

SeeBeyond ICAN Suite

eBAM Studio User's Guide

Release 5.0.2



The information contained in this document is subject to change and is updated periodically to reflect changes to the applicable software. Although every effort has been made to ensure the accuracy of this document, SeeBeyond Technology Corporation (SeeBeyond) assumes no responsibility for any errors that may appear herein. The software described in this document is furnished under a License Agreement and may be used or copied only in accordance with the terms of such License Agreement. Printing, copying, or reproducing this document in any fashion is prohibited except in accordance with the License Agreement. The contents of this document are designated as being confidential and proprietary; are considered to be trade secrets of SeeBeyond; and may be used only in accordance with the License Agreement, as protected and enforceable by law. SeeBeyond assumes no responsibility for the use or reliability of its software on platforms that are not supported by SeeBeyond.

SeeBeyond, e*Gate, e*Way, and e*Xchange are the registered trademarks of SeeBeyond Technology Corporation in the United States and/or select foreign countries. The SeeBeyond logo, SeeBeyond Integrated Composite Application Network Suite, eGate, eWay, eInsight, eVision, eXchange, eView, eIndex, eTL, ePortal, eBAM, and e*Insight are trademarks of SeeBeyond Technology Corporation. The absence of a trademark from this list does not constitute a waiver of SeeBeyond Technology Corporation's intellectual property rights concerning that trademark. This document may contain references to other company, brand, and product names. These company, brand, and product names are used herein for identification purposes only and may be the trademarks of their respective owners.

© 2004 by SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

This work is confidential and proprietary information of SeeBeyond and must be maintained in strict confidence.

Version 20040528143144.

Contents

List of Figures	5
-----------------	---

List of Tables	7
----------------	---

Chapter 1

Introduction	8
Document Purpose and Scope	8
Organization of This Book	8
Intended Audience	9
Writing Conventions	9
Additional Conventions	9
Online Documentation	10
The SeeBeyond Web Site	10
Additions and Changes in This Release	10

Chapter 2

About eBAM Studio	11
Architectural Overview	11
eBAM Architecture in Conjunction With ICAN	12
eBAM Data Flow	12
eBAM Means Business Activity Monitoring	13
What Is Inside an eBAM Application?	13
What Is In a KPI?	13

Chapter 3

Installing eBAM	14
System Requirements	14
Platform Support	14
Prerequisite Products	14
Installation Steps	15

Uploading eBAM to the Repository	15
Updating Enterprise Designer with eBAM	17

Chapter 4

Using eBAM Studio	19
eBAM Applications	19
Components of an eBAM Application	20
Setting Up an eBAM Application’s Data Definition	20
Alert Conditions, Charts, and Queries	25
Setting Up Data Conditions for an Alert, Chart, or Query	25
Using the Condition Editor	26
Setting Up the Application’s Alert Conditions	27
About Charts	33
Setting Up the Application’s Charts	37
About Queries	41
Setting Up the Application’s Queries	42

Chapter 5

Installing and Running the eBAM Sample	46
Last Things First	46
Overview of Steps for Setting Up the Sample Implementation	47
Initial Setup Steps	47
Installing the Sample Files	47
Importing the Sample Project	48
Creating the Sample Environment	48
Starting the Logical Host for the Sample	49
Design, Deployment, and Runtime Steps	49
Validating the OTDs	50
Using the eBAM Application to Preview Sample Data	50
Configuring the eWays	54
Activating and Running the Project	55
Monitoring Key Performance Indicators with Live Data	57
Overview of the InputUpdate Business Process	57
Adding New Data: Exercising the InputUpdate Process, Path A	58
Updating Existing Data: Exercising InputUpdate Paths B and C	60
Executing Stored Queries: Exercising the QuerySales Process	62
Notifying and Alerts: Exercising the AlertUnitSales	63
SQL Reserved Words	64
Index	71

List of Figures

Figure 1	eBAM's Relationship to the ICAN Suite	11
Figure 2	eBAM's Stacked Layers for Task Differentiation	12
Figure 3	Enterprise Manager ADMIN Page	15
Figure 4	DOCUMENTATION Page: Downloading eBAM Sample Files	16
Figure 5	Update Center Wizard: Select Modules to Install	17
Figure 6	Update Center Wizard: Progress Bars	18
Figure 7	Update Center Wizard: Restart Enterprise Designer	18
Figure 8	Setting Up a Data Definition	21
Figure 9	eBAM Editor Showing Completed Data Definition	22
Figure 10	Show Sample: Preview of Field Layout	24
Figure 11	Sample Data Displayed in eBAM Editor	24
Figure 12	Basic Properties of a Data Definition	26
Figure 13	Newly Created Alert Condition	28
Figure 14	Operator Palette for SQL Operations	30
Figure 15	Setting Properties for the Entire Alert Condition	31
Figure 16	Selecting Keys in the Alert Properties Dialog	31
Figure 17	Chart Editor	38
Figure 18	Configuring a Category Dataset Using Operators and Data Definition Fields	39
Figure 19	Chart Properties: "Available Group-By" and "Selected Group-By" Columns	39
Figure 20	Chart Properties: "Available Order-By" and "Selected Order-By" Columns	40
Figure 21	Previewing a Chart	41
Figure 22	Query Editor	43
Figure 23	Configuring a Query's DataSetView Using Operators and Data Definition Fields	43
Figure 24	Query Properties: "Available Group-By" and "Selected Group-By" Columns	44
Figure 25	Query Properties: "Available Order-By" and "Selected Order-By" Columns	44
Figure 26	Adding Input Runtime Arguments to eBAM Queries	44
Figure 27	Bar Chart for Sample	46
Figure 28	Sample Chart barTotalSalesByManufacturer, With Modified Chart Properties	52
Figure 29	Sample Chart pieMarketShare2003, With Modified Chart Properties	54
Figure 30	Connectivity Map for Sample	54
Figure 31	Components from SampleBamProject Assigned to Servers in SampleBamEnv	56

List of Figures

Figure 32	Initial eBAM Charts Viewer (No Data)	57
Figure 33	InputUpdate Business Process	57
Figure 34	Stacked Bar Chart Showing Car Sales Grouped by Manufacturer and Model	58
Figure 35	Pie Chart Showing Comparative Market Share for Model Year 2003	59
Figure 36	Bar and Pie Charts With Added Data for One Category	59
Figure 37	Bar Chart With Added Data for a New Series	60
Figure 38	Bar Chart Reflecting Revision to Complete Data Record	61
Figure 39	Bar Chart Reflecting Revision to Multiple Partial Records	61
Figure 40	QuerySales Business Process	62
Figure 41	AlertUnitSales Business Process	63

List of Tables

Table 1	Writing Conventions	9
Table 2	Metadata for Creating a Data Definition	20
Table 3	Specifying Formatting Type and Encoding for Imported Sample Data	23
Table 4	Parsing Information for Imported Sample Delimited Data	23
Table 5	Parsing Information for Imported Sample Fixed-Width Data	23
Table 6	Tools Provided in the SQL Code Canvas of the Condition Editor	27
Table 7	Tools Provided in the Graphical Canvas of the Condition Editor	27
Table 8	Tools Provided in the Tool Palette of the Alert Editor	29
Table 9	Alert Properties for e-Mail	32
Table 10	Properties Common to All Three Chart Types	33
Table 11	Properties Common to Pie and Bar Charts	34
Table 12	Properties Specific to Pie Charts	34
Table 14	Properties Specific to Meters	35
Table 13	Properties Specific to Bar Charts	35
Table 15	Properties Specific to Trafficlights	36
Table 16	OTDs Included With Sample	50
Table 17	Data Definition (Metadata) for the Sample eBAM Application	51
Table 18	SQL Reserved Words	64

Introduction

BAM (Business Activity Monitoring) involves the collection, aggregation, and presentation of business activity data according to specified Key Performance Indicators (KPIs). eBAM Studio (eBAM) provides the tools for generating custom cross-application graphical dashboards for defining and monitoring KPIs that summarize the aggregated business data collected through the eInsight or eGate application layers. The eBAM Web interface allows business analysts to transform data that has been collected over time into meaningful, rich visual presentations.

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 the business analyst with different views of performance data, enabling the timely identification of business trends.

eBAM Studio 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 the business analyst to quickly translate information into action.

1.0.1 Document Purpose and Scope

The *eBAM Studio User's Guide* explains how to use eBAM 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.

1.1 Organization of This Book

The *eBAM Studio User's Guide* includes the following information:

- An overview of eBAM's application architecture.
- Prerequisites and installation instructions.
- An overview of eBAM's major user interfaces.
- A detailed description of eBAM's features, with step-by-step instructions to guide you through their setup and configuration.
- How to use eBAM, step-by-step, in a sample implementation that collects run-time data, analyzes it, and displays Key Performance Indicators as specified in the setup.
- A listing of special reserved words that should not be used as field names.

1.2 Intended Audience

This guide is intended for experienced computer users who have the responsibility of helping to set up and maintain a fully functioning ICAN Suite system. This person must also understand any operating systems on which eGate will be installed (Windows or UNIX) and must be thoroughly familiar with Web browsers and Windows-style GUI operations.

1.3 Writing Conventions

The following writing conventions are observed throughout this document.

Table 1 Writing Conventions

Text	Convention	Example
Button, file, icon, parameter, variable, method, menu, and object names.	Bold text	<ul style="list-style-type: none"> ▪ Click OK to save and close. ▪ From the File menu, select Exit. ▪ Select the logicalhost.exe file. ▪ Enter the timeout value. ▪ Use the getClassname() method. ▪ Configure the Inbound File eWay.
Command line arguments and code samples	Fixed font. Variables are shown in <i>bold italic</i> .	<code>bootstrap -p <i>password</i></code>
Hypertext links	Blue text	For more information, see "Writing Conventions" on page 9 .

Additional Conventions

Windows Systems

For the purposes of this guide, references to "Windows" will apply to Microsoft Windows Server 2003, Windows XP, and Windows 2000.

Path Name Separator

This guide uses the backslash ("\`) as the separator within path names. If you are working on a UNIX system, please make the appropriate substitutions.`

1.4 Online Documentation

The documentation for the SeeBeyond ICAN Suite is distributed as a collection of online documents. These documents are viewable with the Acrobat Reader application from Adobe Systems. Acrobat Reader can be downloaded from:

<http://www.adobe.com>

When downloading Acrobat Reader, make sure to download the version that includes the option for searching **.pdf** files—Acrobat Reader with Search. This version is required to view the searchable master index.

1.5 The SeeBeyond Web Site

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

<http://www.seebeyond.com>

1.6 Additions and Changes in This Release

Since eBAM 5.0 (released in December 2004 with ICAN 5.0.2), the following features have been added or modified:

- (5.0.1) eBAM Applications now allow you to specify data retention.
- (5.0.1) Alert conditions now allow you to send an e-mail notification.
- (5.0.1) You are now prevented from using SQL keywords as fieldnames.
- (5.0.2) Bar and pie charts now support grouping/subgrouping (by category/series).
- (5.0.2) Two chart types have been added: Trafficlights; and (for category/series) stackable bar charts.
- (5.0.2) A new eBAM application service (**query**) allows the eBAM engine to be queried from within an eInsight business process or an eVision page flow.
- (5.0.2) A new eBAM application service (**upsert**) allows the eBAM engine to capture state change and update data that has already been collected.
- (5.0.2) The User's Guide has been updated to reflect all these changes.

About eBAM Studio

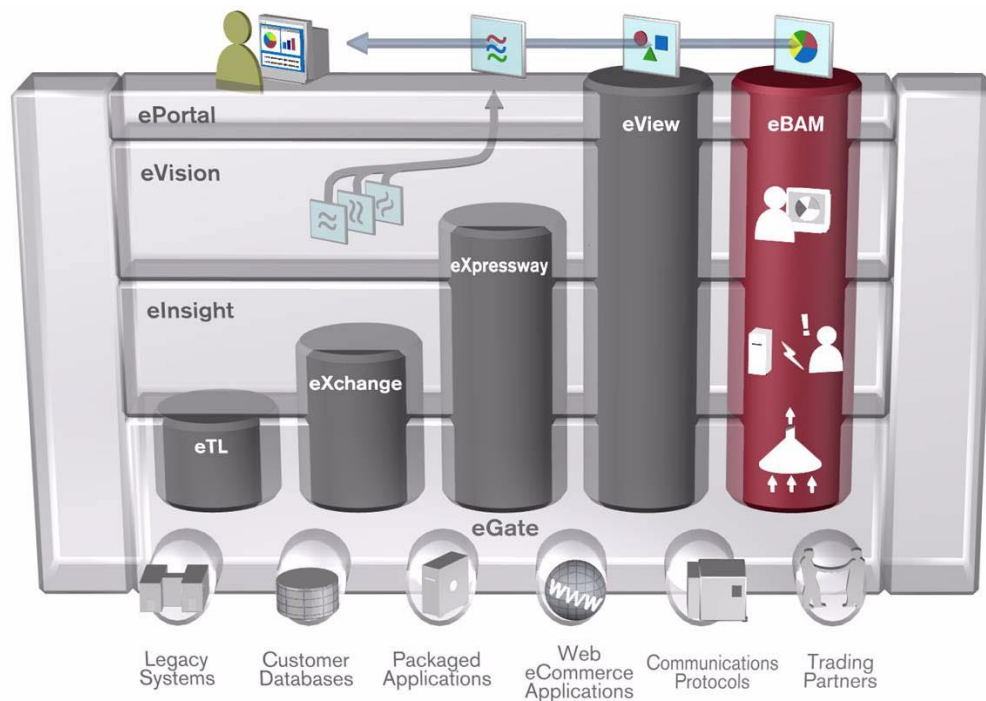
eBAM Studio enables the creation of business activity monitoring applications that intercept the flow of data through ICAN Suite components to produce visual presentations of data analysis based on Key Performance Indicators (KPIs) and alerts.

This chapter provides overviews of eBAM's architecture and overall approach to Business Activity Monitoring.

2.1 Architectural Overview

eBAM takes advantage of the overall J2EE architecture of the ICAN Suite to work hand-in-glove with other ICAN products. As Figure 1 suggests, eBAM can take advantage of eGate and eInsight to leverage your investment in your other ICAN products, such as eVision and ePortal.

Figure 1 eBAM's Relationship to the ICAN Suite



2.1.1 eBAM Architecture in Conjunction With ICAN

Here are some ways that eBAM leverages other ICAN Suite products and features:

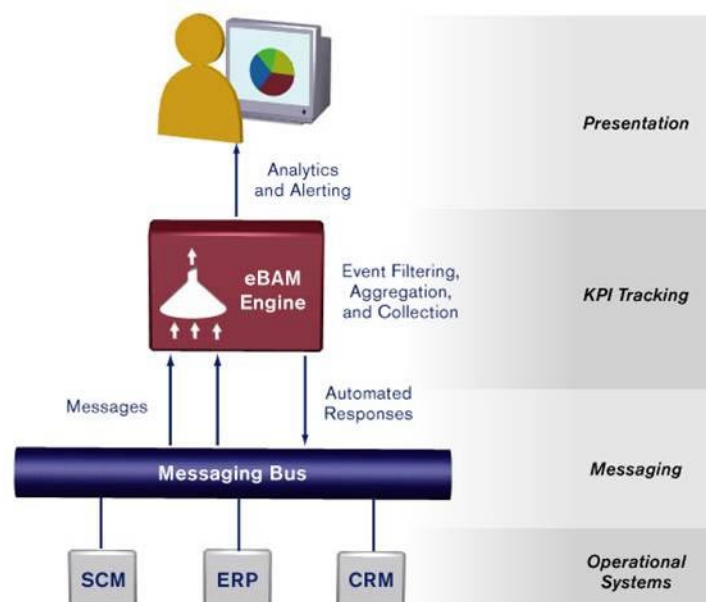
- The Repository stores and controls check-out/check-in and ACL access to eBAM-specific objects such as data definitions, chart configurations, and alert settings.
- Integration Server security provides single-sign-on functionality at design time, and also provides runtime resource management.
- The Enterprise Designer GUI is leveraged for the design of the data collection, graphical display, and notification processes, and the Enterprise Manager GUI allows eBAM to plug in its additional reporting and monitoring functionality.
- The Business Process Modeler is used for alert handling and graphic generation.
- Common deployment experience enabled via ePortal as a natural place for viewing all eBAM applications.

2.1.2 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 eGate's JMS and Collaborations and eInsight's Business Processes.

Figure 2 eBAM's Stacked Layers for Task Differentiation



2.2 eBAM Means Business Activity Monitoring

eBAM takes a simple approach to business activity monitoring: Applications and KPIs.

- Each eBAM *Application* samples the ICAN data traffic and stores results in a way that allows standing queries to update and communicate their findings.
- 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.

2.2.1 What Is Inside an eBAM Application?

An eBAM application requires access to ICAN run-time 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 run-time data from ICAN suite processes.
- User definitions for Key Performance Indicators.
- User definitions for thresholds and conditions—in other words, user-specified conditions that trigger an eBAM response.
- User-defined actions to take for Alerts and Notifications that are triggered when threshold boundaries are crossed.
- 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 ICAN Suite composite applications.

2.2.2 What Is In a KPI?

An eBAM KPI requires you to specify the following attributes:

- Data definitions (the facts that form the basis for the calculation of a KPI value)
- The mathematical formula for calculating the KPI value.
- The dashboard (visual) representation of the KPI.
- The events that are triggered when the value of the KPI crosses a preset threshold.

eBAM provides an intuitive palette of graphical tools for constructing queries of the stored data.

Installing eBAM

This chapter provides the prerequisites and steps for installing eBAM Studio.

3.1 System Requirements

This section lists requirements for operating systems and prerequisite products. The `Readme.txt` file on the product media contains the most up-to-date information on system requirements for each supported platform.

3.1.1. Platform Support

eBAM supports the following operating systems:

- Microsoft Windows Server 2003, Windows XP SP1a, and Windows 2000 SP3 or SP4
- Sun Solaris 8 and 9 with required patches
- IBM AIX 5.1L and 5.2 with required maintenance level patches
- HP-UX 11.0 and 11i (PA-RISC) with required patches and parameter changes
- HP Tru64 V5.1A with required patches
- Red Hat Linux 8 (Intel version) and Linux Advanced Server 2.1 (Intel version)

3.1.2. Prerequisite Products

The prerequisite products for installing eBAM 5.0.2 are: A **Repository** at release **5.0.1** or later, and *one or the other* of the following:

- **eGate Integrator** and **eInsight Business Process Manager**, at release **5.0.1** or later.
- **eInsight Enterprise System Bus (ESB)** at release **5.0.1** or later.

Because of the APIs and GUIs that eBAM requires, `eGate.sar` (or `eInsightESB.sar`) must be uploaded *before* uploading `eBAM.sar`; for details, see the following section.

If you want to run the sample project (provided in `eBAMDocs.sar`) or follow all of the implementation steps provided in [Chapter 6](#), then the **File eWay** at release **5.0.1** or later is also required.

Complete instructions on uploading and downloading `.sar` files and sample files are provided in the following section.

3.2 Installation Steps

The steps for installing eBAM are the same as for other products in the ICAN Suite. You can find general product installation instructions in the *ICAN Suite Installation Guide*, available on the product media and also accessible via Enterprise Manager (Documentation tab).

3.2.1. Uploading eBAM to the Repository

Before you begin

- A Repository server must be running on the machine where you will be uploading the eBAM product files, and a license file (license.sar) and prerequisite product files must have already been uploaded to this Repository.

To upload eBAM product files to the Repository

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

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

where

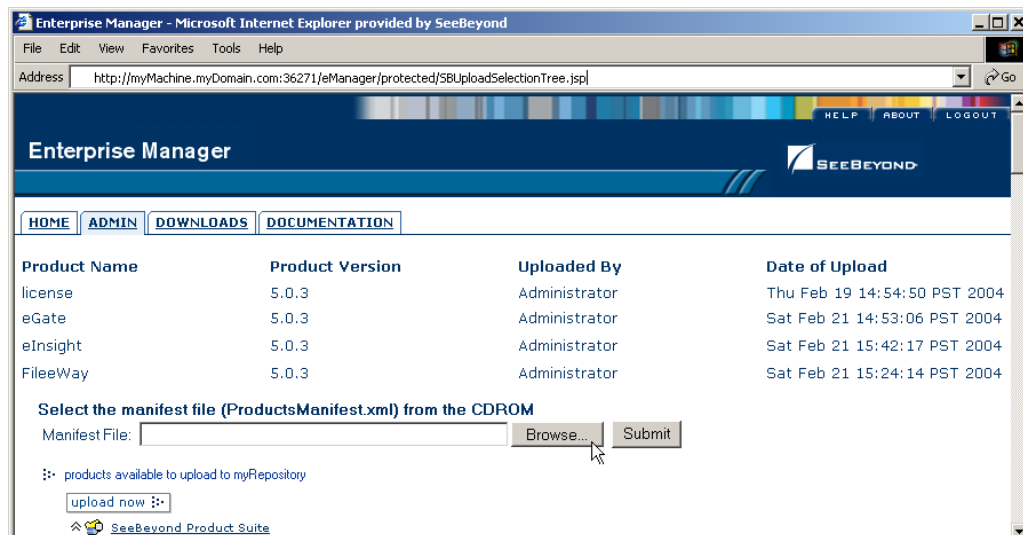
- ♦ `<hostname>` is the name of the machine running the Repository server.
- ♦ `<port>` is the starting port number assigned when the Repository was installed.

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

`http://localhost:12001`
`http://serv1234.company.com:19876`

- 2 On the Enterprise Manager **SeeBeyond Customer Login** page, enter your username and password.
- 3 When Enterprise Manager responds, click the **ADMIN** tab. See Figure 3.

Figure 3 Enterprise Manager ADMIN Page



- 4 In the ADMIN page, click **Browse**.
- 5 In the **Choose file** dialog, click **ProductsManifest.xml**, and then click **Open**.
- 6 In the ADMIN page, click **Submit**.

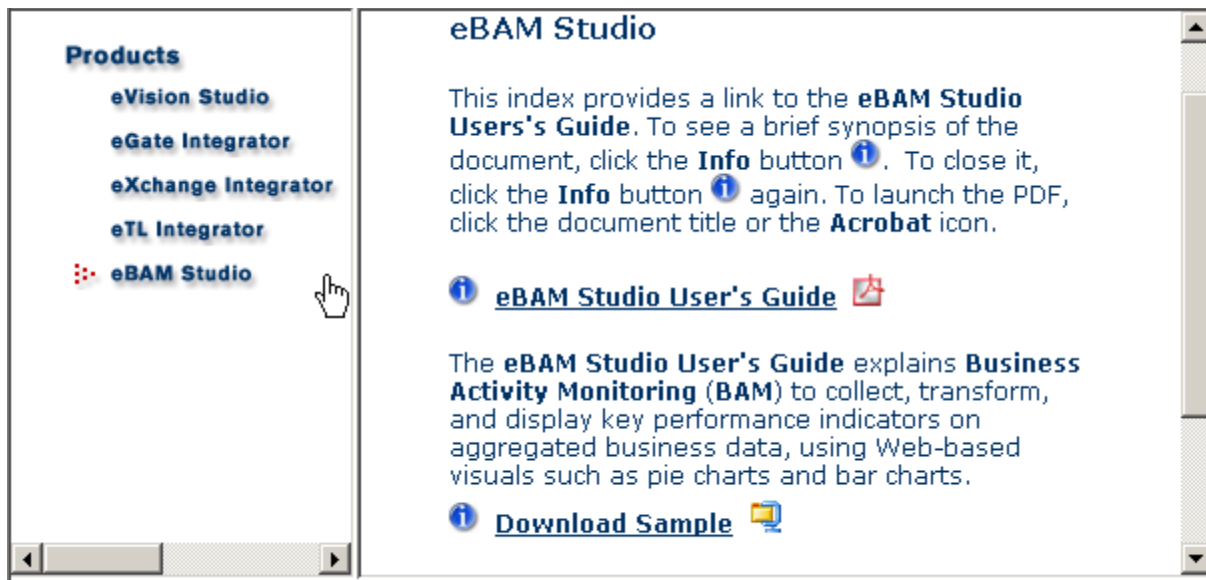
After the manifest uploads, the lower half of the ADMIN page lists the product files you are licensed to upload to this Repository.

- 7 In the Products column, find **eBAM**, and then click the **Browse** button for it.
- 8 In the **Choose file** dialog, click **eBAM.sar**, and then click **Open**.
- 9 Repeat the previous two steps for **eBAMDocs.sar** and (if you haven't already uploaded it, and if you will be doing the sample implementation described in **Chapter 6) FileeWay.sar**.

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.

- 10 In the ADMIN page, click the |upload now ::| button.
- 11 After the files upload, in Enterprise Manager, click the DOCUMENTATION tab.
- 12 In the DOCUMENTATION page, under **Products** (on the lower left), click **eBAM Studio**; then, under eBAM Studio (on the lower right), click **Download Sample**. See Figure 4.

Figure 4 DOCUMENTATION Page: Downloading eBAM Sample Files



Result: Your repository can now serve the files in eBAM.sar to any Enterprise Designer that connects to it and uses the Update Center.

3.2.2. Updating Enterprise Designer with eBAM

Before you begin

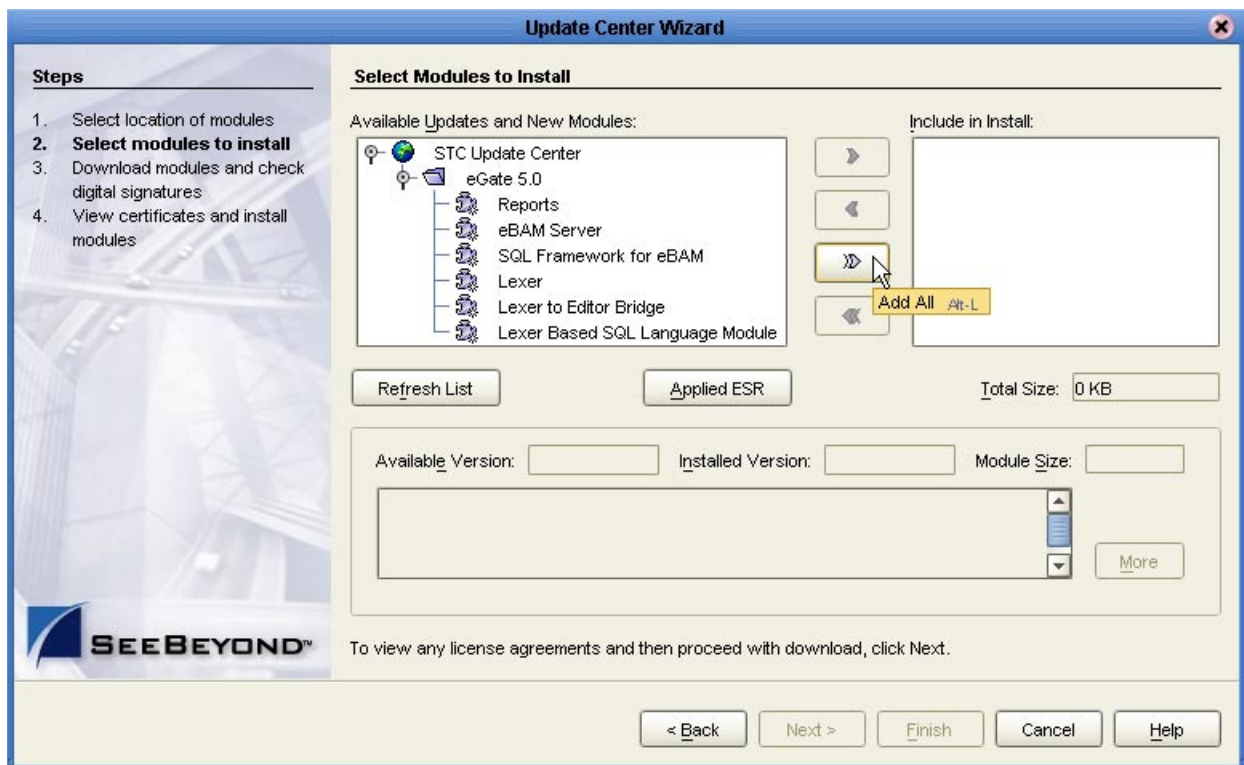
- You must have already downloaded and installed Enterprise Designer.
- A Repository server must be running on the machine where you uploaded the eBAM product files.

To refresh an existing installation of Enterprise Designer

- 1 Start Enterprise Designer.
- 2 On the **Tools** menu, click **Update Center**.

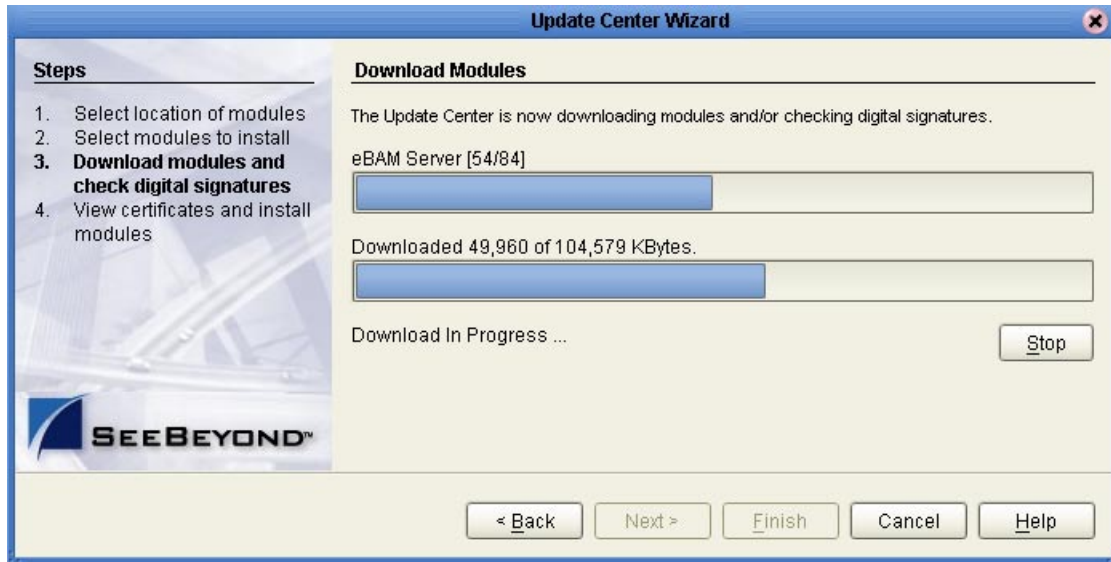
The Update Center shows a list of components ready for updating. See Figure 5.

Figure 5 Update Center Wizard: Select Modules to Install



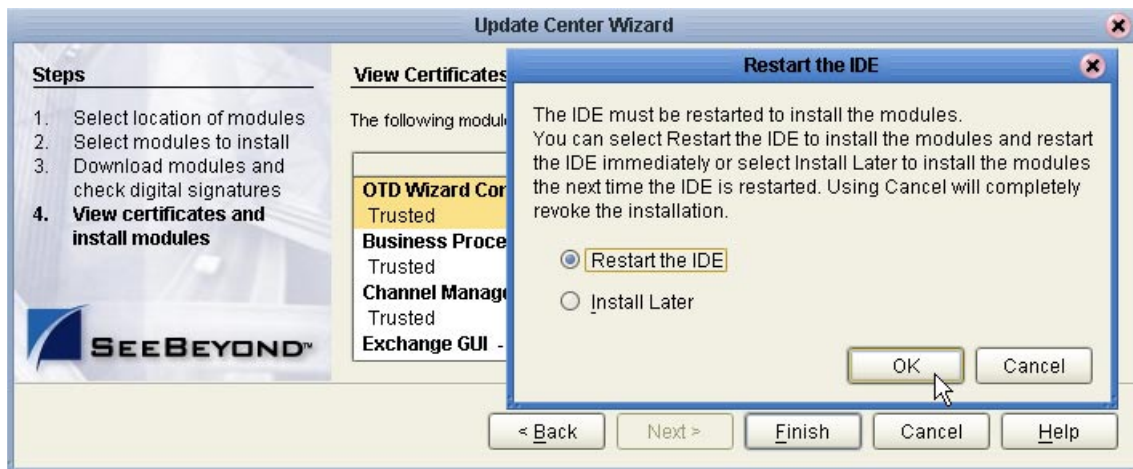
- 3 Click **Add All** (the button with a doubled chevron pointing to the right).
All modules move from the Available/New pane to the **Include in Install** pane.
- 4 Click **Next** and, in the next window, click **Accept** to accept the license agreement.
The wizard shows you the progress of the download. See Figure 6.

Figure 6 Update Center Wizard: Progress Bars



- 5 When the progress bars indicate the download has ended, click **Next**.
- 6 Review the certificates and installed modules, and then click **Finish**.
- 7 When prompted to restart Enterprise Designer, click **OK**. See Figure 7.

Figure 7 Update Center Wizard: Restart Enterprise Designer



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

Using eBAM Studio

This chapter provides information on the eBAM features and step-by-step instructions for using them.

The instructions in this chapter assume you are already familiar with the eGate repository and logical host, and therefore focuses on the concepts and design-time GUI operations specific to eBAM.

4.1 eBAM Applications

When you create an eBAM application, here are the important things 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 [subset of the] data to pass to the eBAM application, and create names for the data elements of interest. Data passes into eBAM via the **add** activity (generated from the data definition).
- *Filtering by condition*—For alerts, and optionally for charts, you set up one or more conditions that must be met for data to be eligible for further processing by eBAM. This is done using the Condition Editor.
- *What to communicate*—For alerts, typically the only item to communicate is simply that the triggering condition was met. For charts, you calculate a metric such as a key performance indicator (KPI), using the Chart Editor to identify one or more fields and how to manipulate them to create the metric you want to communicate.
- *How to communicate*—Alerts and charts communicate in different ways. With alerts, data can be communicated via an e-mail message you design, and/or via the **notify** activity (generated from the alert condition) placed in an eInsight business process. With charts, data is communicated via KPIs. You select a chart type (pie, bar, etc.) 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.

Components of an eBAM Application

Each eBAM Application consists of exactly one data definition, zero or more charts, and zero or more alert conditions. Charts and alerts are based on data definitions.

- The *data definition* is the infrastructure of the application. In it, you specify which data you are interested in, and how to label and organize the data. For step-by-step procedures, see [“Setting Up an eBAM Application’s Data Definition” on page 20](#).
- *Alert conditions* are triggered whenever a condition is met by one of the data items, such as exceeding a threshold. For step-by-step procedures, see [“Setting Up the Application’s Alert Conditions” on page 27](#).
- *Charts* provide real-time feedback on the current data set according to conditions you set up. For further information and step-by-step procedures, see [“About Charts” on page 33](#) and [“Setting Up the Application’s Charts” on page 37](#).
- *Queries* allow you to %%%. For further information and step-by-step procedures, see [“About Charts” on page 33](#) and [“Setting Up the Application’s Charts” on page 37](#).

4.2 Setting Up an eBAM Application’s Data Definition

Before you begin: You must have a clear idea of the data you want to gather before you create the application. You should know the answers to the following questions:

- 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 name? (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.)
- For each data element, what should be the default value? (If not specified, the default default is a null value of the appropriate type.)
- How many hours, days, months, or years should a dataset be retained?

In other words, be prepared with information whose form is similar to that of Table 2:

Table 2 Metadata for Creating a Data Definition

	Name	Default Value	Data Type
1	OrderNum	999999999	integer
2	DateOfOrder	2099-12-31 23:59:59	timestamp
3	CustName1	!! Customer Name Must Be Supplied !!	varchar
(...)	(...)	(...)	(...)
<i>n</i>	TotalDollarsThisOrder		float

Tip: Data definitions, once created, cannot be modified. Therefore, when you reach the end of the wizard, be sure to review your definition carefully before clicking Finish.

To create a new eBAM Application and specify its data definition

- 1 In Enterprise Designer, in the project tree (left side), right-click the project and, on the popup context menu, point at New and click **eBAM Application**.

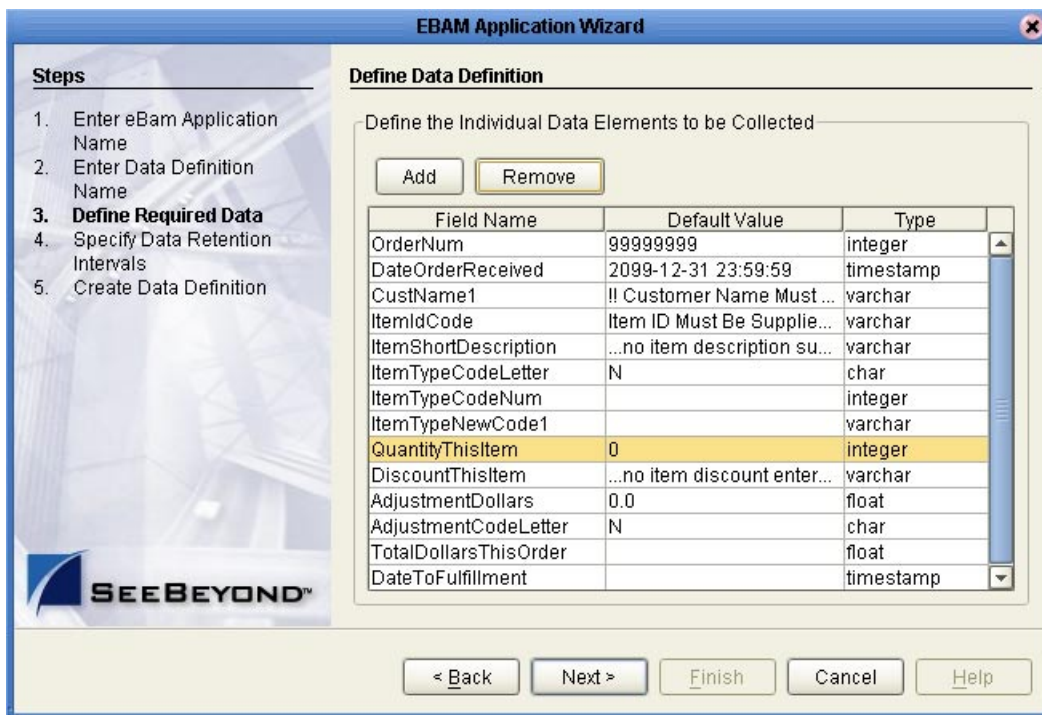
The eBAM Application wizard opens, prompting you to name the application.

- 2 In step 1 of the wizard, type in a name for application, and then click **Next**.
- 3 In step 2 of the wizard, type in a name for the data definition, and then click **Next**.
- 4 In step 3 of the wizard, click **Add** several times, and then supply names, default values, and data types for the data elements.

Tip: If the order of the elements is important to you, double-check frequently to ensure you have not omitted a row—clicking **Add** appends each new row to the end. (However, order is unimportant to the queries, and some of the GUIs alphabetize by field name.)


- 5 As needed, click **Add** to append additional rows or click **Remove** to remove unneeded rows. When done, click **Next**. See Figure 8.

Figure 8 Setting Up a Data Definition



- 6 In the Data Retention step, specify how long data is allowed to age before it expires, and how often to purge expired data from the database.

For example: To keep data for a week, enter [Retain Last:] **7** and **Days**; to purge expired data every five minutes, enter [Data Cleanup Schedule:] **5** and **Minutes**.

- 7 In the completed data definition, review all values carefully before continuing. As needed, click Back and Next to return to a step and make changes.
- 8 Click **Finish** only after you are completely satisfied with the data definition. *After you click Finish, you will be unable to make further changes to the metadata definition (although you can use  Show Properties to adjust data retention parameters).*


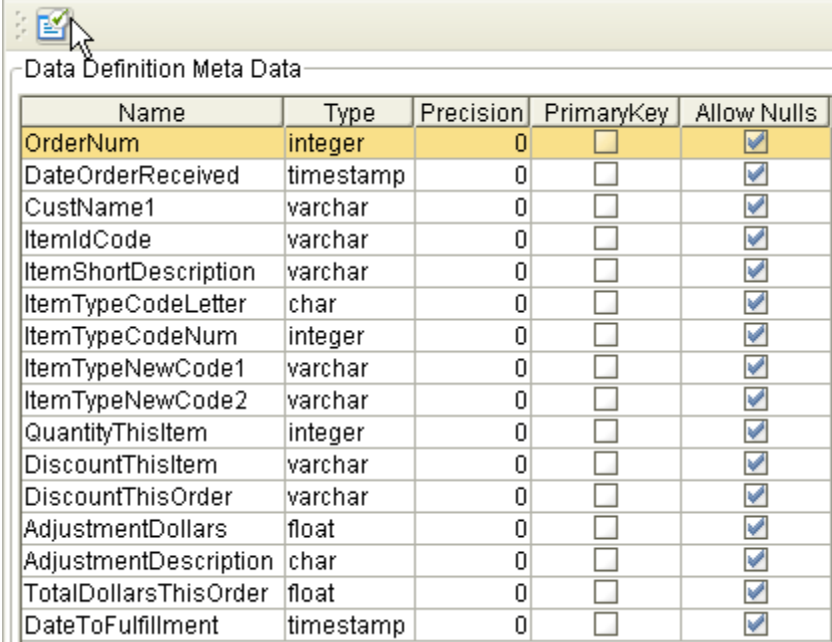

Result: On the left side, the project tree displays an  eBAM application. On the right side (the canvas), the eBAM Editor opens to display the data definition; see Figure 9.



Figure 9 eBAM Editor Showing Completed Data Definition



Name	Type	Precision	PrimaryKey	Allow Nulls
OrderNum	integer	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DateOrderReceived	timestamp	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
CustName1	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemIdCode	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemShortDescription	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemTypeCodeLetter	char	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemTypeCodeNum	integer	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemTypeNewCode1	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
ItemTypeNewCode2	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
QuantityThisItem	integer	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DiscountThisItem	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DiscountThisOrder	varchar	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
AdjustmentDollars	float	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
AdjustmentDescription	char	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
TotalDollarsThisOrder	float	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>
DateToFulfillment	timestamp	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Tip: 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.

To validate the metadata definition

- 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 file and select it.

4 In the formatting step, make appropriate choices (see Table 3), and then click Next.

Table 3 Specifying Formatting Type and Encoding for Imported Sample Data

Item	Choices	Notes
Encoding scheme	ASCII (ISO646-US) UTF-8 EBCDIC: USA-Canada (cp037)	ASCII – 7-bit encoding, roman characters. UTF-8 – 8-bit encoding of Unicode. EBCDIC – for certain mainframe systems
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 appropriate choices for delimited data (see Table 4) or fixed-width data (see Table 5), and then click Next.

Table 4 Parsing Information for Imported Sample Delimited Data

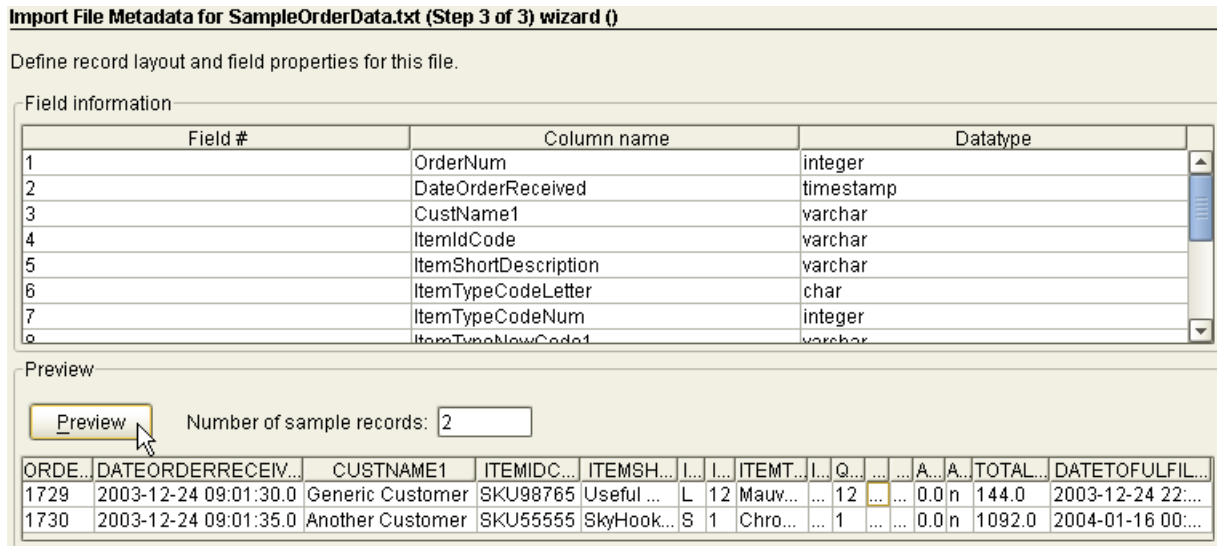
Item	Value	Notes
Default SQL Type	Any of the following: bigint, bit, char, date, decimal, double, float, integer, longvarchar, numeric, real, smallint, time, timestamp, tinyint, varchar	
Record Delimiter	{newline (lf)} {cr} {cr-lf}	newline – each new line starts a new record. cr – each carriage-return starts a new record. cr-lf – each carriage-return+linefeed starts a new record.
	You can type also type in a character or control character; for example, to specify TAB, type in: \t	
Field	{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 quotemarks; only doublequotes or singlequotes are supported.
First line contains field names?	False True	You can indicate whether the first record of the data consists only of labels for the fields, rather than actual data.

Table 5 Parsing Information for Imported Sample Fixed-Width Data

Item	Value	Notes
Default SQL Type	Any of the following: bigint, bit, char, date, decimal, double, float, integer, longvarchar, numeric, real, smallint, time, timestamp, tinyint, varchar	
Record Length	0	To override the default, type in a nonzero number.
HeaderBytesOffset	0	To override the default, type in a nonzero number.
Field Count	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 10, or [Figure 47 on page 74](#).

Figure 10 Show Sample: Preview of Field Layout

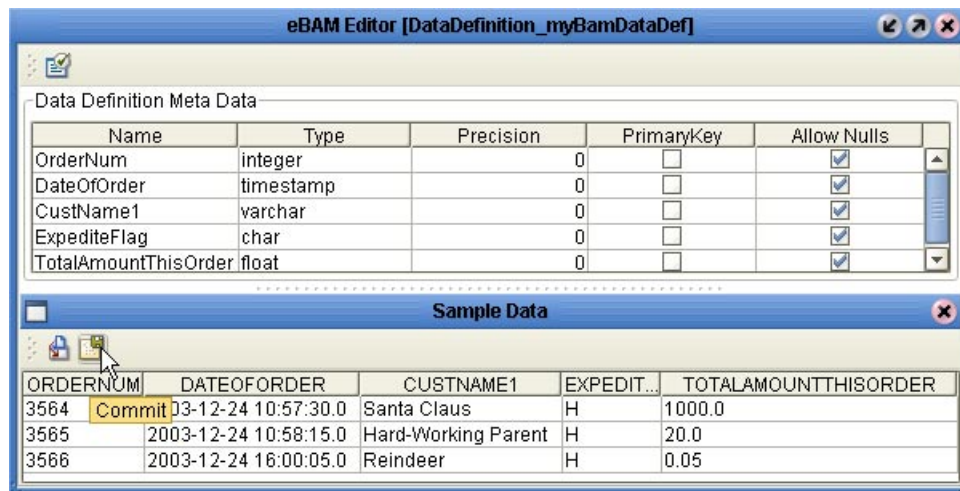


If the sample data is valid (that is, in accord with the layout, encoding, formatting, and properties you specified), the Preview pane displays a table whose rows are the first few records, with each column headed by the data element name.

- 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**.

Result: The entire data set is parsed and displayed in the Sample Data pane. For examples, see [Figure 48 on page 75](#) and Figure 11 below.

Figure 11 Sample Data Displayed in eBAM Editor




- Optionally, if you want to commit this data set to being stored, click **Commit**.

Tip: 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 the data definition has been created, populated, and validated, it is available for use in one or more alert conditions and charts.

4.3 Alert Conditions, Charts, and Queries

Each  alert condition,  chart, or  eBAM query requires:

- A name and an associated data definition. This is set when you run the wizard that creates the alert condition or chart.
- Zero or more conditions set on fields in the data definition instance.
 - ♦ For step-by-step instructions on setting up conditions, see [“Setting Up Data Conditions for an Alert, Chart, or Query” on page 25](#) below, and [“Using the Condition Editor” on page 26](#).
- One or more fields in the dataset view, with appropriate mappings to them from operators and data definition fields.
 - ♦ For instructions on setting up dataset views and mappings for alert conditions, see the [procedure on page 29](#).
 - ♦ For charts, see the [procedure on page 38](#).
- Property settings for the alert, chart, or query as a whole, such as e-mail address, resend frequency, or chart display parameters
 - ♦ For instructions on configuring properties for an alert condition, see the [procedure on page 31](#).
 - ♦ For chart properties, see [Table 10 on page 33](#) through [Table 14 on page 35](#), as well as the [procedure on page 39](#).

4.3.1 Setting Up Data Conditions for an Alert, Chart, or Query

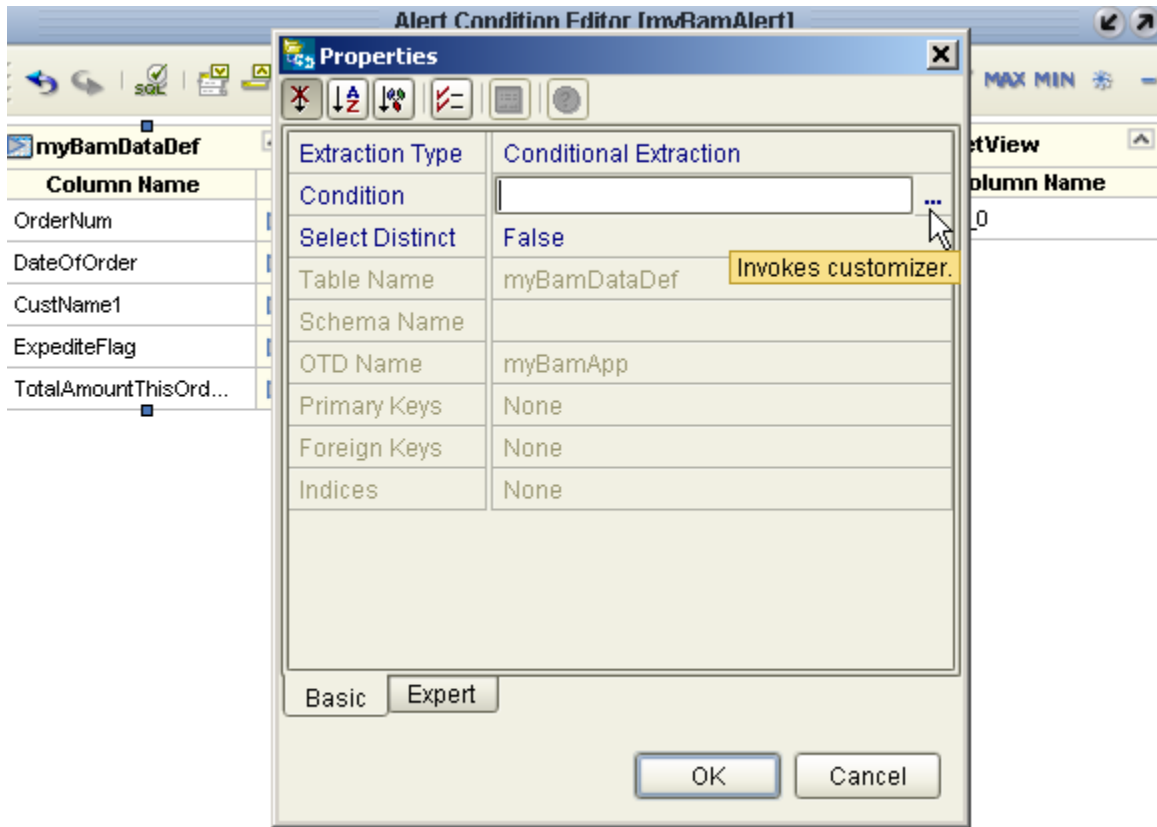
An alert usually depends (and chart or query can depend) on a condition that has been set on its data definition instance. The condition is a property of the data definition instance that appears in the alert, chart, or query; the **Condition** property is set using the **Condition Editor**.

To view or modify a condition on a data definition instance

- 1 In the Alert Editor, Chart Editor, or Query Editor (discussed in detail later), right-click the data definition instance on the left and, on the popup context menu, click **Properties**.

The **Properties** dialog opens, with the **Basic** tab displayed. See Figure 12.

Figure 12 Basic Properties of a Data Definition

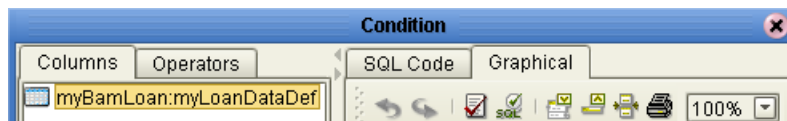


- 2 In the Properties dialog, click **Condition**, and then click the ellipsis [...] to open the **Condition** editor.

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 tree listings (Columns and Operators) and between the two user modes (SQL Code and Graphical).



- The **Columns** tree, which opens via double-click, lists all data elements in the data definition. In both SQL Code and Graphical mode, you drag elements from this tree onto the canvas, after which you specify operations to be performed on them.
- The **Operators** tree lists all SQL operations, providing a quick language reference for SQL Code mode (although you can also drag-and-drop) and providing tools to be dragged onto the Graphical canvas. For any operator placed on the canvas, you can hover your cursor over the title or any field for help on the operator or field.









- The **SQL Code** canvas allows you to enter native SQL commands and/or to drag elements or operators from the trees on the left. Its tool palette provides two tools:


Table 6 Tools Provided in the SQL Code Canvas of the Condition Editor

	Validate checks the syntactical validity of the SQL code listing and reports success or any errors found.
---	--

- 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:

Table 7 Tools Provided in the Graphical Canvas of the Condition Editor

	Undo and Redo allow you travel backward and forward through the sequence of modifications you have made to the graphical canvas.
	
	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. In the Output pane, you can specify which “flavor” of SQL to display, of: Oracle8, Oracle9, SQL Server, internal, or ANSI92.
	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.
	Print Graph allows you to print the graph, using various scaling options.

When the chart editor or alert editor displays a data definition that has a condition, a small “filter” icon——appears to the right of the data element(s) being watched.

4.3.2 Setting Up the Application’s Alert Conditions

eBAM allows you to set up tests whereby an alert (notification) is triggered whenever a condition is met. This 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. You use the Condition editor, discussed earlier, to create queries to detect the condition. In addition to conditions on the data definition, you also set up a dataset view.

An eBAM application can contain many alert conditions, or only one, or none at all. Each alert condition requires the following:

- *Data definition* and name: See [To create a new alert condition](#) on page 28.
- *Filtering by condition*: See [“To configure a condition on a data definition” on page 28](#).

- *What to communicate:* One or more fields in the dataset view, with appropriate mappings to them from operators and data definition fields; see [“To configure the dataset view, mapping operators and data definition fields to its fields” on page 29.](#)
- *How to communicate:* Settings for the alert as a whole, such as resend frequency; see [“To configure properties of the entire alert condition” on page 31.](#)

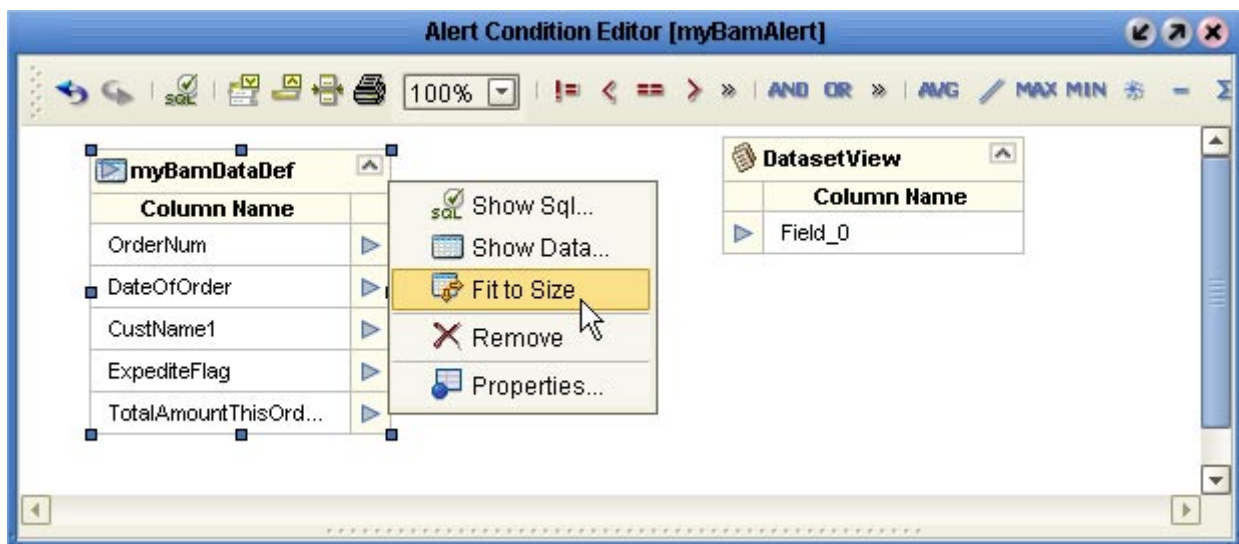
To create a new alert condition

- 1 In the project tree, under the eBAM application, right-click **Alert Conditions** and, on the popup context menu, click: **New Alert Condition**
- 2 In step 1 of the wizard, type in a name for the alert condition, and then click **Next**.
- 3 In step 2, select the check box for the data definition, and then click **Finish**.

Result: The project tree displays the new object under Alert Conditions, and the Alert Editor opens to display an instance of the data definition on the left side of the canvas and an empty dataset view on the right side.

- 4 To see the entire data definition instead of a scrollable view, right-click it and, on the menu, click **Fit To Size**. See Figure 13.

Figure 13 Newly Created Alert Condition



To configure a condition on a data definition

- 1 In the Alert Editor, right-click the data definition; on the popup menu, click **Properties**. In the Properties dialog, click **Condition**, and then click the ellipsis [...]
- 2 In the **Columns** tab, double-click the data definition open it and display its fields.
- 3 See [“Setting Up Data Conditions for an Alert, Chart, or Query” on page 25.](#)

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

- 1 In the Alert Editor, right-click **DatasetView** and, on the menu, click: **Configure**
- 2 In the **Configure Dataset View** dialog, click **Add** as many times as needed to accommodate all the report fields you will need.
- 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 64**.

- 4 In the Alert Editor, drag operators as needed from the Operator Palette (see **Figure 14 on page 30**) into the canvas between the data definition and the dataset view, and then map zero or more fields from the data definition through zero or more operators, with all output ultimately going into the dataset view. For examples of this, see the background of **Figure 16 on page 31**, or see **Figure 53 on page 80**.

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


- 5 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 8.

Table 8 Tools Provided in the Tool Palette of the Alert Editor








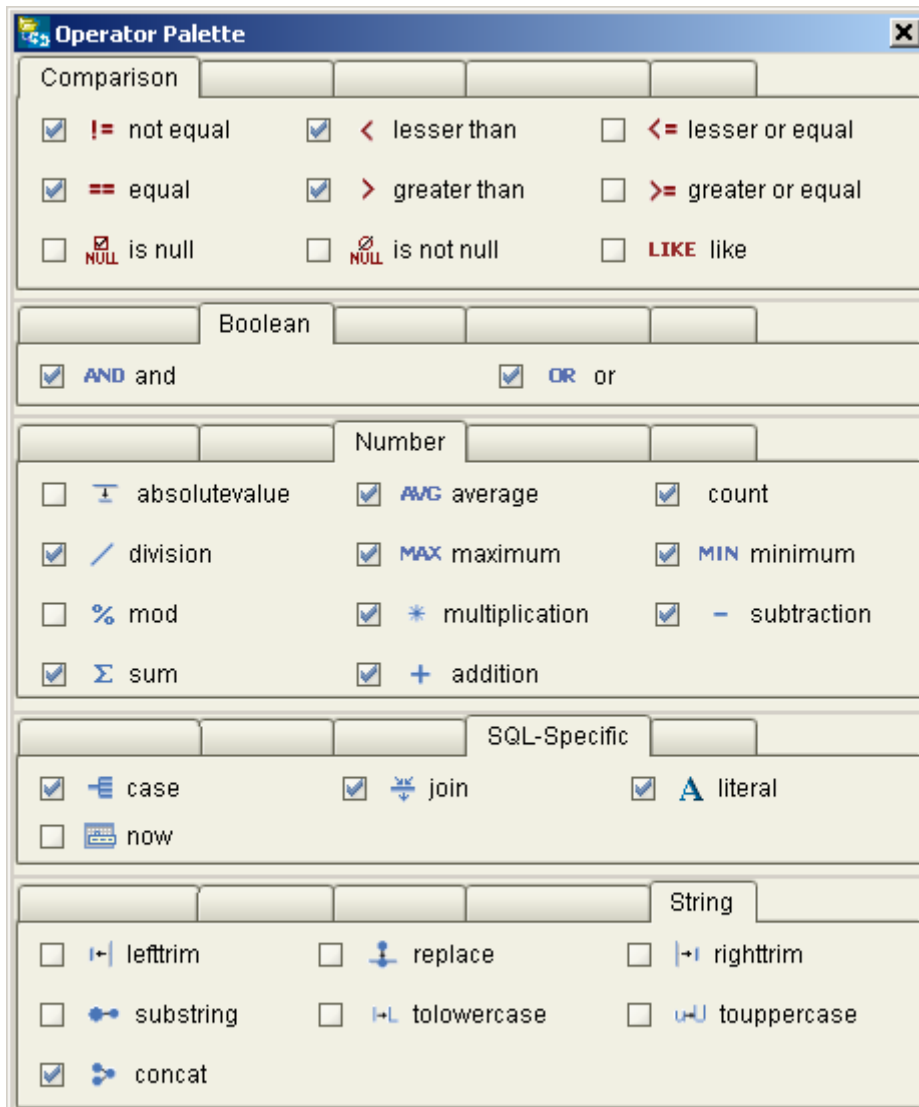
	Undo and Redo allow you travel backward and forward through the sequence of modifications you have made to the graphical canvas.
	
	Show SQL tries to validate the graph and, if successful, displays the corresponding SQL statements for the mapping. In the Output pane, you can specify which “flavor” of SQL to display, of: Oracle8, Oracle9, SQL Server, internal, or ANSI92.
	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.
	Print Graph allows you to print the graph, using various scaling options.

Figure 14 Operator Palette for SQL Operations

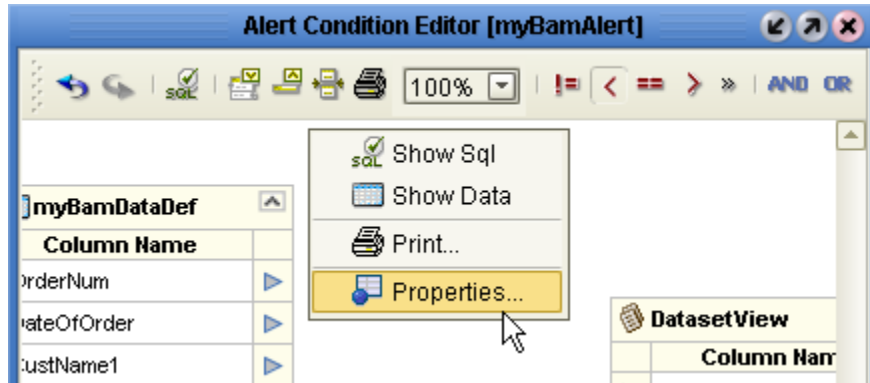


- 6 When you have set up all dataset view fields, save your work.
You are now ready to configure the properties of the alert condition as a whole.

To configure properties of the entire alert condition

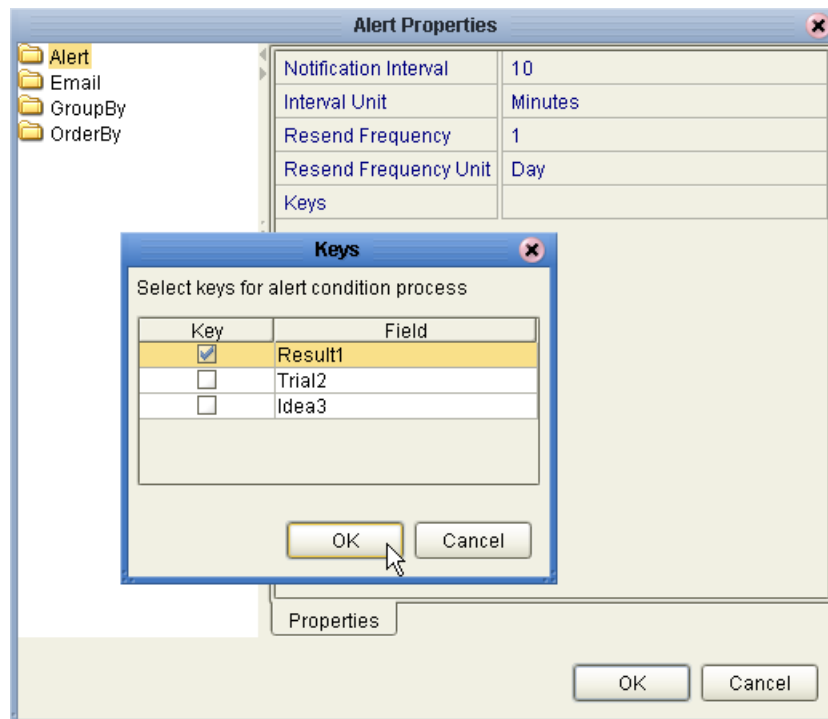
- 1 In the Alert Editor, right-click a blank portion of the canvas itself and, on the popup context menu, click:  **Properties** (see Figure 15).

Figure 15 Setting Properties for the Entire Alert Condition



- 2 In the Alert Properties dialog, select Alert if necessary, and then:
 - A Change the Notification Interval and Resend Frequency to appropriate values and time units: Notification Interval specifies how often to run the process; Resend Frequency specifies how often to send an alert that was previously sent.
 - B Then, for Keys, open the Keys dialog by clicking the ellipsis [...] to the far right. From the dataset view fields listed under Keys, select one or more check boxes, and then click OK. See Figure 16.

Figure 16 Selecting Keys in the Alert Properties Dialog



Note: The key, or a set of keys, is used to prevent duplication. When a record is found that matches the condition, an alert is sent only if its key (or set of keys) mismatches all the records already staged for alert, or if the Resend Frequency has been exceeded.

- 3 If you want an e-mail message sent when an alert condition occurs, click Email, and then supply values appropriate for your site, yourself, and this alert. See Table 9.

Table 9 Alert Properties for e-Mail

Name	Default Value	Comment
Send Email	False	To arrange for e-mails 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 are valid: <ul style="list-style-type: none"> ▪ mySmtpServer ▪ mysmtpserver ▪ mysmtpserver.mydomain.com ▪ 10.18.133.200
SMTP Server Port	25	The port number that this machine uses for e-mail.
Username	(blank)	The login ID of a user who is authorized to send e-mail 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 e-mail, either of the following forms is valid: <ul style="list-style-type: none"> ▪ myname ▪ myname@mydomain.com For external e-mail, a domain must be specified; in other words: <ul style="list-style-type: none"> ▪ myname@mydomain.com
Send To	(blank)	The e-mail 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.

- 4 When you have finished configuring the alert properties, click OK.

4.3.3 About Charts

Charts are graphical representations of KPIs. eBAM provides four types of charts:

- *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. Specific view fields can be “exploded” to highlight them.
Summary charts display one series of data; *category charts* can display several series.
- *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.
- *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. For an example, see [Figure 21 on page 41](#).
- *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. For an example, see %%%.

All charts have common properties, and each has properties specific to its own type:

- [Table 10 on page 33](#) lists properties common to all three chart types.
- [Table 11 on page 34](#) lists properties common to pie and bar charts only.
- [Table 12 on page 34](#) lists properties specific to pie charts.
- [Table 13 on page 35](#) lists properties specific to bar charts.
- [Table 14 on page 35](#) lists properties specific to meters.
- [Table 15 on page 36](#) lists properties specific to trafficlights.

Table 10 Properties Common to All Three Chart Types

Property Name	Value	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.

Table 10 Properties Common to All Three Chart Types (Continued)

Property Name	Value	Notes
Chart Height	420	Height, in pixels, of the entire chart itself.
Data Number Limit	2147483647	The largest number of rows that could be retrieved and used as data in your chart. ($2147483647 = 2^{31} - 1$)
Frequency in seconds	60	How often to update the chart with a new data view.

Table 11 Properties Common to Pie and Bar Charts

Property Name	Value	Notes
3D	True	Whether or not a three-dimensional effect is displayed.
Depth Factor	0.3	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 an elliptical (tilted-circle) chart, keep this set to False . For a circular chart, set this to True .
Null Real	0.0	Value to use for real numbers (float) without data.
Null Integer	0	Value to use for integers without data.
Null String		Value to use for strings (varchar) without data.
Integer Number Format	0	How many digits to display, and whether to display digits in comma-separated groups of three.
Real Number Format	0.000	Whether to display digits in comma-separated groups of three, and whether to display three digits after the decimal point (default) or two, with a dollar sign preceding the value.

Table 12 Properties Specific to Pie Charts

Property Name	Value	Notes
Radius	0.7	Space used by the image (0.0=none; 1.0=all available).
Section Label	Name,Value	Choices consist of: None; Name; Value; Percent; Name,Value; Name,Percent, and 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).
Show Series Labels	False	Whether to display a label for the data series.
Series Label Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Series Label Color		You can specify any color by swatch, HSB, or RGB.
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 13 Properties Specific to Bar Charts

Property Name	Value	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>(Applies only to Category charts, not Summary.)</i> Whether or not to stack the data series.
Category Label Type	None	<i>(Applies only to Category charts, not Summary.)</i> To display the category name(s), set this to Name. To display the category value(s), set this to Value. To suppress the label altogether, set this to None.
Series Label Type	None	<i>(Applies only to Category charts, not Summary.)</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.

Table 14 Properties Specific to Meters

Property Name	Value	Notes
Minimum Value	0.0	Lower bound of the meter.
Maximum Value	0.0	Upper bound of the meter.
Value Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Value Color		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	0.0	Values above this threshold are too high to be normal.
Normal Range Color		Color of the Normal range. You can set this either by picking a swatch or by specifying HSB or RGB.
Minimum Warning Value	0.0	Values below this threshold are too low for warnings.
Maximum Warning Value	0.0	Values above this threshold are too high for warnings.
Warning Range Color		Color of the Warning range. You can set this either by picking a swatch or by specifying HSB or RGB.
Minimum Critical Value	0.0	Values below this threshold are too low to be critical.
Maximum Critical Value	0.0	Values above this threshold are beyond being critical.
Critical Range Color		Color of the Critical range. You can set this either by picking a swatch or by specifying HSB or RGB.
Other/Units	units	

Table 14 Properties Specific to Meters (Continued)

Property Name	Value	Notes
Other/Draw Chart Border	False	Whether or not to display a border around the chart.
Other/Border Type	Normal Range	You can change this to Warning, Critical, or Full Range.
Other/Dial Type	Circle	You can change this to either Pie or Chord.
Other/Dial Border Color		Color of the dial outline. You can set this either by picking a swatch or by specifying HSB or RGB.
Other/Dial Background Color		Color of the dial background. You can set this either by picking a swatch or by specifying HSB or RGB.
Other/Tick Label Type	Value Label	Whether to show marks with values, or just marks.
Other/Tick Label Font		Any of 52 fonts, 9-pt to 72-pt, bold and/or italic or not.
Other/Needle Color		You can specify any color, by swatch, HSB, or RGB.
Other/Meter Angle	270	Angle subtended by the entire range of the meter.

Table 15 Properties Specific to Trafficlights

Property Name	Value	Notes
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		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		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		Color of the Critical range. You can set this either by picking a swatch or by specifying HSB or RGB.
LensDisplayCount	1	Number of lenses to display: 1, 2, or 3.
LensDisplayOrientation	VERTICAL	Which way (vertical or horizontal) the traffic light is oriented.
LensDisplayDirection	Right to Left / Top to Bottom	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.



4.3.4 Setting Up the Application's Charts

In eBAM, charts provide real-time feedback on the current data set according to conditions you set on one or more of the elements in the data definition. As discussed earlier (see [“Using the Condition Editor” on page 26](#)), you use the Condition editor to create queries that detect the condition. In addition to conditions on the data definition, you also set up a dataset view.

An eBAM application can contain many charts, or only one, or none at all. Each chart requires the following:

- A name and an associated data definition; see [“To create a new chart”](#).
- Zero or more conditions set on fields in the data definition; see [“To configure a condition on a data definition field” on page 38](#).
- One or more fields in the dataset view, with appropriate mappings to them from operators and data definition fields; see [“To configure the dataset view, mapping operators and data definition fields to its fields” on page 38](#).
- Settings for the chart's properties as a whole. In addition to generic chart properties discussed earlier (see [“About Charts” on page 33](#)), you can also set chart properties that depend on the dataset view fields of your particular application, such as group-by (aggregation) fields and order-by settings; see [“To configure properties of the entire chart” on page 39](#).

To create a new chart

- 1 In the project tree, under the  eBAM application, right-click  **Charts** and, on the popup context menu, click: **New Chart**
- 2 In step 1 of the Chart wizard, specify a chart type (Pie Chart, Bar Chart, Meter, or Trafficlight) and a name.
- 3 In step 2 of the wizard, select the check box for the data definition.
- 4 In step 3 of the wizard, adjust the common and chart-specific parameters in all three tabs to suit your taste. (Alternatively, you can defer your chart parameter decisions to a later time, and simply accept all defaults for now).
- 5 Click **Finish**.



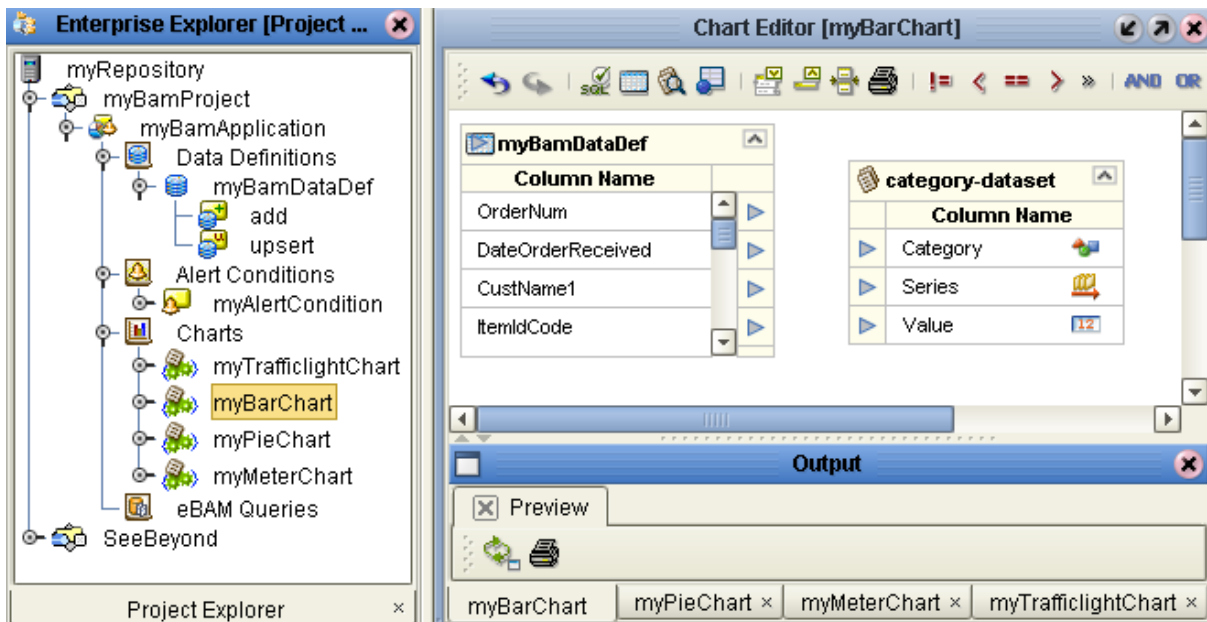
Result: The project tree displays the new  object under  **Charts**, and the Chart Editor opens to display the data definition on the left side of the canvas and an empty dataset view on the right side. See Figure 17.

Figure 17 Chart Editor



Tip: The tool palettes for the Chart Editor and its output window are the same as for the Alert Conditions Editor. [Table 8 on page 29](#) explains the graphical controls, and [Figure 14 on page 30](#) lists the SQL operators.

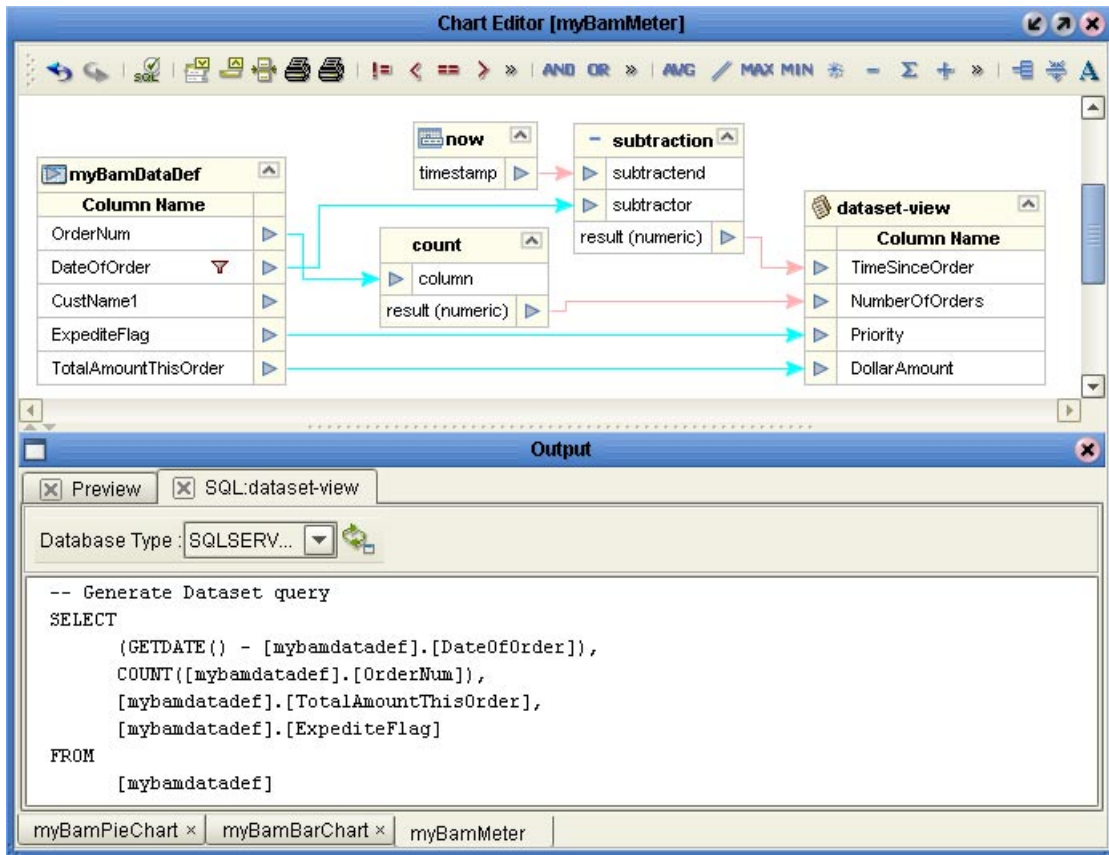
To configure a condition on a data definition field

- 1 In the Chart Editor, right-click the data definition; on the popup menu, click **Properties**. In the Properties dialog, click **Condition**, and then click the ellipsis [...] to open the Condition editor.
- 2 In the **Columns** tab, double-click the data definition open it and display its fields.
- 3 See [“Setting Up Data Conditions for an Alert, Chart, or Query” on page 25](#).

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

- 1 In the Chart Editor, right-click the **category-dataset** or **summary-dataset** and, on the menu, click: **Configure**
- 2 In the **Configure Dataset View** dialog, click **Add** as many times as needed to accommodate all the report fields you will need.
- 3 Double-click each field, edit the name, and press ENTER. When finished, click OK.
- 4 In the Chart Editor, drag operators as needed from the operator palette (see [Figure 14 on page 30](#)) into the canvas between the data definition and the dataset view, and then map zero or more fields from the data definition through zero or more operators, with all output ultimately going into the dataset view. For examples of this, see [Figure 53 on page 80](#) and Figure 18 below.

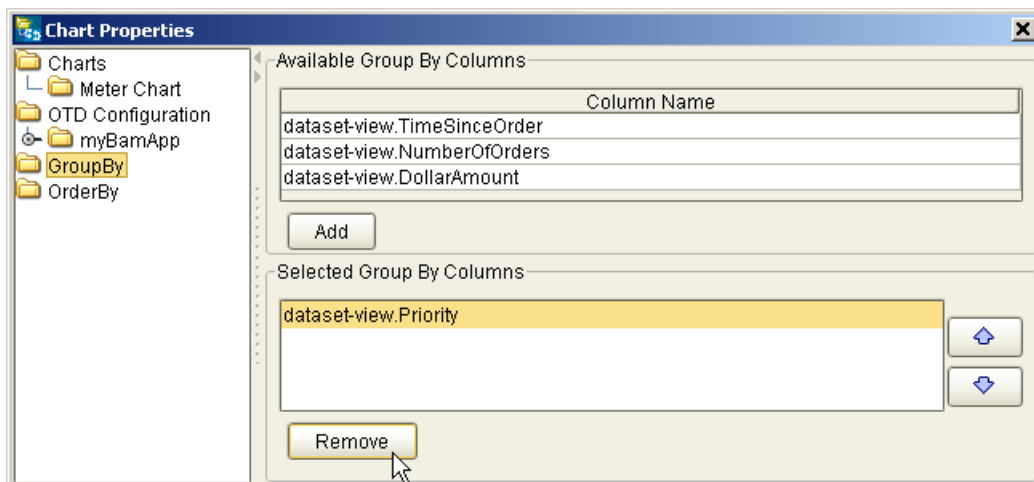
Figure 18 Configuring a Category Dataset Using Operators and Data Definition Fields



To configure properties of the entire chart

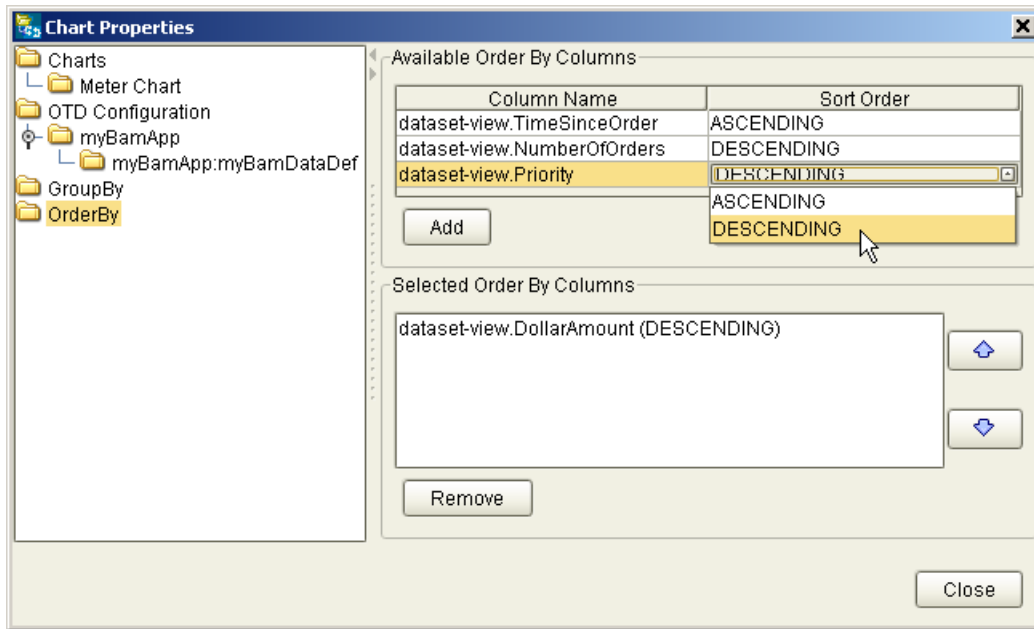
- 1 In the Chart Editor, right-click the blank canvas and, on the menu, click: **Properties**
- 2 In the tree on the left side, select the **GroupBy** folder. Under Available Group-By Columns, select the dataset view element you want to group by, and then click **Add** to move the element to the Selected Group-By Columns pane. See Figure 19.

Figure 19 Chart Properties: “Available Group-By” and “Selected Group-By” Columns



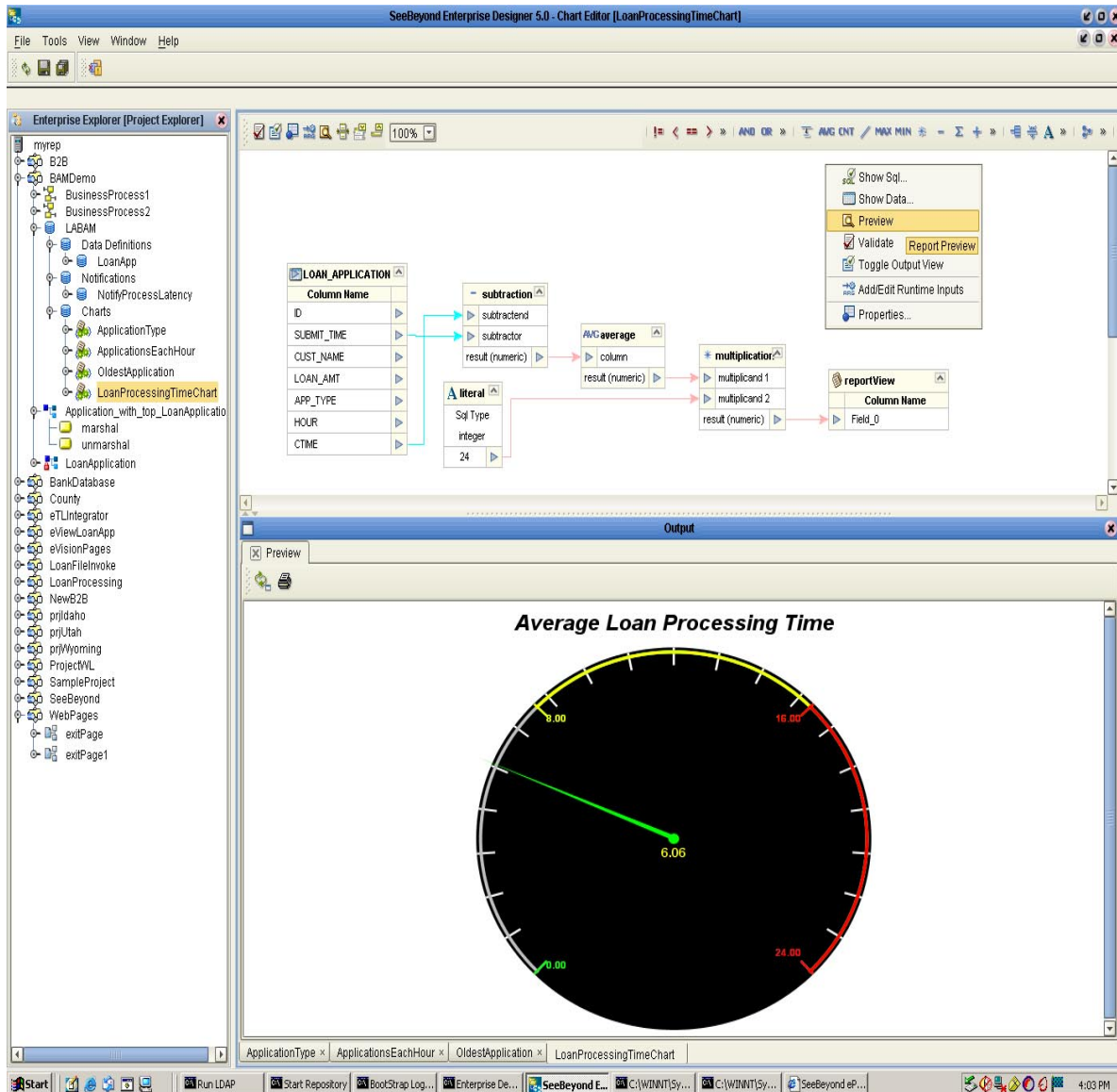
- 3 Optionally, select the **OrderBy** folder, use the Sort Order controls to set the order (ascending or descending) for each view element, select the view element you want to order by, and then click **Add** to move the element to the Selected Order-By Columns pane. See Figure Figure 20.

Figure 20 Chart Properties: “Available Order-By” and “Selected Order-By” Columns



- 4 Optionally select the **Charts** folder and modify any of the common properties or chart-specific properties. (This is the alternative mentioned in step 4 of the [procedure on page 37.](#))
 - 5 When you have configured all chart properties, click **OK**.
 - 6 To preview the chart, right-click the blank canvas and, on the menu, click: **Preview**
- If sample data has been committed previously (see step 8 in the [procedure on page 24](#)), the Output window shows a preview of how the chart for that sample data will appear. For an example, see Figure 21.

Figure 21 Previewing a Chart



7 In the **Output** window, 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.

4.3.5 About Queries

eBAM allows you to %%%:

-

-
-
-

All queries have %%%.



4.3.6 Setting Up the Application's Queries

In eBAM, queries allow you to %%%. As discussed earlier (see [“Using the Condition Editor” on page 26](#)), you use the Condition editor to create queries that detect the condition. In addition to conditions on the data definition, you also set up a dataset view.

An eBAM application can contain many queries, or only one, or none at all. Each query requires the following:

- A name and an associated data definition; see [“To create a new query”](#).
- Zero or more conditions set on fields in the data definition; see [“To configure a condition on a data definition field” on page 43](#).
- One or more fields in the dataset view, with appropriate mappings to them from operators and data definition fields; see [“To configure the dataset view, mapping operators and data definition fields to its fields” on page 43](#).
- Settings for the query's properties as a whole. These consist of group-by (aggregation) fields and order-by settings; see [“To configure properties of the entire query” on page 44](#).

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 a name.
- 3 In step 2 of the wizard, select the check box for the data definition.
- 4 Click **Finish**.



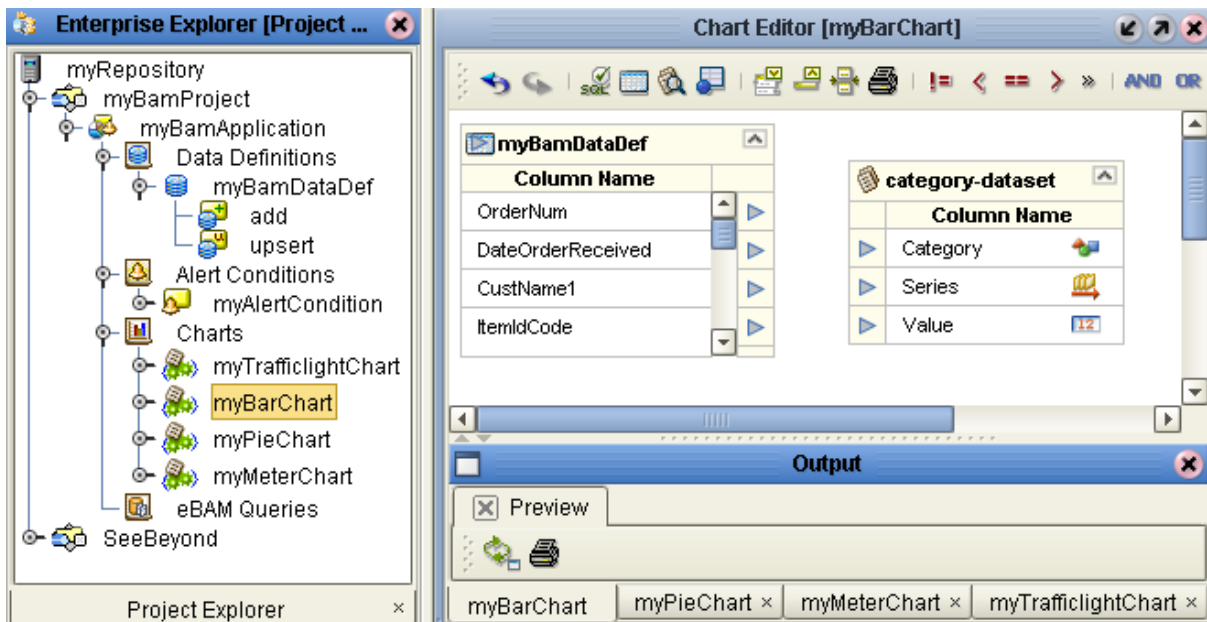

Result: The project tree displays the new  object under  eBAM Queries, and the Query Editor opens to display the data definition on the left side of the canvas and an empty dataset view on the right side. See Figure 17.

Figure 22 Query Editor



Tip: The tool palette for the Query Editor is mostly the same as for the Alert Conditions Editor and Chart Editor. [Table 8 on page 29](#) explains the graphical controls, and [Figure 14 on page 30](#) lists the SQL operators. The one additional tool,  Add/Edit Runtime Inputs, is explained %%%.

To configure a condition on a data definition field

- 1 In the Query Editor, right-click the data definition; on the popup menu, click **Properties**. In the Properties dialog, click **Condition**, and then click the ellipsis [...] to open the Condition editor.
- 2 In the **Columns** tab, double-click the data definition open it and display its fields.
- 3 See [“Setting Up Data Conditions for an Alert, Chart, or Query” on page 25](#).

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

- 1 In the Query Editor, right-click **DatasetView** and, on the menu, click: **Configure**
- 2 In the **Configure Dataset View** dialog, click **Add** as many times as needed to accommodate all the report fields you will need.
- 3 Double-click each field, edit the name, and press ENTER. When finished, click OK.
- 4 In the Query Editor, drag operators as needed from the operator palette (see [Figure 14 on page 30](#)) into the canvas between the data definition and the dataset view, and then map zero or more fields from the data definition through zero or more operators, with all output ultimately going into the dataset view. For examples of this, see [Figure 53 on page 80](#) and Figure 18 below.

Figure 23 Configuring a Query’s DataSetView Using Operators and Data Definition Fields
%%%


To configure properties of the entire query

- 1 In the Query Editor, right-click the blank canvas and, on the menu, click: **Properties**
- 2 In the tree on the left side, select the **GroupBy** folder. Under Available Group-By Columns, select the dataset view element you want to group by, and then click **Add** to move the element to the Selected Group-By Columns pane. See Figure 19.

Figure 24 Query Properties: “Available Group-By” and “Selected Group-By” Columns
%%%

- 3 Optionally, select the **OrderBy** folder, use the Sort Order controls to set the order (ascending or descending) for each view element, select the view element you want to order by, and then click **Add** to move the element to the Selected Order-By Columns pane. See Figure 20.

Figure 25 Query Properties: “Available Order-By” and “Selected Order-By” Columns
%%%

- 4 When you have configured all query properties, click **OK**.
- 5 In the **Output** window, with the SQL:DatasetView tab selected, you can:
 - ♦ Choose a database type, of: Oracle8, Oracle9, internal, SQLServer, and ANSI92.
 - ♦ Use the  **Refresh** button to update the SQL listing.

To add or modify run-time inputs to a configured query


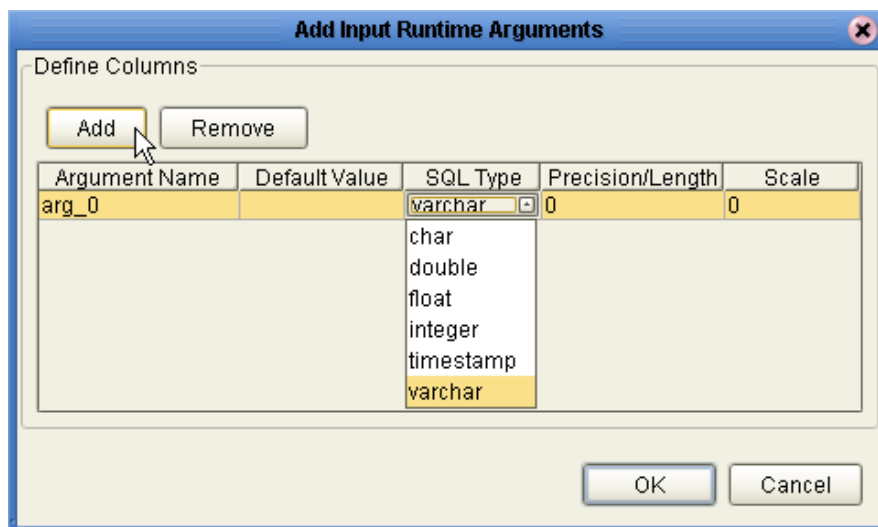
- 1 In the Query Editor, click  **Add/Edit Runtime Inputs**
- 2 In the Add Input Runtime Arguments dialog box (see Figure 26), click **Add** as many times as needed to create additional run-time arguments.

Figure 26 Adding Input Runtime Arguments to eBAM Queries



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

- 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.

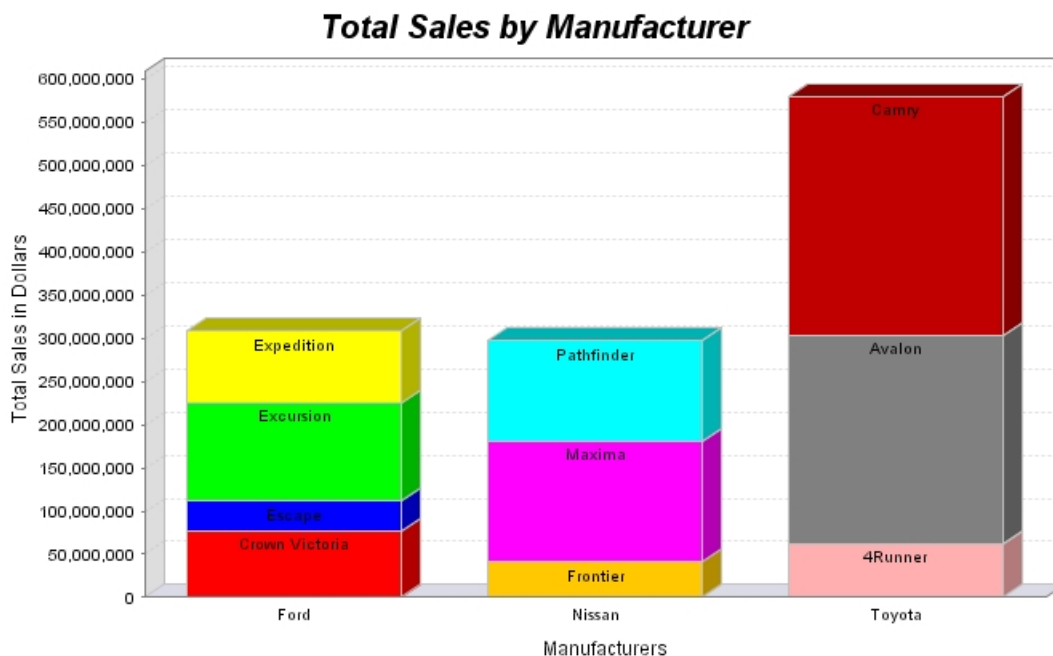
Installing and Running the eBAM Sample

This chapter provides step-by-step instructions for using the sample project and files supplied with the eBAM product file. The instructions in this chapter assume you have installed the prerequisite products (either eInsightESB, or else both eGate and eInsight), and that you have also installed the File eWay, which the sample implementation uses for input/output of sample data.

Last Things First

When you finish this chapter, you will have installed, deployed, run, and monitored the sample eBAM application supplied in eBAMDocs.sar. Upon feeding in sample data, results will include the following: A stacked bar chart that looks like Figure 27, with the display updating every 30 seconds; an updated bar chart that changes according to the updated data you feed in; an output file containing results of a stored query run against the data that has been fed in to date; and an alert file containing notification text that is written (and e-mailed, if you configure it) whenever a particular threshold is reached.

Figure 27 Bar Chart for Sample



5.0.1 Overview of Steps for Setting Up the Sample Implementation

The eBAM Studio product includes a complete sample implementation, included in the **eBAMDocs.sar** file, that allows you to see the end results without having to go through all the design steps. The necessary procedures are gathered into two sections:

Procedures for initial setup

- **Installing the Sample Files** on page 47
- **Importing the Sample Project** on page 48
- **Creating the Sample Environment** on page 48
- **Starting the Logical Host for the Sample** on page 49

Procedures for configuration, deployment, and runtime

- **Configuring the eWays** on page 54
- **Activating and Running the Project** on page 55
- **Monitoring Key Performance Indicators with Live Data** on page 57

5.1 Initial Setup Steps

5.1.1 Installing the Sample Files

These steps assume the existence of a temporary eBAM directory for sample files, such as **C:\temp\eBAM**. You will install the sample files to this directory.

To install the sample files

Before you begin: Your repository must already be running, and you must be logged in to Enterprise Manager. If you have already uploaded the documentation for eBAM, you can skip steps 1 and 2 and start with step 3.

- 1 In the ADMIN tab, if you have not already done so, browse to the [...] \Documentation \ **ProductsManifest.xml** file and submit it.
- 2 In the ADMIN tab, if you have not previously done so, browse to the **eBAMDocs.sar** file, select it, and click the **upload now** button.
- 3 In the DOCUMENTATION tab, in the Products window, click: **eBAM Studio**
- 4 In the window that appears on the right side, click: **Download Sample**
- 5 Preserving file paths, extract the files to your temporary eBAM samples directory.

Result: The following directories and files are created:

```
C:\temp\eBAM\Sample\Projects\SampleBamProject.zip
C:\temp\eBAM\Sample\Data\In\TestAdd*.~in
C:\temp\eBAM\Sample\Data\In\TestRevise*.~in
C:\temp\eBAM\Sample\Data\In\TestQuery*.~in
C:\temp\eBAM\Sample\Data\Out\TestAlertOutput.dat
C:\temp\eBAM\Sample\Data\Out\TestQueryOutput.dat
```

5.1.2 Importing the Sample Project

To install the sample Project

Before you begin: Your repository must already be running, and you must be logged in to Enterprise Designer.

- 1 In Enterprise Designer, save all work and close all canvases. In Project Explorer, right-click the repository and, on the popup context menu, click: **Import**
- 2 In the **Import Manager** dialog, browse to the folder where you installed the sample files (such as C:\temp\EBAM\Sample\Projects), select `SampleBamProject.zip`, and click **Open**.
- 3 Back in the **Import Manager** dialog, click **Import**.
- 4 When the import ends successfully, click **OK**, and then click **Close**.

Result: The sample project has been imported. It is now visible the Project Explorer tree.

5.1.3 Creating the Sample Environment

These steps assume you will use a default integration server running on default ports. If you use anything other than a SeeBeyond Integration Server on ports 18000–18009, make adjustments in step 6 below or in the URL in step 2 in the [procedure on page 57](#).

To create the sample environment

- 1 In Enterprise Designer, near the lower left of the window, click the **Environment Explorer** tab.
- 2 In the environment tree, right-click the repository and, on the popup context menu, click **New Environment**
 - ♦ Rename the newly created environment to **SampleBamEnv**
- 3 Right-click `SampleBamEnv` and, on the menu, click: **New File External System**
 - ♦ Name the new external **extFileIn**, designate it an Inbound File eWay, and click OK.
- 4 Right-click `SampleBamEnv > New File External System`
 - ♦ Name it **extFileOut**, designate it an Outbound File eWay, and click OK.
- 5 Right-click `SampleBamEnv > New Logical Host`
 - ♦ Retain the default name: `LogicalHost1`
- 6 Right-click `LogicalHost1` and click: **New SeeBeyond Integration Server**
 - ♦ Retain the default name: `IntegrationSvr1`

Result: The sample environment now contains the three servers it needs.

5.1.4 Starting the Logical Host for the Sample

These steps assume you have already installed a logical host named LogicalHost1, and that the environment is named SampleBamEnv.

To bootstrap the logical host

- 1 Open a command prompt and change directories to the location of your logical host's bootstrap executables. For example:

```
cd \ican50\logicalhost\bootstrap\bin
```

- 2 Start the bootstrap script using appropriate parameters. For example:

```
bootstrap -r http://myBox:12345/myRep -i myId -p myPassword  
-e SampleBamEnv -l LogicalHost1
```

- For the **-r** (repository) parameter), supply the correct URL with repository name.
- For the **-i** and **-p** (ID and password) parameters, supply the appropriate values.
- For **-e** (environment) parameter, use: **SampleBamEnv**
- For **-l** (logical host name) parameters, use: **LogicalHost1**

Result: The logical host is now running and ready to have a project deployed to it.

5.2 Design, Deployment, and Runtime Steps

This section provides steps for:

- [“Validating the OTDs” on page 50](#), where you can optionally familiarize yourself with the OTDs used in the sample.
- [“Using the eBAM Application to Preview Sample Data” on page 50](#), where you should familiarize yourself with the setup, purpose, and function of the sample eBAM application itself.
- [“Configuring the eWays” on page 54](#), where you will use the Enterprise Designer's Configuration Map Editor to set parameters for inbound and outbound File eWays.
- [“Activating and Running the Project” on page 55](#), where you will use Enterprise Designer to create a deployment profile and activate it.
- [“Monitoring Key Performance Indicators with Live Data” on page 57](#), where you will feed data to the logical host and use the Charts Viewer to monitor results.

5.2.1 Validating the OTDs



This procedure is optional. It helps you get acquainted with the sample OTDs by taking you through steps for viewing and testing them.

The OTDs included with the sample are listed and described in Table 16.

Table 16 OTDs Included With Sample

OTD Name	Purpose	Sample Data to Test
OTDAlertOut	In the AlertUnitSales BP, this OTD marshals data so it can be sent out to the Alert output file.	Out\TestAlertOut.dat
OTDfullDataInput	In the InputUpdate BP, this OTD unmarshals data in Path A (new data records add to the chart), and in Path B (revised data records update the chart).	In\TestAdd*.~in In\TestRevFull*.~in
OTDquerySalesIn	In the QuerySales BP, this OTD unmarshals data so that it a stored query can be executed on it.	In\TestQueryIn.~in
OTDquerySalesOut	In the QuerySales BP, this OTD marshals the data from the query result so it can be written out.	Out\TestQueryOut.dat
OTDupdateUnits	In the InputUpdate BP, this OTD unmarshals data in Path C (revised sales data updates the chart).	In\TestRevUnits*.~in

To validate an OTD

- 1 In Project Explorer, double-click the OTD you want to validate.
The OTD Editor displays the OTD.
- 2 In the top center pane, open each element of the OTD and examine its properties. In particular, notice which nodes are repeating and what its delimiters are.
- 3 On the toolbar of the OTD Editor, click  **Run Test** to open the OTD Tester pane.
- 4 In the toolbar of the OTD Tester, with the Input tab selected, click  Open File
- 5 Browse to the location (under C:\temp\eBAM\Sample\Data) of the file containing the sample data for this OTD (see Table 16) and open it.
- 6 After successful unmarshaling, open the tree in the left pane of the OTD tester to see how the sample data was parsed.

5.2.2 Using the eBAM Application to Preview Sample Data

The following procedures are optional, but highly recommended. They help you get acquainted with the setup of the sample eBAM application by focusing on key aspects of the application and showing you how to preview, commit, and display charts of sample data.

Procedures

- [To view the sample eBAM application’s Data Definition \(metadata\)](#) on page 51
- [To view and manipulate a bar chart of the sample data](#) on page 51
- [To view a pie chart of the sample data](#) on page 53





To view the sample eBAM application’s Data Definition (metadata)

- 1 In Project Explorer, open SampleBamProject > SampleBamApp > **Data Definitions** and double-click **SampleBamDDN**.

The eBAM Data Definition Editor displays metadata as shown in Table 17.

Table 17 Data Definition (Metadata) for the Sample eBAM Application

Name	Type	Meaning
datasource	char	A one-character code indicating which path to follow (A, B, or C) in the InputUpdate BP. Path A is for new data; path B is for complete records of revised data; and path C is for revised units-sold data only.
carModelYear	integer	Model year, such as 2004 or 2002.
carModelName	varchar	Model name, such as “Excursion” or “Camry”.
carManufacturer	varchar	Manufacturer, such as “Ford” or “Toyota”.
carType	varchar	Type of car, such as “SUV” or “car”.
carBasePrice	float	Price, such as 35990.00 or 18990.00
carUnitsSold	integer	Number of cars sold of this type, such as 875 or 1063.

- 2 Click  **Properties** to display the properties for Data Retention: Data is retained for one year, and cleanup of stale data occurs once a month. Click **OK**.
- 3 Click  **Show Sample Data** to open the Sample Data pane; notice how the names of the metadata fields appear as column names across the top.
- 4 In the Sample Data toolbar, click  **Import** to launch the File Import Wizard.
- 5 In the File Import Wizard, browse to C:\temp\eBAM\Sample\Data\In\ and select **TestAddAll.~in**, define it as a delimited file (record delimiters are linefeeds; field delimiters are commas). Click **Preview** to audition the data, and then click **Finish**.
- 6 After checking that the data columns are all of the correct type, click  **Commit**. This loads the sample data so that it can be displayed in a chart.

To view and manipulate a bar chart of the sample data

- 1 In Project Explorer, open SampleBamProject > SampleBamApp > **Charts** and double-click **barTotalSalesByManufacturer**.

The Chart Editor shows the following mappings in graphical form:

- ◆ The value of **carManufacturer** is mapped as a **Category**.
- ◆ The value of **carModelName** is mapped as a **Series**.
- ◆ The values for **carBasePrice** and **carUnitsSold** are multiplied together and summed (to handle multiple records), with the result mapped as a **Value**.

- 2 In the Chart Editor tool palette, click  **Show Selected Data**.

The Output pane displays the data, sorted first by manufacturer (in the Category column), then by model name (in the Series column). The gross sales receipts for each combination are calculated and displayed in the third (Value) column.




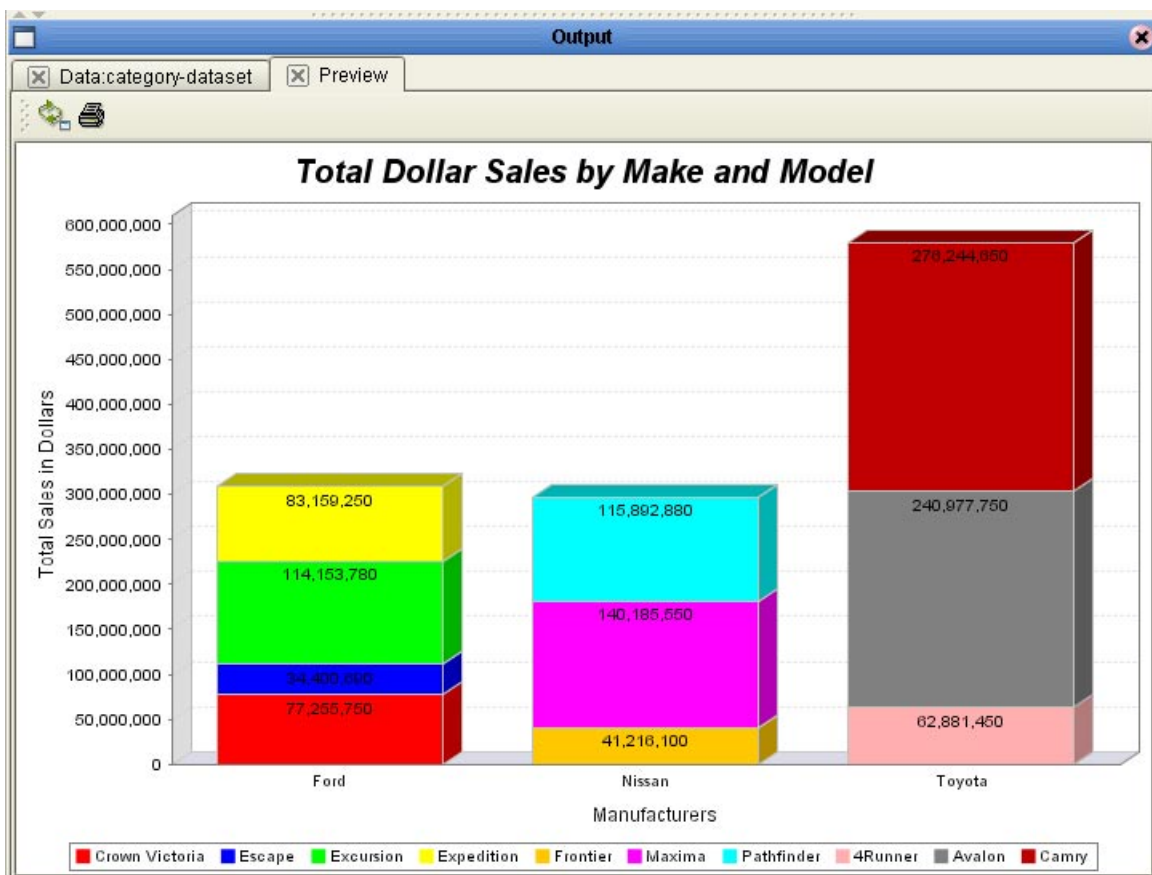
- 3 In the Chart Editor tool palette, click  **Preview Report/Chart**.
The Output pane displays the data in three stacked bars. Each bar represents a manufacturer (in other words, a category), and each block within a bar represents a model (in other words, a series). The height of each block is determined by the gross sales receipts (in other words, the value).
- 4 In the Chart Editor tool palette, click  **Properties** to display the three-tabbed property sheet for the bar chart. Try out changes to various properties, such as:
 - ♦ In the **Common** tab, change the **Title** to: **Total Dollar Sales by Make and Model**
 - ♦ In the **Chart** tab, change the **Include Legend** flag to: **True**
 - ♦ In the **Bar-Specific** tab, change the **Series Label Type** to: **Value**
- 5 Click **OK**, and then, in the Output pane, click  **Refresh**. Notice how the chart title changes, there is now a legend along the lower edge, and each block displays the gross receipts for the model, rather than the model name.
- 6 Repeat the previous two steps as often as you like, to explore how the various chart properties affect the pie chart display. See Figure 29, for example.


Figure 28 Sample Chart barTotalSalesByManufacturer, With Modified Chart Properties



To view a pie chart of the sample data

- 1 In Project Explorer, open SampleBamProject > SampleBamApp > **Charts** and double-click **pieMarketShare**.

The Chart Editor shows the following conditions and mappings in graphical form:

- ♦ A  filter (condition) has been placed on **carModelYear**.
- ♦ As in the bar chart, the value of **carManufacturer** is mapped as a **Category**.
- ♦ As in the bar chart, the values for **carBasePrice** and **carUnitsSold** are multiplied together and summed, with the result mapped as a **Value**.

- 2 Right-click the data source with the filter (condition), open its properties, click the value for its Condition property, and click the ellipsis [...] to the far right to open the Condition dialog.

The filter is, in essence, saying: “In the case where the car’s model year is 2003, ...”.


- 3 Click **OK** to close the Condition dialog; click **OK** again to close the properties.

- 4 In the Chart Editor tool palette, click  **Show Selected Data**.


The Output pane displays the data sorted by manufacturer (Category column). For each manufacturer, gross receipts across all units in model year 2003 are displayed in the second (Value) column.

- 5 In the Chart Editor tool palette, click  **Preview Report/Chart**.

The Output pane displays the data as a pie chart. Each slice represents the market share held by each manufacturer (in other words, a category). The size of the slice is determined by the gross sales receipts for all that manufacturer’s unit sales in model year 2003 (in other words, the value).

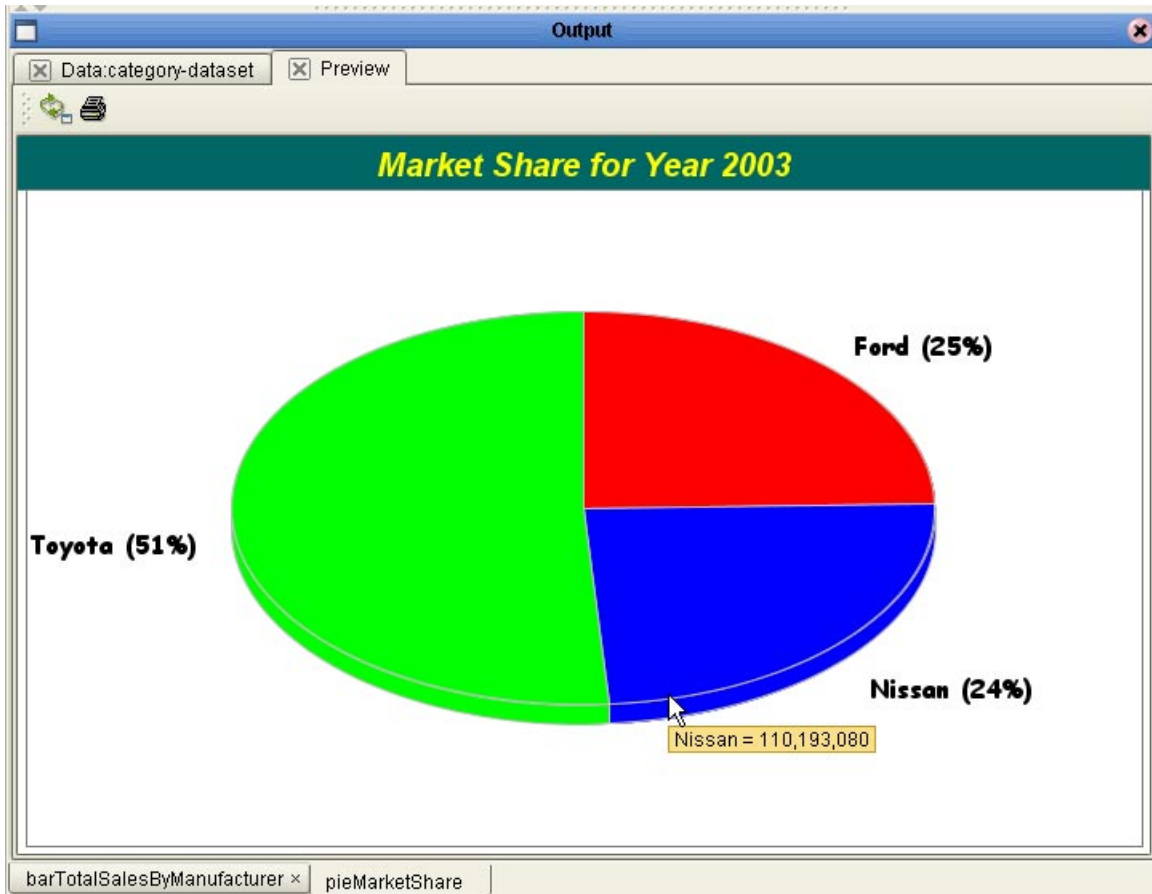
- 6 In the Chart Editor tool palette, click  **Properties** to display the three-tabbed property sheet for the pie chart. Try out changes to various properties, such as:

- ♦ In the **Common** tab, adjust the **Title Color** and **Title Background**.
- ♦ In the **Chart** tab, change the **Depth Factor** to: **0.05** (in other words, very flat)
- ♦ In the **Pie-Specific** tab, change the **Section Label** to: **Name,Percent**. Also adjust the font for the section label.

- 7 Click **OK**, and then, in the Output pane, click  **Refresh**. Notice how the chart title color and background change, the pie flattens to a disk, and the section labels now indicate the percent of market share rather than the gross units sold.

- 8 Repeat the previous two steps as often as you like, to explore how the various chart properties affect the pie chart display. See Figure 29, for example.

Figure 29 Sample Chart pieMarketShare2003, With Modified Chart Properties

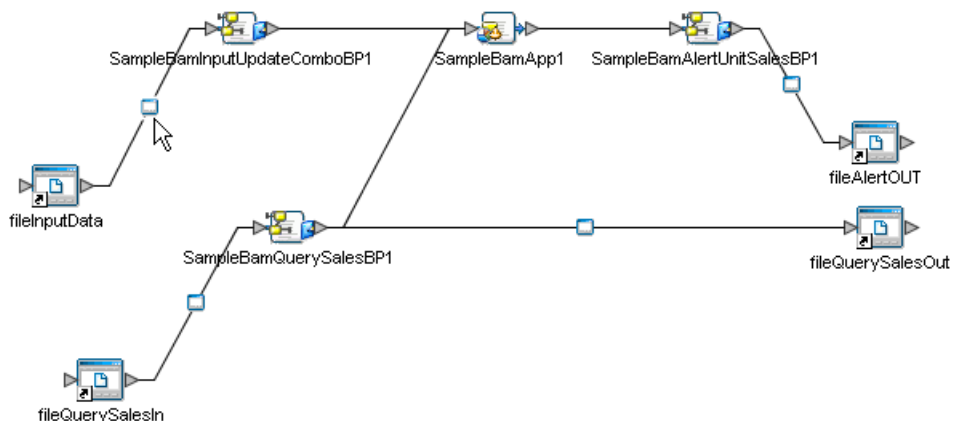


5.2.3 Configuring the eWays

To configure the eWays

- 1 In Project Explorer, in SampleBamProject, double-click **cmptest** to open the Connectivity Map Editor and see the component connections. See Figure 30.

Figure 30 Connectivity Map for Sample



- 2 In the Connectivity Map Editor, double-click the eWay connecting fileInputData with SampleBamApp1, choose the **Inbound** File eWay template, and then make the following changes to the default parameter settings:
 - ◆ **Directory** must point to your input data: **C:\temp\eBAM\Sample\Data\In**
 - ◆ For **Input file name**, change to: **ebamInputData*.txt**
 - ◆ For **Multiple records per file**, change to: **True**
 - ◆ For **Remove EOL**, keep: **True**
- 3 Verify that all parameters are correctly set, and then click OK.
- 4 In the Connectivity Map Editor, double-click the eWay connecting fileQuerySalesIn with SampleBamQuerySalesBP1, choose the **Inbound** File eWay template, and then make the following changes to the default parameter settings:
 - ◆ **Directory** must point to your input data: **C:\temp\eBAM\Sample\Data\In**
 - ◆ For **Input file name**, change to: **ebamQuerySales*.txt**
 - ◆ For **Multiple records per file**, keep: **False**
 - ◆ For **Remove EOL**, change to: **True**
- 5 Verify that all parameters are correctly set, and then click OK.
- 6 Open the eWay connecting SampleBamQuerySalesBP1 with fileQuerySalesOut, choose the **Outbound** File eWay template, and then check the parameter settings:
 - ◆ **Directory** should point to: **C:\temp\eBAM\Sample\Data\Out**
 - ◆ For **Multiple records per file**, change to: **False**
 - ◆ **Output file name** should use this pattern: **querySalesOut%d.dat**
- 7 Verify that all parameters are correctly set, and then click OK.
- 8 Open the eWay connecting SampleBamAlertUnitSalesBP1 with fileAlertOut, choose the **Outbound** template, and then make the following changes:
 - ◆ **Directory** should point to: **C:\temp\eBAM\Sample\Data\Out**
 - ◆ For **Multiple records per file**, change to: **False**
 - ◆ **Output file name** should use this pattern: **alertUnitSalesOut%d.dat**
- 9 Verify that all parameters are set correctly, and then click OK.
- 10 Save your work and close all canvases.

Result: All components are connected and configured. The next step is to create and activate a deployment profile for the project.

5.2.4 Activating and Running the Project

You will create a deployment profile named **SampleBamDP**, which you will activate and deploy to the logical host that is currently running. (If it is not already running, see [“Starting the Logical Host for the Sample” on page 49.](#))

To create the deployment profile

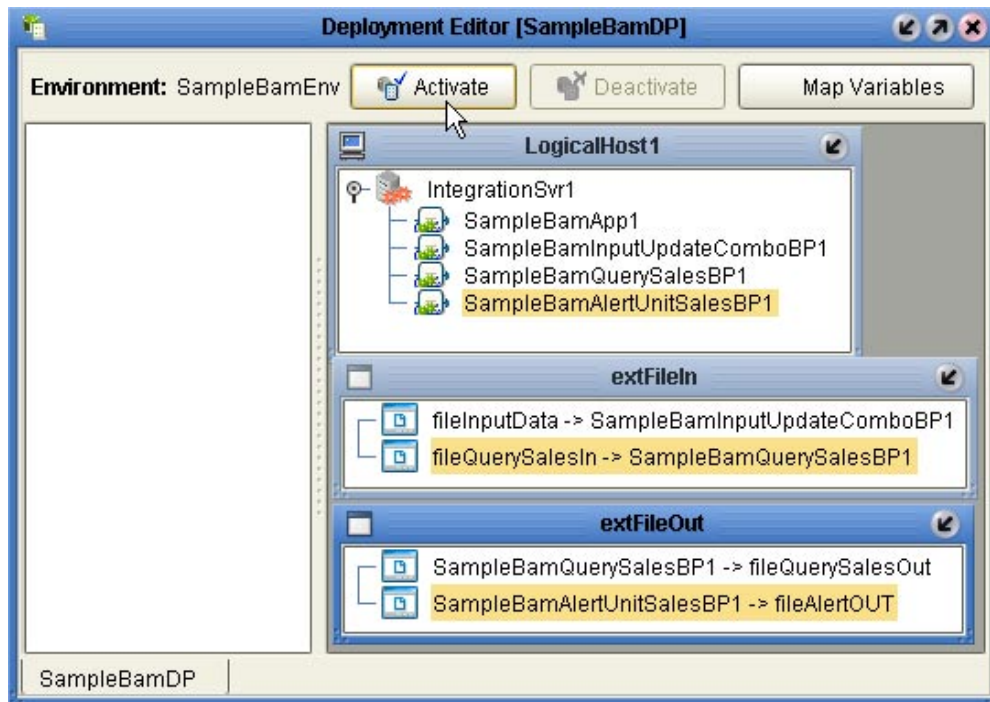
- 1 In the project tree, right-click SampleBamProject and, on the popup context menu, point at New and click: **Deployment Profile**
- 2 In the dialog box, name it **SampleBamDP**, and be sure it references the **SampleBamEnv** environment before clicking OK.

Result: The project tree displays the new object, and the Deployment Editor opens to display the eight components and the three servers to which you will assign them.

To assign components to servers

- 1 One by one, drag the four services (SampleBamApp1, SampleBamInputUpdateBP1, SampleBamQuerySalesBP1, and SampleAlertUnitSalesBP1) into LogicalHost1 and onto **IntegrationSvr1**.
- 2 Drag the two inbound File eWays (fileInputData->SampleBamInputUpdateBP1 and fileQuerySalesIn->SampleBamQuerySalesBP1) into the **extFileIn** server.
- 3 Drag the two outbound File eWays (SampleBamQuerySalesBP1->fileQuerySalesOut and SampleBamAlertUnitSalesBP1->fileAlertOut) into the **extFileOut** server.
- 4 When all components have been assigned (see Figure 31), save your work so far.

Figure 31 Components from SampleBamProject Assigned to Servers in SampleBamEnv



To activate and run the project

- 1 In the Deployment Editor, after all components in SampleBamProject are assigned to the three servers in SampleBamEnv, click **Activate**
 - ♦ Or, if you have previously activated this deployment profile, click **Reactivate**
- 2 After activation is successfully completed, when the Activate dialog box asks whether you want to apply the changes to the logical host immediately, keep the checkbox selected (apply environment updates as well) and click **Yes**

5.3 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 see the chart viewer and how it updates in real time as it receives sample data.

To start monitoring the project

- 1 Open a *new* browser session. (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:

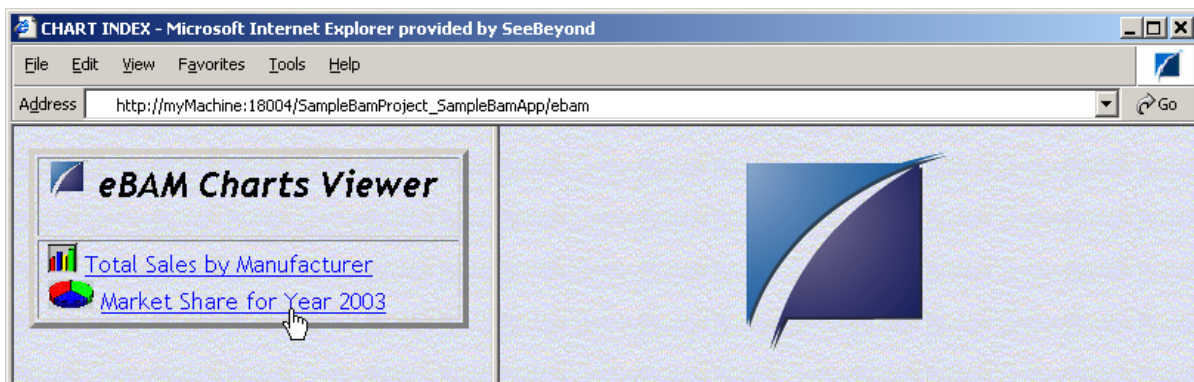
http://logicalhostname:18004/ProjName_AppName/ebam

(This is case-sensitive; the final four letters must be **ebam**, not eBAM.) For example:

http://myMachine:18004/SampleBamProject_SampleBamApp/ebam

If your Integration Server does not use port 18004, then substitute the correct port number in the URL.

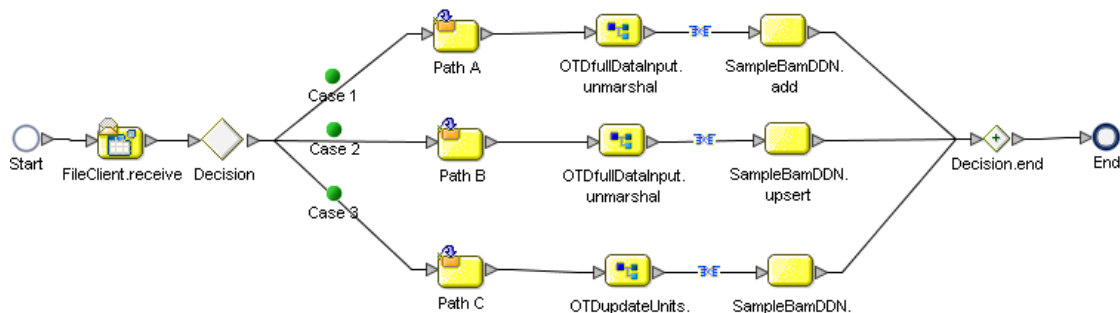
Figure 32 Initial eBAM Charts Viewer (No Data)



5.3.1 Overview of the InputUpdate Business Process

The SampleBamInputUpdateBP process reads the first character of the raw input. Then, if it finds an **A**, it parses complete data records and *adds* them; if it finds a **B**, it parses complete data records and updates the old with the new; and if it finds a **C**, it parses *partial* data records and replaces the old with the new. See Figure 34.

Figure 33 InputUpdate Business Process



5.3.2 Adding New Data: Exercising the InputUpdate Process, Path A

In this section, you will feed live data that is constructed to use the **add** logic (path A) of the SampleBamInputUpdateBP process. It will keep existing data and add new records.

To add new data and see the accumulated results

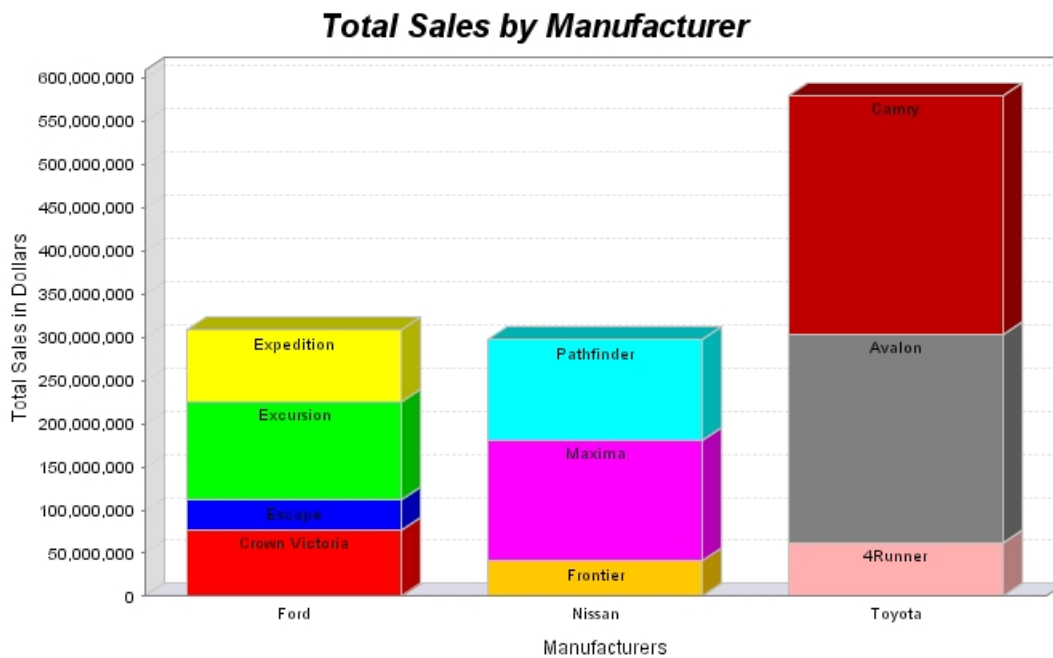
- 1 Browse (or open a command prompt and change directories) to the location where you installed the sample input data, and verify the presence of sample data files (TestAddAll.~in, TestQueryIn.~in, and so forth). For example:

```
dir C:\temp\eBAM\Sample\Data\In
```

- 2 Take a copy of **TestAddAll.~in** and rename it to **ebamInputData_All.txt** and then watch as the file is picked up and renamed to **ebamInputData_All.~in**
- 3 In the Charts Viewer, if you have not already done so, click the **Total Sales by Manufacturer** link.

Result: A bar chart shows car sales (vertical axis) compared between three categories of manufacturer (Ford, Nissan, and Toyota), with each bar consisting of a stacked series of model names. See Figure 34.

Figure 34 Stacked Bar Chart Showing Car Sales Grouped by Manufacturer and Model



Note: Your chart may look somewhat different from the this illustration, if you modified chart properties such as **Title**, **Include Legend**, and so forth.

- To see a pie chart showing market share between manufacturers for 2003 models only, click Market Share for Year 2003. See Figure 35.

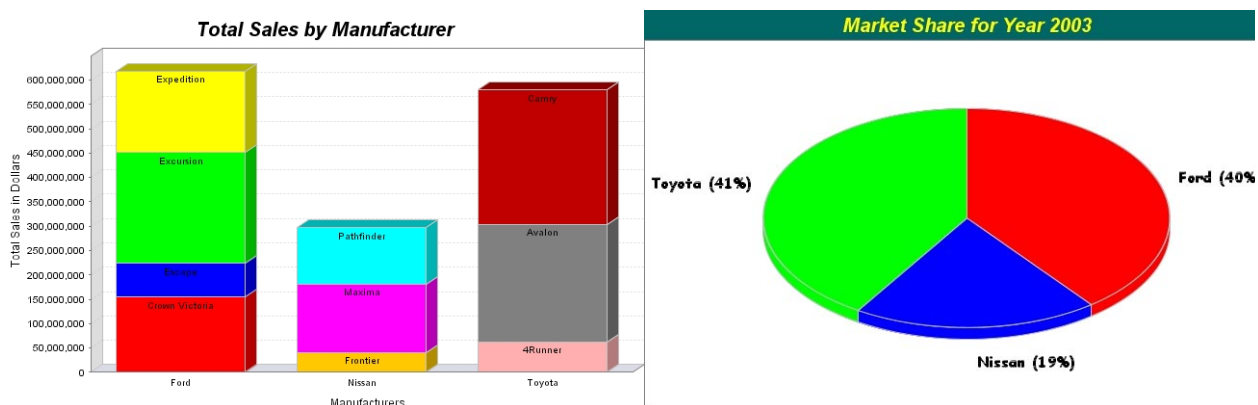
Figure 35 Pie Chart Showing Comparative Market Share for Model Year 2003



Note: Your chart may look somewhat different from the this illustration, if you modified chart properties such as **Title Color**, **Depth Factor**, and so forth.

- See the effect of feeding in additional data from a manufacturer-specific TestAdd* files. For example, copy **TestAddFord.~in** to **ebamInputData_AddFord.txt** and then watch as the file is picked up and Chart Viewer dynamically updates both charts to indicate the added sales and market share for Ford. After this, try the same operation again with a different file (or re-adding the same one). See Figure 36.

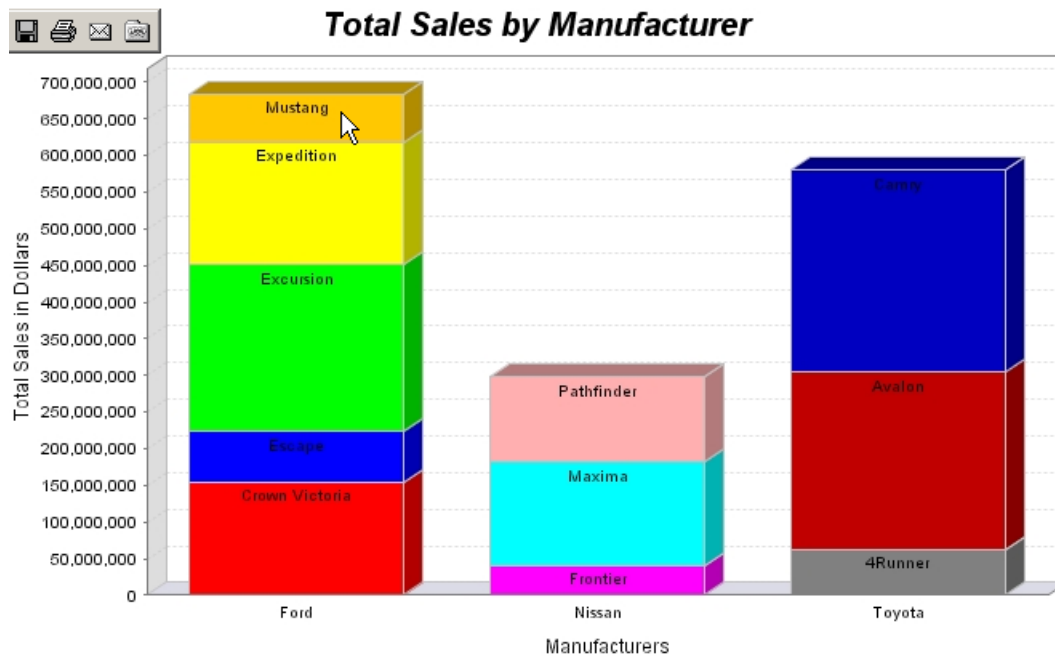
Figure 36 Bar and Pie Charts With Added Data for One Category



- 6 You can also add a new series of data and see the effect. For example, rename **TestAddMustang.~in** to **ebamInputData_AddMustang.txt** and then watch as the file is picked up and Chart Viewer dynamically updates to add the new series.

Result: A new block in a new color appears at the top of the bar for Ford, showing sales for the newly created member of the series (Mustang). See Figure 37.

Figure 37 Bar Chart With Added Data for a New Series



5.3.3 Updating Existing Data: Exercising InputUpdate Paths B and C

In this section, you will feed in live data constructed to use the **upsert** logic of the **SampleBamInputUpdateBP** process, to update old data. Path B requires a data file with a full input record; path C expects a partial data file, and updates only **carUnitsSold**.

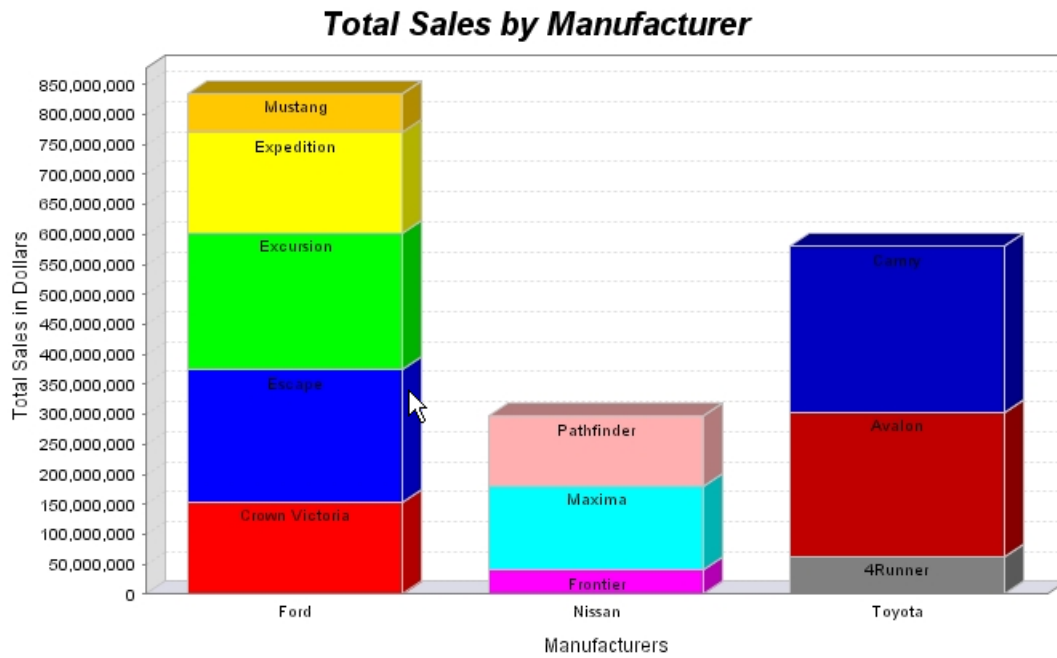
To update existing data using input that contains full data (path B)

- 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 To adjust the current data (which indicates that sales of the 2004 Escape have totaled 450 units) to an updated value of 3000 units, copy **TestReviseFullEscape.~in** to **ebamInputData_RevFullEscape.txt** and then watch as the file is picked up and Chart Viewer dynamically updates to add the new series.

Result: In the left-hand bar (Ford), the size of the medium-blue block (Escape) increases by 2550. See Figure 38.

Figure 38 Bar Chart Reflecting Revision to Complete Data Record

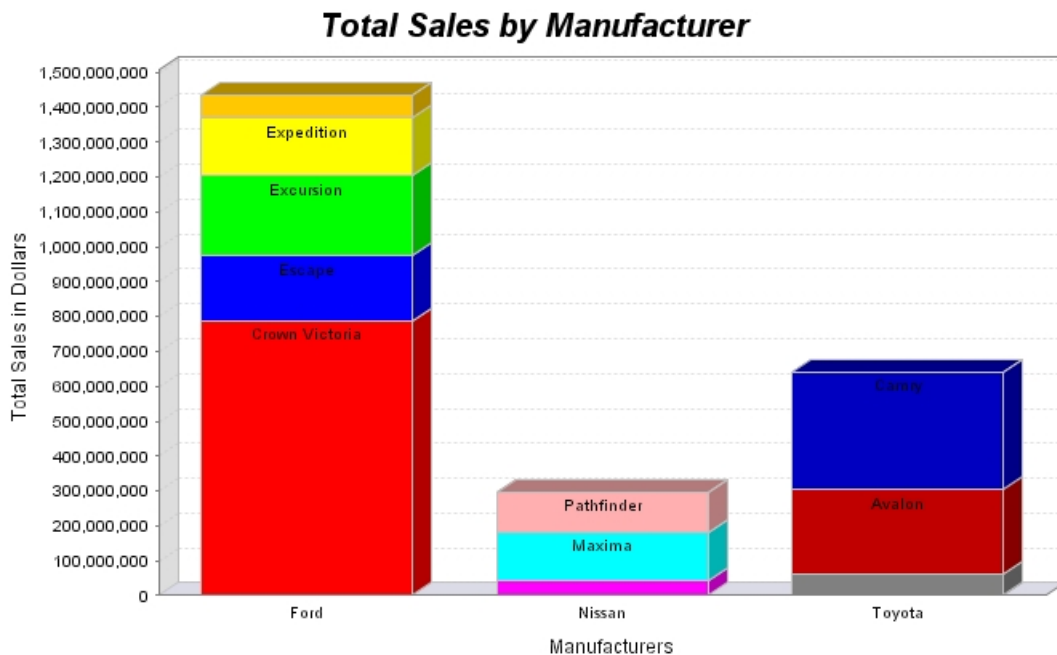


To update existing data using input that contains partial data (path C)

- To adjust the current data with a single file that contains a variety of “correction” records, copy `TestReviseUnitsMany.~in` to `ebamInputData_RevUnitsMany.txt`.

Result: Three blocks change. For Ford, the Crown Victoria block grows and the Escape block shrinks; and for Toyota, the block for Camry increases; but at the same time, the entire chart scales vertically to fit into the available space. See Figure 39.

Figure 39 Bar Chart Reflecting Revision to Multiple Partial Records

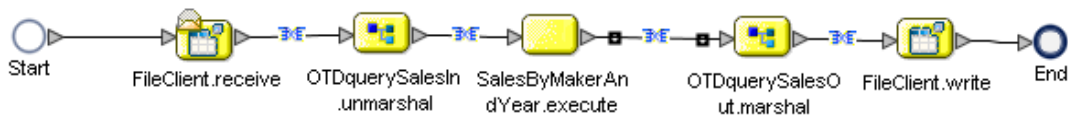


5.3.4 Executing Stored Queries: Exercising the QuerySales Process

The SampleBamQuerySalesBP process parses the input and passes it to an activity (**execute**) that looks in the existing data for records that match the input's specification of manufacturer and year. For all matches found, it calculates the gross sales for each model that combination and hands off the data to be written to a file.

In other words, it answers the question, "Given a <manufacturer> and <modelyear>, what are the gross sales of all models in the data accumulated so far?" See Figure 40.

Figure 40 QuerySales Business Process



In this section, you will feed in a request to find gross sales for each Ford of model year 2003, and receive output that reflects the current state of your data.

To update existing data using input that contains full data (path B)

- 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 Copy **TestQueryFord2003.in** to **ebamQuerySalesFord2003.txt** (its contents are simply **Ford,2003**), and then watch as the file is picked up.

Result: A new file, named **querySalesOut1.dat**, will appear in the output directory, with contents that resemble the following:

```
Ford,2003,Crown Victoria,27260750.0
Ford,2003,Excursion,43284780.0
Ford,2003,Expedition,28638000.0
Ford,2003,Escape,12891190.0
Ford,2003,Mustang,44110000.0
```

- 3 Edit **TestQueryFord2003.in** to change its contents from **Ford,2003** to **Toyota,2002** and then save it under a new name: **ebamQuerySalesToyota2002.txt**

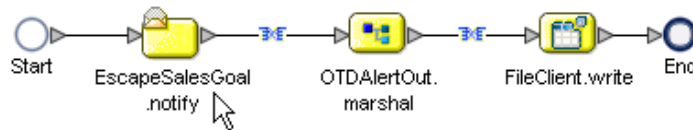
Result: A new file, named **querySalesOut2.dat**, will appear in the output directory, with contents that resemble the following:

```
Toyota,2002,Avalon,86360000.0
Toyota,2002,Camry,99697500.0
```

5.3.5 Notifying and Alerts: Exercising the AlertUnitSales

The SampleAlertUnitSalesBP process checks the data stream for records where more than 2000 Escapes were sold; when a record meets those conditions (*ModelName* = "Escape" and *UnitsSold* > 2000), the **notify** activity is triggered, the information is captured, and a file is written. See Figure 40.

Figure 41 AlertUnitSales Business Process



In this section, you will feed in a data file whose single record reports that 3000 Escapes of model-year 20004 were sold, and that this record should replace the previous one. When the file is read, it will trigger both an **upsert** and a **notify**.

To trigger a notification condition

- 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 Copy **TestRevFullEscape.~in** to **ebamInputData_ReviseEscape.txt** and then watch as the file is picked up.

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

Result 2: A new file, named **alertUnitSalesOut1.dat**, will appear in the output directory, with contents that resemble the following:

```
Threshold has been exceeded for the : Escape,Model Year:
2004,Units Sold = 3000
```

SQL Reserved Words

Because eBAM supports several dialects of SQL, it warns you against from using certain reserved words as eBAM field names. Table 14 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.

Table 18 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
CAST	CATALOG	CATALOG_NAME
CHAIN	CHAR	CHARACTER
CHARACTERISTICS	CHARACTER_LENGTH	CHARACTER_SET_CATALOG
CHARACTER_SET_NAME	CHARACTER_SET_SCHEMA	CHAR_LENGTH
CHECK	CHECKED	CHECKPOINT

Table 18 SQL Reserved Words (Continued)

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
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

Table 18 SQL Reserved Words (Continued)

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
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

Table 18 SQL Reserved Words (Continued)

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
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

Table 18 SQL Reserved Words (Continued)

OWNER	OWNERACCESS	PAD
PARAMETER	PARAMETERS	PARAMETER_MODE
PARAMETER_NAME	PARAMETER_ORDINAL_POSITION	PARAMETER_SPECIFIC_CATALOG
PARAMETER_SPECIFIC_NAME	PARAMETER_SPECIFIC_SCHEMA	PARTIAL
PASCAL	PASSEDBYVALUE	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	REFRESHLINK	REGISTERDATABASE
REINDEX	RELATION	RELATIVE
RENAME	REPAINT	REPAIRDATABASE
REPEATABLE	REPLACE	REPLICATION
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

Table 18 SQL Reserved Words (Continued)

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	SQLEXCEPTION
SQLSTATE	SQLWARNING	STABLE
START	STATE	STATEMENT
STATIC	STATISTICS	STDEV
STDEVP	STDIN	STDOUT
STORAGE	STRICT	STRUCTURE
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_COMMITTED	TRANSACTIONS_ROLLED_BACK	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

Table 18 SQL Reserved Words (Continued)

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 (chart property) 34

A

alert conditions
 configuring 28
 creating 28
 properties 31
 sending e-mail 32

Alert Editor
 (illustrated) 28
 tool palette 29

alerts, preventing duplication of 32
 arguments, adding to eBAM queries 44
 Axis Labels (bar chart properties) 35

B

Background Color (chart property) 33, 36
 bar charts
 described 33
 properties 33, 34, 35

C

Chart Editor
 (illustrated) 37
 properties 39
 tool palettes 38
 using 38
 chart types, listed and described 33
 charts
 configuring 38, 43
 creating 37
 previewing 40
 previewing (illustrated) 41
 properties 33–35, ??–36, 39
 charts, printing 41
 committing sample data 24
 configuring
 charts 38, 43
 conditions on a data definition 28
 data retention 21

 dataset views 29
 conventions
 path name separator 9
 Windows 9
 Critical Range Color (meter property) 35, 36

D

data retention
 configuring 21
 delimited file format 23
 delimiters, setting 23
 Depth Factor (chart property) 34
 Dial Type (meter property) 36
 disallowed field names 29, 64
 document
 conventions 9

E

e-mail
 sending for alert conditions 32
 encoding scheme, setting 23
 execute
 in sample implementation scenario 62
 expired data, purging 21

F

field delimiters, setting 23
 field names, disallowed 29, 64
 file format, setting 23
 fixed-width file format 23

G

GroupBy 39, 44
 (illustrated) 39, 44

I

input run-time arguments for queries 44

K

keys
 (explained) 32
 setting in conditions 31

L

Labels (bar chart properties) 35

M

metadata, validating 24
meters
 described 33
 properties 33
modifying queries 44

N

Needle Color (meter property) 36
new features in this release 10
Normal Range Color (meter property) 35, 36
Notification Interval, setting 31
notify
 in sample implementation scenario 63

O

Operator palette (illustrated) 30
OrderBy 40, 44
 (illustrated) 40, 44
Orientation (bar chart property) 35

P

pie charts
 described 33
 properties 33, 34
preventing duplication of alerts 32
previewing charts 40
 (illustrated) 41
printing charts 41
purging expired data 21

Q

queries
 creating 42
 run-time inputs for 44
queryess
 properties 44
Query Editor
 (illustrated) 42
 properties 44
 tool palettes 43
 using 43, 44

R

Radius (pie chart property) 34
record delimiters, setting 23
Resend Frequency, setting 31
reserved words 29, 64

run-time arguments for eBAM queries 44

S

sample data
 committing 24
 validating 22
sending e-mail for alert conditions 32
SQL Operator palette (illustrated) 30
SQL reserved words 29, 64

T

Title (chart property) 33
tool palettes
 Alert Editor 29
 SQL Operations (illustrated) 30
trafficlights
 described 33

U

upsert
 in sample implementation scenario 60

V

validating
 data definitions 22
 metadata 24

W

Warning Range Color (meter property) 35, 36
what's new in this release 10
writing conventions 9