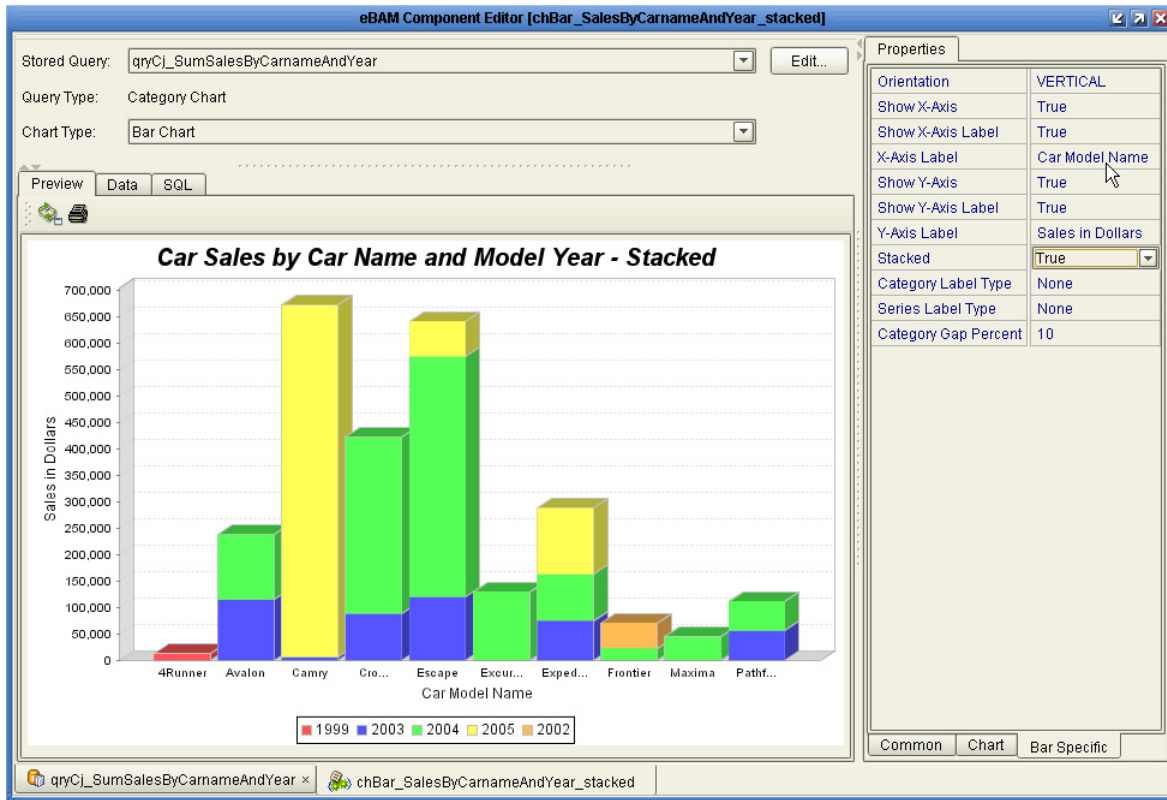
In this category/series bar chart, based on a join query, the bar stacks are models, and each block in the stack is a model year. Total sales are represented by the bar length.

To create the chBar_SalesByCarnameAndYear_stacked

- 1 In the project tree: Open **prjCars** and then **bamCars**, right-click **Charts**, and then click **New Chart**. Name the new chart **chBar_SalesByCarnameAndYear_stacked**.
- 2 In the eBAM Component Editor: For Stored Query, choose **qryCj_SumSalesByCarnameAndYear**; for Chart Type, choose **Bar Chart**.
- 3 In the Properties pane:
 - ♦ For Title (**Common** tab), enter **Sales by Car Name and Model Year - Stacked**.
 - ♦ For Include Legend (**Chart** tab), change to **True**.
 - ♦ For X-Axis Label (**Bar Specific** tab), change to **Car Model Name**.
 - ♦ For Y-Axis Label (**Bar Specific** tab), change to **Sales in Dollars**.
 - ♦ For Stacked (**Bar Specific** tab), change to **True**.
- 4 In the eBAM Component Editor **Preview** tab, click the **Refresh chart** icon.

This is the same data shown in the previous chart, but stacked instead of clustered. See Figure 72 and the lower right chart in Figure 45.

Figure 72 Previewing a Category-Series Bar Chart Using All Committed Sample Data



8.3.7 Creating Navigation Links

The procedure in this section explains how to set up the links for navigating from one chart to another.

To create navigation links

- 1 Open `chPie_SalesByDealer` and set the following Properties in the **Chart** tab:
 - ♦ Change **Display Navigation Buttons** to **True**.
 - ♦ For Navigation Chart Name, enter `chBar_SalesByCarId`.
- 2 Open `chBar_SalesByCarId` and set the following Properties in the **Chart** tab:
 - ♦ Change **Display Navigation Buttons** to **True**.
 - ♦ For Navigation Chart Name, enter `chBar_SalesByCarnameAndYear`.
- 3 Open `chBar_SalesByCarnameAndYear` and set the following Properties in the **Chart** tab:
 - ♦ Change **Display Navigation Buttons** to **True**.
 - ♦ For Navigation Chart Name, enter `chBar_SalesByCarnameAndYear_stacked`.

8.3.8 Creating and Validating Business Processes

These procedures help you get acquainted with the sample business processes (BPs) by taking you through steps for creating, configuring, and validating them.

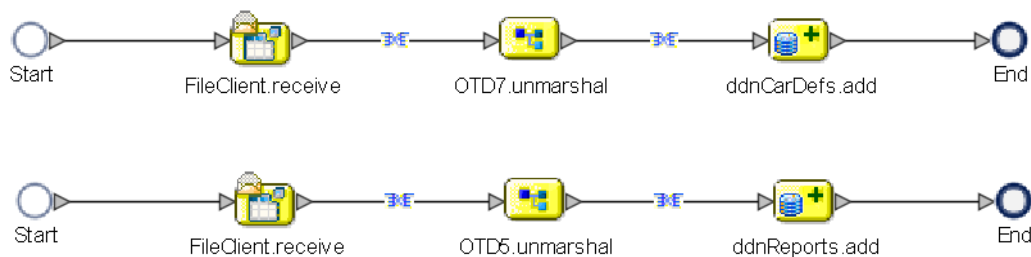
Table 19 Business Processes Included with the eBAM Sample

Name	Purpose	Web Services, Activities, and Methods
bpAddCarDefs	Polls a file, reads data and parses into seven fields, and then adds it to a data definition (ddnCarDefs).	FileClient. receive ; OTD7. unmarshal ; ddnCarDefs. add
bpAddReports	Polls a file, reads data and parses into five fields, and then adds it to a data definition (ddnReports).	FileClient. receive ; OTD5. unmarshal ; ddnReports. add
bpDelTables	Polls a file; if found, deletes data from both data definitions.	FileClient. receive ; ddnCarDefs. delete ; ddnReports. delete
bpQryExtractSalesByCarName	Polls a file, reads a single datum, passes it as a runtime argument to the execute web service, marshals the resulting output and writes to an output file.	FileClient. receive ; qryG_ExtractSalesByCarName. execute ; OTD3. marshal ; FileClient. write
bpAlertBelowMinimum	Samples datastream for a threshold condition; if true, marshals the data and writes to an output file.	alertBelowMinimum. notify ; OTD5. unmarshal ; FileClient. write

Business Processes bpAddCarDefs and bpAddReports

The bpAddCarDefs and bpAddReports business processes operate in identical fashion: Read the data (using the **receive** web service of the FileClient), parse the data (using the **unmarshal** method of the correct OTD), and put the data into the eBAM application (using the **add** web service of the corresponding data definition). See Figure 73.

Figure 73 Business Processes bpAddCarDefs and bpAddReports



To create the bpAddCarDefs business processes

- 1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Business Process**.

The eInsight Business Process Designer appears.

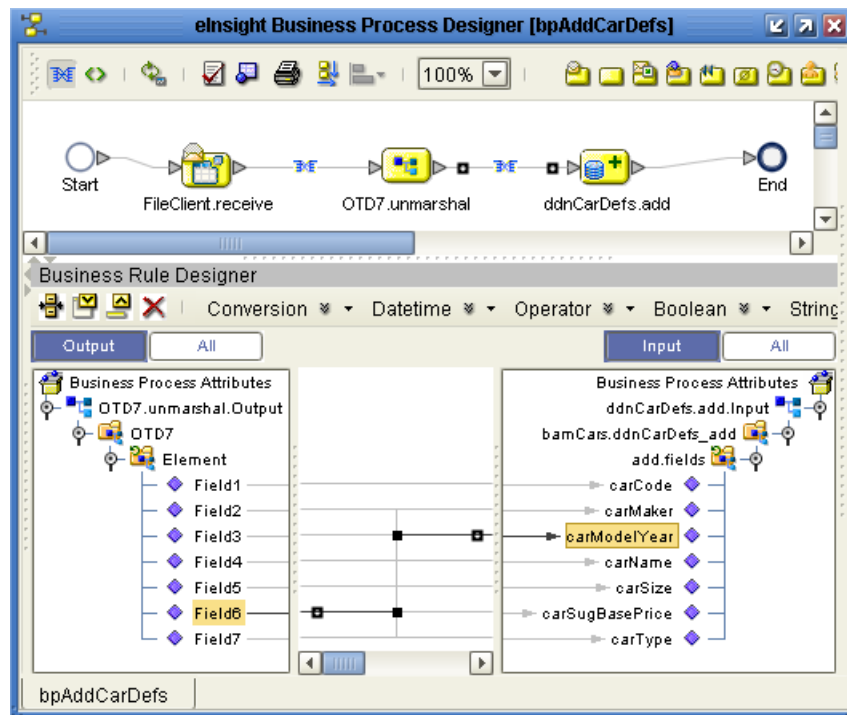
- 2 Rename the new business process from **BusinessProcess1** to **bpAddCarDefs**.
- 3 In the project tree, open **Sun SeeBeyond**, **eWays**, **File**, and **FileClient** and then drag its **receive** activity onto the canvas.
- 4 In the project tree, open **prjCars** and **OTD7** and then drag its **unmarshal** activity onto the canvas.
- 5 In the project tree, open **prjCars**, **bamCars**, **Data Definitions**, and **ddnCarDefs** and then drag its **add** web service onto the canvas.
- 6 Connect the following:
 - A The output of **Start** to the input of **FileClient.receive**.
 - B The output of **receive** to the input of **OTD7.unmarshal** and add a Business Rule.

Note: To add a Business Rule, click the connector between the output and the input and then select **Add Business Rule**.

- C The output of **unmarshal** to the input of **ddnCarDefs.add** and add a Business Rule.
 - D The output of **add** to the input of **End**.
- See the upper half of Figure 73.
- 7 Open the Business Rule Designer and add the following transformation mappings:
 - A For **receive-to-unmarshal**, open both OTDs and then map the **...Output text** element to the **...Input contents** element.
 - B For **unmarshal-to-add**, open **...Output**, **OTD7**, **Element** and **...Input**, **...ddnCarDefs_add**, **add.fields** and then map as follows:
 - ♦ Field1 to carCode
 - ♦ Field2 to carName
 - ♦ Field3 to carMaker
 - ♦ Field4 to carType
 - ♦ Field5 to carSize
 - ♦ Field6 to carModelYear
 - ♦ Field7 to carSugBasePrice

See Figure 74.

Figure 74 Mapping OTD7.unmarshal.Output to ddnCarDefs.add.Input



- 8 In the toolbar of the Business Process Designer, click **Validate Business Process Model**.

The bpAddReports model is valid except for its unused Fault attributes. To clear the unused Fault attributes: Right-click the business process in the project tree, open its properties and click the **Business Process Attributes** tab, and then delete the Fault attributes whose **In Use** value is No.

To create the bpAddReports business processes

- 1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Business Process**.
The eInsight Business Process Designer appears.
- 2 Rename the new business process from BusinessProcess1 to **bpAddReports**.
- 3 In the project tree, open **Sun SeeBeyond**, **eWays**, **File**, and **FileClient** and then drag its **receive** activity onto the canvas.
- 4 In the project tree, open **prjCars** and **OTD5** and then drag its **unmarshal** activity onto the canvas.
- 5 In the project tree, open **prjCars**, **bamCars**, **Data Definitions**, and **ddnReports** and then drag its **add** web service onto the canvas.
- 6 Connect the following:
 - A The output of **Start** to the input of FileClient.receive.
 - B The output of **receive** to the input of OTD5.unmarshal and add a Business Rule.

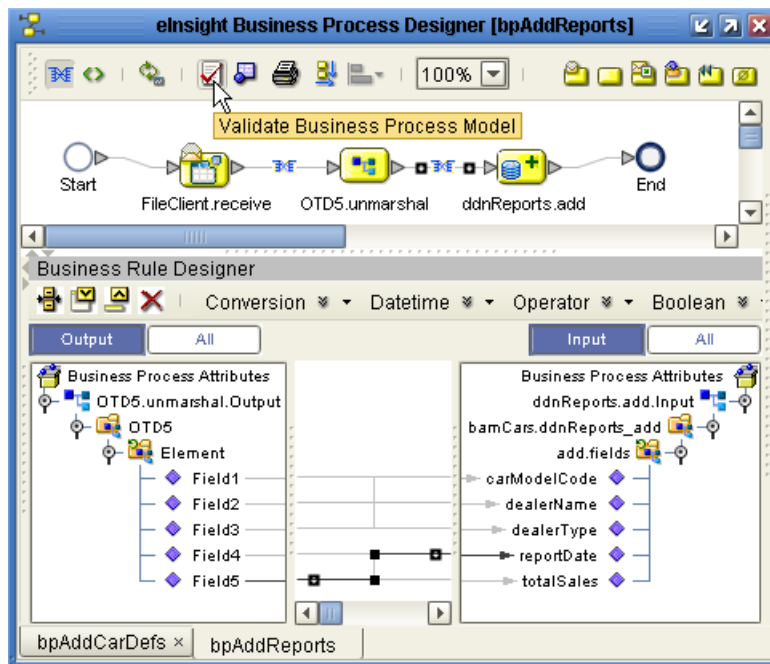
- C The output of **unmarshal** to the input of **ddnReports.add** and add a business rule.
- D The output of **add** to the input of **End**.

See the lower half of Figure 73.

- 7 Open the Business Rule Designer and add the following transformation mappings:
 - A For **receive-to-unmarshal**, open both OTDs and then map the **...Output text** element to the **...Input.contents** element.
 - B For **unmarshal-to-add**, open **...Output**, **OTD5, Element** and **...Input**, **ddnReports_add**, **add.fields** and then map as follows:
 - ♦ Field1 to dealerName
 - ♦ Field2 to dealerType
 - ♦ Field3 to carModelCode
 - ♦ Field4 to totalSales
 - ♦ Field5 to reportDate

See Figure 75.

Figure 75 Mapping OTD5.unmarshal.Output to ddnReports.add.Input



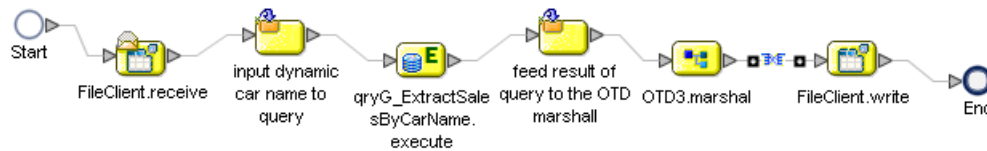
- 8 In the toolbar of the Business Process Designer, click **Validate Business Process Model**.

The bpAddReports model is valid except for its unused Fault attributes.

Business Process bpQryExtractSalesByCarName

The bpQryExtractSalesByCarName illustrates using dynamic input to a query. It reads data (using the **receive** web service of the FileClient). See Figure 76.

Figure 76 Business Process bpQryExtractSalesByCarName

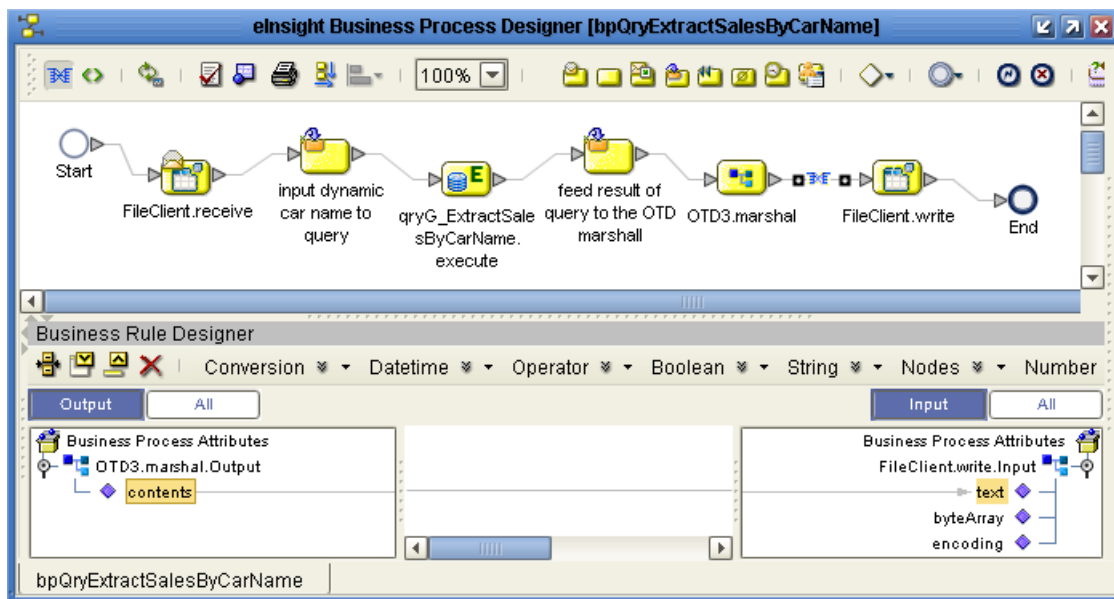


To create the bpQryExtractSalesByCarName business processes

- 1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Business Process**.
The eInsight Business Process Designer appears.
- 2 Rename the BP from **BusinessProcess1** to **bpQryExtractSalesByCarName**.
- 3 In the project tree, open **Sun SeeBeyond**, **eWays**, **File**, and **FileClient** and then drag its **receive** activity onto the canvas.
- 4 From the tool palette, drag a **Business Rule** onto the canvas and rename it to **input dynamic car name to query**.
- 5 In the project tree, open **prjCars**, **bamCars**, **eBAM Queries**, and **qryG_ExtractSalesByCarName** and then drag its **execute** web service onto the canvas.
- 6 From the tool palette, drag a **Business Rule** onto the canvas and rename it to **feed result of query to the OTD marshal**.
- 7 In the project tree, open **prjCars** and **OTD3** and then drag its **marshal** activity onto the canvas.
- 8 In the project tree, from **Sun SeeBeyond**, **eWays**, **File**, and **FileClient**, and then drag its **write** activity onto the canvas.
- 9 Connect the following:
 - A The output of **Start** to the input of **FileClient.receive**.
 - B The output of **receive** to the input of the first Business Rule ("**input dynamic ...**").
 - C The output of the first Business Rule ("**input dynamic ...**") to the input of **qryG_...execute**.
 - D The output of **qryG_...execute** to the input of the second Business Rule ("**feed result ...**").
 - E The output of the second Business Rule ("**feed result ...**") to the input of **OTD3.marshall**.
 - F The output of **marshal** to the input of **FileClient.write** and add a business rule.
 - G The output of **write** to the input of **End** (see Figure 78).

- 10 Open the Business Rule Designer and add the following transformation mappings:
 - A For **input dynamic car name to query**, open both sides and then map **...Output text** to **...RuntimeArgs rg_carName**.
 - B For **feed result of query to the OTD marshal**, open both sides and then map **Result_carName** to **Field 1**, **Result_carModelYear** to **Field 2**, and **Result_SumofSales** to **Field 3**.
 - C For **marshal-to-write**, open both sides and then map the **...Output contents** element to the **...Input text** element. See Figure 80.

Figure 77 Mapping OTD3.marshal.Output to FileClient.write.Input



- 11 In the toolbar of the Business Process Designer, click **Validate Business Process Model**.

The bpQryExtractSalesByCarName model is valid except for its unused Fault attributes.

Business Process bpDelTables

The bpDelTables business process operates even more simply as the bpAdd* processes, but with the opposite result: It reads data (using the **receive** web service of the FileClient), and then invokes the **delete** web services of the two data definitions (see Figure 78).

Figure 78 Business Process bpDelTables



To create the bpDelTables business processes

- 1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Business Process**.
The eInsight Business Process Designer appears.
- 2 Rename the new business process from **BusinessProcess1** to **bpDelTables**.
- 3 In the project tree, open **Sun SeeBeyond**, **eWays**, **File**, and **FileClient** and then drag its **receive** activity onto the canvas.
- 4 In the project tree, open **prjCars**, **bamCars**, **Data Definitions**, and **ddnCarDefs** and then drag its **delete** web service onto the canvas.
- 5 In the project tree, open **prjCars**, **bamCars**, **Data Definitions**, and **ddnReports** and then drag its **delete** web service onto the canvas.
- 6 Connect the following:
 - A The output of **Start** to the input of **FileClient.receive**.
 - B The output of **receive** to the input of **ddnCarDefs.delete**.
 - C The output of the first **delete** to the input of **ddnReports.delete**.
 - D The output of the second **delete** to the input of **End** (see Figure 78).
- 7 In the toolbar of the Business Process Designer, click **Validate Business Process Model**.

The bpDelTables model is valid except for its unused Fault attributes.

Business Process bpAlertBelowMinimum

The bpAlertBelowMinimum business process operates as simply as the others, but does not introduce any change. It watches the data and, when the filtering condition in its **when** clause is satisfied by the data, invokes its **notify** web service to pass data (via the **marshal** method of OTD5) out to the **write** web service of the FileClient. See Figure 79.

Figure 79 Business Process bpAlertBelowMinimum



To create the bpAlertBelowMinimum business processes

- 1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Business Process**.
The eInsight Business Process Designer appears.
- 2 Rename the new business process to **bpAlertBelowMinimum**.
- 3 In the project tree, open **prjCars**, **bamCars**, **Alert Conditions**, and **alertBelowMinimum** and then drag its **notify** web service onto the canvas.

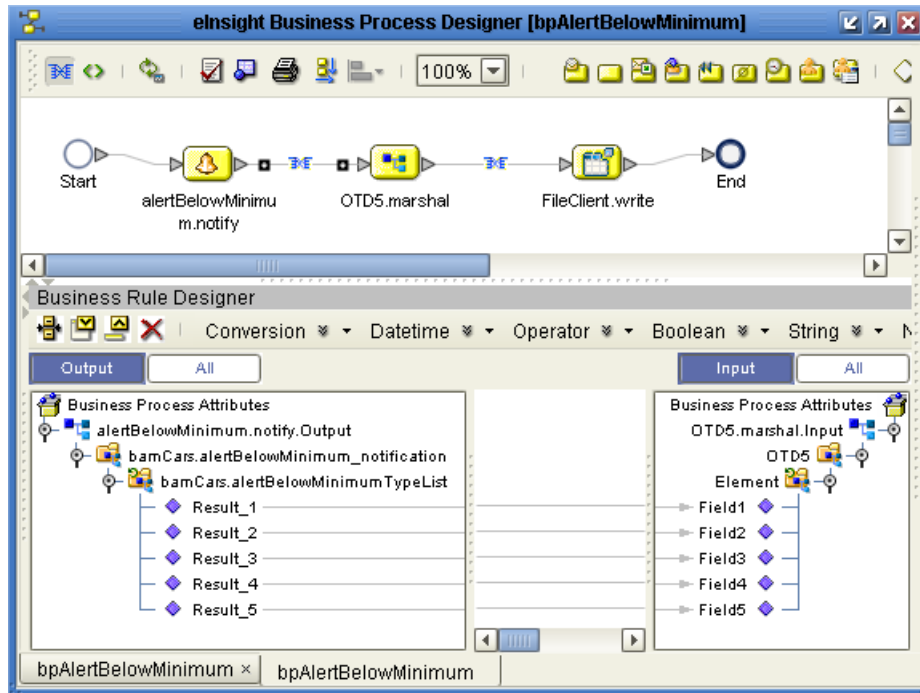
- 4 In the project tree, open **prjCars** and **OTD5** and then drag its **marshal** activity onto the canvas.
- 5 In the project tree, open **Sun SeeBeyond**, **eWays**, **File**, and **FileClient** and then drag its **write** activity onto the canvas.
- 6 Connect the following:
 - A The output of **Start** to the input of **alertBelowMinimum.notify**.
 - B The output of **notify** to the input of **OTD5.marshal** and add a Business Rule.
 - C The output of **marshal** to the input of **FileClient.write** and add a Business Rule.
 - D The output of **write** to the input of **End**.

See Figure 75.

- 7 Open the Business Rule Designer and add the following transformation mappings:
 - A For **notify-to-marshal**, open **...Output**, **..._notification**, and **...TypeList** on the left and **...Input**, **OTD5**, and **Element** on the right and then map as follows:
 - ♦ Result_1 to Field1,
 - ♦ Result_2 to Field2,
 - ♦ Result_3 to Field3,
 - ♦ Result_4 to Field4, and
 - ♦ Result_5 to Field5.

See Figure 80.

Figure 80 Mapping OTD5.unmarshal.Output to ddnReports.add.Input



B For **marshal-to-write**, open both sides and then map the ...**Output** contents element to the ...**Input** text element.

8 In the toolbar of the Business Process Designer, click **Validate Business Process Model**.

The bpAlertBelowMinimum model is valid except for unused Fault attributes.

8.3.9 Creating and Configuring Connectivity Maps

In the Connectivity Map you create, **cmCars**, you connect instances of all the components in your Project and configure the connections. For a preview of the result, see Figure 81.

To create a Connectivity Map

1 In the project tree, right-click **prjCars** and, on the pop-up menu, point at **New** and click **Connectivity Map**.

The Connectivity Map Editor appears.

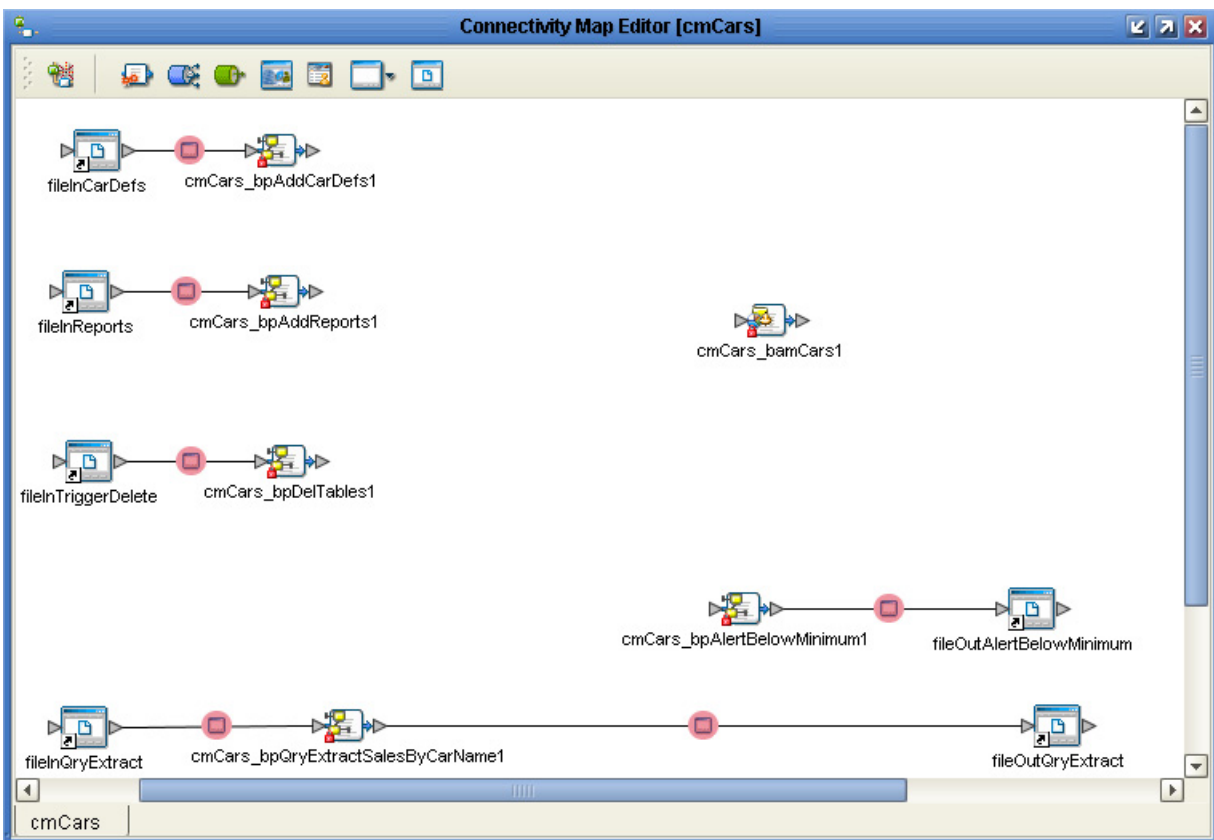
2 Rename the new Connectivity Map to **cmCars**.

3 From **External Applications** icon on the tool palette, drag six instances of the **File External Application** onto the canvas, naming them as follows:

- ♦ **fileInCarDefs** (on the far left).
- ♦ **fileInReports** (on the far left, just below the preceding).
- ♦ **fileInTriggerDelete** (on the far left, just below the preceding).

- ♦ **fileInQryExtract** (on the far left, well below the preceding three).
 - ♦ **fileOutQryExtract** (on the far right, at the same level as the preceding).
 - ♦ **fileOutAlertBelowMinimum** (on the far right, above the preceding).
- 4 From the project tree, drag the following five Business Processes onto the canvas:
 - ♦ **bpAddCarDefs** (just to the right of fileInCarDefs).
 - ♦ **bpAddReports** (just to the right of fileInReports).
 - ♦ **bpDelTables** (just to the right of fileInTriggerDelete).
 - ♦ **bpQryExtractSalesByCarName** (just to the right of fileInQryExtract).
 - ♦ **bpAlertBelowMinimum** (just to the left of fileOutAlertBelowMinimum).
 - 5 From the project tree, drag the eBAM application itself (**bamCars**) to the center of the map.
 - 6 Connect the external file components as shown in Figure 81.

Figure 81 Connectivity Map for Sample



Before you can build your application, you must configure the File eWays. You do not need to worry about the directory paths where input is read and output is written, since you configured the extFileBam external already, in the [procedure on page 78](#).

To configure the eWays in a Connectivity Map

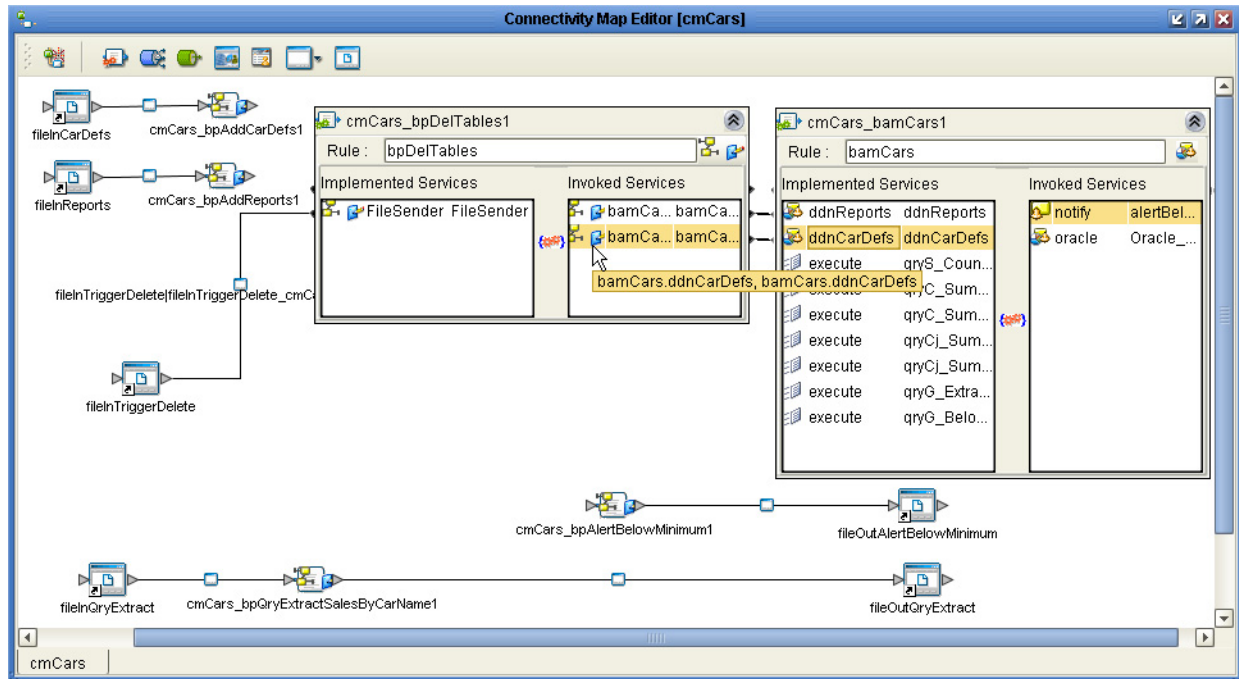
- 1 In the Connectivity Map Editor, double-click each inbound eWay connecting fileIn* to cmCars_bp* and ensure the following parameters are set correctly:
 - ♦ **Input file name** should be as follows:
 - ♦ For fileInCarDefs, **inputCarDefs*.txt**.
 - ♦ For fileInReports, **inputReports*.txt**.
 - ♦ For fileInTriggerDelete, **TriggerDelete.txt**.
 - ♦ For fileInQryExtract, **inputQryName*.txt**.
 - ♦ **Remove EOL** should be **False**.
 - ♦ **Multiple records per file** should be **False**.
 - ♦ Verify that all parameters are correctly set for each inbound eWay connecting fileIn*, and then click **OK**.
- 2 Double-click each outbound eWay connecting cmCars_bp* to fileOut* and ensure the following parameters are set correctly:
 - ♦ **Output file name** should be as follows:
 - ♦ For fileAlertOutBelowMinimum, **outputAlertBelowMinimum%d.dat**.
 - ♦ For fileOutQryExtract, **outputQryNameResults%d.dat**.
 - ♦ **Add EOL** should be **True**.
 - ♦ **Multiple records per file** should be **False**.
 - ♦ Verify that all parameters are correctly set for each outbound eWay connecting fileOut*, and then click **OK**.

All eWays have been configured. The next step is to link the services between the eBAM application and each of the Business Processes that communicate with it.

To link the services in a Connectivity Map

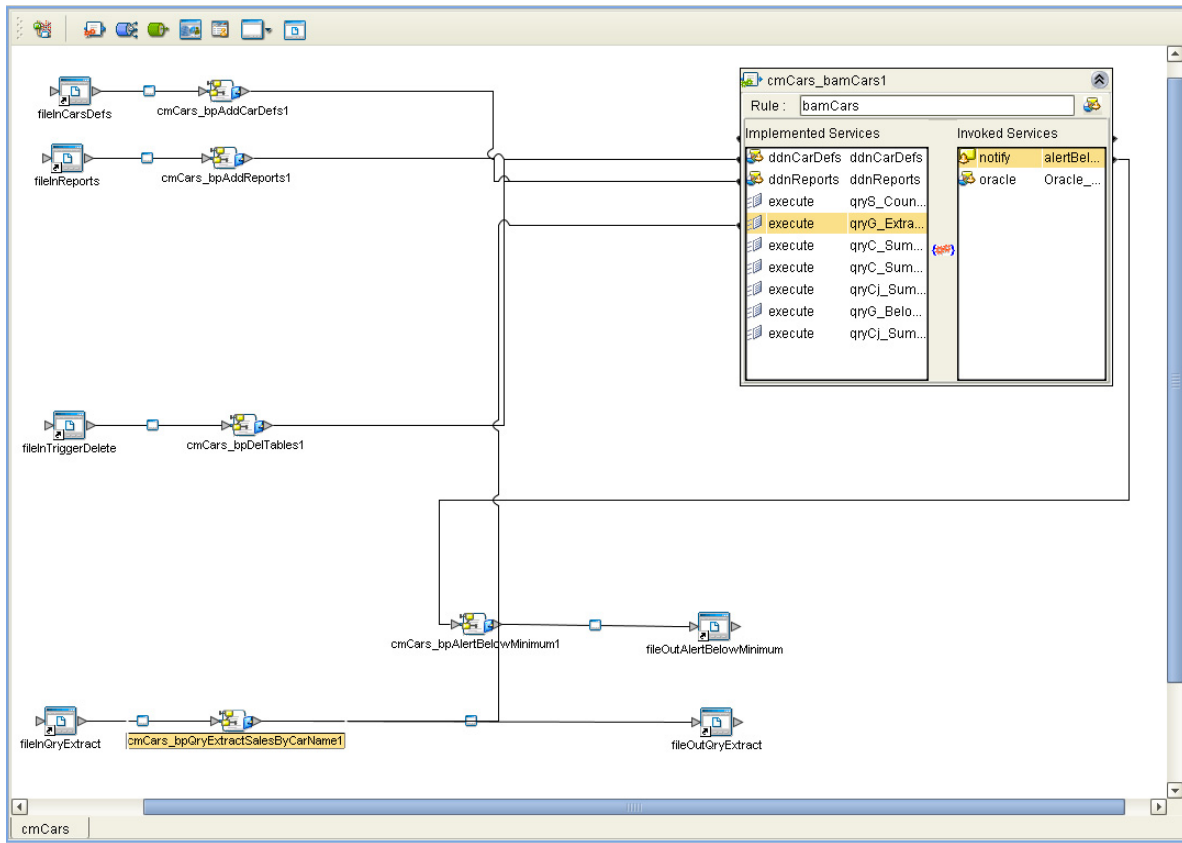
- 1 In the Connectivity Map Editor, double-click the eBAM application instance (**cmCars_bamCars1**) to open it, and then double-click the bpDelTables instance (**cmCars_bpDelTables1**) to open it.
- 2 Drag the invoked service **bamCars.ddnReports** onto the implemented service **ddnReports**, and then drag the invoked service **bamCars.ddnCarDefs** onto the implemented service **ddnCarDefs** (see Figure 82).

Figure 82 Connecting Both Services of bpDelTables to bamCars



- 3 Collapse the **cmCars_bpDelTables1** instance, and then, one by one, open each of the other three Business Processes on the left and connect the correct invoked service to the corresponding implemented service in the eBAM application:
 - ♦ Connect **bpAddCarDefs** (bamCars.ddnCarDefs) to **ddnCarDefs**.
 - ♦ Connect **bpAddReports** (bamCars.ddnReports) to **ddnReports**.
 - ♦ Connect **bpQryExtractSalesByCarNames** (...qryG_ExtractSalesByCarName) to **qryG_ExtractSalesByCarName_execute**.
- 4 From the eBAM application's invoked services, connect the **notify** service to the **cmCars_bpAlertBelowMinimum1** instance (see Figure 83).

Figure 83 All Connections Completed Between bamCars and BPs



- 5 Save your work and close all canvases.

All parameters have been configured. The next step is to start a Domain (if you have not already done so), and then create and activate a deployment profile for the Project.

8.3.10 Starting the Domain to Run the Sample

These steps assume you have already created a Domain named **domain1**, and that a script named **start-domain1** has started it.

To start the Domain via the command line

- Open a command prompt and change directories to the location of your Logical Host. Then start the Domain using the appropriate parameters. For example:

```
cd <JavaCAPS513>\logicalhost
.\start_domain1
```

Note: You can also start the Domain by using **domainmgr.bat**, which is located in your **\logicalhost** directory.

As soon as the Domain is running, it is ready to have a Project deployed to it.

8.4 Building, Deploying, and Monitoring a Project

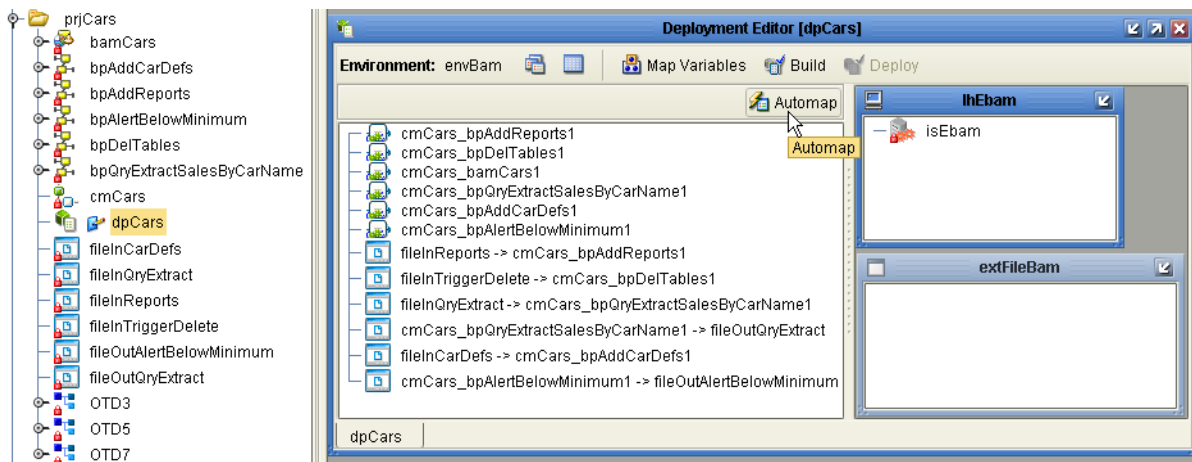
After the data definitions, two charts, and an alert condition are set up, you are ready to create a deployment profile named **dpCars**, and then activate and deploy it to the Domain that is currently running. (If a Domain is not already running, see [“Starting the Domain to Run the Sample” on page 119.](#))

To create the deployment profile and then build and deploy the Project

- 1 In Project Explorer, right-click **prjCars** and, on the pop-up context menu, point at **New** and click **Deployment Profile**.
- 2 In the dialog box, name it **dpCars**, and be sure it references the **envBam** environment and the **cmCars** Connectivity Map before clicking **OK**.

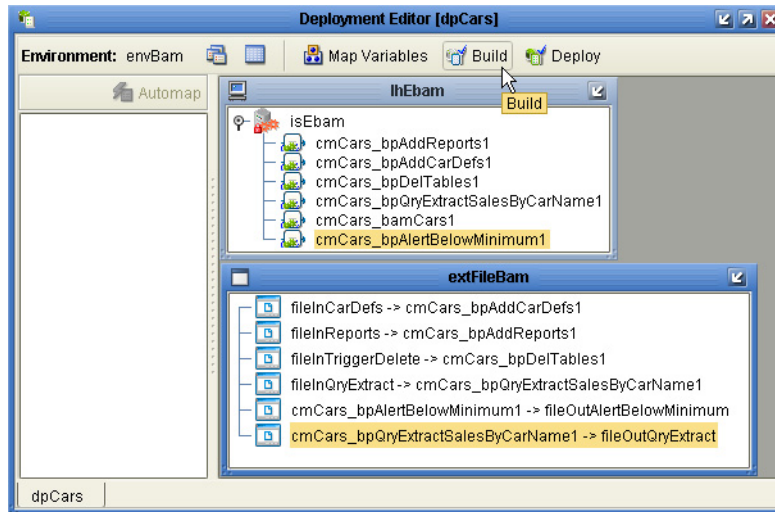
The project tree displays the new object, and the Deployment Editor shows the components and the servers to which you will assign them. See Figure 84.

Figure 84 Deployment Profile for Sample, Before Component Mapping



- 3 To assign the components to their servers, click **Automap**. See Figure 85.

Figure 85 Deployment Profile for Sample, After Component Mapping



- 4 To build the Project, click **Build**.
- 5 After the Project builds successfully, click **Deploy** and then click **Yes**.

Or

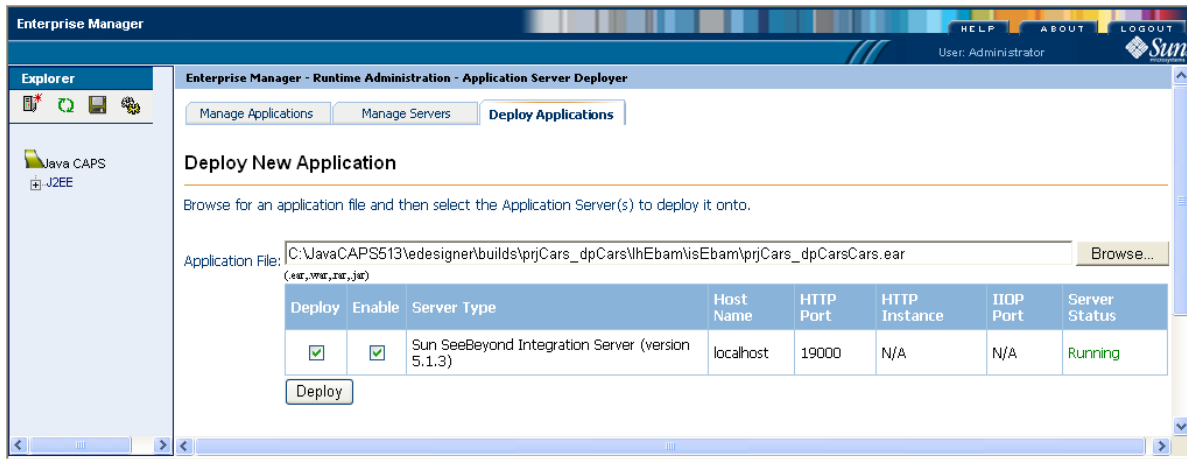
Use Enterprise Manager to deploy and enable the Project's .ear file—**prjCarsdpCars.ear**—in `<edesigner>\builds\prjCarsdpCars\lhEbam\isEbam\`. See Figure 86.

- A Start Enterprise Manager.

Note: *A Logical Host must be running.*

- B Double-click **J2EE** in the Explorer pane.
- C Enter the information to add an Application Server and click **Connect to Server** (see Figure 86).
- D When the Current Application Server List appears, click **Deploy Applications**.

Figure 86 Using Enterprise Manager to Deploy and Enable the Application



- 6 Browse to the EAR file, select it, and then click **Deploy**.

Note: Make sure that the *Deploy* and *Enable* boxes are selected.

To monitor the Project

- 1 Open a new browser session.

Important: Do not clone a new window from an existing session.

- 2 Point your browser at the URL for monitoring the eBAM charts for this project. This takes the following form:

`http://Lhname:18001/PrjnameAppinstancenameDpname/`

where:

- ♦ **Lhname** is the computer name specified in the Integration Server URL for isEbam. For example (if you are running on your computer): **localhost**.
- ♦ **PrjnameAppinstancenameDpname** uniquely identifies the deployment profile and eBAM application instance, with underscores stripped out:
 - ♦ **Prjname** is the name of the Project that contains the deployment profile; for this sample, it is **prjCars**.
 - ♦ **Appinstancename** is the instance name of the eBAM application as it appears on the Connectivity Map (but with underscores stripped out); for this sample, it is **cmCarsbamCars1**.
 - ♦ **Dpname** is the name of the deployment profile; for this sample, it is **dpCars**.
- ♦ The entire string is case-sensitive, and any underscores must be stripped out; for example, the instance name must be **cmCarsbamCars1** (not cmCars_bamCars1).

Thus, if you have retained all defaults and are running the integration server on your local machine, this is the URL:

`http://localhost:18001/prjCarscmCarsbamCars1dpCars/`

If your integration server is running on another computer, or if it does not use port 18001 for HTTP, then substitute the correct computer name and port in the URL.

At first, since no data has been fed in, the chart view is blank. Continue with [“Monitoring Key Performance Indicators with Live Data” on page 123](#).

8.5 Monitoring Key Performance Indicators with Live Data

Now that the data definitions, two charts, and an alert condition are set up, you are ready to look at the Chart Viewer to see how it updates in real time as it receives sample data.

8.5.1 Adding New Data

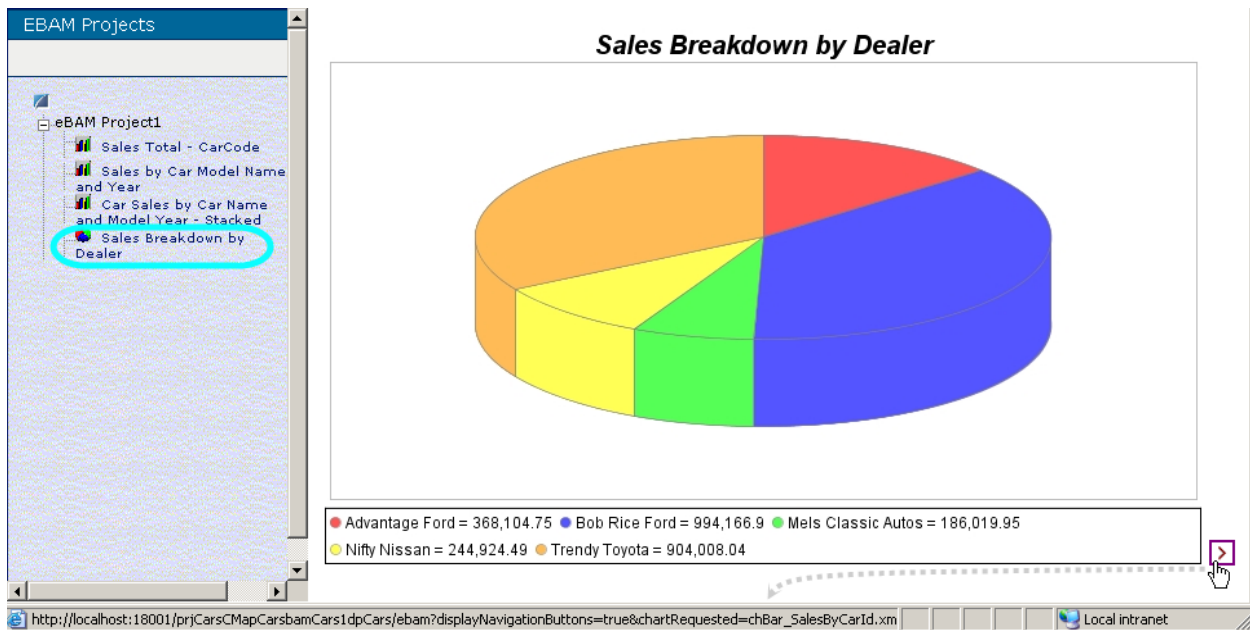
In this section, you will feed live data that is constructed to use the **add** services of the two bpAdd* business processes. The process keeps existing data and adds new records.

To add new data and see the accumulated results

- 1 Browse (or open a command prompt and change directories) to the location where you installed the sample input data, and verify the presence of sample data files (inputCarDefs.txt.~in, inputReports_all.txt.~in, and so forth). For example:

```
cd C:\temp\eBAM\Sample\Data\In
```
- 2 Take a copy of **inputReports5_original.~in** and rename it to **inputReports001.txt** and then watch as the file is picked up and renamed to **inputReports001.txt.~in**.
- 3 In the Chart Viewer, if you have not already done so, click the pie-chart item (the fourth link in the tree, labeled **Sales Breakdown by Dealer**. For the result, see Figure 87.

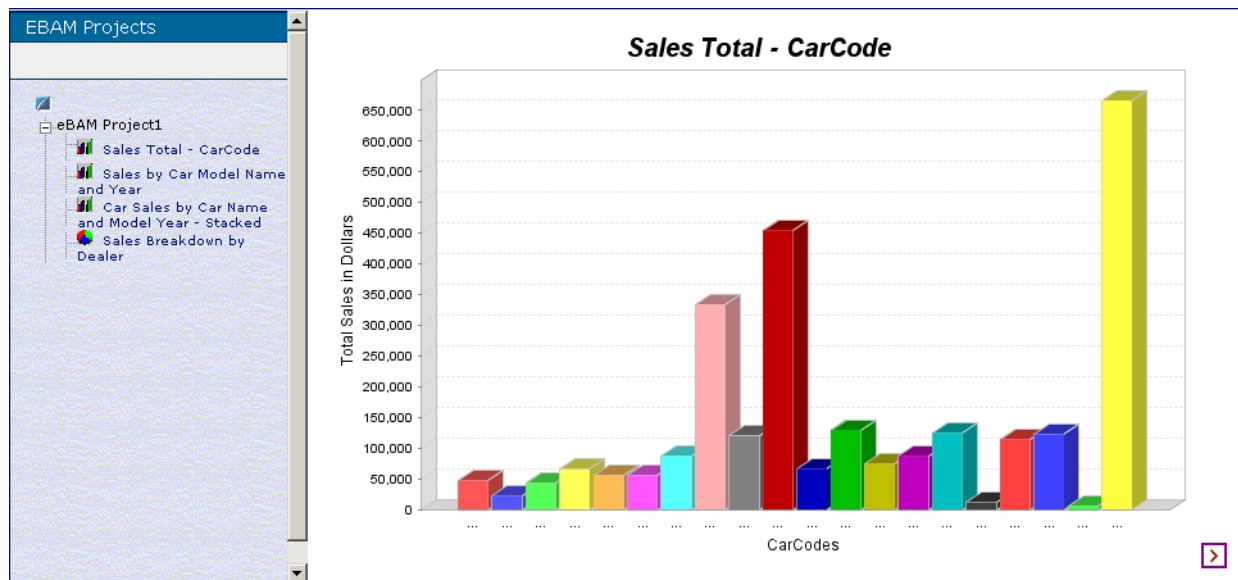
Figure 87 Pie Chart Showing Car Sales Breakdown by Dealer



Note: If you modified chart properties such as **Title**, **Include Legend**, and so forth, your charts may look somewhat different from these illustrations.

- 4 In the right pane, below the chart, click the right-pointing chevron [>] link to navigate to the next chart. For the result, see Figure 88.

Figure 88 Bar Chart Showing Sales Totals by Car Code



- 5 In the left pane, click the other two charts to see that they do not yet have data. (Remember that these two charts use queries with joined data definitions.)

- 6 Take a copy of **inputCarDefs7_original.~in** and rename it to **inputCarDefs001.txt**, and then watch as the file is picked up and renamed to **inputCarDefs001.txt.~in**; then repeat step 5 and see that the charts are now populated. Compare the stacked and unstacked visual representations of the same data.
- 7 See the effect of feeding in additional data from a modified input file. For example, edit a copy of **inputReports001.txt.~in** and delete the four lines for “Bob Rice Ford.” Save the modified file as **inputReports_noBobRice.txt** and, after the file is read, see how the charts indicate the decreased sales and market share for Ford. After this, try the same operation again with different numbers.

8.5.2 Sending Alerts and Notifications

In this section, you will trigger a business process that writes an alert to a file. This exercises the **notify** logic of the **bpAlertBelowMinimum** process.

To trigger a notification condition

- 1 Browse (or open a command prompt and change directories) to the location where you installed the sample output data. For example:

```
dir C:\temp\eBAM\Sample\Data\Out
```
- 2 Copy the file **outputAlert_comparison.dat** to ... \Sample\Data\In\, rename it to **inputReports_WayShort.txt**, and then watch as the file is picked up and renamed to **inputReports_WayShort.txt.~in**.

This file contains data reporting unrealistically low car sales—so low as to fall below the \$1000 threshold set in “[Creating eBAM Sample Alerts](#)” on page 97, and thus triggering an alert notification.

Result 1: The display in the Chart Viewer changes to reflect the updated data.

Result 2: A new file, named **outputAlertBelowMinimum1.dat**, eventually appears in the output directory, with contents that echo the current total sales. For as long as the KPI remains below the threshold, alert notifications are re-issued at regular intervals.

8.5.3 Using a Dynamic Query

In this section, you will provide a keyword (in this case, a car model name) that is used to filter the data.

To execute a dynamic query

- 1 Browse (or open a command prompt and change directories) to the location where you installed the sample input data. For example:

```
dir C:\temp\eBAM\Sample\Data\In
```
- 2 Take a copy of **inputQryName_original.~in** (contains the single word “Camry”) and rename it to **inputQryName_Camry.txt**, and then watch as the file is picked up and renamed to **inputQryName_Camry.txt.~in**.

- 3 In the output directory, verify that **outputQryNameResults1.dat** is written out, and that its contents are restricted to data on sales of Toyota Camrys.
- 4 Open **inputQryName_Camry.txt.~in**, change its contents to “Explorer,” and save it as **inputQryName_Explorer.txt**. Then watch as the file is picked up and renamed to **inputQryName_Explorer.txt.~in**.
- 5 In the output directory, verify that **outputQryNameResults1.dat** is written out, and that its contents are restricted to data on sales of Ford Explorers.

8.5.4 Deleting Existing Data

In this section, you will delete all data previously written. This exercises the **delete** logic of the bpDelTables process.

To update existing data using input that contains full data

- 1 Browse (or open a command prompt and change directories) to the location where you installed the sample input data. For example:

```
dir C:\temp\eBAM\Sample\Data\In
```
- 2 Take a copy of **TriggerDel_original.~in** and rename it to **TriggerDelete.txt**, and then watch as the file is picked up and renamed to **TriggerDelete.txt.~in**
- 3 In the Chart Viewer, verify that all charts have lost data and gone dim.

Using eBAM Charts in Portals

Projects can be set up so that a Sun Java System Portal Server Version 6 (2005Q4) or Version 7 displays your eBAM charts. The material and instructions discussed here assume that you have defined at least one environment that includes a Sun Java Application Server.

What's in This Chapter

- [“About Portals and Portlets” on page 127](#)
- [“About Java CAPS Application Files” on page 127](#)
- [“About Sun Java Portal Server 6 and 7” on page 128](#)
- [“Using Sun Java System Portal Server” on page 129](#)

9.1 About Portals and Portlets

A *portal* is a web site that serves as a gateway for web-based services and applications.

A *portlet* is a Java-based web component that runs inside a portal.

Java Specification Request (JSR) 168 is a standard that ensures interoperability between portals and portlets. You can generate a JSR 168-compliant portlet application in eBAM Studio, and then expose the portlet in Sun Java System Portal Server.

An *inline frame* (iFrame) web application allows portlet simulation by supplying a simple URL; the portal server creates a virtual browser window for displaying the application, but there is no other intercommunication between portal and application.

9.2 About Java CAPS Application Files

When you use Enterprise Designer (Deployment Editor, Build button) or a command-line utility to build a Java CAPS application, an **.ear** file named `<projName><dpName>.ear` is generated under `<JavaCAPS513>\edesigner\builds\`. For example, if your project is named **prj2**, deployment profile **dp8**, logical host **lh3**, and Application Server **as81**, it would be `<JavaCAPS513>\edesigner\builds\prj2dp8\lh3\as81\prj2dp8.ear`. This **.ear** file

includes a **.war** file named `<projName><dpInstanceName>.war` (in the example cited, this would be **prj2dp8.war**).

This **.ear** file, or the corresponding **.war** file that it contains, is called an *application file*. Once it is built, it can be *deployed* to an application server in several different ways.

9.3 About Sun Java Portal Server 6 and 7

In Sun Java Portal Server 6, Java CAPS applications must be deployed as simple URL-based inline frames (iFrames). Portal Server 7 allows Java CAPS portlet applications to be deployed either as simple URL-based iFrames or as JSR 168 portlets; however, using the JSR 168 option requires extra steps.

Table 20 explains which procedures are required for which type of deployment.

Table 20 Procedures for Generating and Deploying Applications to Sun Java Portal Server

To display eBAM charts on this platform...	... follow these procedures
In a web browser	No special steps are required: Build and deploy your Java CAPS project so as to generate an application for any supported integration server or application server, and then open a web browser and point it at the appropriate URL for the deployed eBAM application. For example: <ul style="list-style-type: none"> ▪ <code>http://localhost:18001/ProjnameAppinstancenameDpname/ebam</code>
In an iFrame on Portal Server 6	<ol style="list-style-type: none"> 1 Build and deploy your Java CAPS project so as to generate an application for Sun Java System Application Server. 2 Refer to “Running eBAM Studio Applications with Portal Server 6 2005Q4” on page 132.
In an iFrame on Portal Server 7	<ol style="list-style-type: none"> 1 Build and deploy your Java CAPS project so as to generate an application for Sun Java System Application Server. 2 Follow the steps in “Setting Up an IFrame Channel” on page 156.
In a JSR 168 portlet on Portal Server 7	<ol style="list-style-type: none"> 1 Be sure you meet the requirements described in “Running eBAM Studio Applications with Portal Server 7” on page 155. In particular, be sure you have reviewed the Release Notes for Portal Server 7 and that you have installed any patches that it indicates are required. 2 Build and deploy your Java CAPS project so as to generate an application for Sun Java System Application Server. 3 Follow the steps in “Setting Up a JSR 168 Portlet Channel” on page 169.

9.4 Using Sun Java System Portal Server

Sun Java™ System Portal Server delivers capabilities and components necessary for advanced portal solutions. Using Portal Server, you are able to access and find relevant applications and information and personalize your portal environments to best meet your needs.

Portal Server Benefits

Portal Server's major benefits include:

- Aggregation
- Personalization
- Security, including single sign-on (SSO)
- Collaboration and Communities
- Secure remote access
- Mobile access
- Search
- Content management and delivery
- You can refer to Sun's Portal Server Web site for information on additional available features at the following URL:

http://www.sun.com/software/products/portal_srvr/index.xml

See also **Reference Documentation** on page 130.

These benefits and their related features securely deliver the Portal Server capabilities that you require. The overall result is that you can work more efficiently and flexibly, doing a variety of business-related activities, for example, allowing:

- Employees to collaborate
- Sales personnel to access and update customer information on the road
- Partners to get advance access to the latest product designs

Portal Server and the Java CAPS

You can integrate Portal Server with the Java CAPS, which supports the following Portal Server releases:

- Version 6 2005Q4
- Version 7
- Version 7.1

Portal Server is part of the Sun Java Enterprise System (Java ES). Use Portal Server Version 6 2005Q4 when deploying on a Windows platform. Versions 7 and 7.1 are used for all other platforms.

The rest of this section explains operations you need to perform to allow Portal Server to work seamlessly with eBAM Studio applications.

9.4.1 Reference Documentation

This section provides general information on Portal Server features and procedures relevant to the Java CAPS. For more information on Portal Server, Java ES, and Sun Java System Application Server, you can refer to the appropriate Sun Microsystems documentation.

You can use the Documentation Web site's Search feature, located in the upper right corner of the page, to find the appropriate documentation. This feature allows you to search by a variety of methods, for example, document title, key word, or product name.

Sun Documentation Web Page for Portal Server 6 2005Q4

To access Portal Server 6 2005Q4 and related documentation, see the following Sun Web page:

<http://docs.sun.com/app/docs/coll/1293.1>

Scroll down the page to find the product documentation.

You can also use this Web page as a convenient list of the Portal Server-related applications and systems. The following information helps you navigate this Web page:

- Because Portal Server operates with Java ES, its documentation is integrated with the suite of user's guides for Java ES.
- This documentation set includes installation instructions for overall Java ES, including all its component products, such as Sun Java System Application Server, and Portal Server.

Sun Documentation Web Page for Portal Server 7

To access Portal Server 7 and related documentation, see the following Sun Web page:

<http://docs.sun.com/coll/1303.1>

You can also use this Web page as a convenient list of the Portal Server-related applications and systems. The following information helps you navigate this Web page:

- This documentation set includes installation instructions for overall Java ES, including all its component products, such as Sun Java System Application Server, and Portal Server.
- This documentation set includes a technical note on integrating Portal Server with Java CAPS. See [Necessary Patches](#) on page 131 for more information.

Sun Documentation Web Page for Portal Server 7.1

To access Portal Server 7.1 and related documentation, see the following Sun Web page:

<http://docs.sun.com/app/docs/coll/1483>

You can also use this Web page as a convenient list of the Portal Server-related applications and systems. The documentation set includes installation instructions for overall Java ES, including all its component products, such as Sun Java System Application Server, and Portal Server.

9.4.2 Before Installing Portal Server

Before you install Portal Server, ensure that:

- eGate Integrator and eBAM Studio (and any related applications, as needed) are installed and operational.
- You have installed the necessary patches for Portal Server and Java CAPS.
- You have created or imported at least one Project for eBAM Studio using Enterprise Designer; not necessary but recommended.
- You have read the eBAM Studio **Readme.txt** file for information on pre-installation requirements, for example, patches or Portal Server prerequisites.

For complete instructions on how to create and/or import a Project, using Enterprise Designer, see the *eGate Integrator User's Guide*.

Necessary Patches

You must install patches for your system, to ensure the proper operation of Portal Server. These patches are necessary to:

- Allow Portal Server to operate correctly with Java CAPS.
- Allow Java CAPS to operate correctly with the Sun Java System Application Server version 8.2.

The patch you need depends on your operating system and the version of Portal Server being used. You can access the patches directly from SunSolve Online. The URL for SunSolve is:

<http://www.sunsolve.sun.com>

Because patches are frequently revised or superseded, do not install any patch until you have checked that it is the latest available.

You can find specific information on locating and accessing these patches as follows:

- **Portal Server:** See the Portal Server technical note "Integrating Java CAPS With Sun Java System Portal Server." This technical note provides the patch numbers you need to search for these patches on SunSolve. You can find this technical note at the following URL:

<http://docs.sun.com/source/819-6117/index.html>

- **Sun Java System Application Server:** See the main Java CAPS **Readme.txt** file, which provides the patch numbers you need to search for these patches on SunSolve.

Portal Server Version Compatibility

Regardless of whether you install either Portal Server 6 2005Q4, Portal Server 7, or Portal Server 7.1, all other related systems and applications are compatible with these Portal Server versions (see the list on the Web page cited under [Sun Documentation Web Page for Portal Server 6 2005Q4](#) on page 130).

Release Notes and Additional Information

Before you begin installing Portal Server for use with Java CAPS, be sure to read the release notes that accompany the documentation. For general information on how to obtain Portal Server documentation, see **Reference Documentation** on page 130.

It is especially recommended that you read the following documents:

- “Integrating Java CAPS With Sun Java System Portal Server” (see the URL for the Portal Server technical note referenced under **Sun Documentation Web Page for Portal Server 7** on page 130).
- “Sun Java System Portal Server Release Notes for Microsoft Windows, Version 6 2005Q4” at the following URL:

<http://docs.sun.com/source/819-4270/index.html>

- *Sun Java System Portal Server 7 Installation Guide*, “Chapter 1: Pre Installation Requirements for the Sun Java System Portal Server 7 Software” at the following URL:

<http://docs.sun.com/app/docs/doc/819-3027>

- “Sun Java System Portal Server Release Notes for Microsoft Windows, Version 7.1” at the following URL:

<http://docs.sun.com/app/docs/doc/819-4986>

9.4.3 Running eBAM Studio Applications with Portal Server 6 2005Q4

This section explains how to set up Sun Java System Portal Server 6 2005Q4 to run with the Java CAPS applications, including eBAM Studio.

Before You Start

Take care to ensure you have completed the prerequisite actions listed under **Before Installing Portal Server** on page 131.

System Requirements

You must run Portal Server 6 2005Q4 in:

- Sun Java System Application Server, version 8.2

If you want to utilize a Windows XP operating system, use this version of Java ES and Portal Server and display your Web interfaces (frames) on eBAM Studio using an IFrame channel.

IFrame channels can reference any URL accessible to your Sun Java System Application Server installation. As a result, eBAM Studio Projects displayed by IFrames do not have to be deployed on the same server as Portal Server.

For more information on supported operating systems, see the **Readme.txt** file that accompanies eBAM Studio.

Supported Operating Systems

For specific information on all operating systems supported by Portal Server 6 2005Q4 in conjunction with the Java CAPS and eBAM Studio, see the **Readme.txt** file for eBAM Studio.

Portal Server 6 2005Q4 on Windows: Procedural Overview

This section explains procedures you need to know, to successfully set up eBAM Studio applications to display on Portal Server 6 2005Q4, utilizing IFrames on Windows.

The following list provides an overview of these procedures:

- Configuring the Sun Java System Application Server and Java ES so that Portal Server deploys on the correct application server.
- Deploying eBAM Studio applications to the Sun Java System Application Server.
- Creating users in the Sun Java System Application Server using Access Manager.

Note: For more information on Access Manager, see the following URL:
<http://docs.sun.com/app/docs/coll/1292.1>

- Creating new IFrame channels using Access Manager.
- Displaying the Portal Desktop.

Important

The port numbers provided in all access URL examples are the system defaults. The actual port numbers used by your system's configuration may be different. Before you begin using the procedures provided in the rest of this section, make sure you have verified the correct port number for any URL referenced by a given procedure.

Installing Portal Server 6 2005Q4 to Run in Sun Java System Application Server on Windows

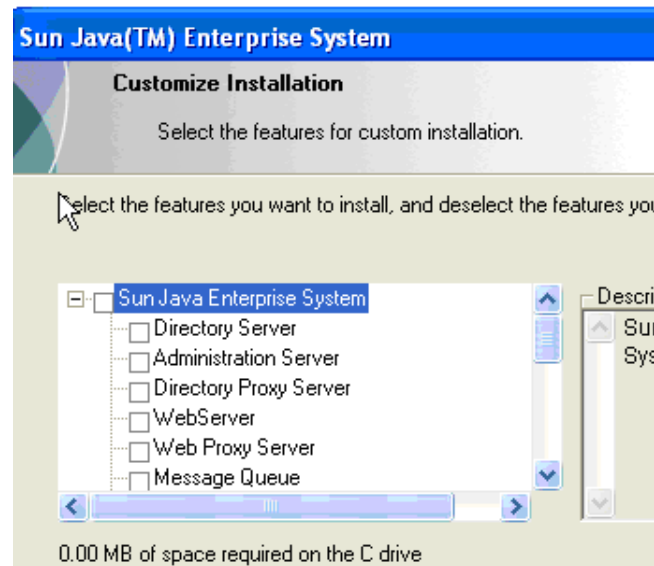
Portal Server 6 2005Q4 uses the Java ES installer and, by default, is configured to run in the Java ES Web Server. You must change this default configuration.

To install

- 1 Start the Java ES installer.
- 2 Choose **Configure Automatically**.
- 3 Choose the **Custom Installation** option.

The Customize Installation user interface appears. See Figure 89.

Figure 89 Customize Installation User Interface



- 4 Deselect **Java ES**. See Figure 89.
- 5 Select **Sun Java System Application Server**, **Directory Server**, **Access Manager**, and **Portal Server** from the list shown in Figure 89.

Automatically, several other items are selected, including the Java ES Web server, which you do not want installed.
- 6 Expand **Sun Java System Application Server**; all the boxes are checked by default.
- 7 Deselect **Load Balancer Plugin**.
- 8 Make sure that **Web Server** is not selected.
- 9 Click **Next** to continue.
- 10 Finish the installation procedure. You are prompted appropriately as other components are needed.

Important

During the installation, you are asked to create and enter an administrator user name and password. Make sure you keep a record of these exact entries.

Starting Sun Java System Application Server on Windows

This section explains how to start Sun Java System Application Server and the Node Agent on Windows.

To Start Sun Java System Application Server and the Node Agent

- 1 On the Windows **Start > All Programs** menu, choose **Sun Microsystems > Application Server > Start Administration Server**.

A command line window displays, and you are prompted to enter your password.

Whenever you are prompted for your administrator user name and/or password, be sure to use the ones you created during your initial Java ES Portal Server installation.

- 2 After you enter the password, click **Enter**.

There is no confirmation the password is accepted. Several minutes may pass before the server finishes starting.

- 3 On the Windows **Start > All Programs** menu, choose **Sun Microsystems > Application Server > Start Node Agent**.

A command line window displays, and you are prompted to enter your password.

- 4 After you enter the password, click **Enter**.

Again, there is no confirmation the password is accepted. Several minutes may pass before the node agent finishes starting.

Deploying eBAM Studio Applications

This section explains how to deploy eBAM Studio applications on Sun Java System Application Server.

To deploy eBAM Studio applications

- 1 Make sure the Sun Java System Application Server is running.
- 2 Java CAPS applications require special permissions. Make sure the appropriate permissions have been set up correctly in the **server.policy** file. For more information on permissions in Java CAPS, see the *eGate Integrator System Administration User's Guide*.
- 3 Deploy the eBAM Studio **.ear** file to the application server on which Portal Server is running. You can do this operation using:
 - ♦ Enterprise Designer
 - ♦ Enterprise Manager
 - ♦ A Sun Java System Application Server's administrator interface, either the Admin Console or the command line

A command-line example follows:

```
./asadmin deploy --user admin /tmp/prj2dp8.ear
```

See the *eGate Integrator User's Guide* and *eGate Integrator System Administration Guide* for details on how to deploy an **.ear** file using Enterprise Designer and Enterprise Manager. See the [procedure on page 136](#) for information on how to deploy an **.ear** file using Sun Java System Admin Console.

To access the Application Server Admin Console

- 1 On the Windows **Start > All Programs** menu, choose **Sun Microsystems > Application Server > Admin Console**.

You can also access this page by entering either of the following URLs, using your Web browser:

- ♦ On any machine:

`https://<host name>:4850/`

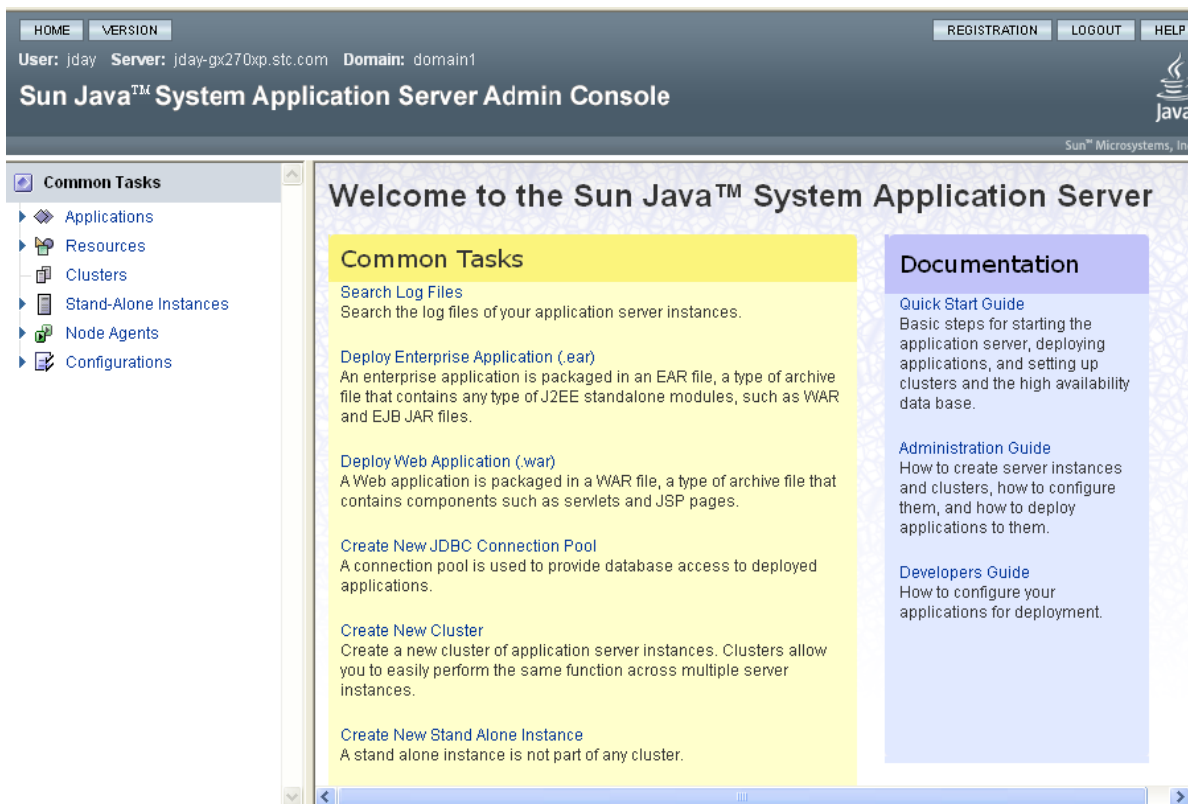
- ♦ On your own machine:

`https://localhost:4850/`

- 2 In the resulting dialog box, enter your administrator user name and password, and click **Enter**.

The Admin Console page displays. See Figure 90.

Figure 90 Admin Console Page

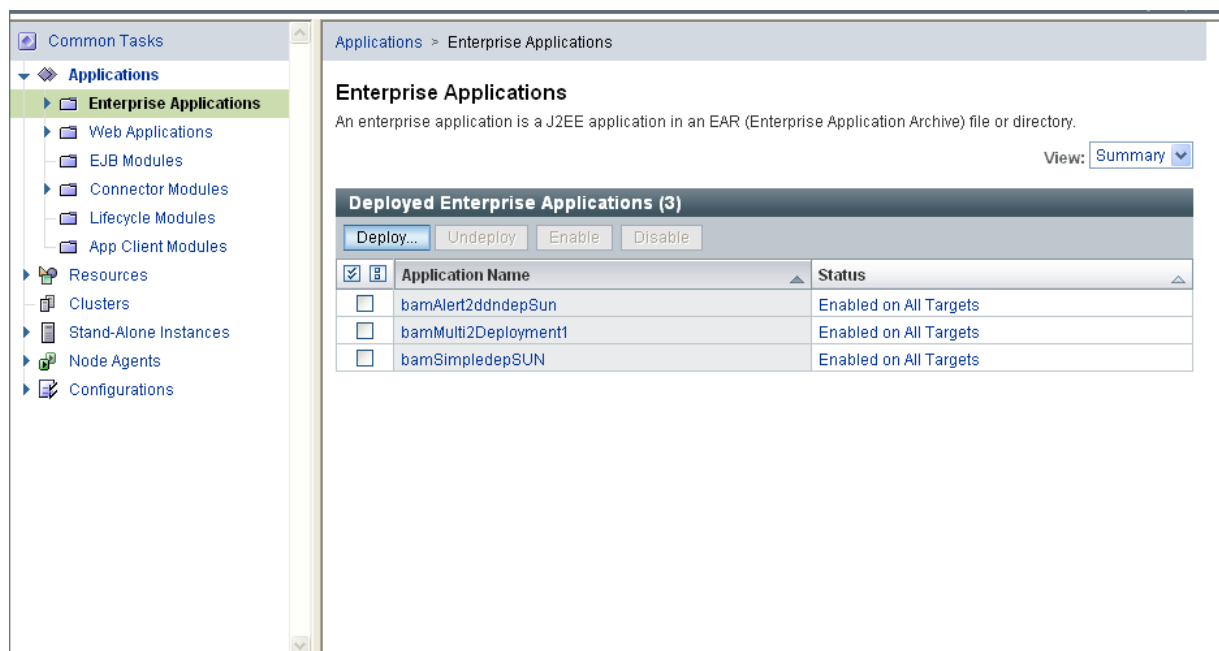


To deploy eBAM Studio applications using Admin Console

- 1 On the **Common Tasks** frame (on the left), select **Applications** in the tree, to show the available operations under this option.
- 2 Select **Enterprise Applications** under **Applications**.

The **Enterprise Applications** frame appears on the right. See Figure 91.

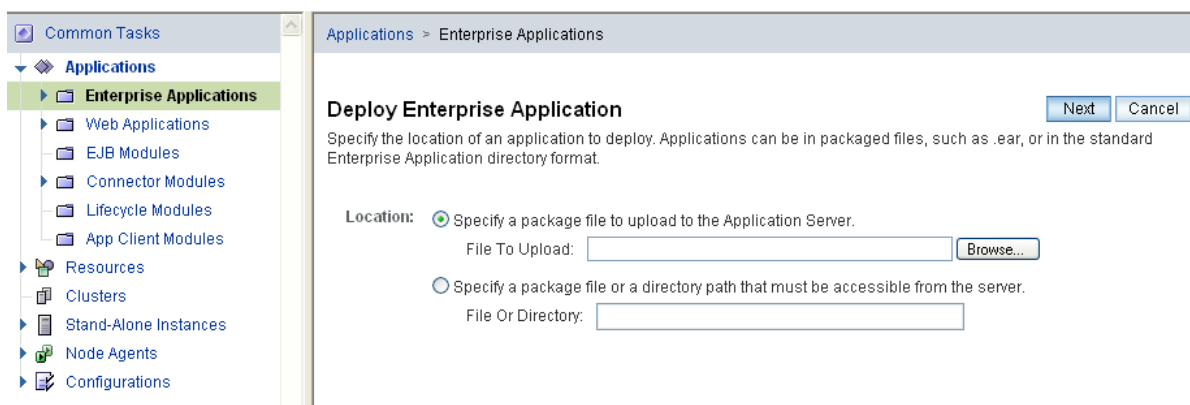
Figure 91 Admin Console Page With Enterprise Applications Frame



3 Click **Deploy**.

The **Deploy Enterprise Application** frame appears. See Figure 92.

Figure 92 Deploy Enterprise Application Frame: Initial

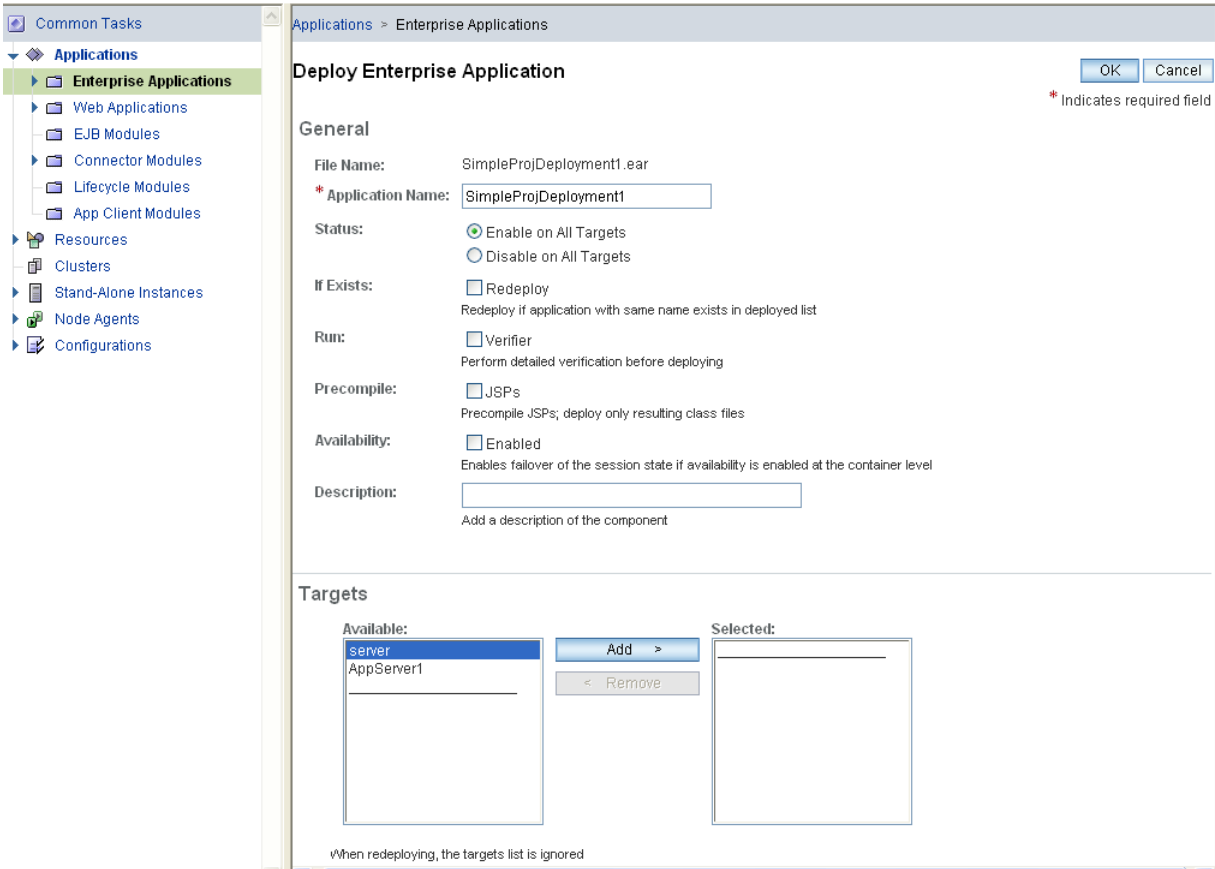


4 Browse to the **.ear** file you generated using Enterprise Designer.

5 Select the file and click **Next**.

The **Deploy Enterprise Application** frame refreshes to display another page, as shown in Figure 93.

Figure 93 Deploy Enterprise Application Frame: After .ear File Selection



- 6 Scroll to the bottom of this frame, to **Targets**, if this section is not already displayed.
- 7 Under **Available**, click **server** then **Add** to move your selected file to the **Selected** column on the right.

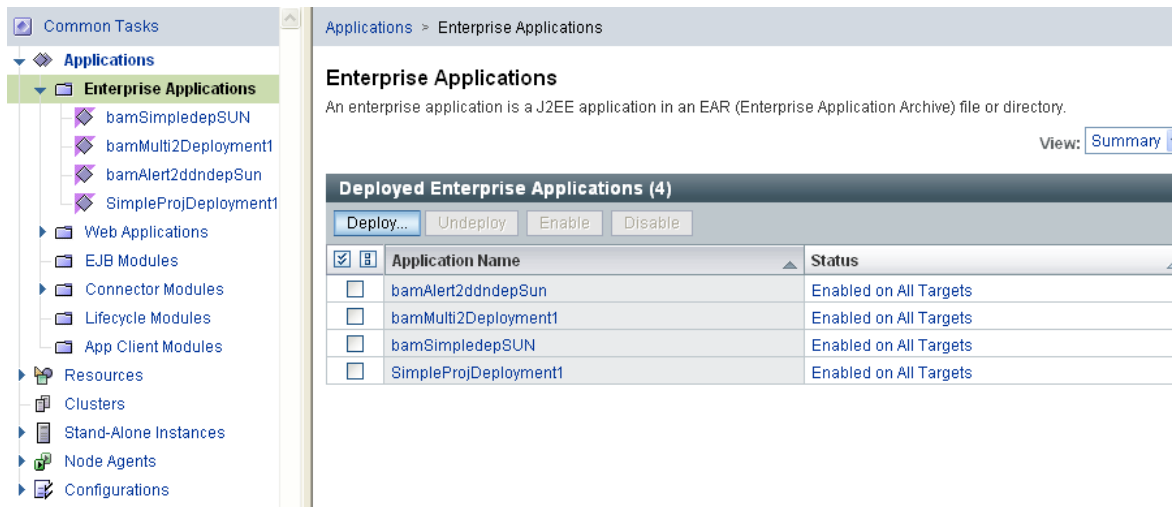
Note: *If you have deployed this application before, select the **Redeploy** check box.*

- 8 Click **OK**.

You have now finished deploying the current application. The **Deploy Enterprise Application** frame refreshes to display the initial frame, **Enterprise Applications**.

The name of your application appears in the list of deployed applications on this frame. See Figure 94.

Figure 94 New Enterprise Applications List



Creating New Portal Server Users

Use Sun Java System Access Manager to create a new Portal Server user.

To view the Access Manager Console page

- 1 Perform either of the following operations:
 - ♦ On the Windows **Start > All Programs** menu, choose **Sun Microsystems > Access Manager > Administration Console**.
 - ♦ Enter the following URL on your Web browser:
`http://<host name>:38080/amconsole`
- 2 Use the default user name **amadmin**, and enter your administrator password.
- 3 Click **Enter**.

The Access Manager Console page appears. See Figure 95.

Figure 95 Access Manager Console Page

The screenshot displays the Sun Java System Access Manager console interface. At the top, there is a header bar with the Sun logo, the text "Sun Java™ System Access Manager", and navigation links for "Search", "Logout", and "Help". Below the header, a tabbed interface shows "Identity Management" as the active tab, with other tabs for "Service Configuration", "Current Sessions", and "Federation Management". The main content area is divided into two panes. The left pane, titled "stc", shows a "View: Organizations" dropdown and a list of organizations (currently empty, showing "There are no entries."). The right pane, also titled "stc", shows a "View: General" dropdown and "General Properties" for a selected organization. These properties include "Domain Name", "Organization Status" (set to "Active"), and "Organization Aliases" (containing "jday-gx270xp.stc.com" and "stc"). There are "Add" and "Remove" buttons for the aliases. At the bottom of the right pane, there is a section for "DNS Alias Names" with another "Add" and "Remove" button. The browser's status bar at the bottom shows "Done" and "Internet".

To create a new user

- 1 On the Access Manager Console page, make sure the **Identity Management** tab content is displayed.

If there are multiple organizations displayed, choose the organization that has the appropriate permissions for you to use the Portal Desktop.

- 2 From the **View** drop-down list choose **Users**.

A frame appears on the right, allowing you to edit user properties, as well as create new users. See Figure 96.

Figure 96 Access Manager Console With User General Properties

The screenshot displays the Sun Java System Access Manager console. At the top, there is a header bar with the Sun logo, the text 'Sun Java System Access Manager', and links for 'Search', 'Logout', and 'Help'. Below this is a navigation bar with tabs: 'Identity Management' (selected), 'Service Configuration', 'Current Sessions', and 'Federation Management'. The main content area is divided into two panes. The left pane, titled 'stc', shows a 'View: Users' dropdown and a section for 'Organizations (0 items)' with 'New...', 'Delete', and 'Search' buttons. The right pane, also titled 'stc', shows the 'General Properties' tab. It includes a 'View: General' dropdown, 'Save' and 'Reset' buttons, and fields for 'Domain Name', 'Organization Status' (set to 'Active'), 'Organization Aliases' (containing 'jday-gx270xp.stc.com' and 'stc'), and 'DNS Alias Names'. Each of these fields has an 'Add' and 'Remove' button. The bottom of the screen shows a Windows taskbar with an 'Internet' icon.

3 Click **New**.

A frame appears, allowing you to select the appropriate services for the new user. The right frame also displays a list of users. See Figure 97.

Figure 97 Available Services for New User

Identity Management | Service Configuration | Current Sessions | Federation Management

stc ▾

View: Users ▾

Users (5 items)

New... Delete

User ID	Full Name
<input type="checkbox"/> amAdmin	amAdmin ▾
<input type="checkbox"/> anonymous	anonymous ▾
<input type="checkbox"/> authlessanonymous	authlessanon ▾
<input type="checkbox"/> jKoenig	John Q. Koenig ▾
<input type="checkbox"/> jUser	Joe User ▾

[Advance](#)

New User - Step 1 of 2

Select the services to be assigned to the user.

Available Services

<input type="checkbox"/> Authentication Configuration
<input type="checkbox"/> Mobile Address Book
<input type="checkbox"/> Mobile Calendar
<input type="checkbox"/> Mobile Mail
<input type="checkbox"/> NetMail
<input type="checkbox"/> Portal Desktop
<input type="checkbox"/> SSO Adapter
<input type="checkbox"/> Subscriptions

Back Next Cancel

- 4 Select the services you want. Be sure to include **Portal Desktop**. For an example, see Figure 98.

Figure 98 Available Services for New User List: Selected

Identity Management | Service Configuration | Current Sessions | Federation Management

stc ▾

View: Users ▾

Users (5 items)

New... Delete

User ID	Full Name
<input type="checkbox"/> amAdmin	amAdmin ▾
<input type="checkbox"/> anonymous	anonymous ▾
<input type="checkbox"/> authlessanonymous	authlessanon ▾
<input type="checkbox"/> jKoenig	John Q. Koenig ▾
<input type="checkbox"/> jUser	Joe User ▾

[Advance](#)

New User - Step 1 of 2

Select the services to be assigned to the user.

Available Services

<input checked="" type="checkbox"/> Authentication Configuration
<input type="checkbox"/> Mobile Address Book
<input type="checkbox"/> Mobile Calendar
<input type="checkbox"/> Mobile Mail
<input type="checkbox"/> NetMail
<input checked="" type="checkbox"/> Portal Desktop
<input checked="" type="checkbox"/> SSO Adapter
<input checked="" type="checkbox"/> Subscriptions

Back Next Cancel

- 5 Click **Next**.

A frame appears, allowing you to enter new user information. See Figure 99.

Figure 99 Information for New User List

Identity Management | Service Configuration | Current Sessions | Federation Management

stc ▾

View: Users ▾

Users (5 items)

[Advanced Search](#)

User ID	Full Name
<input type="checkbox"/> amAdmin	amAdmin ▾
<input type="checkbox"/> anonymous	anonymous ▾
<input type="checkbox"/> authlessanonymous	authlessanonymous ▾
<input type="checkbox"/> jKoenig	John Q. Koenig ▾
<input type="checkbox"/> jUser	Joe User ▾

New User - Step 2 of 2

Enter Required User Attributes

* Indicates required field

User

* User ID:

First Name:

* Last Name:

* Full Name:

* Password:

* Password (confirm):

* User Status: ▾

- 6 Enter the necessary user information and click **Finish**.

You are finished creating the new user. The list of users refreshes and the Access Manager Console page now shows the new user.

Creating a New Channel

Use Access Manager Console to create a new channel.

To view the Access Manager Console page

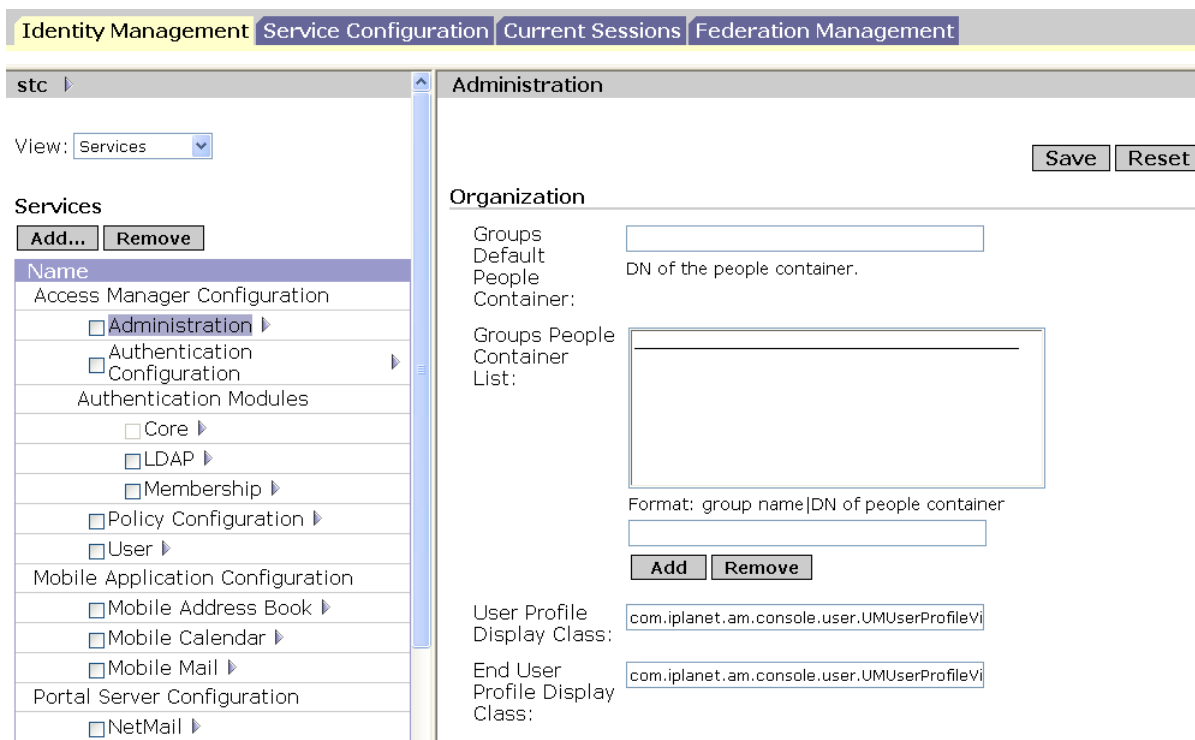
- Use the [procedure on page 139](#).

To create a new channel

- 1 On the Access Manager page, make sure the **Identity Management** tab content is displayed.
- 2 From the **View** drop-down list choose **Services**.

A frame appears on the right, allowing you to manage Portal Server services, under Identity Management. See Figure 100.

Figure 100 Initial Window: Portal Server Services (Identity Management)



- 3 Scroll down the **Services** (left) frame to display the **Portal Desktop** options and click the pointer icon to the right of **Portal Desktop**.

The **Portal Desktop** frame appears on the right. See Figure 101.

Figure 101 Access Manager Page With Portal Desktop

The screenshot shows the Access Manager configuration page. The top navigation bar includes 'Identity Management', 'Service Configuration', 'Current Sessions', and 'Federation Management'. The left sidebar, titled 'Services', contains a list of configuration categories with expandable sub-items. The main content area is titled 'Portal Desktop' and includes a 'Dynamic' section with the following fields:

- Conflict Resolution Level:
- Default Channel Name:
- Default Edit Channel Name:
- Portal Desktop Type:
- Display Profile:
- Show Portal Desktop Service Attributes: ☒

Buttons for 'Save', 'Reset', and 'Delete' are present at the top right and bottom right of the configuration area.

- On the **Portal Desktop** frame, click the **Manage Channels and Containers** link.

The **Portal Desktop: Channels** frame appears, which allows you to create and edit Portal Desktop channels, along with related features. See Figure 102.

Figure 102 Access Manager Page: Portal Desktop - Channels

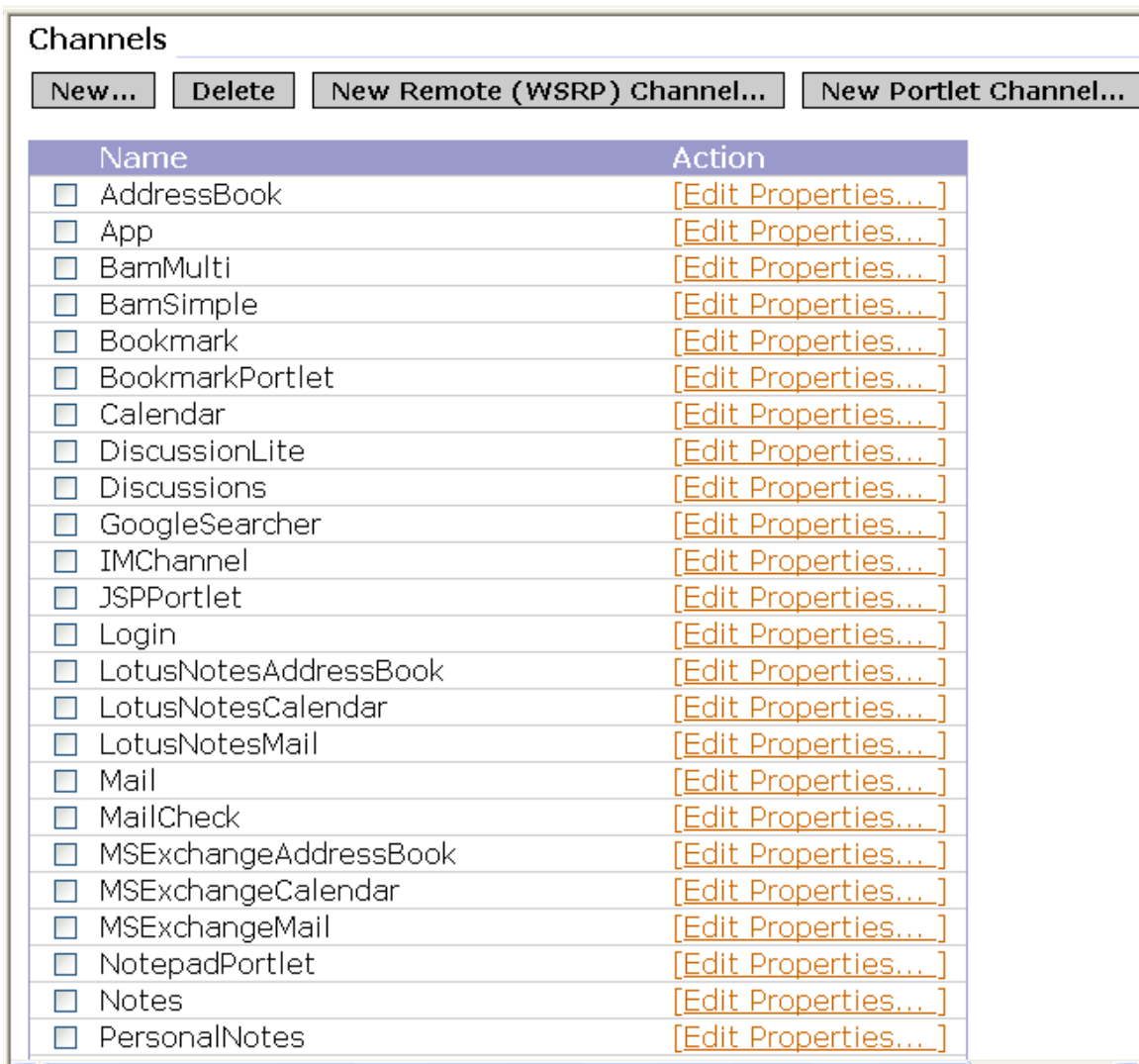
The screenshot shows the 'Portal Desktop - Channels' configuration page. The left sidebar is identical to Figure 101. The main content area is titled 'Portal Desktop - Channels' and includes the following elements:

- Container Path:
-
- Display Profile Top Level:
- Container Channels section with and buttons.
- A table listing container channels:

Name	Action
CollaborationTabPanelContainer	[Edit Properties...]
FrameCustomTableContainer	[Edit Properties...]
FrameTabContainer	[Edit Properties...]
JSPNativeContainer	[Edit Properties...]
JSPRenderingContainer	[Edit Properties...]
JSPTabContainer	[Edit Properties...]
JSPTabCustomTableContainer	[Edit Properties...]
JSPTableContainer	[Edit Properties...]
MyFrontPageFramePanelContainer	[Edit Properties...]

- 5 Scroll down the **Portal Desktop - Channels** frame to the **Channels** section, which displays a list of existing channels. See Figure 103.

Figure 103 List of Existing Channels



Name	Action
<input type="checkbox"/> AddressBook	[Edit Properties...]
<input type="checkbox"/> App	[Edit Properties...]
<input type="checkbox"/> BamMulti	[Edit Properties...]
<input type="checkbox"/> BamSimple	[Edit Properties...]
<input type="checkbox"/> Bookmark	[Edit Properties...]
<input type="checkbox"/> BookmarkPortlet	[Edit Properties...]
<input type="checkbox"/> Calendar	[Edit Properties...]
<input type="checkbox"/> DiscussionLite	[Edit Properties...]
<input type="checkbox"/> Discussions	[Edit Properties...]
<input type="checkbox"/> GoogleSearcher	[Edit Properties...]
<input type="checkbox"/> IMChannel	[Edit Properties...]
<input type="checkbox"/> JSPPortlet	[Edit Properties...]
<input type="checkbox"/> Login	[Edit Properties...]
<input type="checkbox"/> LotusNotesAddressBook	[Edit Properties...]
<input type="checkbox"/> LotusNotesCalendar	[Edit Properties...]
<input type="checkbox"/> LotusNotesMail	[Edit Properties...]
<input type="checkbox"/> Mail	[Edit Properties...]
<input type="checkbox"/> MailCheck	[Edit Properties...]
<input type="checkbox"/> MExchangeAddressBook	[Edit Properties...]
<input type="checkbox"/> MExchangeCalendar	[Edit Properties...]
<input type="checkbox"/> MExchangeMail	[Edit Properties...]
<input type="checkbox"/> NotepadPortlet	[Edit Properties...]
<input type="checkbox"/> Notes	[Edit Properties...]
<input type="checkbox"/> PersonalNotes	[Edit Properties...]

- 6 Click **New**.
The **New Channel** frame appears.

- 7 Enter the name of your new channel and choose **IFrameProvider** on the **Provider** pull-down menu. Enter the **Channel Name** of your new channel. In this example, the name **DefaultSearchEngine** is used. See Figure 104.

Figure 104 New Channel Frame With IFrame Provider

Portal Desktop - New Channel

Container Path: Top

Channel Name: *

Channel name may contain only letters (a-z,A-Z) and digits (0-9)

Provider:

* Indicates a required field

OK Cancel

- 8 Click **OK**.
Portal Desktop - Channels frame appears again. See [Figure 102 on page 145](#).
- 9 Scroll down this frame to the **Channels** section again. See [Figure 103 on page 146](#).
However, the section now contains the name of the new channel you just created.
- 10 Locate this channel's name. If you used the example shown in Figure 104, the name is **DefaultSearchEngine**.
- 11 Click the **Edit Properties** link to the right of **DefaultSearchEngine**, or the alternate channel name you entered, if applicable.

The **Edit Properties** frame for the channel appears, displaying the channel's default properties. See Figure 105.

Figure 105 Edit Properties Frame for New Channel

Portal Desktop - Edit Channel Properties
Container Path: [Top](#) > DefaultSearchEngine

Save
Reset

Edit Properties -- DefaultSearchEngine [Provider: IFrameProvider]

New...
Delete
Filter: Basic

Name	Value	Category	State
<input type="checkbox"/> client=HTML		basic	default
<input type="checkbox"/> contentPage	<input type="text" value="iframe.jsp"/>	basic	default
<input type="checkbox"/> description	<input type="text" value="*** This Provider uses"/>	basic	default
<input type="checkbox"/> fBorder	<input type="text" value="0"/>	basic	default
<input type="checkbox"/> fHeight	<input type="text" value="400"/>	basic	default
<input type="checkbox"/> fontFace1	<input type="text" value="Sans-serif"/>	basic	default
<input type="checkbox"/> fWidth	<input type="text" value="100%"/>	basic	default
<input type="checkbox"/> productName	<input type="text" value="Sun Java™ System Pc"/>	basic	default
<input type="checkbox"/> scrolling	<input type="text" value="yes"/>	basic	default
<input type="checkbox"/> showExceptions	<input type="checkbox"/>	basic	default
<input type="checkbox"/> srcURL	<input type="text" value="http://www.sun.com"/>	basic	default
<input type="checkbox"/> title	<input type="text" value="*** IFrame Provider **"/>	basic	default
<input type="checkbox"/> width	<input type="text" value="thick"/>	basic	default

- 12 Edit the properties, as necessary. Make sure you enter the appropriate access URL for your new channel, using the **srcURL** text box.

It is recommended that you enter a descriptive **title** property name of your own choosing, to replace the default. This name appears in the new channel's frame title bar.

- 13 Click **Save**.

Your new channel is now ready to be placed on the Portal Desktop. See [Placing a New Channel Into a Portal Desktop Tab](#) on page 149 for an explanation of how to associate a channel with a tap on the Portal Desktop.

Placing a New Channel Into a Portal Desktop Tab

Use Access Manager Console to place a new channel into a tab on the Portal Desktop.

To view the Access Manager Console page

- Use the [procedure on page 139](#).

To place a channel on the Portal Desktop

- 1 On the Access Manager page, make sure the **Identity Management** tab content is displayed.
- 2 From the **View** drop-down list select **Services**. See [Figure 100 on page 144](#)
- 3 Scroll down the **Services** (left) frame to display the **Portal Desktop** options and click the pointer icon to the right of **Portal Desktop**.

The **Portal Desktop** frame appears on the right. See [Figure 101 on page 145](#).

- 4 On the **Portal Desktop** frame, click the **Manage Channels and Containers** link.

The **Portal Desktop - Channels** frame appears. See [Figure 102 on page 145](#).

- 5 Scroll down the **Portal Desktop - Channels** frame to the **Container Channels** section (if it is not already displayed).

This section displays a list of existing Portal Desktop tab container channels. See Figure 106.

Figure 106 List of Container Channels

The screenshot shows a web application window titled "Portal Desktop - Channels". Below the title bar, it says "Container Path: Top". On the right side, there is a button labeled "Back To Portal Desktop". Below this, there is a label "Display Profile Top Level:" followed by a button labeled "Edit Properties...". The main section is titled "Container Channels" and contains two buttons: "New..." and "Delete". Below these buttons is a table with two columns: "Name" and "Action". The table lists ten container channels, each with a checkbox in the "Name" column and a link labeled "[Edit Properties...]" in the "Action" column.

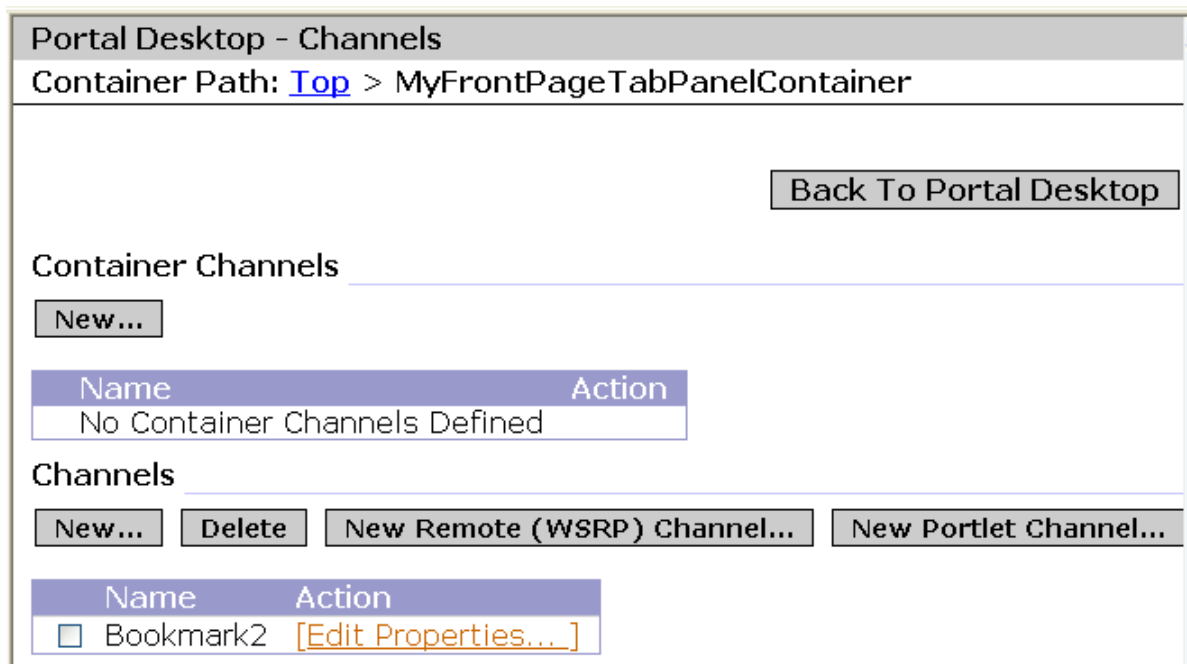
Name	Action
<input type="checkbox"/> CollaborationTabPanelContainer	[Edit Properties...]
<input type="checkbox"/> FrameCustomTableContainer	[Edit Properties...]
<input type="checkbox"/> FrameTabContainer	[Edit Properties...]
<input type="checkbox"/> JSPNativeContainer	[Edit Properties...]
<input type="checkbox"/> JSPRenderingContainer	[Edit Properties...]
<input type="checkbox"/> JSPTabContainer	[Edit Properties...]
<input type="checkbox"/> JSPTabCustomTableContainer	[Edit Properties...]
<input type="checkbox"/> JSPTableContainer	[Edit Properties...]
<input type="checkbox"/> MyFrontPageFramePanelContainer	[Edit Properties...]
<input type="checkbox"/> MyFrontPageTabPanelContainer	[Edit Properties...]
<input type="checkbox"/> MyFrontPageTemplatePanelContainer	[Edit Properties...]

6 Click **MyFrontPageTabPanelContainer**.

Note: Do *not* click the *Edit Properties* link.

- 7 A frame appears, which allows you to manage the channels that appear on your **Front Page** tab. See Figure 107.

Figure 107 Initial MyFrontPageTabPanelContainer Frame



- 8 Scroll down this frame to the **Channel Management** section. See Figure 108.

Figure 108 Channel Management Section: Initial

Channel Management

Save **Reset**

Ready For Use:

- Multi
- BookmarkPortlet
- CollaborationTabPanelContainer
- DefaultSearchEngine
- DiscussionLite
- Discussions

Add **Remove**

Available to End Users on the Content Page:

Add **Remove**

Visible on the Portal Desktop:

- UserInfo
- MailCheck
- App
- Bookmark
- MyFrontPageTabPanelContainer/Bookmark2
- SampleJSP

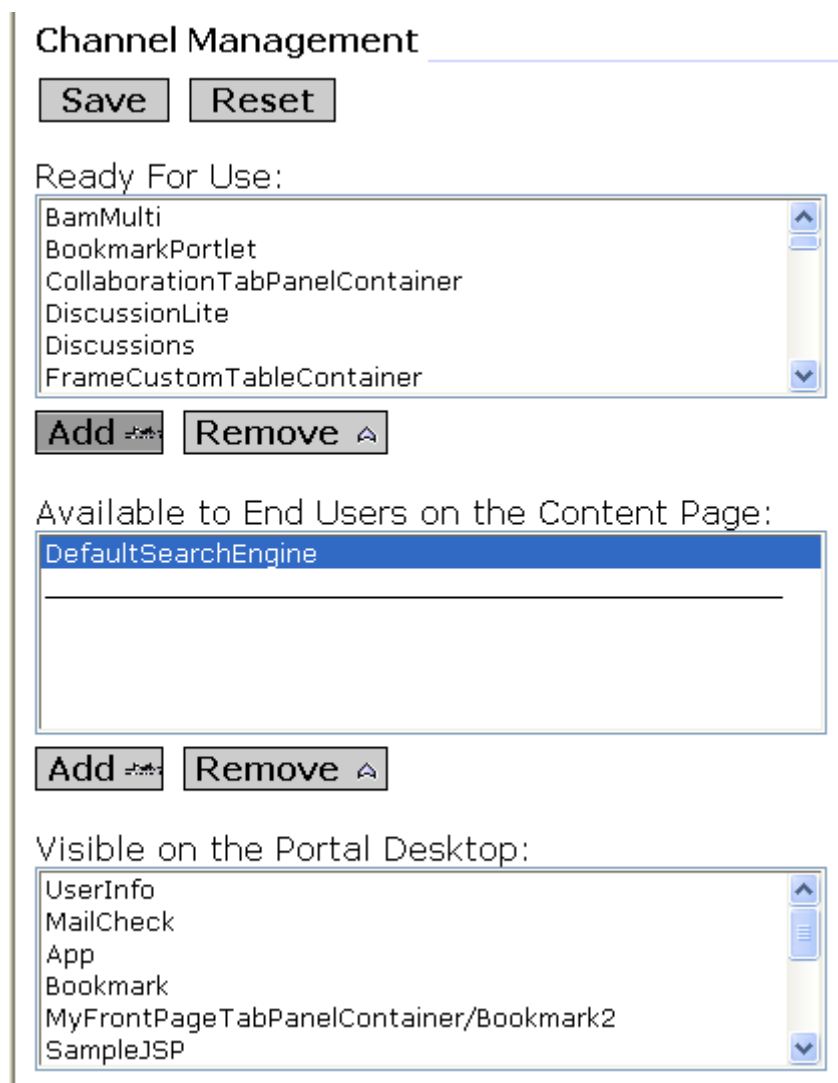
The **Ready for Use** pane contains a list of existing channels.

The section shown in the figure is an example. Your **Channel Management** section contains the Portal Server defaults plus any additional channels that have been created for your system.

- 9 Under **Ready for Use**, select the new channel you created under the [procedure on page 143](#). In the example, the new channel is **DefaultSearchEngine**.
- 10 Click **Add**.

Your new channel name appears under **Available to End Users on the Content Page**. See Figure 109.

Figure 109 Channel Management Section: First Add



The screenshot shows the 'Channel Management' section of a web application. It has a title bar 'Channel Management' with a close button. Below the title are 'Save' and 'Reset' buttons. The first section is 'Ready For Use:', which contains a list box with the following items: 'BamMulti', 'BookmarkPortlet', 'CollaborationTabPanelContainer', 'DiscussionLite', 'Discussions', and 'FrameCustomTableContainer'. Below this list box are 'Add' and 'Remove' buttons. The second section is 'Available to End Users on the Content Page:', which contains a list box with 'DefaultSearchEngine' selected. Below this list box are 'Add' and 'Remove' buttons. The third section is 'Visible on the Portal Desktop:', which contains a list box with the following items: 'UserInfo', 'MailCheck', 'App', 'Bookmark', 'MyFrontPageTabPanelContainer/Bookmark2', and 'SampleJSP'. Below this list box are 'Add' and 'Remove' buttons.

Channel Management

Save Reset

Ready For Use:

BamMulti
BookmarkPortlet
CollaborationTabPanelContainer
DiscussionLite
Discussions
FrameCustomTableContainer

Add Remove

Available to End Users on the Content Page:

DefaultSearchEngine

Add Remove

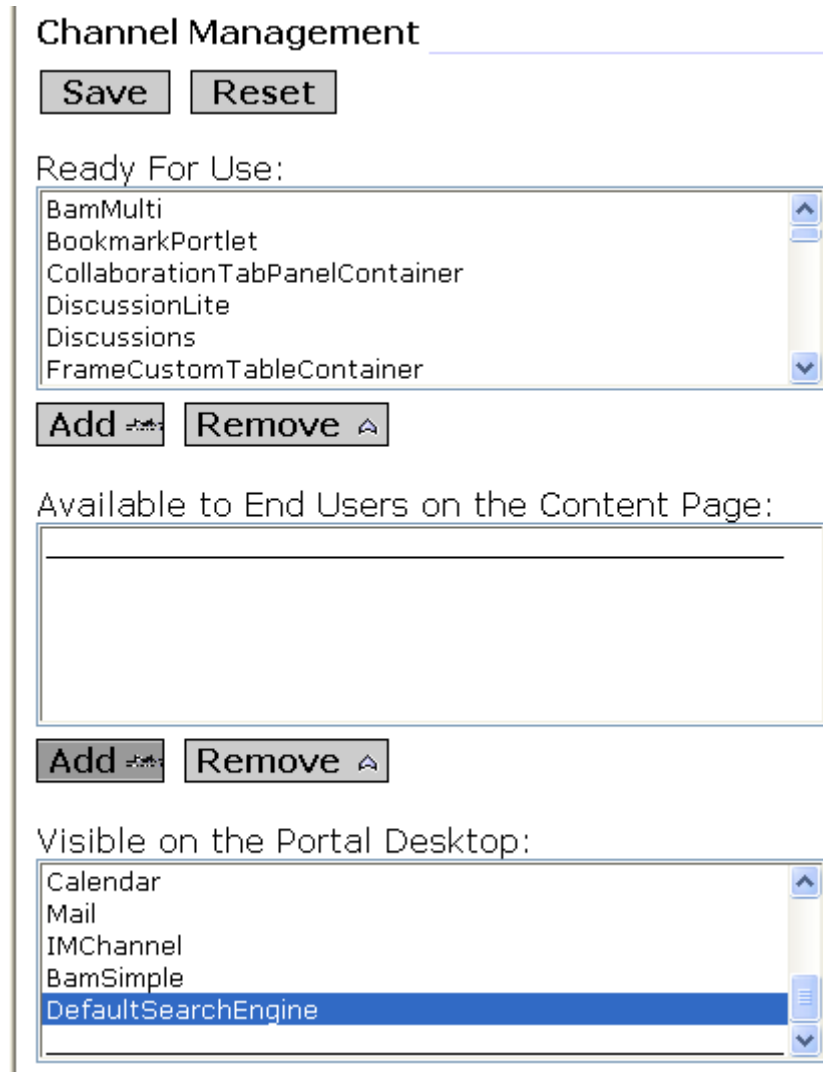
Visible on the Portal Desktop:

UserInfo
MailCheck
App
Bookmark
MyFrontPageTabPanelContainer/Bookmark2
SampleJSP

- 11 Select the new channel (if necessary) and click **Add**.

Your new channel name appears under **Visible on the Portal Desktop**. The name appears at the end of the list, so you may need to scroll down the pane to view it. See Figure 110.

Figure 110 Channel Management Section: Second Add



The figure shows a web interface titled "Channel Management". At the top, there are two buttons: "Save" and "Reset". Below these, there is a section labeled "Ready For Use:" containing a list box with the following items: "BamMulti", "BookmarkPortlet", "CollaborationTabPanelContainer", "DiscussionLite", "Discussions", and "FrameCustomTableContainer". Below the list box are two buttons: "Add" (with a plus icon) and "Remove" (with a minus icon). The next section is labeled "Available to End Users on the Content Page:" and contains an empty list box. Below this list box are also "Add" and "Remove" buttons. The final section is labeled "Visible on the Portal Desktop:" and contains a list box with the following items: "Calendar", "Mail", "IMChannel", "BamSimple", and "DefaultSearchEngine". The "DefaultSearchEngine" item is currently selected and highlighted in blue.

- 12 Click **Save**.

Your new channel is now a part of the **Front Page** tab. This channel appears the next time a user with permission to view it accesses the Portal Desktop and selects the **Front Page** tab.

Viewing a New Channel

Using this procedure, you can place any existing channel on any existing tab panel page you choose, and then view that channel.

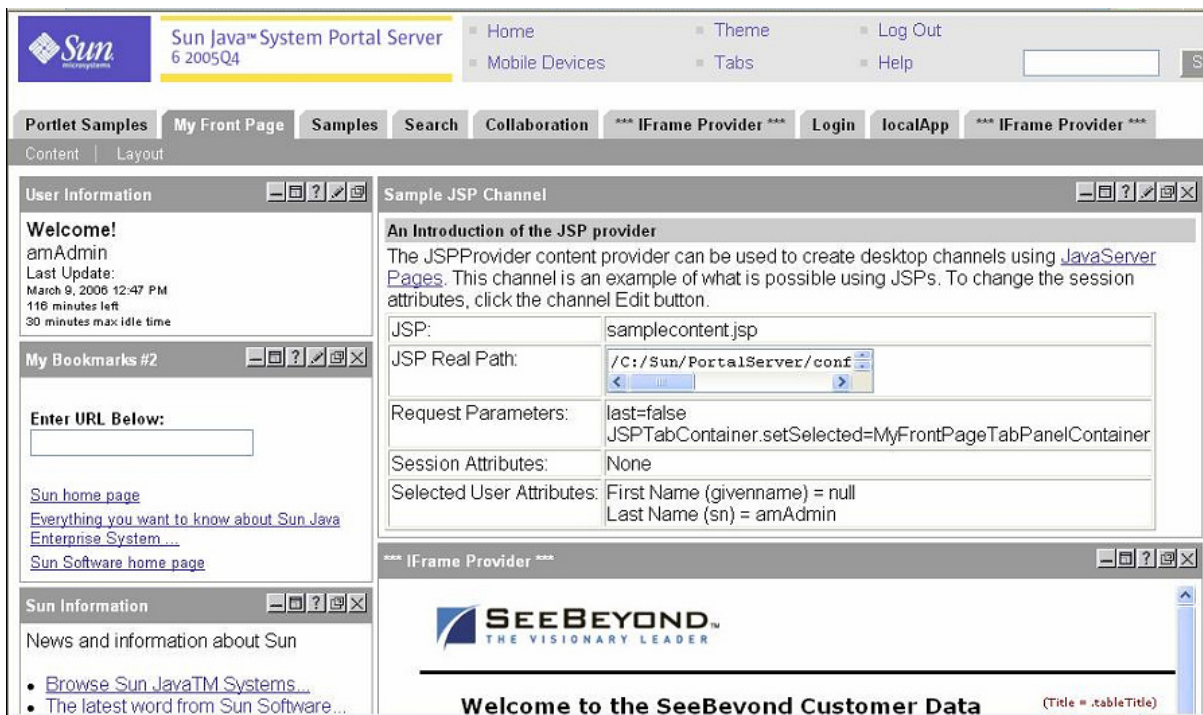
To view your new channel on the Portal Desktop

- 1 Perform either of the following operations:
 - ♦ On the Windows **Start > All Programs** menu, choose **Sun Microsystems > Portal Server > Portal Desktop**.
 - ♦ Enter the following URL on your Web browser:

`http://<host name>:38080/amserver/portal/dt`

The Portal Desktop Welcome page appears with the **My Front Page** tab displayed. See Figure 111, which shows an example.

Figure 111 Portal Server 6 Portal Desktop Welcome Page



- 2 You can scroll down this page to view your new Portal Server channel.

9.4.4 Running eBAM Studio Applications with Portal Server 7

This section explains how to set up Sun Java System Portal Server 7 to run with the Java CAPS applications, including eBAM Studio.

Before You Start

Take care to ensure you have completed the prerequisite actions listed under **Before Installing Portal Server** on page 131.

System Requirements

You must run Portal Server 7 in:

- Sun Java System Application Server, version 8.2

If you want to deploy JSR 168 portlets with eBAM Studio, you must deploy Portal 7 together with the application server on any of the following platforms:

- Solaris 10 OS for SPARC and x86 platforms
- Solaris 9 OS for SPARC and x86 platforms
- Solaris 8 OS for the SPARC platform
- Red Hat Enterprise Linux WS/AS/ES 2.1 to 2.1U6
- Red Hat Enterprise Linux WS/AS/ES 3.0 to 3.0U4

Supported Operating Systems

For specific information on all operating systems supported by Portal Server 7 in conjunction with the Java CAPS and eBAM Studio, see the **Readme.txt** file for eBAM Studio.

Important

The port numbers provided in all access URL examples are the system defaults. The actual port numbers used by your system's configuration may be different. Before you begin using the procedures provided in the rest of this section, make sure you have verified the correct port number for any URL referenced by a given procedure.

Installation

For complete instructions on installation, see the Sun Microsystems *Sun Java System Portal Server 7 Installation Guide*. See [Reference Documentation](#) on page 130 for an explanation of how to find this document.

Setting Up an IFrame Channel

This section describes how to create and configure an IFrame channel for Portal Server 7.

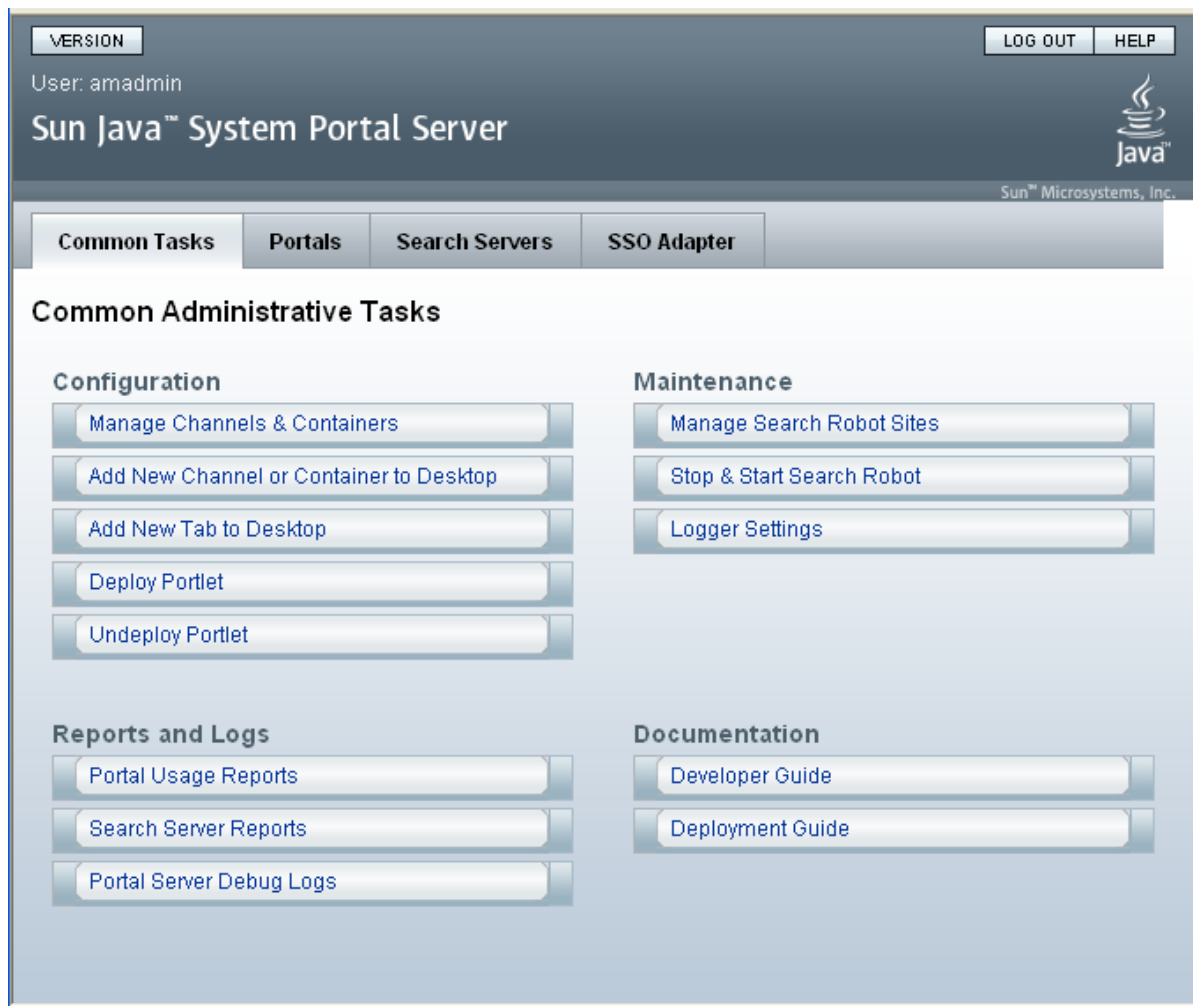
To set up an IFrame channel

- 1 Log on to Portal Server 7 Console (**psconsole**) with Access Manager privileges, using the following URL:

```
http://<host name>:38080/psconsole
```

The Portal Server 7 Console page appears. See Figure 112.

Figure 112 Portal Server 7 Console Page



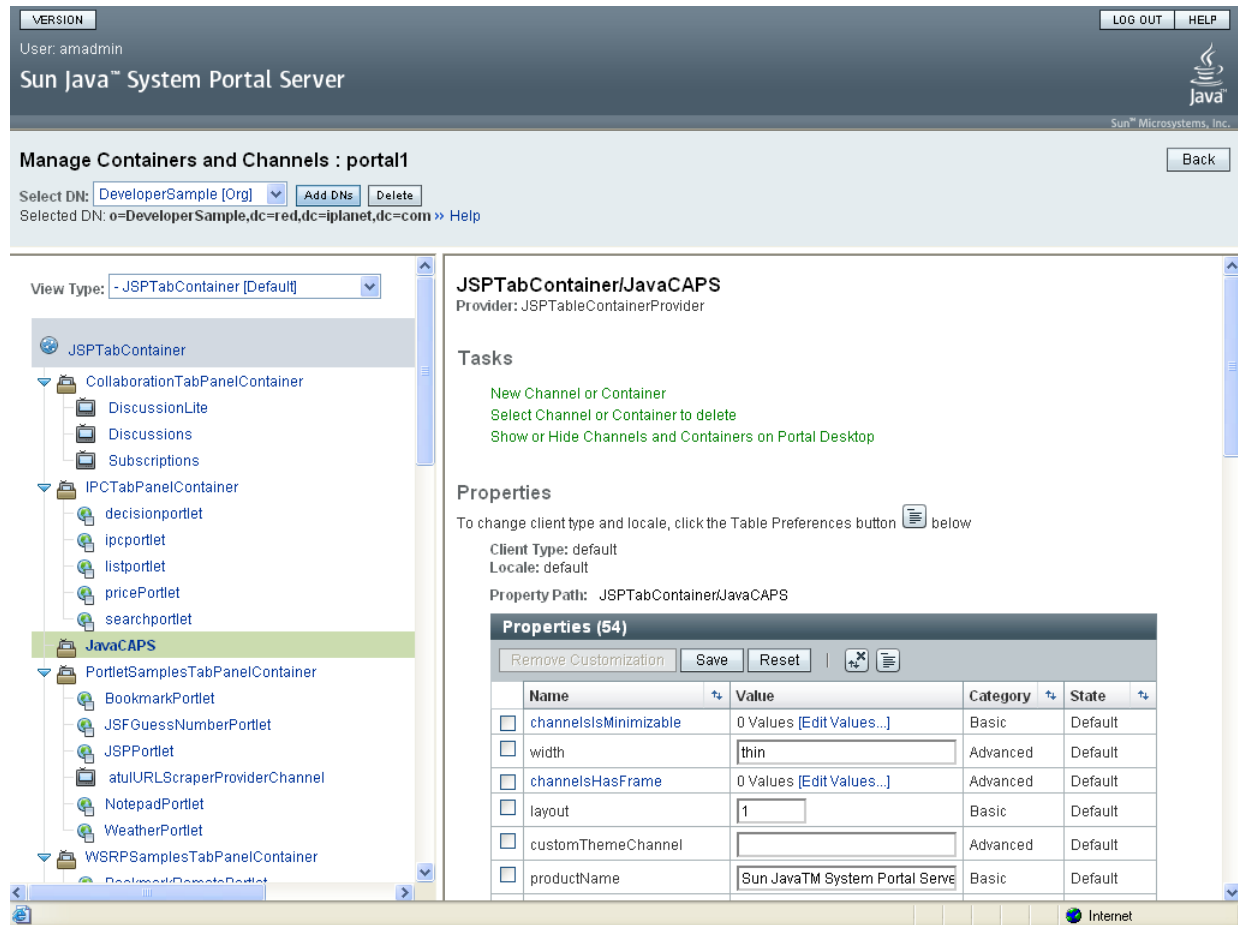
- 2 On the **Common Tasks** tab, under **Configuration**, click **Manage Channels And Containers**.

A window appears that allows you to choose the portal and distinguished name (DN) you want to create the channel for.

- 3 For this example, choose **portal1** for the portal and **DeveloperSample [Org]** for the DN, and click **OK**.

The Manage Containers and Channels: portal1 page appears. See Figure 113.

Figure 113 Manage Containers and Channels : portal1 Page: Initial



Using this page, you can choose an existing tab or create a new one. For an explanation of how to create a new tab, see the Sun Microsystems **Sun Java System Portal Server 7 Technical Overview** document.

- 4 In the left frame, select the container in which you want to create the new channel. In the example in Figure 113 you select the **JavaCAPS** container.
- 5 In the right frame, select **New Channel or Container**.

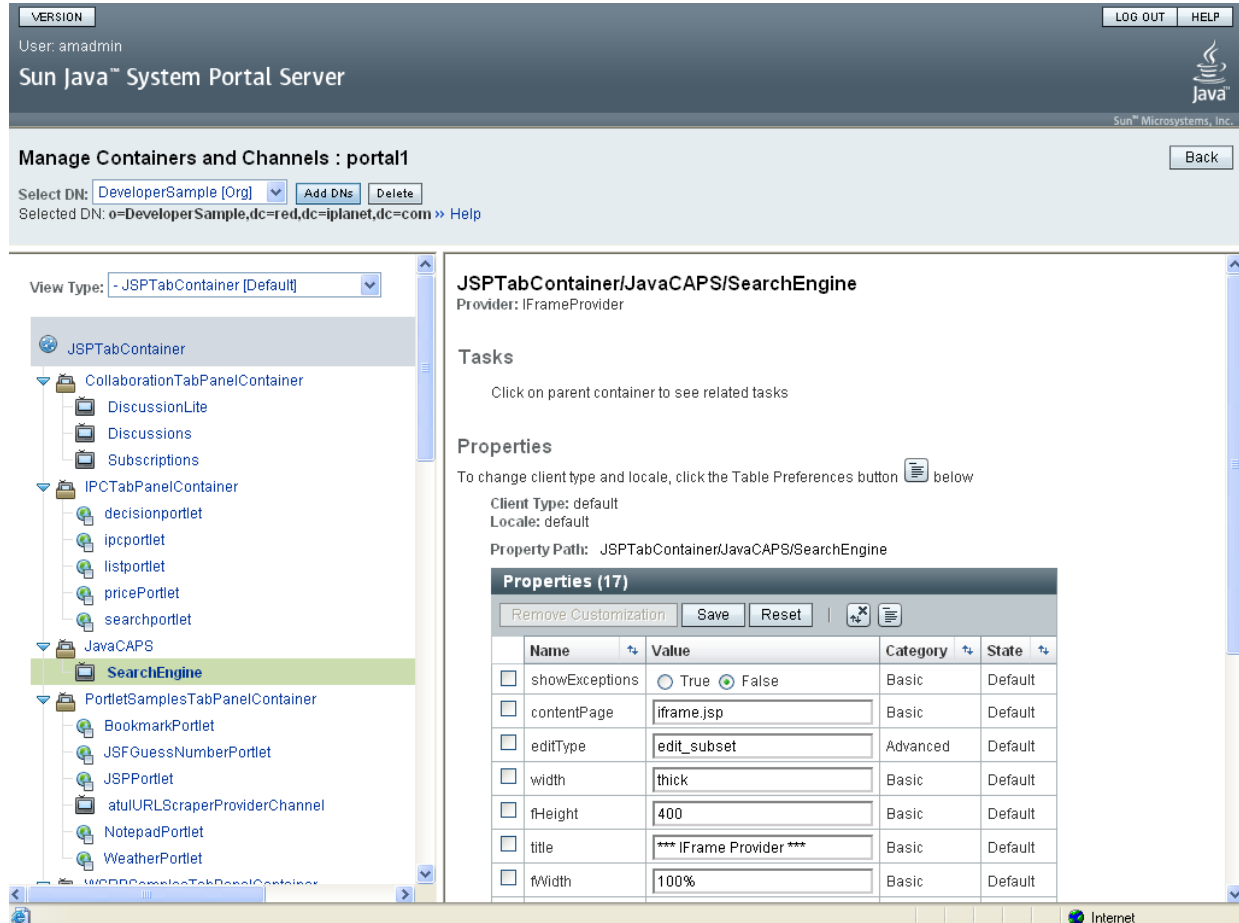
A new window appears. Using that window, perform the following steps:

- E Select the Portal instance and DN you want and click **Next**.
- F On the next window, choose **Provider Channel** for the **Channel Type** and click **Next**.
- G On the next window, choose **IFrameProvider** for the **Provider** and click **Next**.
- H On the next window, enter a name for your channel, **SearchEngine** for this example, and click **Next**.

- I In the next window, view the information you have entered to make sure it is correct, and then click **Finish**.
- J On the next window, which indicates you have successfully created a new channel, click **Close**.

The Manage Containers and Channels: portal1 page appears again. The page now displays properties for your new channel, in the right frame. See Figure 114.

Figure 114 Manage Containers and Channels : portal1 Page: With New Channel



- 6 On the right side of the page, scroll down the pane to display all the new channel's properties. See Figure 115.

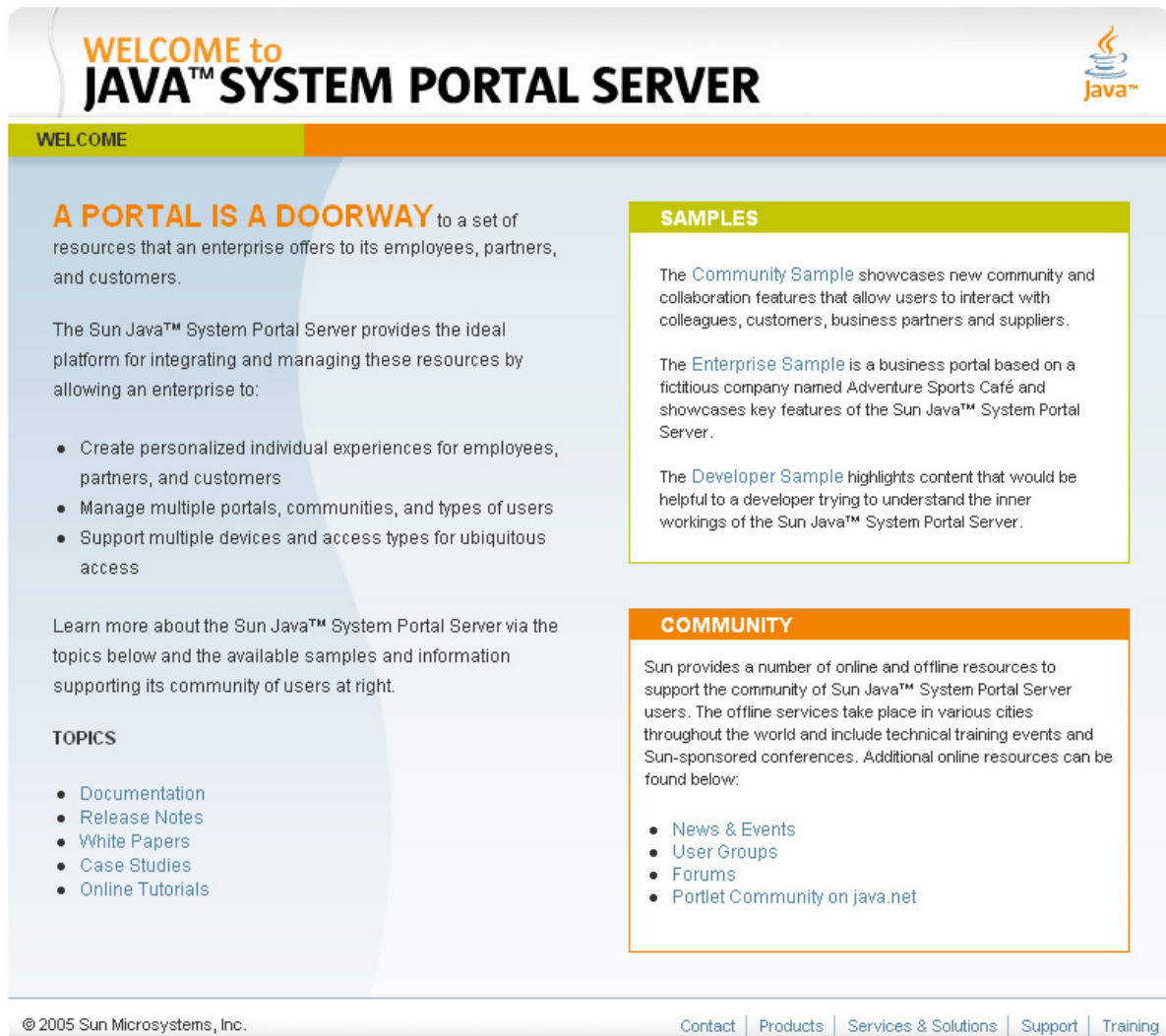
Figure 115 Channel Properties Pane

Properties (17)				
<input type="button" value="Remove Customization"/> <input type="button" value="Save"/> <input type="button" value="Reset"/>				
	Name	Value	Category	State
<input type="checkbox"/>	showExceptions	<input type="radio"/> True <input checked="" type="radio"/> False	Basic	Default
<input type="checkbox"/>	contentPage	<input type="text" value="iframe.jsp"/>	Basic	Default
<input type="checkbox"/>	editType	<input type="text" value="edit_subset"/>	Advanced	Default
<input type="checkbox"/>	width	<input type="text" value="thick"/>	Basic	Default
<input type="checkbox"/>	fHeight	<input type="text" value="400"/>	Basic	Default
<input type="checkbox"/>	title	<input type="text" value="*** IFrame Provider ***"/>	Basic	Default
<input type="checkbox"/>	fWidth	<input type="text" value="100%"/>	Basic	Default
<input type="checkbox"/>	productName	<input type="text" value="Sun Java™ System Portal Serve"/>	Basic	Default
<input type="checkbox"/>	refreshTime	<input type="text" value="0"/>	Advanced	Default
<input type="checkbox"/>	helpURL	<input type="text" value="en/desktop/iframechann.htm"/>	Advanced	Default
<input type="checkbox"/>	fBorder	<input type="text" value="0"/>	Basic	Default
<input type="checkbox"/>	isTopLevel	<input type="radio"/> True <input checked="" type="radio"/> False	Advanced	Default
<input type="checkbox"/>	isEditable	<input type="radio"/> True <input checked="" type="radio"/> False	Advanced	Default
<input type="checkbox"/>	description	<input type="text" value="*** This Provider uses IFrames *"/>	Basic	Default
<input type="checkbox"/>	scrolling	<input type="text" value="yes"/>	Basic	Default
<input type="checkbox"/>	fontFace1	<input type="text" value="Sans-serif"/>	Basic	Default
<input type="checkbox"/>	srcURL	<input type="text" value="http://www.sun.com"/>	Basic	Default

- 7 Be sure to enter the URL for your new channel under **srcURL** and give your new channel a **title**.
- 8 Supply other configuration parameters, as needed.
- 9 When you are finished, click **Save**.
- 10 Access the Portal Desktop, using `http://<host name>:38080/portal`, to view your new channel.

The Portal Desktop Welcome page appears. See Figure 116.

Figure 116 Portal Server 7 Portal Desktop Welcome Page



- 11 On the Welcome page, select the **Developer Sample** link.

The Developer Sample page appears.

- 12 In this example, your new channel appears under the **Java CAPS** tab.

Setting Up a JSR 168 Portlet Channel

This section describes how to create and configure a JSR 168 portlet channel for Portal Server 7.

Portal Server 7 patches are required to register Java CAPS JSR 168 portlets. See the Portal Server technical note “Integrating Java CAPS With Sun Java System Portal Server” referenced under [Necessary Patches](#) on page 131, for installation instructions.

Before You Start

Before creating a JSR 168 channel, you must deploy the eBAM Studio **.ear** file to the application server on which Portal Server is running and register the same **.ear** file with Portal Server.

To deploy eBAM Studio applications

- 1 Make sure the Sun Java System Application Server is running.
- 2 Java CAPS applications require special permissions. Make sure the appropriate permissions have been set up correctly in the **server.policy** file.
- 3 Deploy the eBAM Studio **.ear** file to the application server on which Portal Server is running. You can do this operation using:
 - ♦ Enterprise Designer
 - ♦ Enterprise Manager
 - ♦ A Sun Java System Application Server's administrator interface, either Admin Console or the command line; a command-line example follows:

```
./asadmin deploy --user admin /tmp/prj2dp8.ear
```

See the *eGate Integrator User's Guide* and *eGate Integrator System Administration Guide* for details on how to deploy an **.ear** file using Enterprise Designer and Enterprise Manager. See the [procedure on page 136](#) for information on how to deploy an **.ear** file using Admin Console.

Registering eBAM Studio JSR 168 application with Portal Server 7

Use the **psadmin register-portlet** command in the system's command line to register the portlet **.ear** file to a particular portal and DN. The following text shows an example of this command:

```
./psadmin register-portlet -u amadmin -f pfile -p portal1  
-d o=Sample,dc=sun,dc=com  
/tmp/168/simplejsr168dpsun.ear
```

Note: For details on how to use the command line, see the Sun Microsystems **Sun Java System Portal Server 7 Command-Line Reference** document.

After you have done this action, the portlet becomes available in the list of portlets for use with a JSR 168 channel.

Note: If you redeploy an application without changing the application **.ear** file name, the *ServletContext* name and the Portlet name, you do not need to perform registration again.

To create a JSR 168 portlet channel

- 1 Follow steps 1 through 4 under [Setting Up an IFrame Channel](#) on page 156.
- 2 In the right frame, select **New Channel or Container**.
A new window appears. Using that window, perform the following steps:
 - A Select the Portal instance and DN you want and click **Next**.

- B On the next window, choose **JSR 168 Channel** for the **Channel Type** and click **Next**
- C On the next window, choose one of the portlet providers you have registered, from the drop-down list.

After registering the portlet with **psadmin**, a provider for your portlet is available from the list of portlet providers. Your provider appears as `<servlet-context-name>.<portlet-name>`. For example, in eVision, `<portlet-name>` defaults to **PageFlowPortlet**. So, if you generate an eVision portlet using **myContext** for `<servlet-context-name>`, the portlet provider is listed as **myContext.PageFlowPortlet**.

- D On the next window, enter a name for your channel and click **Next**.
- E In the next window, view the information you have entered to make sure it is correct, and then click **Finish**.
- F On the next window, which indicates you have successfully created a new channel, click **Close**.

The Manage Containers and Channels: portal1 page appears again. The page now displays your new JSR 168 channel.

- 3 On the right side of the page, scroll down the pane to display all the new channel's properties.
- 4 Enter the **title** you want.
- 5 Supply other configuration parameters, as needed.
- 6 When you are finished, click **Save**.
- 7 Access the Portal Desktop, using `http://<host name>:38080/portal`, to view your new channel.

Unregistering eBAM Studio JSR 168 application with Portal Server 7

When appropriate, you can use the **psadmin unregister-portlet** command to unregister a portlet. This step requires the name of the **.ear** file that was registered, for example:

```
./psadmin unregister-portlet -u amadmin -f pfile -p portal1  
-d o=Sample,dc=sun,dc=com simplejsr168dpsun
```

9.4.5 Running eBAM Studio Applications with Portal Server 7.1

This section explains how to set up Sun Java System Portal Server 7.1 to run with the Java CAPS applications, including eBAM Studio.

Before You Start

Take care to ensure you have completed the prerequisite actions listed under **Before Installing Portal Server** on page 131.

System Requirements

You must run Portal Server 7.1 in:

- Sun Java System Application Server, version 8.2

If you want to deploy JSR 168 portlets with eBAM Studio, you must deploy Portal 7.1 together with the application server on any of the following platforms:

- Solaris 9 or 10 OS for SPARC
- Solaris 9 or 10 OS for x86
- Red Hat Enterprise Linux 3.0 Update 3 or 4.0 on x86

Supported Operating Systems

For specific information on all operating systems supported by Portal Server 7.1 in conjunction with the Java CAPS and eBAM Studio, see the **Readme.txt** file for eBAM Studio.

Important

The port numbers provided in all access URL examples are the system defaults. The actual port numbers used by your system's configuration may be different. Before you begin using the procedures provided in the rest of this section, make sure you have verified the correct port number for any URL referenced by a given procedure.

Installation

For complete instructions on installation, see the Sun Microsystems *Sun Java System Portal Server 7.1 Installation Guide*. See [Reference Documentation](#) on page 130 for an explanation of how to find this document.

Setting Up an IFrame Channel

This section describes how to create and configure an IFrame channel for Portal Server 7.1.

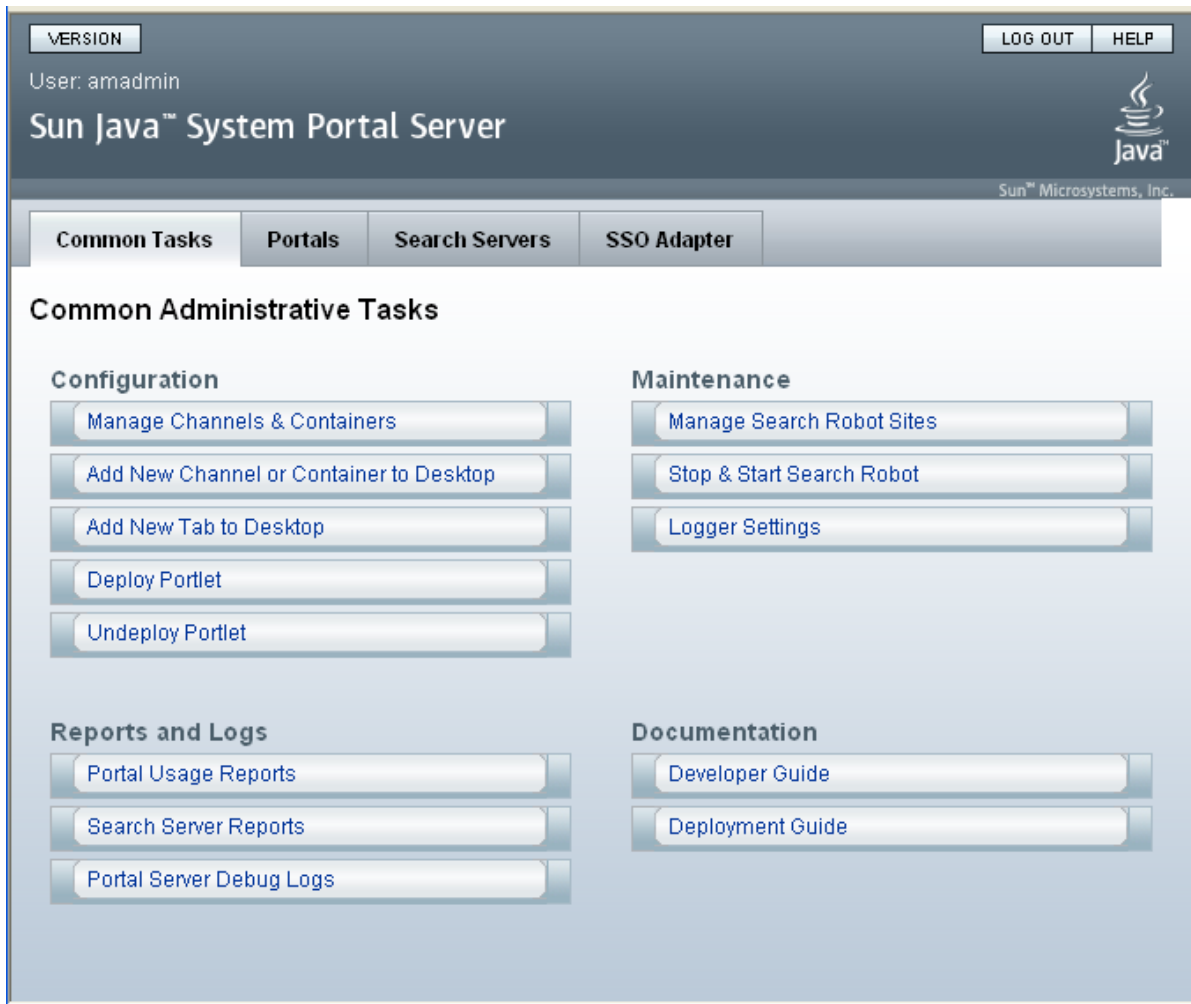
To set up an IFrame channel

- 1 Log on to Portal Server 7.1 Console (**psconsole**) with Access Manager privileges, using the following URL:

```
http://<host name>:38080/psconsole
```

The Portal Server 7 Console page appears. See Figure 112.

Figure 117 Portal Server 7.1 Console Page



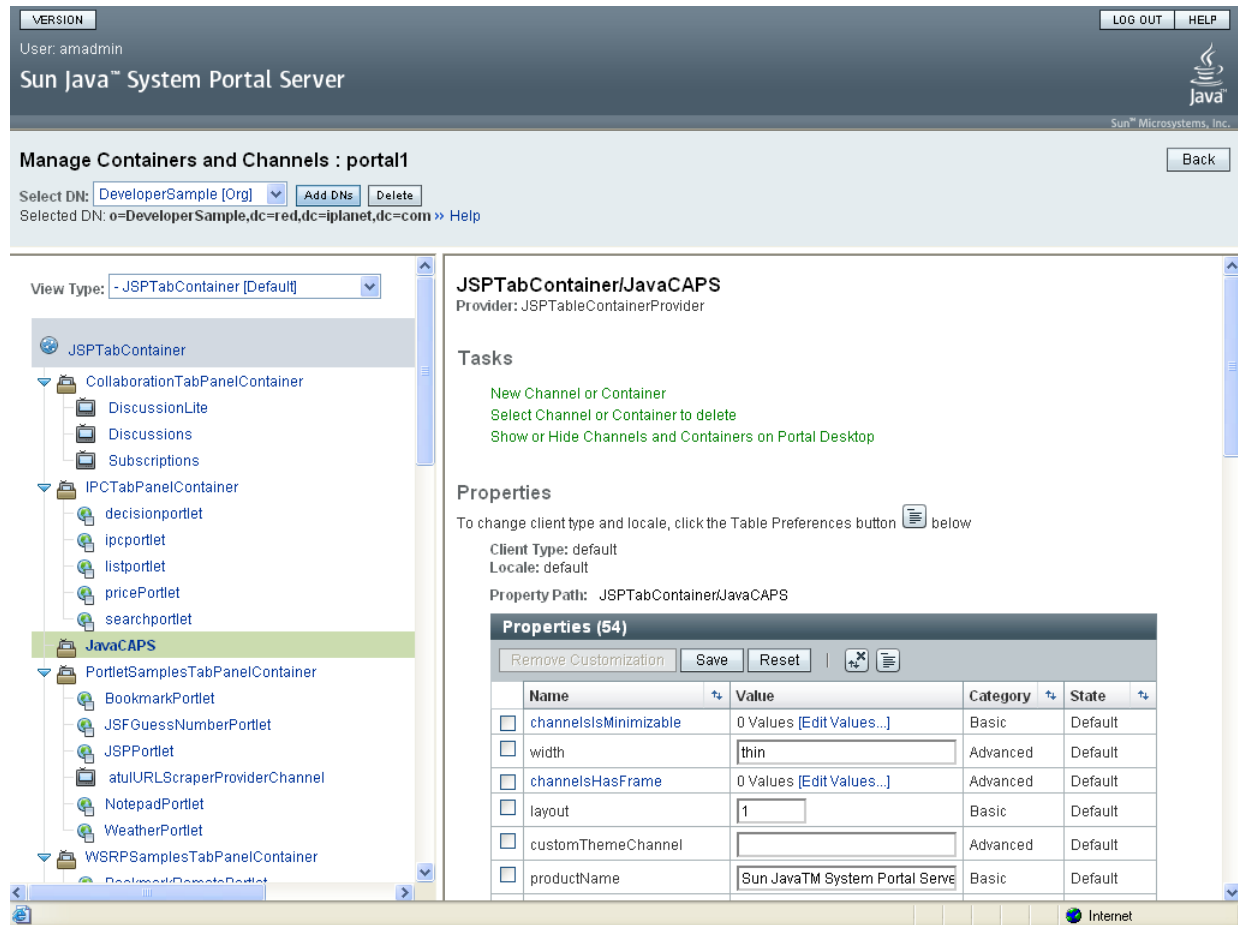
- 2 On the **Common Tasks** tab, under **Configuration**, click **Manage Channels And Containers**.

A window appears that allows you to choose the portal and distinguished name (DN) you want to create the channel for.

- 3 For this example, choose **portal1** for the portal and **DeveloperSample [Org]** for the DN, and click **OK**.

The Manage Containers and Channels: portal1 page appears. See Figure 113.

Figure 118 Manage Containers and Channels: portal1 Page: Initial



Using this page, you can choose an existing tab or create a new one. For an explanation of how to create a new tab, see the Sun Microsystems **Sun Java System Portal Server 7 Technical Overview** document.

- 4 In the left frame, select the container in which you want to create the new channel. In the example in Figure 113 you select the **JavaCAPS** container.
- 5 In the right frame, select **New Channel or Container**.

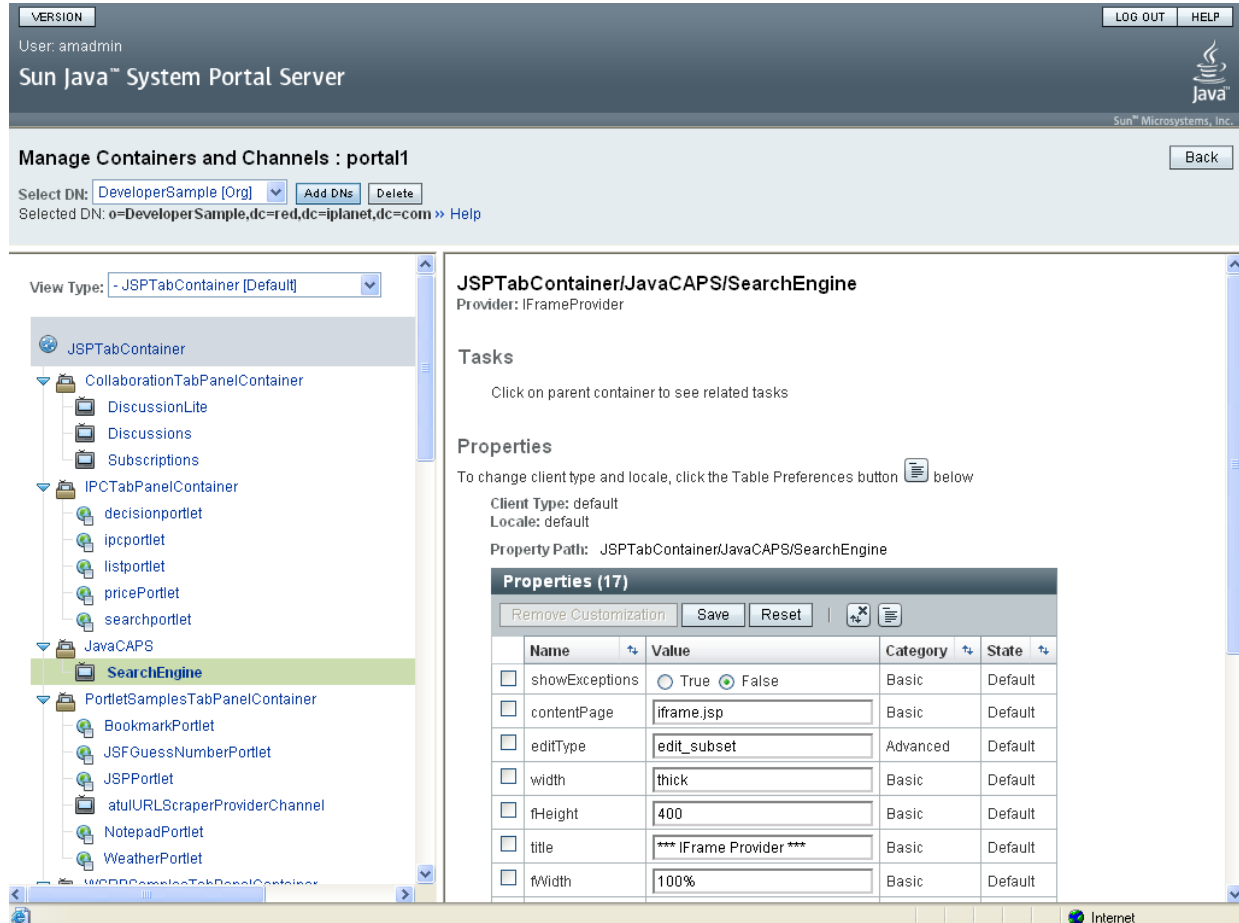
A new window appears. Using that window, perform the following steps:

- G Select the Portal instance and DN you want and click **Next**.
- H On the next window, choose **Provider Channel** for the **Channel Type** and click **Next**.
- I On the next window, choose **IFrameProvider** for the **Provider** and click **Next**.
- J On the next window, enter a name for your channel, **SearchEngine** for this example, and click **Next**.

- K In the next window, view the information you have entered to make sure it is correct, and then click **Finish**.
- L On the next window, which indicates you have successfully created a new channel, click **Close**.

The Manage Containers and Channels: portal1 page appears again. The page now displays properties for your new channel, in the right frame. See Figure 114.

Figure 119 Manage Containers and Channels: portal1 Page: With New Channel



- 6 On the right side of the page, scroll down the pane to display all the new channel's properties. See Figure 115.

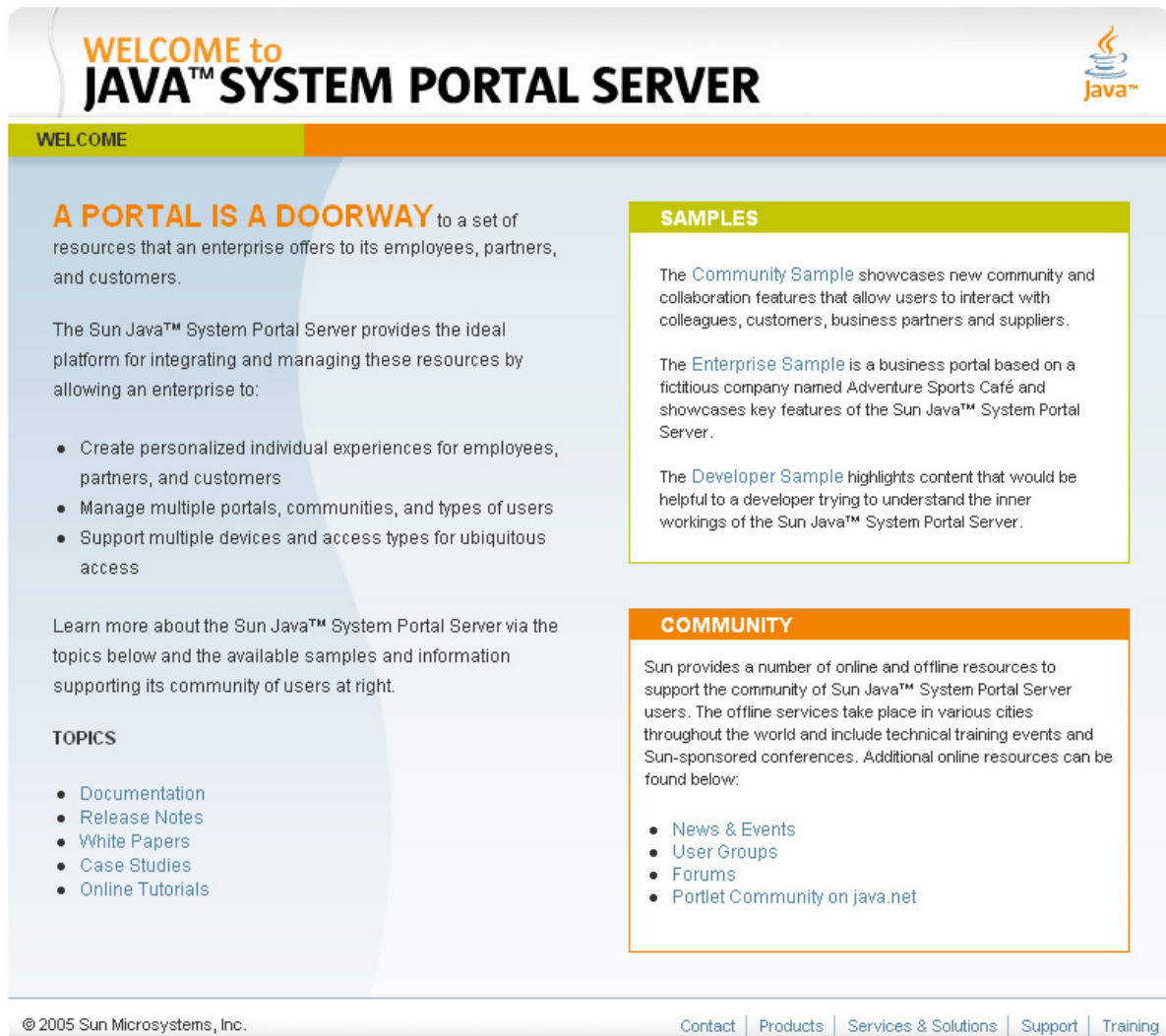
Figure 120 Channel Properties Pane

Properties (17)				
<input type="button" value="Remove Customization"/> <input type="button" value="Save"/> <input type="button" value="Reset"/>				
	Name	Value	Category	State
<input type="checkbox"/>	showExceptions	<input type="radio"/> True <input checked="" type="radio"/> False	Basic	Default
<input type="checkbox"/>	contentPage	<input type="text" value="iframe.jsp"/>	Basic	Default
<input type="checkbox"/>	editType	<input type="text" value="edit_subset"/>	Advanced	Default
<input type="checkbox"/>	width	<input type="text" value="thick"/>	Basic	Default
<input type="checkbox"/>	fHeight	<input type="text" value="400"/>	Basic	Default
<input type="checkbox"/>	title	<input type="text" value="*** IFrame Provider ***"/>	Basic	Default
<input type="checkbox"/>	fWidth	<input type="text" value="100%"/>	Basic	Default
<input type="checkbox"/>	productName	<input type="text" value="Sun Java™ System Portal Serve"/>	Basic	Default
<input type="checkbox"/>	refreshTime	<input type="text" value="0"/>	Advanced	Default
<input type="checkbox"/>	helpURL	<input type="text" value="en/desktop/iframechann.htm"/>	Advanced	Default
<input type="checkbox"/>	fBorder	<input type="text" value="0"/>	Basic	Default
<input type="checkbox"/>	isTopLevel	<input type="radio"/> True <input checked="" type="radio"/> False	Advanced	Default
<input type="checkbox"/>	isEditable	<input type="radio"/> True <input checked="" type="radio"/> False	Advanced	Default
<input type="checkbox"/>	description	<input type="text" value="*** This Provider uses IFrames *"/>	Basic	Default
<input type="checkbox"/>	scrolling	<input type="text" value="yes"/>	Basic	Default
<input type="checkbox"/>	fontFace1	<input type="text" value="Sans-serif"/>	Basic	Default
<input type="checkbox"/>	srcURL	<input type="text" value="http://www.sun.com"/>	Basic	Default

- 7 Be sure to enter the URL for your new channel under **srcURL** and give your new channel a **title**.
- 8 Supply other configuration parameters, as needed.
- 9 When you are finished, click **Save**.
- 10 Access the Portal Desktop, using `http://<host name>:38080/portal`, to view your new channel.

The Portal Desktop Welcome page appears. See Figure 116.

Figure 121 Portal Server 7.1 Portal Desktop Welcome Page



- 11 On the Welcome page, select the **Developer Sample** link.

The Developer Sample page appears.

- 12 In this example, your new channel appears under the **Java CAPS** tab.

Setting Up a JSR 168 Portlet Channel

This section describes how to create and configure a JSR 168 portlet channel for Portal Server 7.1.

Portal Server 7.1 patches are required to register Java CAPS JSR 168 portlets. See the Portal Server technical note “Integrating Java CAPS With Sun Java System Portal Server” referenced under [Necessary Patches](#) on page 131, for installation instructions.

Before You Start

Before creating a JSR 168 channel, you must deploy the eBAM Studio **.ear** file to the application server on which Portal Server is running and register the same **.ear** file with Portal Server.

To deploy eBAM Studio applications

- 1 Make sure the Sun Java System Application Server is running.
- 2 Java CAPS applications require special permissions. Make sure the appropriate permissions have been set up correctly in the **server.policy** file.
- 3 Deploy the eBAM Studio **.ear** file to the application server on which Portal Server is running. You can do this operation using:
 - ♦ Enterprise Designer
 - ♦ Enterprise Manager
 - ♦ A Sun Java System Application Server's administrator interface, either Admin Console or the command line; a command-line example follows:

```
./asadmin deploy --user admin /tmp/prj2dp8.ear
```

See the *eGate Integrator User's Guide* and *eGate Integrator System Administration Guide* for details on how to deploy an **.ear** file using Enterprise Designer and Enterprise Manager. See the [procedure on page 136](#) for information on how to deploy an **.ear** file using Admin Console.

Registering eBAM Studio JSR 168 application with Portal Server 7

Use the **psadmin register-portlet** command in the system's command line to register the portlet **.ear** file to a particular portal and DN. The following text shows an example of this command:

```
./psadmin register-portlet -u amadmin -f pfile -p portal1  
-d o=Sample,dc=sun,dc=com  
/tmp/168/simplejsr168dpsun.ear
```

Note: For details on how to use the command line, see the Sun Microsystems **Sun Java System Portal Server 7.1 Command-Line Reference** document.

After you have done this action, the portlet becomes available in the list of portlets for use with a JSR 168 channel.

Note: If you redeploy an application without changing the application **.ear** file name, the *ServletContext* name and the Portlet name, you do not need to perform registration again.

To create a JSR 168 portlet channel

- 1 Follow steps 1 through 4 under [Setting Up an IFrame Channel](#) on page 156.
- 2 In the right frame, select **New Channel or Container**.

A new window appears. Using that window, perform the following steps:

 - A Select the Portal instance and DN you want and click **Next**.

- B On the next window, choose **JSR 168 Channel** for the **Channel Type** and click **Next**
- C On the next window, choose one of the portlet providers you have registered, from the drop-down list.

After registering the portlet with **psadmin**, a provider for your portlet is available from the list of portlet providers. Your provider appears as `<servlet-context-name>.<portlet-name>`. For example, in eVision, `<portlet-name>` defaults to **PageFlowPortlet**. So, if you generate an eVision portlet using **myContext** for `<servlet-context-name>`, the portlet provider is listed as **myContext.PageFlowPortlet**.

- D On the next window, enter a name for your channel and click **Next**.
- E In the next window, view the information you have entered to make sure it is correct, and then click **Finish**.
- F On the next window, which indicates you have successfully created a new channel, click **Close**.

The Manage Containers and Channels: portal1 page appears again. The page now displays your new JSR 168 channel.

- 3 On the right side of the page, scroll down the pane to display all the new channel's properties.
- 4 Enter the **title** you want.
- 5 Supply other configuration parameters, as needed.
- 6 When you are finished, click **Save**.
- 7 Access the Portal Desktop, using `http://<host name>:38080/portal`, to view your new channel.

Unregistering eBAM Studio JSR 168 application with Portal Server 7.1

When appropriate, you can use the **psadmin unregister-portlet** command to unregister a portlet. This step requires the name of the **.ear** file that was registered, for example:

```
./psadmin unregister-portlet -u amadmin -f pfile -p portal1  
-d o=Sample,dc=sun,dc=com simplejsr168dpsun
```

Chart Types

eBAM provides four chart types. Appendix A discusses those charts and describes their properties.

What's in This Chapter

- [“Chart Types” on page 172](#)

A.1 Chart Types

The chart types eBAM provides include:

- *Pie charts* display elements in the dataset view as wedge-shaped segments (slices) comprising an entire disk (pie). Each segment's relation to the pie and to other segments is cued visually by its apparent area, based on its subtended angle. For an example, see [Figure 67 on page 101](#).
- *Bar charts* display elements in the dataset view as rectangles (bars) in a rectilinear setup where each bar's relation to the total and to other bars is cued visually by its apparent area, based on its length. Bar charts can have labels on either or both axes to facilitate quantitative readings. For an example, see [Figure 72 on page 106](#).
- *Meters*, also called *meter charts*, display conditions as a needle on a dial, similar to a tachometer or clock face. At designated thresholds on the range, differing colors at the outside of the dial are used to signify when the measured condition is in normal range, warning range, or critical range.
- *Trafficlights* convey conditions at a glance, using one, two, or three indicators, using the metaphor of a traffic signal: a light in a green, amber, or red lens indicates a normal, warning, or critical condition. (Colors can be tailored to suit.) For an example of a two-dimensional trafficlight array, see [Figure 70 on page 104](#).

All charts have common properties, and each has properties specific to its own type:

- [“Properties Common to All Chart Types” on page 173](#) lists properties common to all four chart types.
- [“Properties Common to Pie and Bar Charts” on page 173](#) lists properties common to pie and bar charts only.
- [“Properties Specific to Pie Charts” on page 174](#) lists properties specific to pie charts.

- [“Properties Specific to Bar Charts” on page 174](#) lists properties specific to bar charts.
- [“Properties Specific to Meters” on page 174](#) lists properties specific to meters.
- [“Properties Specific to Trafficlights” on page 176](#) lists properties specific to trafficlights.

Table 21 Properties Common to All Chart Types

Property Name	Default	Notes
Display Title	True	Whether or not the chart title is displayed.
Title		Any string; set initially to the name of the chart as created.
Title Font		You can use any of 52 fonts, ranging in size from 9-point to 72-point, with or without bold and/or italic attributes.
Title Alignment	CENTER	You can change the default (CENTER) to LEFT or RIGHT.
Title Color		You can either pick a color swatch or specify an exact color: <ul style="list-style-type: none"> ▪ Specify percentages for HSB: Hue, Saturation, Brightness ▪ Specify values (0-255) for RGB: Red, Green, Blue.
Title Background		
Draw Border	False	Whether or not to draw a border around the chart.
Border Color		You can specify any color either by picking a swatch or by specifying HSB or RGB.
Background Color		
Image Width	680	Width, in pixels, of the image portion of the chart.
Image Height	420	Height, in pixels, of the image portion of the chart.
Chart Width	680	Width, in pixels, of the entire chart itself.
Chart Height	420	Height, in pixels, of the entire chart itself.
Frequency in seconds	60	How often to update the chart with a new data view.

Table 22 Properties Common to Pie and Bar Charts

Property Name	Default	Notes
3D	True	Whether or not a three-dimensional effect is displayed.
Depth Factor	0.3	(for Pie chart) Amount by which the chart seems three-dimensional.
Include Legend	False	Whether or not to display a legend with the chart.
Legend Anchor	SOUTH	Where to position the starting point for the chart legend.
Circular	False	(for Pie chart) For an elliptical (titled-circle) chart, keep the set to False . For a circular chart, set this to True .
Number Format	0	How many digits to display, and whether to display digits in comma-separated groups of three.
Display Navigation Buttons	False	Whether the chart viewer displays a chevron [>] as a hyperlink to the chart specified in the next field.
Navigation Chart Name		Name of the “next” chart targeted by the hyperlink in the chart viewer.

Table 23 Properties Specific to Pie Charts

Property Name	Default	Notes
Show Section Labels	False	Whether to display a label name for the data series. (<i>Only Category queries with one or more Series values.</i>)
Section Label	Name, Value	Choices consist of: Name; Value; Percent; Name, Value; Name, Percent; Value, Percent; and Name, Value, Percent.
Section Label Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Section Label Color		You can set this either by picking a swatch or by specifying HSB or RGB.
Section Label Gap	0.3	Space used by the section label gap (0.0=none; 1.0=all).
Direction	Clockwise	Each successively larger value is displayed either right (clockwise) or left (anticlockwise) of the next smaller.
Interior Gap	0.3	Space to be used by the interior gap (0.0=none; 1.0=all).

Table 24 Properties Specific to Bar Charts

Property Name	Default	Notes
Orientation	VERTICAL	Which way (vertical or horizontal) the bars run.
Show X-Axis	True	Whether or not to display the X axis.
Show X-Axis Label	True	Whether or not to display the label for the X axis.
X-Axis Label	domain	The text to display as a label for the X axis.
Show Y-Axis	True	Whether or not to display the Y axis.
Show Y-Axis Label	True	Whether or not to display the label for the Y axis.
Y-Axis Label	range	The text to display as a label for the Y axis.
Stacked	False	(<i>Only Category queries with one or more Series values.</i>) Whether or not to stack the data series.
Category Label Type	None	(<i>Only Category queries with one or more Series values.</i>) To display the category name, set this to Name. To display the category values, set this to Value. To suppress the label altogether, set this to None.
Series Label Type	None	(<i>Only Category queries with one or more Series values.</i>) To display the series names, set this to Name. To display the series values, set this to Value. To suppress the label altogether, set this to None.
Category Gap Percent	10	(<i>Only Category queries with one or more Series values.</i>) To set the Category Gap percent between the series of Bars.

Table 25 Properties Specific to Meters

Property Name	Default	Notes
Minimum Value	0.0	Lower bound of the meter.

Table 25 Properties Specific to Meters (Continued)

Property Name	Default	Notes
Maximum Value	3.0	Upper bound of the meter.
Value Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Value Color		To set the color of the value labels. You can set this either by picking a swatch or by specifying HSB or RGB.
Minimum Normal Value	0.0	Values below this threshold are too low to be normal.
Maximum Normal Value	1.0	Values above this threshold are too high to be normal.
Normal Range Color		To set the color of the Normal range. You can set this either by picking a swatch or by specifying HSB or RGB.
Minimum Warning Value	1.0	Values below this threshold are too low for warnings.
Maximum Warning Value	2.0	Values above this threshold are too high for warnings.
Warning Range Color		To set the color of the Warning range. You can set this either by picking a swatch or by specifying HSB or RGB.
Minimum Critical Value	2.0	Values below this threshold are too low to be critical.
Maximum Critical Value	3.0	Values above this threshold are beyond being critical.
Critical Range Color		To set the color of the Critical range. You can set this either by picking a swatch or by specifying HSB or RGB.
Number Format	0.000	How many digits to display, and whether to display digits in comma-separated groups of three.
Units	units	To specify the units value.
Draw Chart Border	False	Whether or not to display a border around the chart.
Dial Type	Circle	You can change this to either Circle, Pie, or Chord.
Dial Background Color		To set color of the dial background. You can set this either by picking a swatch or by specifying HSB or RGB.
Tick Label Type	Value Label	Whether to show marks with values, or just marks.
Tick Label Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Needle Color		You can specify any color, by swatch, HSB, or RGB.
Meter Angle	270	Angle subtended by the entire range of the meter.
Display Navigation Buttons	False	Whether the chart viewer displays a chevron [>] as a hyperlink to the chart specified in the next field.
Navigation Chart Name		Name of the “next” chart targeted by the hyperlink in the chart viewer.

Table 26 Properties Specific to Trafficlights

Property Name	Default	Notes
Display Housing	True	Whether or not to displays the Housing background in the chart.
Housing Margin	0	How much additional space to provide around the Housing.
Extend Housing To Edge	False	Whether or not to display the extended Housing edges in the chart.
Maintain Housing Proportionality	True	Whether to retain the ratio of height to width.
Housing Color	Need to select	To set the Color of the Housing. You can set this either by picking a swatch or by specifying HSB or RGB.
Linearize Date	False	Whether or not to linearize the data in the chart specified.
Use Fixed Horizontal Display Limit	True	Whether or not to use the fixed horizontal display limit.
Fixed Horizontal Display Limit	1	To set the value for fixed horizontal display limit.
Use Fixed Vertical Display Limit	True	Whether or not to use the fixed vertical display limit.
Fixed Vertical Display Limit	1	To set the value for the fixed vertical display limit.
Lens Display Count	3	Number of lenses to display: 1, 2, or 3.
Lens Display Orientation	VERTICAL	Which way (vertical or horizontal) the traffic light is oriented.
Lens Display Direction	Right to Left/ Bottom to Top	In horizontal orientation, successively worse states are displayed either right-to-left or left-to-right. In vertical orientation, successively worse states are displayed either top-to-bottom or bottom-to-top.
Housing Label Relationship	in	Whether the labels are inside or outside the housing.
Housing Label Source	none	Choices consist of: none; category; series; category, series; value; category, value; series, value; and category, series value.
Housing Label Position	top	You can change this to either top, left, right, or bottom.
Housing Label Font	San Serif 10 Plain	You can select any of the 52 fonts with 3-pt to 48-pt size.
Housing Label Color	(white)	To set the color of the housing labels. You can set this either by picking a swatch or by specifying HSB or RGB.
Center Labeled States	False	Whether or not to center the labels for the states (labels such as "Normal", "Warning", and "Critical").
State Label Source	none	Choices consist of: none; static label, category; series; category, series; value; static label, value; category, value; series, value; and category, series value.

Table 26 Properties Specific to Trafficlights (Continued)

Property Name	Default	Notes
State Label Position	bottom	You can change this to either top, left, right, or bottom.
State Label Font	San Serif 10 Plain	You can select any of the 52 fonts with 3-pt to 48-pt size.
State Label Color	(white)	To set the color of the state labels. You can set this either by picking a swatch or by specifying HSB or RGB.
Lens Label Source	none	Choices consist of: none; static label, category; series; category, series; value; static label, value; category, value; series, value; and category, series value.
Lens Label Font	San Serif 10 Plain	You can select any of the 52 fonts with 3-pt to 48-pt size.
Lens Label Color	(white)	To set the color of the lens labels. You can set this either by picking a swatch or by specifying HSB or RGB
Lens Shape	Round	You can change this to either Round or Square.
Lens Style	SmallRidges	You can change this to either Plain, Glow, SmallRidges, BigRidges, or Elevator.
Normal Static Label	Normal	To set the name of normal static label in the chart.
Minimum Normal Value	0.0	Values below this threshold are too low to be normal.
Maximum Normal Value	1.0	Values above this threshold are too high to be normal.
Normal Range Color	(red)	To set the color of the normal range. You can set this either by picking a swatch or by specifying HSB or RGB.
Warning Static Label	Warning	To set the name of warning static label in the chart.
Minimum Warning Value	1.0	Values below this threshold are too low for warnings.
Maximum Warning Value	2.0	Values above this threshold are too high for warnings.
Warning Range Color	(yellow)	To set the color of the Warning range. You can set this either by picking a swatch or by specifying HSB or RGB.
Critical Static Label	Critical	To set the name of critical static label in the chart.
Minimum Critical Value	2.0	Values below this threshold are too low to be critical.
Maximum Critical Value	3.0	Values above this threshold are beyond being critical.
Critical Range Color	(green)	To set the color of the Critical range. You can set this either by picking a swatch or by specifying HSB or RGB.

Persistence and Monitoring

Persistence and monitoring features are optional. If you use them, they require a different database—the *eInsight engine* database—to collect and persist data from your business processes (BPs).

See the *Sun SeeBeyond eInsight Business Process Manager User's Guide* for eInsight engine database requirements.

What's in This Chapter

- [“Overview” on page 178](#)
- [“Setting Up the Database Schema for eInsight Engine” on page 179](#)
- [“Configuring the eInsight Engine for Runtime” on page 180](#)
- [“Monitoring” on page 183](#)
- [“Logging” on page 186](#)

B.1 Overview

eInsight provides scripts to create the eInsight database schema, which can be used to collect and persist data from your BPs. (The eInsight database schema is optional, and is independent of any databases used by eBAM.)

After the eInsight engine database schema has been created and configured for use by a particular DB username, the instance must be referenced by the BPs whose activities are to be persisted by it. It must also be referenced by all Integration Servers that host persisted BPs.

Once a specific BP has been set up to so that its data is persisted by an instance of an eInsight engine database, you can additionally configure the BP so that its activities are monitored by Enterprise Manager.

As needed, you can set up several different eInsight database instances.

Procedures

The following steps are required for eInsight database and engine setup:

- [“Setting Up the Database Schema for eInsight Engine” on page 179](#)
- [“Configuring the eInsight Engine for Runtime” on page 180](#)

After the schema and engine(s) have been set up, follow these steps for each BP:

- [“Configuring Specific BPs to Use Persistence” on page 182](#)
- [“Turning Monitoring On or Off for Specific BPs” on page 183](#)

For instructions on using Enterprise Manager to monitor and log BP activity, see:

- [“Monitoring” on page 183](#)
- [“Logging” on page 186](#)

B.2 Setting Up the Database Schema for eInsight Engine

To create the runtime recoverability database schema, you must extract and run the eInsight database scripts that are automatically installed with eInsight.

B.2.1 Exporting and Extracting Database Scripts for eInsight

eInsight engine database scripts are supplied as compressed files. These files cannot be run in place, and must instead be exported and extracted.

To export and extract the eInsight engine database schema scripts

- 1 In Enterprise Explorer, in the project tree, expand the following folders:
Sun SeeBeyond > eInsight > Download Database Scripts
- 2 Right-click the **.zip** file associated with the appropriate database (oracle.zip, db2.zip, sybase.zip, or sqlserver.zip) and, on the popup context menu, click **Export**.
- 3 Save the **.zip** file to a local folder.
- 4 Extract the **.zip** file contents to a local folder, which will contain:
 - ♦ **install_db.bat** or **install.sh**—Creates the tablespace, users, tables, stored procedures, and initial values.
 - ♦ **uninstall_db.bat** or **uninstall_db.sh**—Reverses what the install_db script creates; that is, it drops tables and users, and deletes stored procedures.
 - ♦ **clear_db.bat** or **clear_db.sh**—Truncates all tables without performing any other uninstall actions.
 - ♦ *database-specific.sql scripts*—Called by the install_db and uninstall_db scripts. For example:, create_tables.sql, drop_tables.sql.
 - ♦ **Readme.txt**—Additional instructions specific to your database application.
- 5 For additional information, read the material in the Readme.txt file.

B.2.2 Running Database Scripts for eInsight

To create, modify, or delete the runtime recoverability database schema, run the corresponding eInsight database script that you extracted in the previous procedure. The database user that executes this script must have permission to create tables.

To install, truncate, or uninstall the eInsight engine database schema

- 1 Open a command prompt (on Windows) or (on UNIX) shell.
- 2 Change to the directory where you extracted the eInsight engine database scripts.
- 3 Enter one of the following commands.
 - ♦ To create the tablespace, users, tables, stored procedures, and initial values:
install_db <user> <password> <tnsname>
 - ♦ To truncate the database tables created by the install_db script:
clear_db <user> <password> <tnsname>
 - ♦ To undo the effects of the install_db script (in other words, to drop the tables, users, and stored procedures that were created):
uninstall_db <user> <password> <tnsname>

In each case:

- ♦ <user> is the database username
- ♦ <password> is the password for this database user
- ♦ <tnsname> is the database or TNS name

Note: The default user and password created from these scripts is: **einsight**. You can modify the user, password, and disk space allocated for tables and user permissions. Do not modify the table and column definitions.

B.3 Configuring the eInsight Engine for Runtime

To configure the eInsight engine in an Integration Server

- 1 In Environment Explorer, open the environment and Logical Host (checking it out, if necessary), right-click the **Integration Server (IS)**, and, on the popup context menu, click **Properties**.
- 2 In the Properties dialog, under Configuration click **eInsight Engine Configuration**. See Figure 122.

Figure 122 eInsight Engine Configuration Properties

Properties	
Application Mode	Single Engine
Automatic Execution Of Database Scripts	false
Database	Oracle 9i
Database Connection Max Idle Time (Seconds)	600
Database Connection Pool Size	60
Database Connection Retries	Infinity
Database Connection Retry Interval (milliseconds)	10000
Database Connection Steady Pool Size	20
Database Host	<host>
Database Instance/Schema	<sid>
Database Password	
Database Port	1521
Database User	<user>
Debug	false
Debug Port	4865
Enable Monitoring	true
Engine Expiry Interval (seconds)	120
Failover Grace Period (seconds)	30
Invocation Allocation Ratio (%)	80
Monitoring Thread Buffer Size	2
Monitoring Thread Buffer Time Lag (seconds)	30
Monitoring Thread Sleep Time (milliseconds)	5000
Persistence Mode	Persist to Database
Receive Timeout (milliseconds)	15000
Recovery Batch Size	10
Recovery Enabled	true
Reporting Thread Sleep Time (milliseconds)	180000
Work Item Submit Limit	500

Description (Application Mode)

Single Engine - application can be deployed in one IS/LH connecting to one DB

Multi Engine - application can be deployed on multiple IS/LHs connecting to one DB

Comments (Application Mode)

OK Cancel

- 3 Set the following parameters appropriately for your database connection:
 - A Application Mode—Select one of the following (compare Persistence Mode):
 - ♦ Multiple Engine—allows for both recovery and failover
 - ♦ **Single Engine**—allows for recovery but not failover
 - B Automatic Execution of Database Scripts—If you have already run these scripts (see [“Running Database Scripts for eInsight” on page 179](#)), retain the default value: false
 - C **Database**—Select Oracle 9i, Oracle 8.1.7, DB2, Sybase 12.5, or SQL Server 2000.
 - D **Database Connection Pool Size**—Enter a positive integer to set an upper limit on the number of runtime threads spawned by the eInsight engine for activity generated by Java CAPS components or Web services. The default setting is 60; thus, if 300 requests are received, the engine processes them in five groups of 60.

- E Database Host**—Enter the name of the machine where your database resides.
- F Database Instance/Schema**—Enter the database name or SID.
- G Database Port**—Enter your database connection port number (default=1521).
- H Database User Name**—Enter the user name for your database.
- I Enable Monitoring**—For monitoring, change this from the default to: **true**.
- J Password**—Enter the password for your database user.
- K Persistence Mode**—For persistence, change this to one of the following:
 - ♦ Persist to Database - Multiple Engines (Recovery, Failover)
 - ♦ Persist to Database - Single Engines (Recovery)
- L Recover During Startup**—For persistence, change this from the default to: **true**.

Note: Leave other settings with their default values if you have no reason to override them.

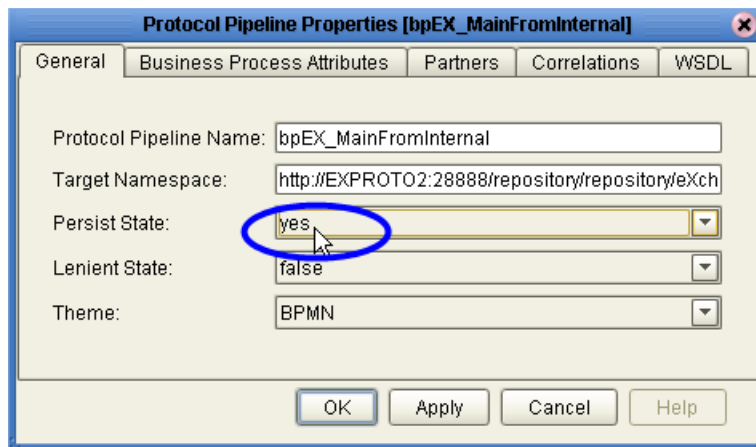
B.3.1 Configuring Specific BPs to Use Persistence

Persistence is set for each business process individually. The default setting is **no**.

To set persistence for a specific business process

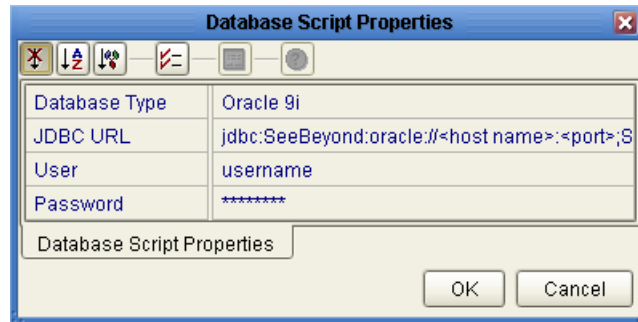
- 1 In Enterprise Designer, in the project tree, right-click the BP (first use **Check Out**, if necessary) and, on the popup context menu, click **Properties**.
- 2 In the **General** tab (see Figure 123), change **Persist State** to **yes** and then click **OK**.

Figure 123 Setting Persistence for a Business Process



- 3 On the main Enterprise Designer toolbar, click **Save All**.
The project tree displays a new **Database Scripts** under your business process. In Figure 124, see the highlighted item in the project tree.
- 4 Right-click **Database Scripts** and, on the popup context menu, click **Properties**.
- 5 In the Business Process Database Script Properties, supply the correct information for your database. See Figure 124.

Figure 124 Setting Database Script Properties for BPs



B.3.2 Turning Monitoring On or Off for Specific BPs

Monitoring is turned on or off for each business process individually, and only applies to BPs that have been configured to use persistence.

To start running a database script for a particular business process

- 1 In the project tree, open the BP and its **Database Scripts** folder.
If the BP does not have a Database Scripts folder, follow the [procedure on page 182](#).
- 2 Right-click the appropriate database **Install** script and, on the popup context menu, click **Run**.
Running this script completes the setup process for this BP, and allows you to use Enterprise Manager to monitor the BP at runtime.

To stop running a database script for a business process

- 1 In the project tree, open the BP and its **Database Scripts** folder (see Figure 124).
- 2 Right-click the appropriate database **Uninstall** script and, on the pop-up context menu, click **Run**.

B.4 Monitoring

Enterprise Manager allows users to quickly identify problems with components or systems. From Enterprise Manager, you can double-click a BP component to go directly to the problem.

From Enterprise Manager (Monitoring interface), you can:

- Filter the list of displayed instances to quickly identify exceptions.
- Navigate to particular versions of a BP to monitor the progress of instances.
- Use a Web-based interface allows users to securely access the monitoring environment over the Internet.

About the Adobe SFV plug-in

See the *eGate Integrator System Administration Guide* and the *Readme.txt* file for complete information on the Adobe SVG plug-in. Although this plug-in is not required to see the graphic model in the Enterprise Manager or to use monitoring, it must be uploaded and installed if you want to view and use the special tools for the BP state diagram in Enterprise Manager.

B.4.1 Using Enterprise Manager to Monitor BP Activity

Once persistence is configured and the project is running, you can use Enterprise Manager to monitor the activity of any BP instance for which monitoring has been enabled.

To monitor a BP

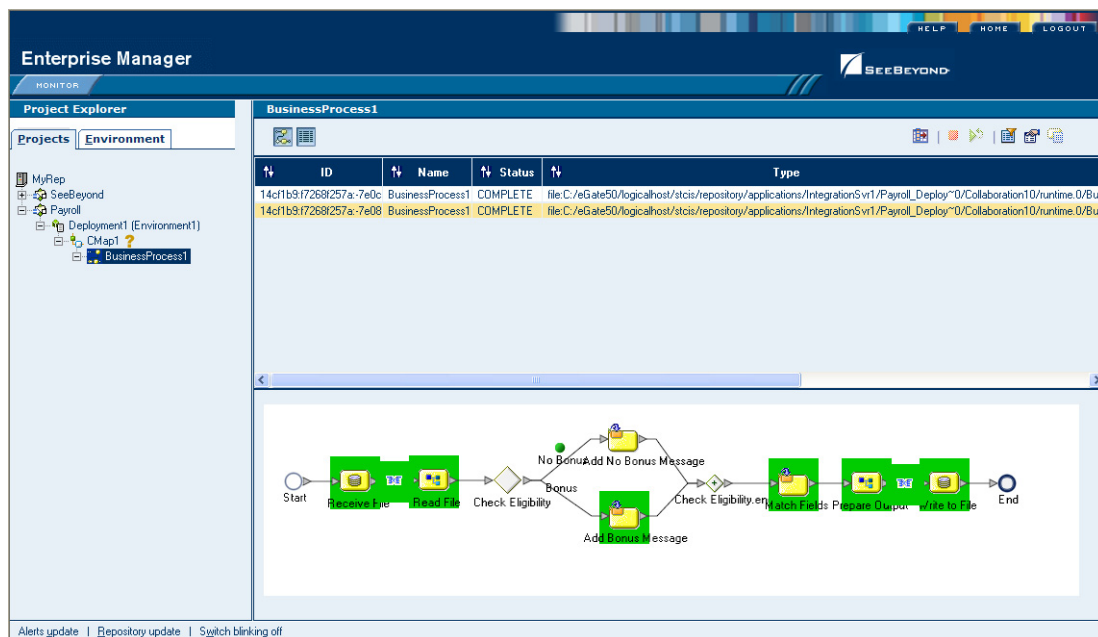
Before you begin: In Enterprise Designer, ensure each BP you want to monitor has been set up to use persistence and monitoring and that it references a valid instance of an eInsight engine database. Also, all Integration Servers hosting the BPs must reference the same database instance. If necessary, see the procedures earlier in this chapter.

- 1 Activate a deployment profile and bootstrap a Logical Host to run the project.
- 2 Start Enterprise Manager. If necessary, log in.
- 3 In the **Home** tab, click the Enterprise Monitor icon.

Project Explorer displays a tree structure of active projects and environments.

- 4 In the **Projects** tab, open the Project, Deployment Profile, and Connectivity Map, and click the name of the BP whose instance(s) you want to monitor. See Figure 125.

Figure 125 Monitor View



B.4.2 Using the Monitor Tool Palettes

Using the monitor console, you can view and interact with BP instances in both Project and Environment views.

- In Project view, you can start or stop a BP instance or set the cache.
- In Environment view, you can start, stop, and examine BP instances using the graphical model of the BP.





BP Monitoring Tools

Interactive monitoring tools allow you to control the view of BP instances, and manage the display of instance details. The monitor console provides controls in toolbar format, across the top of the **Details** (right-hand) pane of the window.

Tools for Controlling Display Modes

The display of BP instances and the Instance List in the console viewer are controlled by the buttons in the upper *left* of the Details window, described in the following table.

Table 27 Display Modes

	Hide Business Process hides the rendered image of a BP instance so it does not appear the Details pane.
	Show Business Process renders the image of a BP instance in the Details pane.
	Show Instance List displays the attributes of the current BP instance in list format and adds the tools described in the following table.
	Hide Instance List hides the attributes of the current BP instances and removes the instance tools from the interface.

Controlling the Display of Instance Data

When the monitor is in instance monitoring mode, you can manipulate the view of instance data using the buttons in the upper *right* of the Details window, as described in Table 28.

Table 28 Monitor: Display Instance Data







	Choose Preferences allows you to add, move, and sort the columns in the BP instance.
	Start starts a stopped BP instance.

Table 28 Monitor: Display Instance Data (Continued)

	Stop stops a BP instance.
	Filter Instances allows you to set criteria to display a specific instance or group of Instances.
	Business Process Instance Attributes displays the attributes of an Instance when the instance is selected in the Instance List .
	Activity Details allows you to see the details of an Activity. Defines a step within a particular Business Process. when the Activity is selected in the Instance List .

Note: *Online help for Enterprise Manager has additional information about monitoring business processes.*

B.5 Logging

The eInsight engine coordinates all BP-related activity of a deployed project. The engine runs within the SeeBeyond Integration Server.

B.5.1 Setting Logging Levels

You cannot set the logging level of the eInsight Engine from within the Business Process Instance Monitor. To set the log level, follow these steps.

To set the eInsight logging level

- 1 Using a text editor, open the **log4j.properties** file in the following folder:
<JavaCAPS513_logicalhost>/logconfigs/IS_integration-server-name
- 2 Add the following line:

```
log4j.category.com.stc.bpms.bpelImpl=<loglevel>
```

The values for **<loglevel>** are:

- ♦ Debug
- ♦ Information
- ♦ Warning
- ♦ Error
- ♦ Fatal

SQL Reserved Words

Because eBAM supports several dialects of SQL, it warns you against from using certain reserved words as eBAM field names. Table 29 lists the words that are disallowed; it is a superset of several lists of words reserved by SQL and/or SQLPlus.

Using any of these words as field name, whether in uppercase, lowercase, or mixed case, will result in a warning message.

C.1 SQL Reserved Words

Table 29 SQL Reserved Words

ABORT	ABS	ABSOLUTE
ACCESS	ACTION	ADA
ADD	ADMIN	AFTER
AGGREGATE	ALIAS	ALIGNMENT
ALL	ALLOCATE	ALPHANUMERIC
ALTER	ANALYSE	ANALYZE
AND	ANY	ARE
ARRAY	AS	ASC
ASENSITIVE	ASSERTION	ASSIGNMENT
ASYMMETRIC	AT	ATOMIC
AUTHORIZATION	AUTOINCREMENT	AVG
BACKUP	BACKWARD	BASETYPE
BEFORE	BEGIN	BETWEEN
BIGINT	BINARY	BIT
BITVAR	BIT_LENGTH	BLOB
BOOLEAN	BOTH	BREADTH
BREAK	BROWSE	BULK
BY	C	CACHE
CALL	CALLED	CARDINALITY
CASCADE	CASCADED	CASE

Table 29 SQL Reserved Words (Continued)

CAST	CATALOG	CATALOG_NAME
CHAIN	CHAR	CHARACTER
CHARACTERISTICS	CHARACTER_LENGTH	CHARACTER_SET_CATALOG
CHARACTER_SET_NAME	CHARACTER_SET_SCHEMA	CHAR_LENGTH
CHECK	CHECKED	CHECKPOINT
CLASS	CLASS_ORIGIN	CLOB
CLOSE	CLUSTER	CLUSTERED
COALESCE	COLLATE	COLLATION
COLLATION_CATALOG	COLLATION_SCHEMA	COLUMN
COLUMNS	COLUMN_NAME	COMMAND_FUNCTION
COMMAND_FUNCTION_CODE	COMMENT	COMMIT
COMMITTED	COMMUTATOR	COMPACTDATABASE
COMPLETION	COMPUTE	CONDITION_NUMBER
CONNECT	CONNECTION	CONNECTION_NAME
CONSTRAINT	CONSTRAINTS	CONSTRAINT_CATALOG
CONSTRAINT_NAME	CONSTRAINT_SCHEMA	CONSTRUCTOR
CONTAINER	CONTAINS	CONTAINSTABLE
CONTINUE	CONVERSION	CONVERT
COPY	CORRESPONDING	COUNT
CREATE	CREATEDB	CREATEFIELD
CREATEGROUP	CREATEINDEX	CREATEOBJECT
CREATEPROPERTY	CREATERELATION	CREATETABLEDEF
CREATEUSER	CREATEWORKSPACE	CROSS
CUBE	CURRENCY	CURRENT
CURRENT_DATE	CURRENT_PATH	CURRENT_ROLE
CURRENT_TIME	CURRENT_TIMESTAMP	CURRENT_USER
CURSOR	CURSOR_NAME	CYCLE
DATA	DATABASE	DATE
DATETIME_INTERVAL_CODE	DATETIME_INTERVAL_PRECISION	DAY
DDBC	DEALLOCATE	DEC
DECIMAL	DECLARE	DEFAULT
DEFERRABLE	DEFERRED	DEFINED
DEFINER	DELETE	DELIMITER
DELIMITERS	DENY	DEPTH
DEREF	DESC	DESCRIBE

Table 29 SQL Reserved Words (Continued)

DESCRIPTOR	DESTROY	DESTRUCTOR
DETERMINISTIC	DIAGNOSTICS	DICTIONARY
DISALLOW	DISCONNECT	DISK
DISPATCH	DISTINCT	DISTINCTROW
DISTRIBUTED	DO	DOMAIN
DOUBLE	DROP	DUMMY
DUMP	DYNAMIC	DYNAMIC_FUNCTION
DYNAMIC_FUNCTION_CODE	DYNASET	EACH
ELSE	ELEMENT	ENCODING
ENCRYPTED	END	END-EXEC
EQV	EQUALS	ERROR
ERRLVL	ESCAPE	EVERY
EXCEPT	EXCEPTION	EXCLUSIVE
EXEC	EXECUTE	EXISTING
EXISTS	EXIT	EXPLAIN
EXTENDED	EXTERNAL	EXTRACT
FALSE	FETCH	FIELD
FILE	FILLCACHE	FILLFACTOR
FINAL	FINALFUNC	FIRST
FLOAT	FLOAT4	FLOAT8
FOR	FORM	FORMS
FORCE	FOREIGN	FORTRAN
FORWARD	FOUND	FREE
FREETEXT	FREETEXTTABLE	FREEZE
FROM	FULL	FUNCTION
G	GENERAL	GENERATED
GET	GETOBJECT	GETOPTION
GOTOPAGE	GLOBAL	GO
GOTO	GRANT	GRANTED
GROUP	GROUPING	GTCMP
GUID	HANDLER	HASHES
HAVING	HIERARCHY	HOLD
HOLDLOCK	HOST	HOURL
IDENTITY	IDENTITYCOL	IDENTITY_INSERT
IDLE	IEEEDOUBLE	IEEESINGLE
IF	IGNORE	ILIKE

Table 29 SQL Reserved Words (Continued)

IMMEDIATE	IMMUTABLE	IMP
IMPLEMENTATION	IMPLICIT	IN
INCREMENT	INDEX	INDICATOR
INFIX	INHERITS	INITCOND
INITIALIZE	INITIALLY	INNER
INOUT	INPUT	INSENSITIVE
INSERT	INSTANCE	INSTANTIABLE
INSTEAD	INT	INTEGER
INTEGER2	INTEGER4	INTERNALLENGTH
INTERSECT	INTERVAL	INTO
INVOKER	IS	ISNULL
ISOLATION	ITERATE	JOIN
K	KEY	KEY_MEMBER
KEY_TYPE	KILL	LANCOMPILER
LANGUAGE	LARGE	LAST
LATERAL	LEADING	LEFT
LEFTARG	LENGTH	LESS
LEVEL	LIKE	LIMIT
LINENO	LISTEN	LOAD
LOCAL	LOCALTIME	LOCALTIMESTAMP
LOCATION	LOCATOR	LOCK
LOGICAL	LOGICAL1	LONGBINARY
LONGTEXT	LONGVARCHAR	LOWER
LTCMP	M	MAIN
MAP	MATCH	MAX
MAXROWS	MAXVALUE	MEMO
MEMORYSET	MERGES	MESSAGE_LENGTH
MESSAGE_OCTET_LENGTH	MESSAGE_TEXT	MIRROREXIT
MIN	MINUTE	MINVALUE
MOD	MODE	MODIFIES
MODIFY	MODULE	MONEY
MONTH	MORE	MOVE
MUMPS	NAME	NAMES
NATIONAL	NATURAL	NCHAR
NCLOB	NEGATOR	NEW
NEWPASSWORD	NEXT	NO
NOCHECK	NOCREATEDB	NOCREATEUSER

Table 29 SQL Reserved Words (Continued)

NONCLUSTERED	NONE	NOT
NOTHING	NOTIFY	NOTNULL
NULL	NULLABLE	NULLIF
NUMBER	NUMERIC	OBJECT
OCTET_LENGTH	OF	OFF
OFFSETS	OIDS	OLD
OLEOBJECT	ON	ONCE
ONLY	OPEN	OPENDATASOURCE
OPENQUERY	OPENRECORDSET	OPENROWSET
OPENXML	OPERATION	OPERATOR
OPTION	OPTIONS	OR
ORDER	ORDINALITY	OUT
OUTER	OUTPUT	OVER
OVERLAPS	OVERLAY	OVERRIDING
OWNER	OWNERACCESS	PAD
PARAMETER	PARAMETERS	PARAMETER_MODE
PARAMETER_NAME	PARAMETER_ORDINAL_POSITION	PARAMETER_SPECIFIC_CATALOG
PARAMETER_SPECIFIC_NAME	PARAMETER_SPECIFIC_SCHEMA	PARTIAL
PASCAL	PASSED BY VALUE	PASSWORD
PATH	PENDANT	PERCENT
PERM	PERMANENT	PIPE
PIVOT	PLACING	PLAIN
PLAN	PLI	POSITION
POSTFIX	PRECISION	PREFIX
PREPARE	PRESERVE	PRIMARY
PRIOR	PRINT	PRIVILEGES
PROC	PROCEDURAL	PROCEDURE
PUBLIC	QUIT	RAISERROR
READ	READS	READTEXT
REAL	RECALC	RECHECK
RECONFIGURE	RECORDSET	RECURSIVE
REF	REFERENCES	REFERENCING
REFRESH	REFRESH LINK	REGISTER DATABASE
REINDEX	RELATION	RELATIVE
RENAME	REPAINT	REPAIR DATABASE

Table 29 SQL Reserved Words (Continued)

REPEATABLE	REPLACE	REPLICATION
REPORTS	REQUERY	RESET
RESIGNAL	RESTORE	RESTRICT
RESULT	RETURN	RETURNED_LENGTH
RETURNED_OCTET_LENGTH	RETURNED_SQLSTATE	RETURNS
REPORTS	REQUERY	RESET
RESIGNAL	RESTORE	RESTRICT
RESULT	RETURN	RETURNED_LENGTH
RETURNED_OCTET_LENGTH	RETURNED_SQLSTATE	RETURNS
REVOKE	RIGHT	RIGHTARG
ROLE	ROLLBACK	ROLLUP
ROUTINE	ROUTINE_CATALOG	ROUTINE_NAME
ROUTINE_SCHEMA	ROW	ROWS
ROWCOUNT	ROWGUIDCOL	ROW_COUNT
RULE	SAVEPOINT	SCALE
SCHEMA	SCHEMA_NAME	SCOPE
SCREEN	SCROLL	SEARCH
SECOND	SECTION	SECURITY
SELECT	SELF	SENSITIVE
SEQUENCE	SERIALIZABLE	SERVER_NAME
SESSION	SESSION_USER	SET
SETFOCUS	SETOF	SETOPTION
SETS	SETUSER	SHARE
SHORT	SHOW	SHUTDOWN
SIMILAR	SIMPLE	SINGLE
SIZE	SMALLINT	SOME
SORT1	SORT2	SOURCE
SPACE	SPECIFIC	SPECIFICTYPE
SPECIFIC_NAME	SQL	SQLCA
SQLCODE	SQLERROR	SQLLEXCEPTION
SQLSTATE	SQLWARNING	STABLE
START	STATE	STATEMENT
STATIC	STATISTICS	STDEV
STDEVP	STDIN	STDOUT
STORAGE	STRICT	STRUCTURE

Table 29 SQL Reserved Words (Continued)

STYLE	SUBCLASS_ORIGIN	SUBLIST
SUBSTRING	SUM	SYMMETRIC
SYSID	SYSTEM	SYSTEM_USER
TABLE	TABLEDEF	TABLEDEFS
TABLEID	TABLE_NAME	TABLES
TABLESET	TEMP	TEMPLATE
TEMPORARY	TERMINATE	TEXTSIZE
THAN	THEN	TIME
TIMESTAMP	TIMEZONE_HOUR	TIMEZONE_MINUTE
TO	TOAST	TOP
TRAILING	TRAN	TRANSACTION
TRANSACTION_COMMITTE D	TRANSACTIONS_ROLLED_B ACK	TRANSACTION_ACTIVE
TRANSFORM	TRANSFORMS	TRANSLATE
TRANSLATION	TREAT	TRIGGER
TRIGGER_CATALOG	TRIGGER_NAME	TRIGGER_SCHEMA
TRIM	TRUE	TRUNCATE
TRUSTED	TSEQUAL	TYPE
UNCOMMITTED	UNDER	UNENCRYPTED
UNION	UNIQUE	UNKNOWN
UNLISTEN	UNNAMED	UNNEST
UNTIL	UPDATE	UPDATETEXT
UPPER	USAGE	USE
USER	USING	VACUUM
VALID	VALIDATOR	VALUE
VALUES	VARBINARY	VARCHAR
VARIABLE	VARP	VARYING
VERBOSE	VERSION	VIEW
VOLATILE	WAITFOR	WHEN
WHENEVER	WHERE	WHILE
WITH	WITHOUT	WORK
WRITE	WRITETEXT	XOR
YEAR	YES	YESNO
ZONE		

Index

Numerics

3D

- bar chart property 173
- pie chart property 173

A

activities

- add 32
- delete 32
- upsert 32

add web service 32, 44

- add.fields node 44

add.fields node 44

alert conditions

- creating 68
- email properties 69
- properties 69
- sending email 68

Alert Editor

- tool palette 63

alerts 22

App Server 8.2 EE

- (support for) 15

application servers

- (support for) 15

arguments

- adding to eBAM queries 57

Axis Labels (bar chart properties) 174

B

Background Color

- chart property 173, 175

bar charts

- described 172
- properties 173, 174
 - 3D 173
 - Depth Factor 173
 - Orientation 174

BEA WebLogic

- (support for) 15

Business Process Attributes tab 109

Business Process Display Mode

controlling 185

Business Rule

- adding 108

C

chart types

- listed and described 172

charts 21

- about 70
- creating and configuring 71
- properties 172
 - Background Color 173, 175
 - common to pie and bar charts 173
 - specific to bar charts 174
 - specific to meters 174
 - specific to trafficlighs 176
 - Title 173

columns

- in data definitions 39

command-line CodeGen

- (support for) 16

committing sample data 85

Component Editor

- editing queries 59, 60
- using 56, 57, 58

configuring

- data retention 36, 82
- queries 59

conventions, text 18

copy-and-paste of eBAM applications

- (support for) 16

Critical Range Color

- meters property 175, 177

D

data definitions 21

- creating 33
- setting up 33

data flow

- of eBAM Studio 23

data retention

- add web service 43
- configuring 36, 82

data retention properties

- modifying 39

database considerations

- "virtual database" 49

delete web service 32, 47

deleting expired data 36

delimited file format 40, 83

delimiters, setting 41, 83

Depth Factor

- bar chart property 173
- pie chart property 173
- Dial Type
 - meters property 175
- disallowed field names 58, 60, 187
- domainmgr.bat 119
- DVD
 - part numbers 28

E

- eBAM Studio
 - and Java CAPS 22
 - application
 - components of 32
 - important considerations 31
 - requirements 20
 - data flow 23
 - overview 13
 - summary of components 21
 - updating Enterprise Designer 29
 - uploading to Repository 25
- email
 - sending for alert conditions 68
- email notifications 69
- encoding scheme, setting 40, 83
- Enterprise Designer
 - updating with eBAM 29
- execute web service 32
- expired data, removing 36

F

- Fault attributes
 - clearing 109
- field delimiters, setting 41, 83
- field names
 - disallowed 58, 60
- field names, disallowed 187
- file format, setting 40, 83
- fixed-width file format 40, 83
- flatfile
 - database considerations 49
 - used as a "virtual database" 49

G

- Group-By
 - in queries 60

I

- input runtime arguments for queries 57

- instance data
 - controlling the display of 185
- instance monitoring
 - Activity Details 186
 - Business Process Instance Attributes 186
 - Choose Preferences 185
 - Filter Instances 186
 - Hide Business Process 185
 - Hide Instance List 185
 - Show Business Process 185
 - Show Instance List 185
 - Start 185
 - Stop 186
- ISO Images
 - file names 28

J

- Java CAPS
 - and eBAM Studio 22
 - prerequisites 24

K

- Key Performance Indicators 13, 21

L

- Labels (bar chart properties) 174
- logging 186
- logs
 - setting levels 186

M

- metadata
 - validating 85
- metadata definition
 - validating 39
- meters
 - described 172
 - properties 174
 - Critical Range Color 175, 177
 - Dial Type 175
 - Needle Color 175
 - Normal Range Color 175, 177
 - Warning Range Color 175, 177
- modifying queries 57
- monitoring
 - options 185
 - setting up 184
- monitoring tools for business processes 185

N

- navigation links
 - (support for) 16
- Needle Color
 - meters property 175
- new features in this release 14
- Normal Range Color
 - meters property 175, 177
- notification by email 69
- notify web service 32

O

- Operator palette (illustrated) 64
- Oracle
 - (support for) 15
 - alternative to 49
 - configuring 50
 - database considerations 50
 - eWay 15
 - external system 51, 52
 - service 15, 53
- Order-By
 - in queries 60
- Orientation
 - bar chart property 174
- overview
 - of eBAM Studio 13

P

- part numbers
 - DVD 28
- patches for Portal Server 131
- patches for Sun Java System Application Server 131
- pie charts
 - described 172
 - properties 173, 174
 - 3D 173
 - Depth Factor 173
- Portal Server
 - (support for) 16
 - installation
 - information 132
 - installation, before 131
 - release notes 132
 - versions, differences 131
 - reference documentation
 - overview 130
 - Web site 130
 - running applications with version 6 2005Q4
 - Access Manager Console 139, 143, 149
 - Admin Console (application server) 135

- before you start 132
- creating a new channel 143
- creating new Portal Server users 139
- deploying applications 135
- installing to run in server on Windows 133
- introduction 132
- new channel into Portal Desktop tab 149
- Portal Desktop 145, 147, 149, 155
- procedural overview 133
- starting application server on Windows 134
- supported operating systems 133
- system requirements 132
- viewing a new channel 154
- running applications with version 7
 - before you start 155, 163
 - creating JSR 168 portlet channel 162, 170
 - deploying applications 162, 170
 - IFrame channel, setting up 156, 164
 - installation 156, 164
 - introduction 155, 163
 - JSR 168 portlet channels, setting up 161, 169
 - Portal Desktop 160, 161, 168, 169
 - Portal Server 7 Console 156, 164
 - registering JSR 168 applications 162, 170
 - supported operating systems 156, 164
 - system requirements 156, 164
 - unregistering JSR 168 applications 163, 171
- prerequisites
 - Java CAPS Products 24

Q

- queries 21
 - configuring 59
 - configuring conditions on 58
 - creating 56
 - run-time input arguments for 57
 - run-time inputs for 57
- query-editing canvas
 - (illustrated) 56
 - Group-By and Order-By 60
 - mapping operators 59
 - tool palettes 57

R

- record delimiters, setting 41, 83
- removing expired data 36
- Repository
 - uploading eBAM to 25
- reserved words 58, 60, 187
- runtime arguments for eBAM queries 57

S

- sample data
 - committing 85
 - validating 38, 83
- screenshots 18
- sending email for alert conditions 68
- SQL Operator palette (illustrated) 64
- SQL reserved words 58, 60, 187
- SQL types 41
- Sun Java System Application Server 8.2 EE
 - (support for) 15
- Sun Java System Portal Server
 - (support for) 16
- Sun Java System Portal Server, overview 129
- supporting documents 18

- (support for) 15
- what's new in this release 14

T

- text conventions 18
- timestamp 41
- Title
 - chart property 173
- tool palettes
 - Alert Editor 63
 - SQL Operations (illustrated) 64
- trafficlights
 - described 172

U

- upsert web service 32, 45

V

- validating
 - data definitions 38, 83
 - metadata 85
- version control
 - (support for) 16
- virtual database
 - alternative to Oracle 49

W

- Warning Range Color
 - meters property 175, 177
- web services
 - add 32, 44
 - delete 32, 47
 - execute 32
 - notify 32
 - upsert 32, 45
- WebLogic