SUN SEEBEYOND

eTL™ INTEGRATOR USER'S GUIDE

Release 5.1.2

Sun microsystems

Part Number: 819-7468-10

Version 20061010145209

Contents

List of Figures

Chapter 1

Introduction

This chapter provides an overview of this user's guide, including its contents and writing conventions.

What's in This Chapter

1.1 About This Document

This section describes the contents of this document as well as its conventions and intended audience.

1.1.1 What's in This Document

This guide contains the following information:

- **Chapter 2 "Overview of eTL Integrator" on page 13** provides an overview of eTL™ Integrator and provides a brief overview of the steps necessary to build and deploy eTL Projects.
- **Chapter 3 "Installing eTL Integrator" on page 17** describes how to install eTL Integrator, its documentation, and its sample Projects.
- **Chapter 4 "Building and Deploying eTL Projects" on page 22** describes how to build eTL Projects.
- **Chapter 5 "Building Business Logic for eTL Projects" on page 43** describes how to build the business logic for eTL Projects.
- **Chapter 6 "Tutorial: Building eTL Projects" on page 78** provides a tutorial for the step-by-step process of creating Projects with eTL Collaborations.
- **Chapter 7 "Managing Runtime eTL Projects" on page 104** describes how to monitor and reconfigure deployed eTL Projects.
- **Appendix A "Data Type and Operator Support" on page 109** provides information about the data types and operators that eTL Integrator supports.

1.1.2 Intended Audience

This user's guide is intended for Java™ Composite Application Platform Suite (CAPS) Project developers who have experience with Extract, Transform, and Load (ETL) concepts and SQL.

1.1.3 Text Conventions

The following conventions are observed throughout this document.

Table 1 Text Conventions

Text Convention	Used For	Examples
Bold	Names of buttons, files, icons, parameters, variables, methods, menus, and objects	▪ Click **OK**. ▪ On the **File** menu, click **Exit**. ▪ Select the **eGate.sar** file.
`Monospaced`	Command line arguments, code samples; variables are shown in ***bold italic***	`java -jar ***filename***.jar`
Blue bold	Hypertext links within document	See **Text Conventions** on page 11
Blue underlined	Hypertext links for Web addresses (URLs) or email addresses	http://www.sun.com

1.1.4 Screenshots

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

1.2 Related Documents

For more information about eGate™ Integrator, eInsight™ Partner Manager, and the eWays™ used for the sample Projects, refer to the following documents:

- *Sun Java Composite Application Platform Suite Installation Guide*
- *Sun SeeBeyond eGate Integrator User's Guide*
- *Sun SeeBeyond eGate Integrator System Administrator Guide*
- *Sun SeeBeyond eInsight Business Process Manager User's Guide*
- *Sun SeeBeyond File eWay Intelligent Adapter User's Guide*
- *Sun SeeBeyond DB2 Universal Database eWay Intelligent Adapter User's Guide*
- *Oracle eWay Intelligent Adapter User's Guide*
- *Sybase eWay Intelligent Adapter User's Guide*

- *SQL Server eWay Intelligent Adapter User's Guide*
- *Sun SeeBeyond JDBC/ODBC Adapter User's Guide*
- *Sun SeeBeyond eWay Adapter for WebLogic User's Guide*

1.3 Sun Microsystems, Inc. Web Site

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

http://www.sun.com

1.4 Documentation Feedback

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

CAPS_docsfeedback@sun.com

Chapter 2

Overview of eTL Integrator

This chapter provides an overview of eTL Integrator.

What's in This Chapter

2.1 About eTL Integrator

Extraction, Transform, and Load (ETL) is a data integration methodology that extracts data from data sources, transforms and cleanses the data, then loads the data in a uniform format into one or more target data sources.

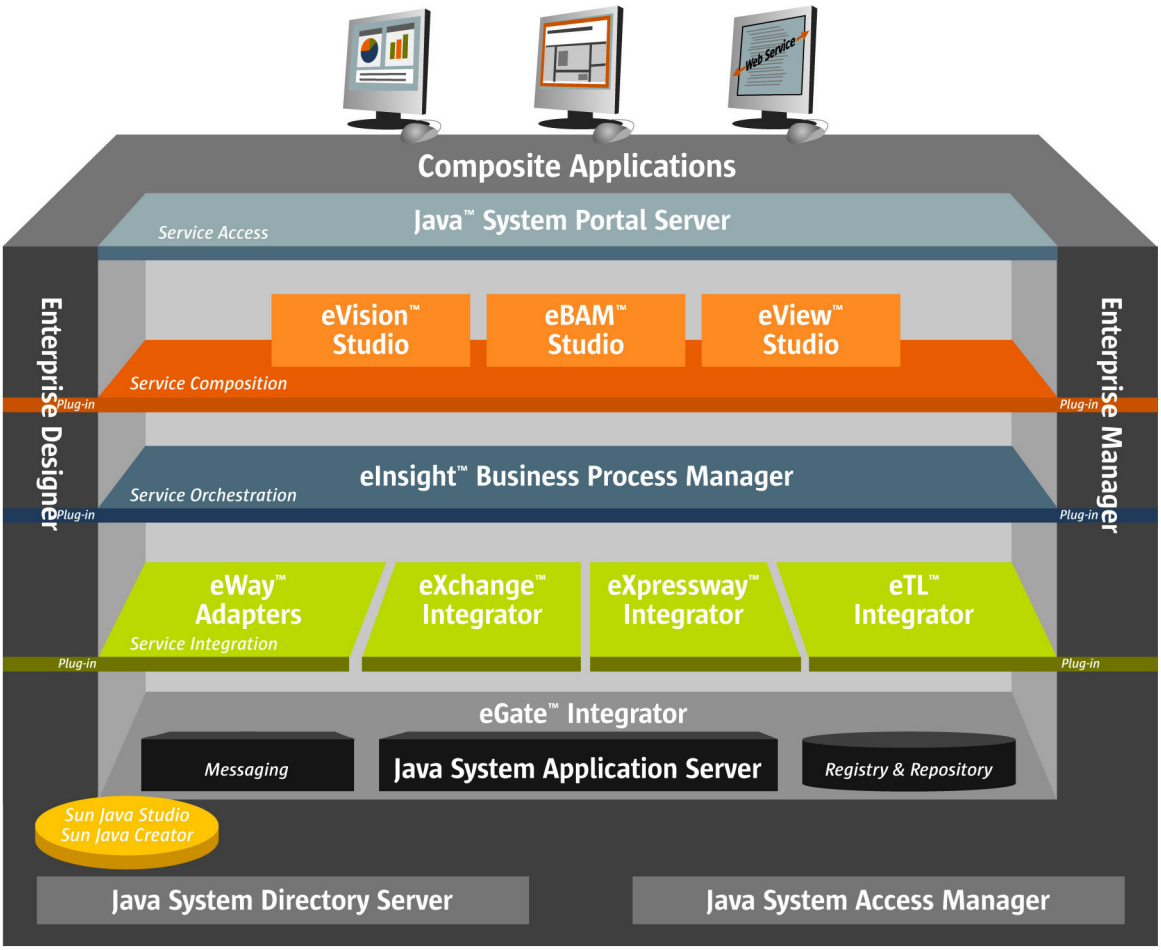
eTL Integrator provides high-volume extraction and loading of tabular data sets for Java CAPS Projects, or as a standalone product. You can use eTL Integrator to acquire a temporary subset of data for reports or other purposes, or acquire a more permanent data set for the population of a data mart or data warehouse. You can also use eTL for database type conversions or to migrate data from one database or platform to another.

eTL Integrator applies the following ETL methodology:

1. *Extraction:* the input data is extracted from data sources. Using eTL Integrator, the data can be filtered and joined from multiple, heterogeneous sources, which results in a desired subset of data suitable for transformation.
2. *Transformation*: eTL Integrator applies the operators specified for the process to transforms and cleanse the data to the desired state.
3. *Load*: The transformed data is loaded into one or multiple databases or data warehouses.

Figure 1 shows the integration between eTL Integrator and other Java CAPS products.

Figure 1 eTL Integrator and Other Java CAPS Products



2.2 Supported File Formats

eTL Integrator supports fixed and delimited tabular ASCII files, including comma-separated value (CSV) files.

2.3 Supported Data Types

eTL Projects can handle many data types; some data types can be transformed, others are merely passed through without transformation.

The list below shows the supported data types for flat file Projects:

- varchar (default)
- numeric
- time
- timestamp

If a flat file is created using the time or timestamp datatype, the data must follow one of these formats:

- yyyy-MM-dd HH:mm:ss.SSS
- yyyy-MM-dd HH:mm:ss
- yyyy-MM-dd
- MM-dd-yyyy
- HH:mm:ss

2.4 Supported Operators and Data Types

For a list of supported operators and data types, refer to **"Data Type and Operator Support" on page 109**.

2.5 Supported Databases

The JDBC eWay is now supported. See **Support for JDBC eWay** on page 77. For information about all supported databases, refer to the **eTL_Readme.txt**.

2.6 Collaboration Support

eTL Integrator supports multiple eTL Collaborations or multiple instances of the same Collaboration, running simultaneously, writing to flat files with unique names. This applies to Java CAPS Environments executing the Collaborations on one or multiple Integration Servers.

eTL Integrator does not support multiple eTL Collaborations writing the same flat file.

2.7 eWay Properties Support

eTL Integrator supports the eWay configurations; the properties defined for the eWay in the Java CAPS Environment and Project are used.

2.8 New Features in this Release

- Supports the ability to manage temporary tables.

Chapter 3

Installing eTL Integrator

This chapter describes how to install eTL Integrator, its documentation, and the sample eTL Projects. This chapter also describes how to install the eWays that are supported by eTL Integrator.

What's in This Chapter

3.1 System Requirements

The system requirements for eTL Integrator are the same as those for eGate Integrator and eInsight Business Process Manager. For information, refer to the *Sun Java Composite Application Platform Suite Installation Guide*.

3.2 Supported Operating Systems

For information about supported operating systems, refer to the **eTL_Readme.txt**.

3.3 Required Java CAPS Products

To use eTL Integrator, you must install Sun SeeBeyond eGate Integrator.

3.4 Supported Java CAPS Products

eTL Integrator currently supports the following CAPS products, available for purchase from Sun:

- Sun SeeBeyond eGate Integrator
- Sun SeeBeyond eInsight™ Business Process Manager
- Sun SeeBeyond DB2 Universal Database eWay™ Intelligent Adapter
- Sun SeeBeyond Oracle eWay Intelligent Adapter
- Sun SeeBeyond SQL Server eWay Intelligent Adapter
- Sun SeeBeyond Sybase eWay Intelligent Adapter
- Sun SeeBeyond JDBC/ODBC Adapter
- Sun SeeBeyond eWay Adapter for WebLogic

3.5 Installing eTL Integrator and Sample Projects

The procedure below describes an overview of how to install eTL Integrator. For detailed installation instructions, refer to the *Sun Java Composite Application Platform Suite Installation Guide*.

Before you install the eTL Integrator, install and download the following items using the Java CAPS Installer:

- Repository
- eGate Integrator
- Enterprise Designer
- Enterprise Manager
- Logical Host

The procedure below describes how to install the following items for eTL Integrator:

- the software
- the plug-in that enables you to monitor eTL Integrator Projects in the Enterprise Manager
- the documentation
- the sample Projects
- the File eWay (necessary to run the sample Projects)

To install eTL Integrator

1. In the **Java Composite Application Platform Suite Installer**, click **Click to install additional products**.

2 On the resulting display page, select the following items:

- **Core Product > eTL** (to install the eTL Integrator software)
- **Documentation > eTLDocs** (optional—to install the eTL Integrator documentation and sample Projects)
- **eWay > FileeWay** (optional—to install the File eWay required for sample Projects)

Click **Next** and browse to select the proper .sar files.

- eTL.sar
- eTLDocs.sar
- FileeWay.sar

3 To view the eTL Integrator documentation and download the sample Projects, click **Documentation** > **Core Products** > **Sun SeeBeyond eTL Integrator**.

You can select the following documents from the list:

- eTL Integrator User's Guide
- eTL Integrator Release Notes
- Readme
- Sample Projects

4 To download the sample Projects, click **Sample Projects** (icon), and select a location to save the **.zip** file.

3.6 Deploying the eTL Enterprise Manager Plug-in

eTL Integrator comes with a plug-in to Enterprise Manager to enable you to monitor runtime eTL Projects. The plug-in is the **eTLMonitor.emr** file downloaded during the eTL Integrator installation as described in the section above.

The procedure below describes how to deploy the plug-in using Enterprise Manager.

To deploy the eTL Enterprise Manager plug-in

- First make sure you have clicked the **Download** tab (from the Java Composite Application Platform Suite Installer), and selected **eTL Enterprise Manager plug-in** from the list.
- Click **Save.** (Save to any location. For example: `C:\JavaCAPS51\emanager`, so that **eTLMonitor.emr** is available).

Then:

1 Log into Enterprise Manager as a user with a Manager role.

2 In the Explorer pane, click the **Configuration** icon and click the **Web Applications Manager** tab.

3 Click the **Manage Applications** tab.

4 Browse to **eTLMonitor.emr** and click **Deploy**

OR

1 Click **Auto-Install from Repository**, and enter the connection properties for the Repository where you installed eTL Integrator, and click **Connect**.

The available plug-ins appear.

2 Select all the listed applications by checking the boxes.

See also: *Sun SeeBeyond eGate Integrator System Administration Guide*.

Figure 2 Management Applications Available for Installation from the Repository

Management Applications deployed on this Enterprise Manager Server

Applications	Physical Location on Server	Sessions	Status	Available Actions			
/	D:\JavaCAPS51\emanager\server\webapps\ROOT	1	running		Stop	Reload	Undeploy
/EMServices	D:/JavaCAPS51/emanager/server/webapps/EMServices	0	running		Stop	Reload	Undeploy
/FileeWayMonitor	D:\JavaCAPS51\emanager\server\webapps\FileeWayMonitor	0	running		Stop	Reload	Undeploy
/JDBCeWayMonitor	D:\JavaCAPS51\emanager\server\webapps\JDBCeWayMonitor	0	running		Stop	Reload	Undeploy
/WSeWayMonitor	D:\JavaCAPS51\emanager\server\webapps\WSeWayMonitor	0	running		Stop	Reload	Undeploy
/aaamanagementagent	D:/JavaCAPS51/emanager/server/webapps/aaamanagementagent	0	running		Stop	Reload	Undeploy
/admin	D:/JavaCAPS51/emanager/server/server/webapps/admin	0	running		Stop	Reload	Undeploy
/alerts	D:/JavaCAPS51/emanager/server/webapps/alerts	0	running		Stop	Reload	Undeploy
/balancer	balancer	0	running		Stop	Reload	Undeploy
/bannermanager	D:/JavaCAPS51/emanager/server/webapps/bannermanager	1	running		Stop	Reload	Undeploy
/cm	D:/JavaCAPS51/emanager/server/webapps/cm	0	running		Stop	Reload	Undeploy
/configuration	D:/JavaCAPS51/emanager/server/webapps/configuration	1	running		Stop	Reload	Undeploy
/deployer	D:/JavaCAPS51/emanager/server/webapps/deployer	0	running		Stop	Reload	Undeploy
/eManager	D:/JavaCAPS51/emanager/server/webapps/eManager	2	running		Stop	Reload	Undeploy
/eTLMonitor	D:\JavaCAPS51\emanager\server\webapps\eTLMonitor	0	running		Stop	Reload	Undeploy
/eWayMonitor	D:\JavaCAPS51\emanager\server\webapps\eWayMonitor	0	running		Stop	Reload	Undeploy
/graphicsRenderer	D:/JavaCAPS51/emanager/server/webapps/graphicsRenderer	0	running		Stop	Reload	Undeploy
/jmsMonitor	D:/JavaCAPS51/emanager/server/webapps/jmsMonitor	0	running		Stop	Reload	Undeploy
/logging	D:/JavaCAPS51/emanager/server/webapps/logging	0	running		Stop	Reload	Undeploy

3.7 Updating Enterprise Designer for eTL Integrator

After you installed the eTL Integrator software, update Enterprise Designer to incorporate eTL Integrator components. The procedure below describes how to update Enterprise Designer.

To update Enterprise Designer for eTL Integrator

1 (Re)start Enterprise Designer and click **Update Center** on the **Tools** menu.

2 Click **Check for Available Updates** and click **Next**.

3 Click the double-right-arrow icon and click **Next**.

This updates the Repository with applications needed for eTL Integrator.

Chapter 4

Building and Deploying eTL Projects

This chapter describes how to build and deploy eTL Projects.

What's in This Chapter

4.1 Quick Start Guide to Building eTL Projects

This section provides a quick overview of the overall process of building and deploying eTL Projects.

1. Create a Project as described in **"Creating Java CAPS Projects" on page 23**.
2. Create the flat file and/or database OTDs as described in **"Creating Flat File Database Definitions" on page 23** and **"Creating Database OTDs" on page 30**.
3. Create the eTL Collaboration Definition as described in **"Creating eTL Collaboration Definitions" on page 31**.
4. Create the Business Logic for the Business Processes or Collaborations as described in **"Building Business Logic for eTL Projects" on page 43**.
5. For eInsight users only, create the eTL Business Process as described in **"Creating Business Processes (eInsight Only)" on page 34**.

6 Create a Connectivity Map as described in **"Creating eTL Connectivity Maps" on page 36**.

7 Add the Project components to the Connectivity Map as described in **"Adding Connectivity Map Components" on page 36**.

8 Link the Connectivity Map components as described in **"Linking Connectivity Map Components" on page 38**.

9 Configure the external applications as described in **"Configuring the External Applications" on page 39**.

10 Create the eTL Collaboration Definition as described in **"Creating eTL Collaboration Definitions" on page 31**.

11 Create an Environments as described in **"Creating Java CAPS Environments for eTL Projects" on page 39**.

12 Create and activate the Deployment Profile as described in **"Creating Deployment Profiles" on page 41**.

4.2 Creating Java CAPS Projects

To create Java CAPS Projects

1 In the **Project Explorer** tab of the Enterprise Designer, right-click the Repository, and click **New Project**.

This adds a new Project under the Repository.

2 To rename the Project, right-click, **Rename**, and enter a new name.

3 Press **Enter**.

4.3 Creating Flat File Database Definitions

The procedure describes how to create flat file database definition using the Flat File OTD wizard. eTL Integrator supports fixed and delimited tabular ASCII files, including comma-separated value (CSV) files.

Note: *When a large amount of data is fed to a flat file database, a stack overflow can occur and this may in turn cause an error, "an index already exists." You should set the Thread Stack size for the JVM. This is done in the Integration Server properties by adding the JVM Args: -Xss. For example, to increase the Thread Stack size to 5 MB, the following JVM Arg would be added" -Xss5M.*

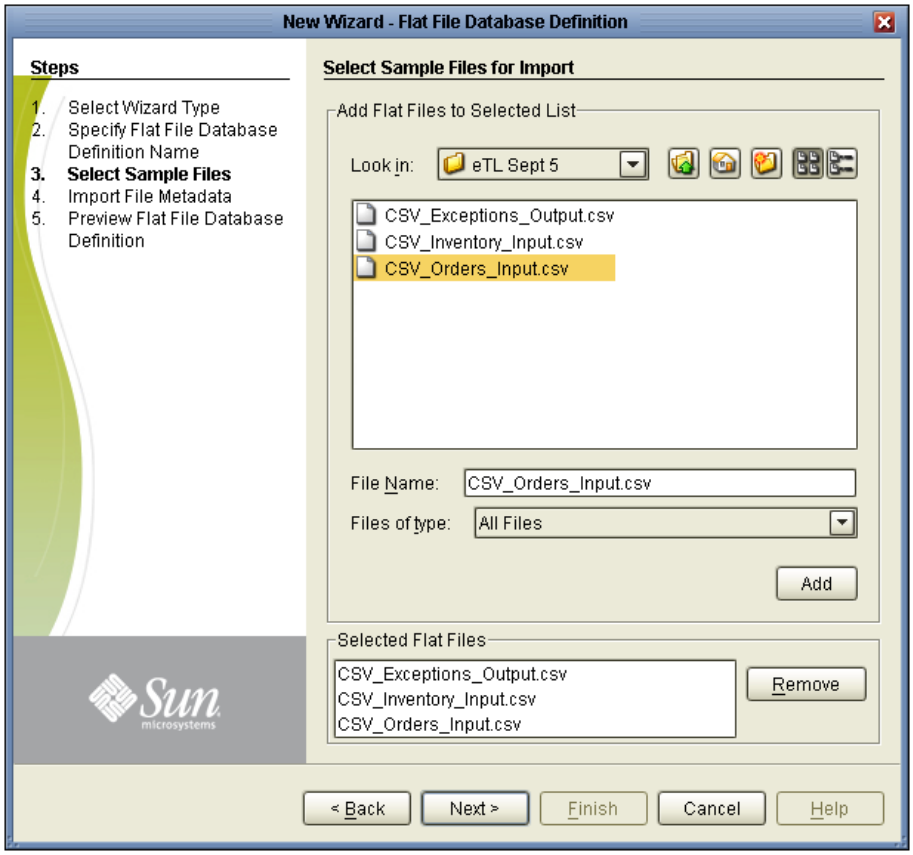
Before you start the procedure, create sample files for the input and output data. These sample files must be a small example of the runtime flat files (less than 5 MB). You must also have write permissions to the files.

To create flat file database definitions

1. In the **Project Explorer** tab of the Enterprise Designer, right-click the eTL Project, click **New**, and click **Object Type Definition**. The **New Object Type Definition** dialog box appears as shown below.
2. Click **Flat File Database Definition** and click **Next**.
3. In the **New Flat File Database Name** box, enter the name for the flat file database definition and click **Next**. The **Select Sample Files for Import** page appears.
4. In the **Look In** box, navigate to the folder where the sample input and output flat files reside. You must have write permission to the files.

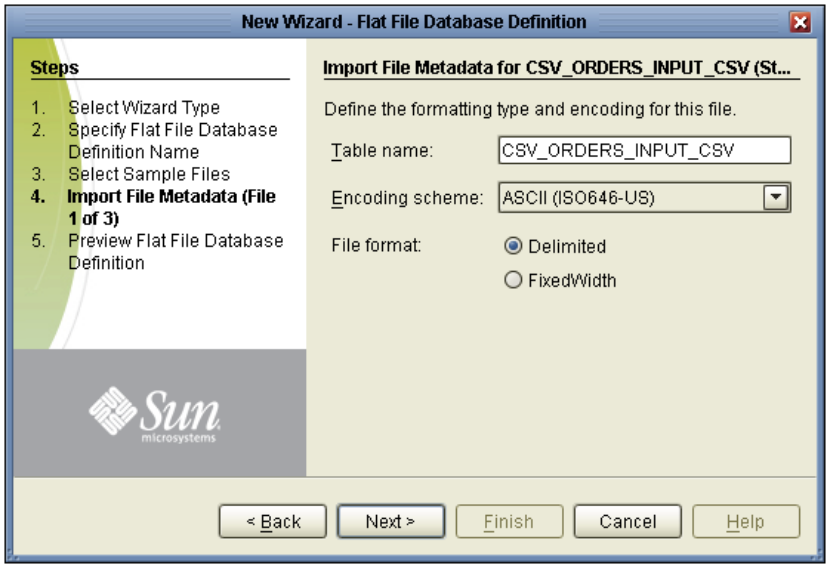
   Note: *The sample files must be small examples of the actual input and output files; the Flat File OTD wizard cannot handle file sizes over 5 MB.*
5. Double-click the files for your Project as shown in the figure below. You can add multiple flat files. This adds the files under **Selected Flat Files**.

Figure 3 Selecting Input and Output Sample Files



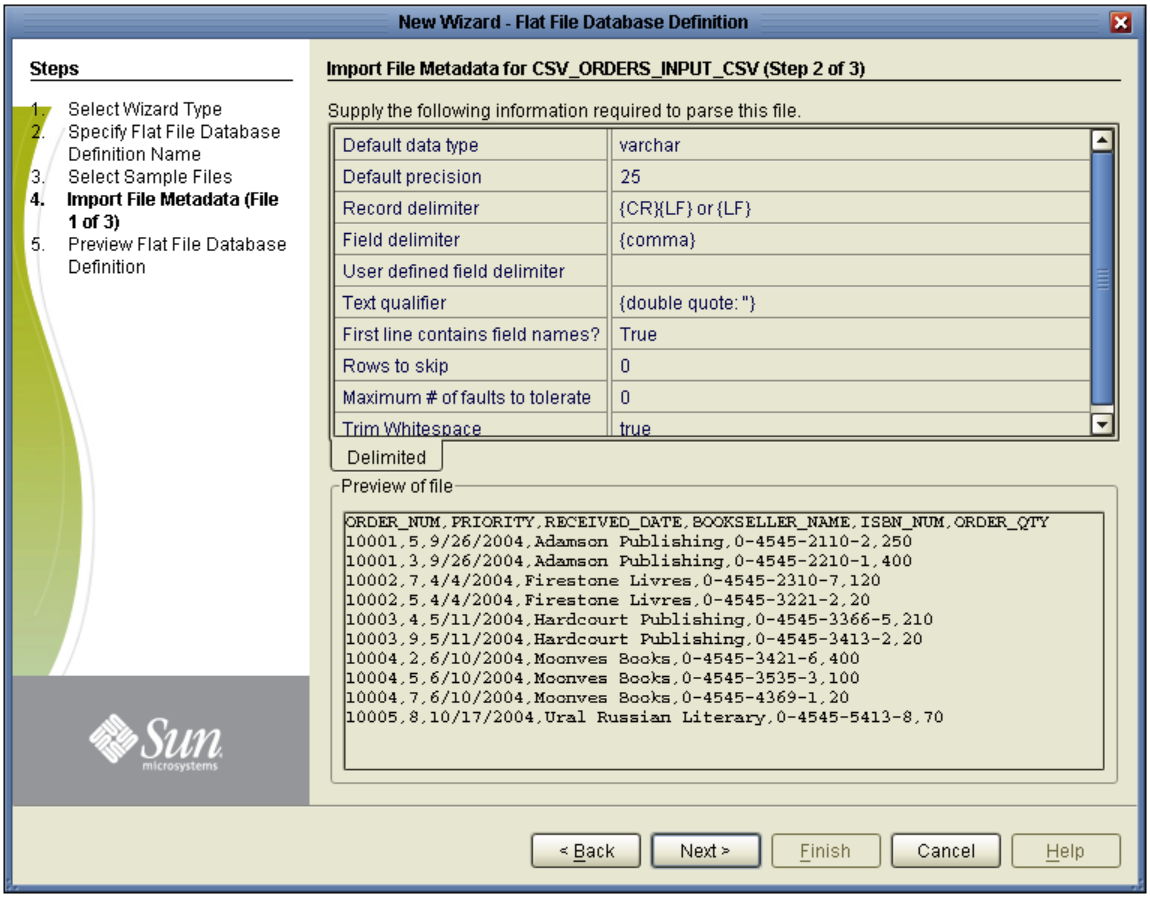
6. Click **Next**. The **Import File Metadata (Step 1)** page appears.

Figure 4 Importing File Metadata—Step 1



7 Click **Next**. The **Import File Metadata (Step 2)** page appears.

Figure 5 Import File Metadata (Step 2 of 3)



Refer to the following tables and supply the parsing data. (The previous figure already has the correct settings for this sample.)

8 Click **Next**.

9 For a delimited file format (or fixed file format), set the options based on the table below to indicate how to parse the flat file and click **Next**. Comma delimited is the default.

Table 2 Metadata Settings and Options – Delimited Format

This Setting	Indicates	Options
Default data type	The default data type used for all elements in the flat file database structure (you can change the default in subsequent steps). For information about supported data types and the formats required, refer to the readme text file.	*varchar (default)* numeric time timestamp
Default precision	The precision allowed for the datatype.	20 (default) For numeric data types, enter <= 38 For time/timestamp data types, enter the length of the format
Record delimiter	The delimiter that separates flat file records	newline (LF) carriage return (CR) CR LF *CR LF or LF (default)* semicolon (;) comma (,) tab pipe (|)
Field delimiter	The delimiter that separates fields in flat file records	*comma (default)* tab semicolon pipe User Defined
User-defined field delimiter	The delimiter that separates files in flat file records. Use this field when you select the **User Defined** option in the **Field delimiter** setting.	
Text qualifier	The qualifier used to indicate text	none *double quote: " (default)* single quote: '
First line contains field names?	Whether the names specified in the header row are used as field names (**True**) or whether eTL Integrator assigns default field names (**False**)	*True (default)* False
Rows to skip	The number of rows to skip before the starting row	*>= 0* *0 (default)*

Table 2 Metadata Settings and Options – Delimited Format (Continued)

This Setting	Indicates	Options
Maximum # of faults to tolerate	The upper limit of the number of faults before which an error message occurs	>= *0* *0 (default)*
Trim Whitespace	Strip whitespace and tabs from the beginning and end of a string	*True (default)* False

Table 3 Metadata and Options – Fixed-width Format

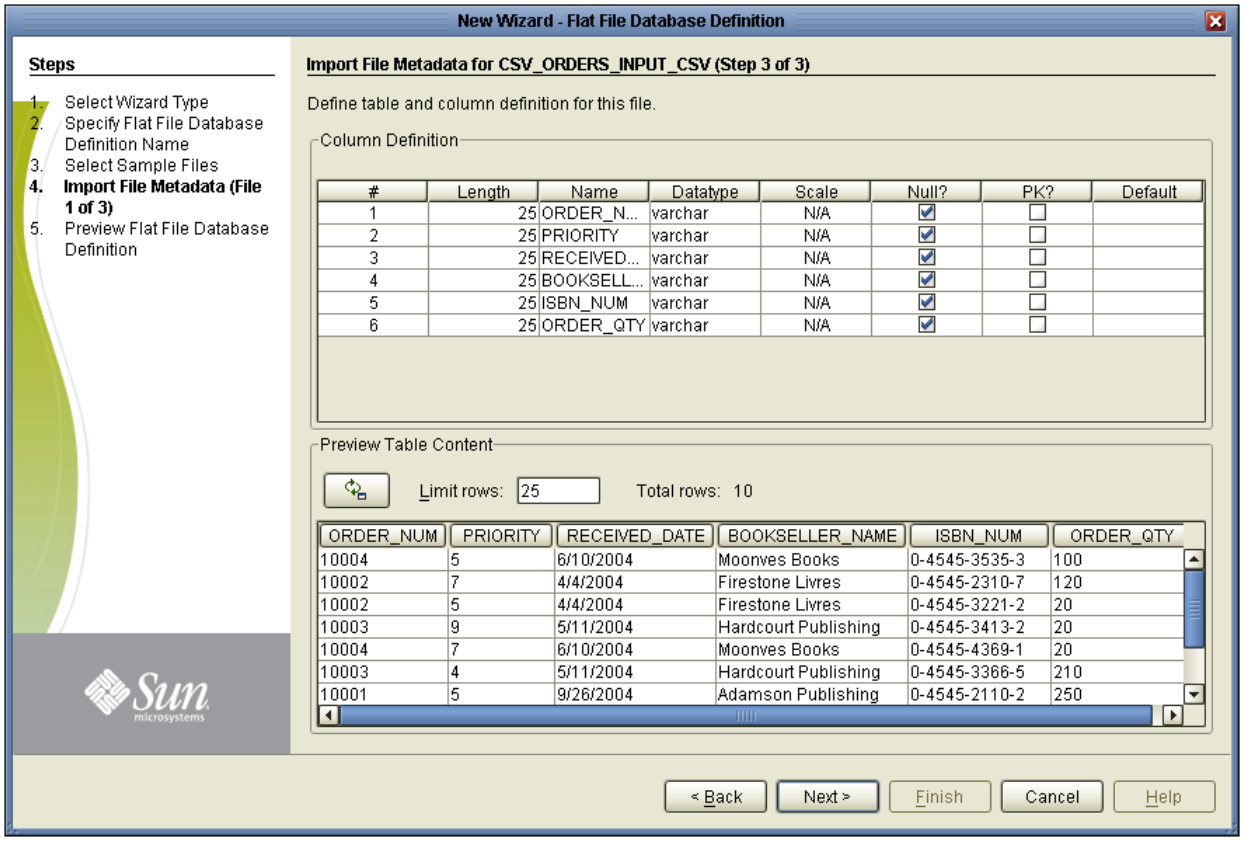
This Setting	Indicates	Options
Default data type	The default data type used for all elements in the flat file database structure (you can change the default in subsequent steps). For information about supported data types and the formats required, refer to the readme text file.	*varchar (default)* numeric time timestamp
Record length	The maximum length of a record in number of characters. This option must be appropriate for the selected data type and must be the same for all fields.	> 0 *default depends on the flat file used*
Field count	The number of fields per record	> 1 *default depends on the flat file used*
Record delimiter	The delimiter that separates flat file records	*CR LF (default)* none newline (LF) carriage return (CR)
First line contains field names?	Whether the names specified in the header row are used as field names (**True**) or whether eTL Integrator assigns default field names (**False**)	*True (default)* False
Header offset	The number of bytes to skip before reaching the start of the first record; value is ignored if the **First line contains field names** setting is set to **True**	>= 0
Records to skip	The number of records to skip before the starting record	>= *0* *0 (default)*
Maximum # of faults to tolerate	The upper limit of the number of faults before which an error message occurs	>= *0* *0 (default)*
Trim Whitespace	Strip whitespace and tabs from the beginning and end of a string	*True (default)* False

Table 4 Metadata – Column Definition

This Setting	Indicates	Options
#	Column number	
Length	The maximum number of digits to use in representing a number field; for example, the number 9876.543 has a precision of 7	> 0 (must be appropriate for the selected data type)
Name	The name of the column	unlimited number of characters
Datatype	Type of data	*varchar (default)* time numeric timestamp
Scale	The number of digits to the right of the decimal point in a number field; for example, 9876.543 has a scale of 3	> 0
Null	Whether the field can be null	Select (the field can be null) or deselect (the field cannot be null)
PK	Primary Key	Select or deselect Primary Key
Default	Option to add default data to a column	

The **Import File Metatdata (Step 3)** dialog appears. See the following figure.

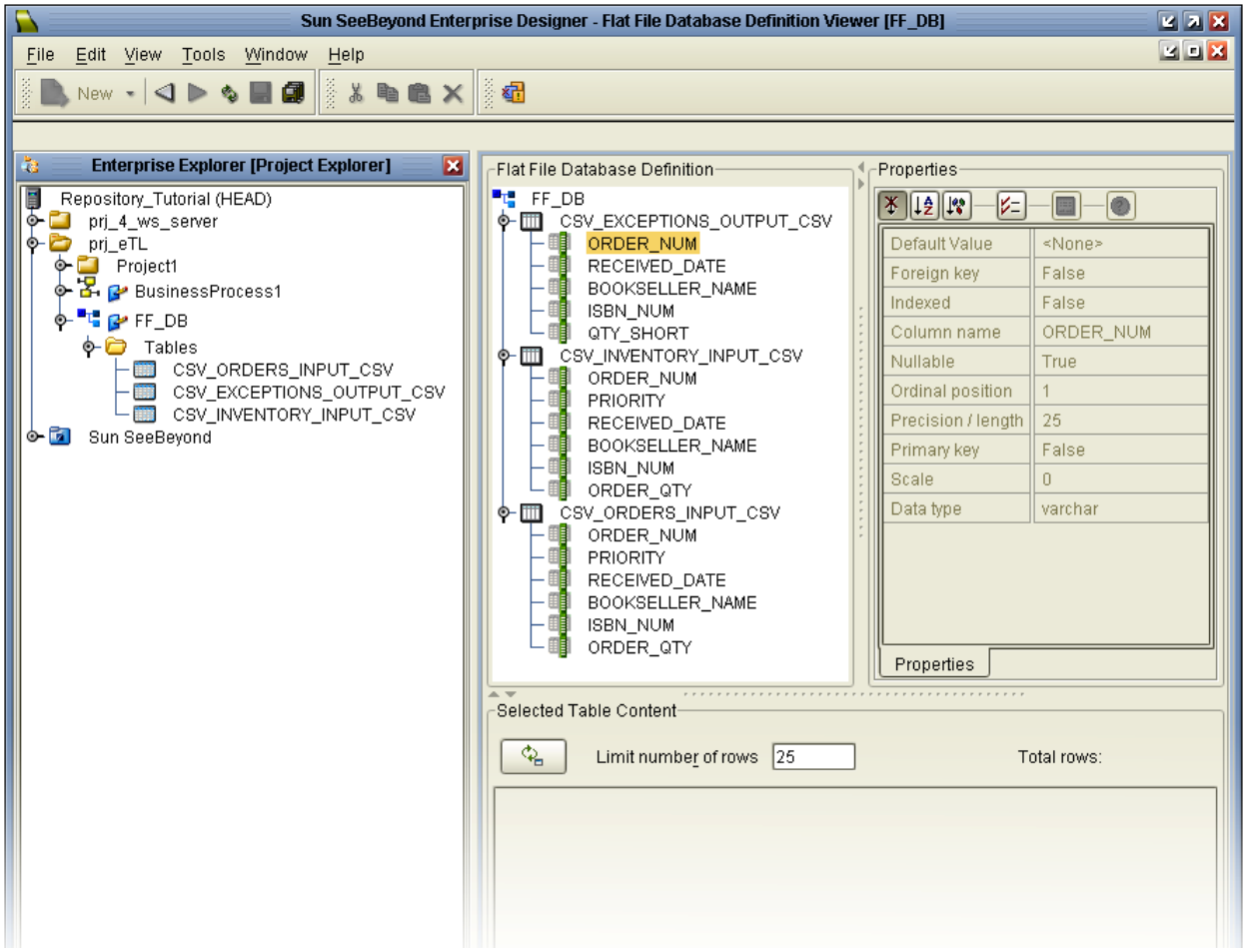
Figure 6 Import File Metadata (Step3 of 3)



10 Click **Next**.

After you have finished importing the metadata for the first file, do the same for the remaining two flat files in the sample.

Figure 7 Verifying the Flat File Database Definition



11 Click **Finish**. The new OTD displays in the **Project Explorer** tab in the Enterprise Designer.

12 Expand the OTD and the flat file definitions, and click each record to verify that the properties are correct.

Records are marked with either green or black icons. A green icon indicates that a record is nullable. Records marked with black icons cannot contain a null value.

4.4 Creating Database OTDs

eTL Integrator supports several database types as described in the readme file. You must have the eWay for the particular database type installed. For instructions on creating an OTD for a database, refer to the eWay user's guide for that database. For a list of eWay documents, refer to **"Related Documents" on page 11**.

4.5 Creating eTL Collaboration Definitions

Once you have created the OTDs for the eTL Project, create the eTL Collaboration Definitions as described in the procedure below. The eTL Collaboration Definition defines the source tables, the source files to be joined, and the target tables.

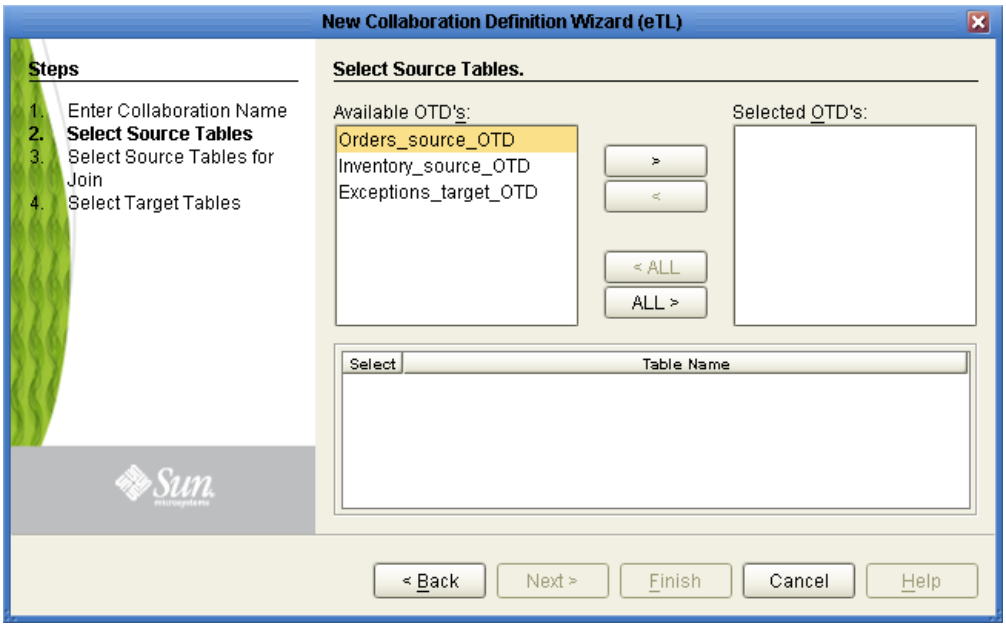
To create eTL Collaboration Definitions

1. In the **Project Explorer** tab of the Enterprise Designer, expand the eTL Project, right-click the eTL Project, click **New**, and click **Collaboration Definition (eTL)**.

   The **New Collaboration Definition Wizard (eTL)** dialog box appears.

2. Enter a name for the Collaboration and click **Next**. The **Select Source Tables** page appears.

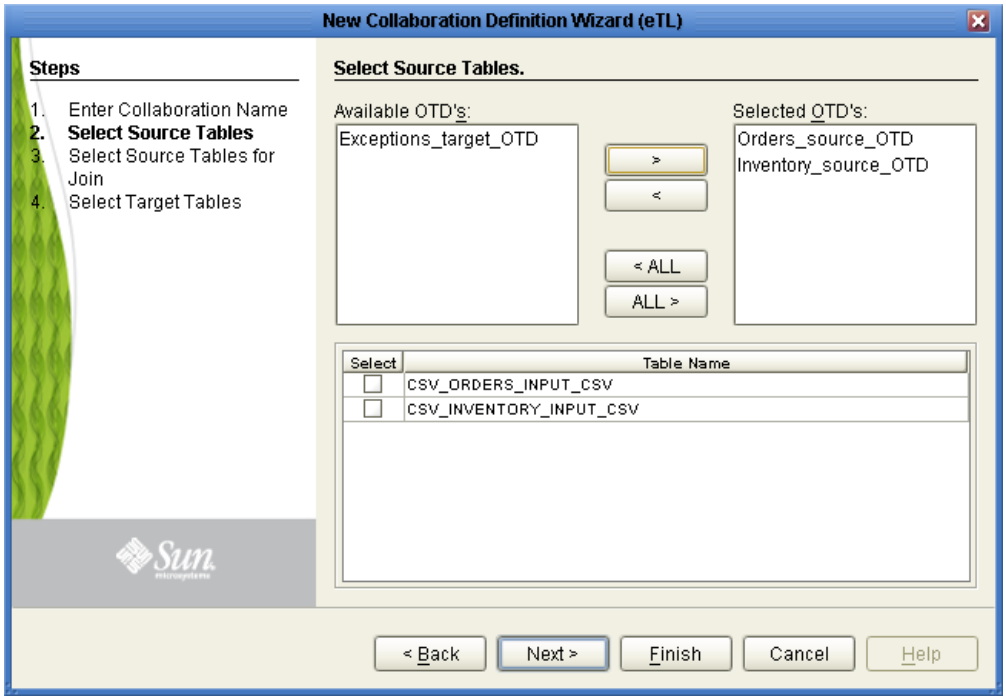
Figure 8 Selecting Source Tables



3. Click the OTDs under **Available OTDs** and click the right arrow to move them to under **Selected OTDs**.

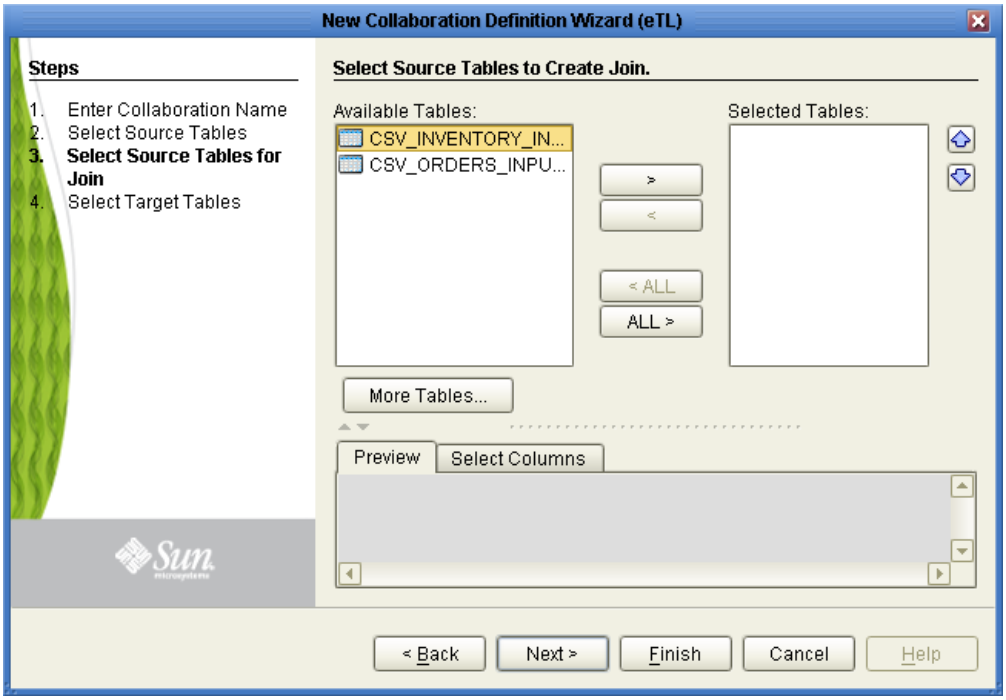
4. Under **Select**, select the tables to be used as source tables. The figure below shows an example of selected source tables.

Figure 9 Selecting Source Tables



5 Click **Next**. If there are multiple tables, the **Select Source Tables for Join** page appears.

Figure 10 Selecting Tables for Joining



6 Under **Available Tables**, select the tables to be joined and click the right arrow to move them under **Selected Tables**. (If not joining, click Next.)

7 Click **Next**. The **Select Target Tables** page appears.

8 Click the OTDs that contain the target tables under **Available OTDs** and click the right arrow to move them to **Selected OTDs**.

9 Under **Select**, select the tables to be used as target tables.

10 Click **Finish**. The Collaboration displays in the eTL Collaboration Editor window.

4.6 Viewing Flat File Database Definitions

To view an existing flat file database definition, follow the procedure below.

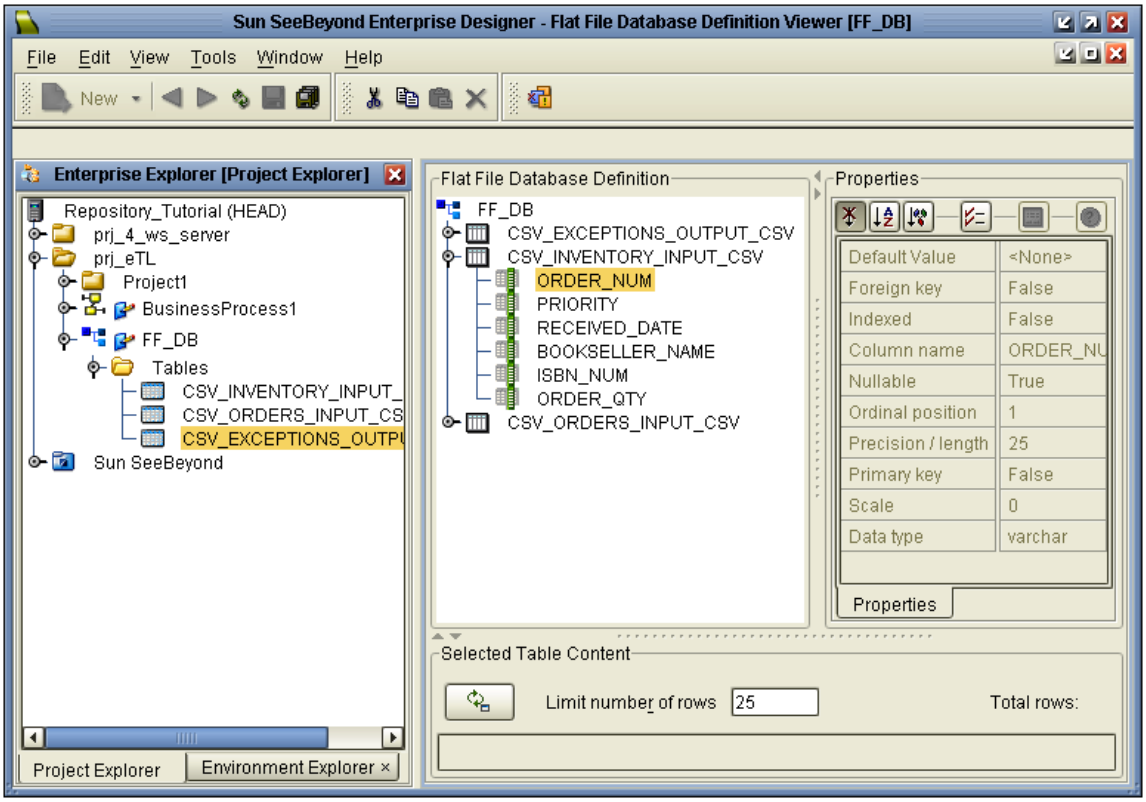
To view flat file database definitions

- In the **Project Explorer** tab of the Enterprise Designer, expand the Repository and the eTL Project, right-click the OTD to be viewed, and click **Open**.

This displays the Flat File Database Definition Viewer. The figure below shows an example.

To view table content, click the table name in the **Flat File Database Definition** panel and click **Show data** in the **Selected Table Content** panel. To change the number of rows displayed from the default 25, enter the value in **Limit number of rows** and refresh the window.

Figure 11 Viewing Flat File Database Definitions



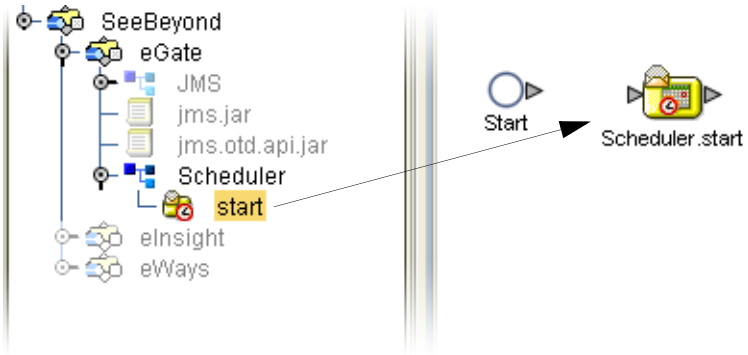
4.7 Creating Business Processes (eInsight Only)

If you are using eGate Integrator with eInsight Business Process Manager, use the procedure below to create the eTL Business Process.

To add Connectivity Map components

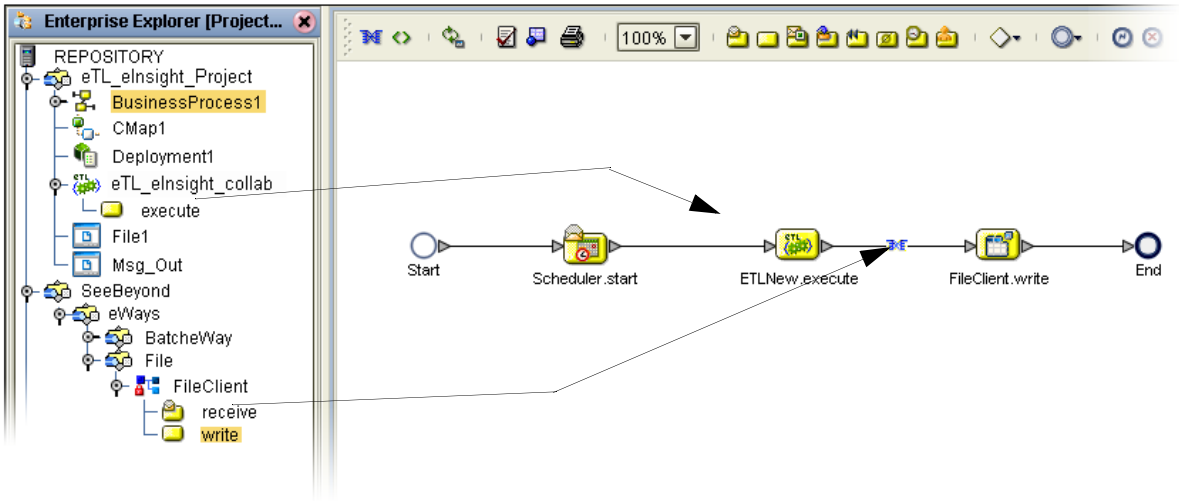
1. In the **Project Explorer** tab of the Enterprise Designer, right-click the eTL Project, click **New**, and click **Business Process**.
2. To rename the Business Process, right-click, **Rename** and press **Enter**.
3. Click and expand the **Sun SeeBeyond** services.
4. Click and expand your **eGate** Service.
5. Click and expand your **Scheduler** Service.
6. Click and drag **start** to the Business Process designer pane.

Figure 12 Scheduler for Business Process



7 Drag in the other Services as shown in the following figure.

Figure 13 Business Process Map



8 Drag the eTL Collaboration's subfolder **execute** to the designer map and place it just to the right of the **FileClient.receive** Service.

9 Drag the **Write** Service to the map and place it to the right of the eTL Collaboration.

10 Connect the nodes to all components from Start to End.

11 Click **Save**.

To create a File Client Write Business Rule

1 Add a Business Rule between the eTL Collaboration and the **FileClient.write** Service. This rule formats the output for the status and count messages.

2 Right-click the node between the **eTL Collaboration Service** and the **FileClient.write Service** and select Add Business Rule.

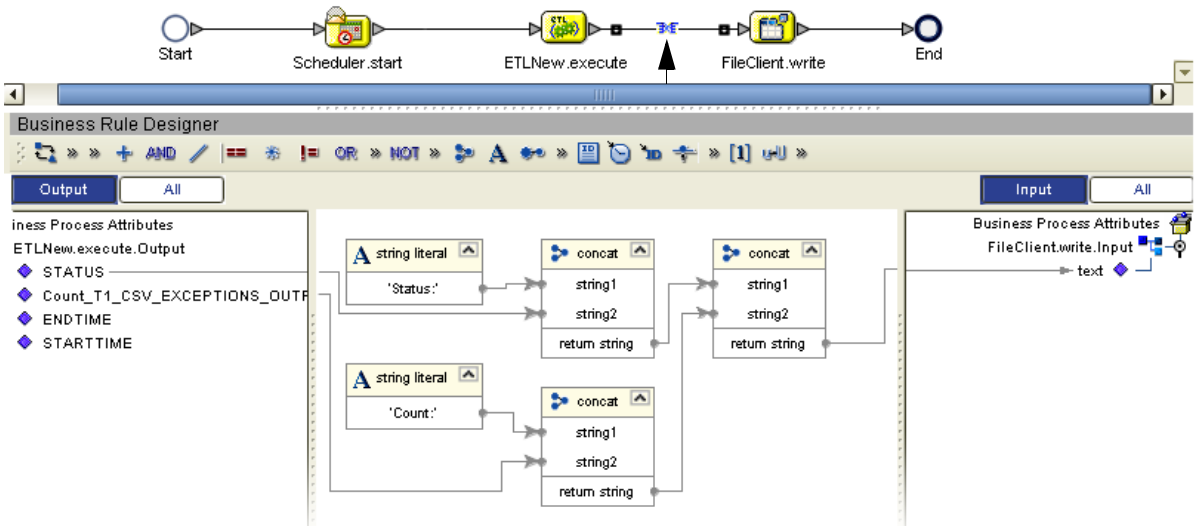
3 Double-click the Business Rule icon to display the Business Rules Designer pane.

Figure 14 Business Rule Icon

You will create the business rules by dragging operators to this work space.

4 Connect the nodes in the Business Rule Designer as shown in Figure 15 to create the Business Rule.

Figure 15 Business Rule



4.8 Creating eTL Connectivity Maps

To create eTL Connectivity Maps

- In the **Project Explorer** tab of the Enterprise Designer, right-click the eTL Project, click **New**, and click **Connectivity Map**. This adds a new Connectivity Map under the eTL Project.

To rename the Connectivity Map, right-click, **Rename** and press **Enter**.

4.9 Adding Connectivity Map Components

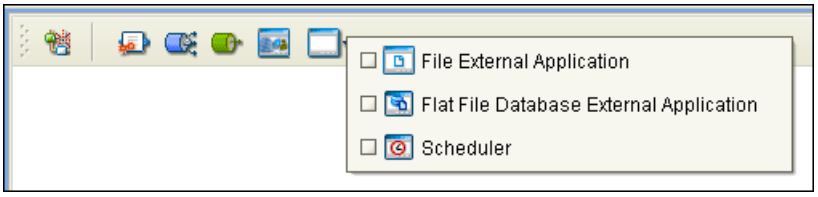
To add Connectivity Map components

1 In the **Project Explorer** tab of the Enterprise Designer, click the Connectivity Map for the eTL Project.

2 Add necessary components such as schedulers and Services.

3 To add a flat file application, click the **External Applications** icon and click **Flat File External Application**.

Figure 16 Selecting the Flat File Application

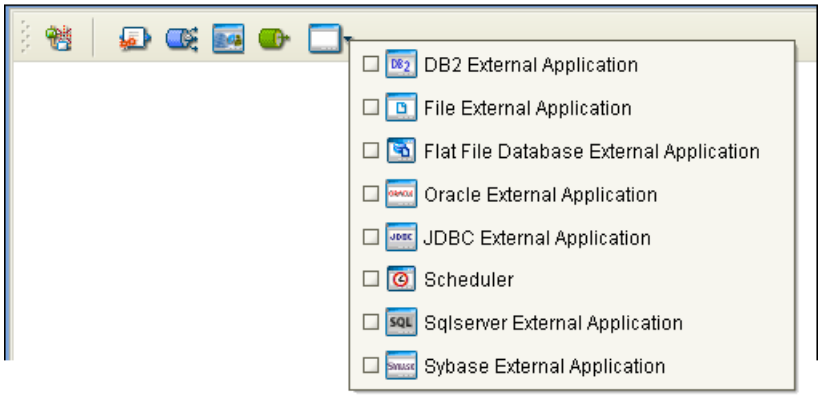


4 Drag the **Flat File External Application** icon from the Connectivity Map toolbar onto the Connectivity Map canvas.

5 To add eTL supported and installed eWay components to the Connectivity Map, click the **External Applications** icon, click any of following options, and drag the icon to the canvas:

- **DB2 External Application** for DB2 Universal Database eWays
- **File External Application** for File eWays
- **Oracle External Application** for Oracle eWays
- **SqlServer External Application** for SQL Server eWays
- **Sybase External Application** for Sybase eWays
- **JDBC External Application** for JDBC eWays

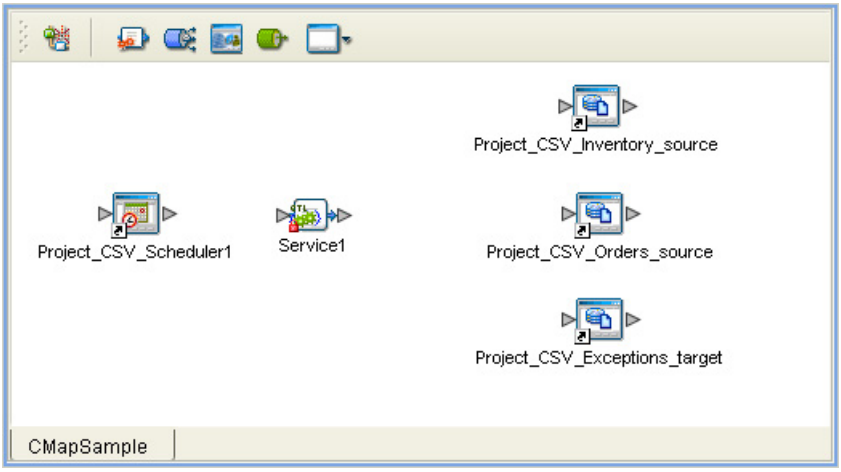
Figure 17 Selecting Supported eWay Components



6 To rename an external application, right-click the icon, click **Rename**, and enter a new name.

The figure below shows an example of a Connectivity Map for an eTL Project.

Figure 18 Connectivity Map for an eTL Project



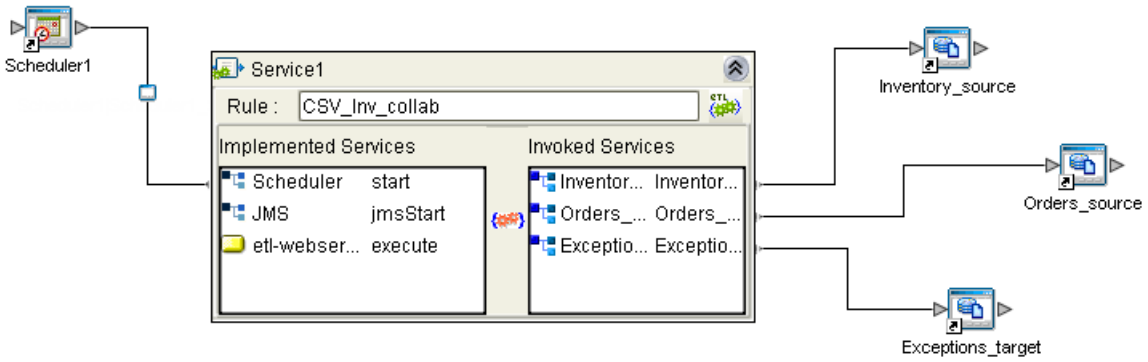
Note: *eTL Collaborations can be invoked by a Scheduler, Business Process, Topic, or Queue.*

4.10 Linking Connectivity Map Components

To link Connectivity Map components

1. In the **Project Explorer** tab of the Enterprise Designer, click the Connectivity Map for the eTL Project.
2. Open the eTL Collaboration container.

Figure 19 Linking Connectivity Map Components



3. Link from eTL Collaboration to Scheduler, JMS Topic or Queue.
4. Link from the Collaboration to the flat file eWay, database eWay, etc.

4.11 Configuring the External Applications

You configure the logical properties of the Scheduler and eWays in the Connectivity Map. Physical properties are configured in the Enterprise Designer Environment. For information about configuring the supported eWays, refer to the eWay documentation.

4.12 Creating Java CAPS Environments for eTL Projects

To create Java CAPS Environments for eTL Projects

1. In the **Environment Explorer** tab of Enterprise Designer, right-click the Repository and click **New Environment**.
2. Right-click the Environment and click **New Logical Host**.
3. Right-click the Logical Host and click **New SeeBeyond Integration Server.**
4. To add a flat file external application, right-click the Environment, click **New Flat File Database External System**, enter the name for the flat file database application, and click **OK**.
5. To add database eWays, right-click the Environment, click any of the database external systems as applicable to your Project design, enter the component name, select inbound or outbound, and click **OK**.

   For detailed information, refer to the eWay documentation. The following eWays are supported:

   - Outbound Oracle, Sybase, and SQLServer eWay (not XA)
   - Outbound DB2 eWay for Windows/UNIX
   - Outbound JDBC eWay
6. Configure the Logical Host, Integration Server, and database eWays by right-clicking the component in the **Environment Explorer** tab and clicking **Properties**. (If you have a Topic or Queue, you must configure the Scheduler, Logical Host, and JMS as well.)

Note: *You must set the Application Workspace Directory property for the Integration Server or Java System Application Server. A folder <Application Workspace Directory>/eTL/m/$eTL_collab_svc_Id> is created and removed whenever a Collaboration is deployed and undeployed. This folder may not be deleted during undeployment of a Collaboration if a collaboration is running. If this happens, remove the folder before deploying the same Collaboration again.*

A sub folder under <Application Workspace Directory>/eTL/i/> may be created and removed for each Collaboration instance that is executed. If this folder is not empty when none of the eTL Collaborations are running, it is safe to remove content of this folder.

7 Enter the properties and click **OK**.

For information about eWay configurations, refer to the eWay documentation. For information about Logical Host and Integration Server configurations, refer to the eGate documentation.

8 To configure the flat file external system, refer to the section below.

4.13 Configuring the Flat File Database External Application

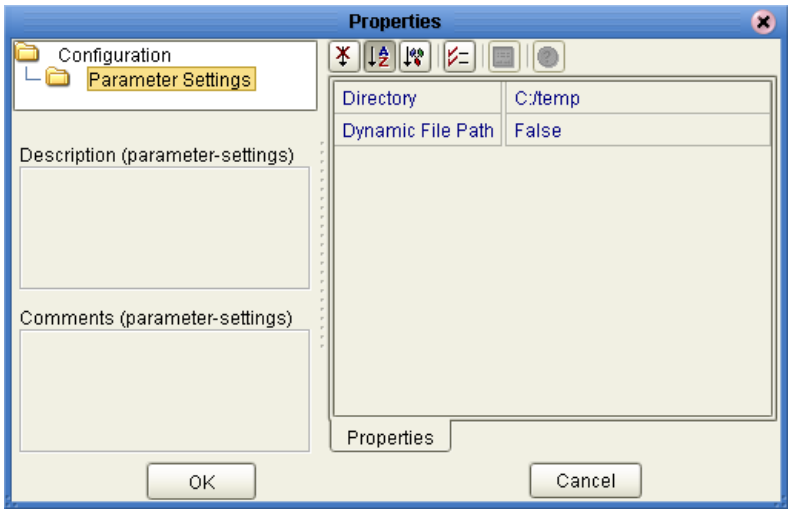
To configure the flat file database external application, follow the procedure below. The application is configured in the Java CAPS Environment. To create a Java CAPS Environment, refer to **"Creating Java CAPS Environments for eTL Projects" on page 39**.

To configure flat file database external applications

1 In the **Environment Explorer** tab of the Enterprise Designer, expand the eTL Environment and right-click the flat file database external application.

2 Click **Properties**. The **Properties** dialog box appears as shown below.

Figure 20 Configuring Flat File External Applications



3 In the **Directory** box, enter folder where the flat files reside.

4 To use a file path for dynamic files, set the **Dynamic File Path** property to **True**. This property is disabled by default.

If the **Dynamic Flat File** property is disabled, the **Default Value** setting for the runtime argument is used for the table name, and the **Directory** property is used as the path. If the filename is passed in the Business Process, the **Directory** property is used and the Business Process information is appended.

If the **Dynamic Flat File** property is enabled, the **Directory** property setting is ignored. eTL looks for the fully qualified path and filename in the Business Process.

If it cannot find those, eTL searches for the path and filename in the **Default Value** setting for the runtime argument.

Passing filenames dynamically at runtime requires the filenames to be specified as described in **"Passing Flat Filenames at Runtime Inputs" on page 56**.

5 Click **OK**.

4.14 Creating Deployment Profiles

Deployment Profiles indicate which service runs on which system in the eTL Environment you created. Before linking the services to the systems, you must have created the Environment as described in **"Creating Java CAPS Environments for eTL Projects" on page 39**.

To create Deployment Profiles

1 In the **Project Explorer** tab of the Enterprise Designer, right-click the Project, click **New**, and click **Deployment Profile**.

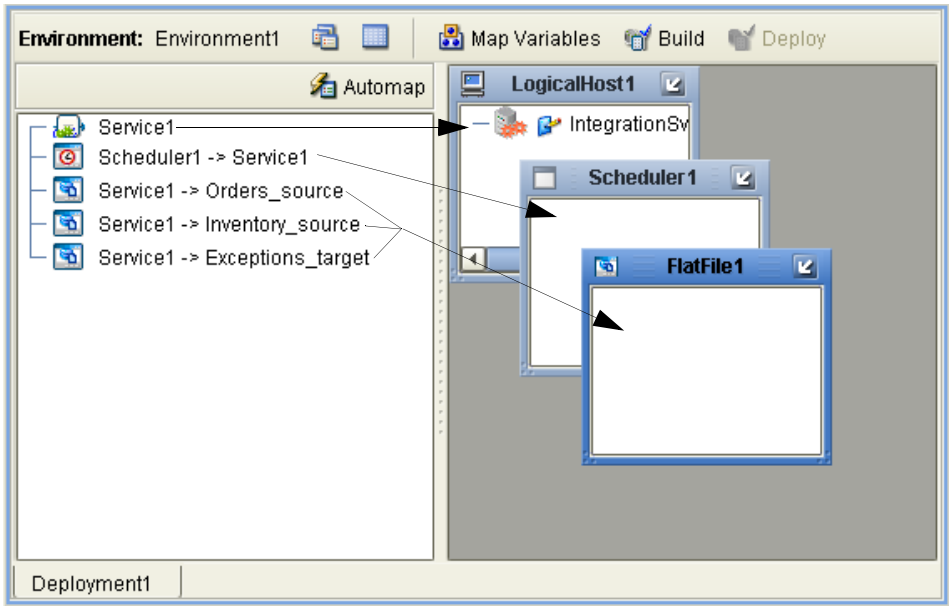
2 Enter the name for the Deployment Profile, click the Environment you created for the Project, and click **OK**.

3 Drag the services into the systems or use Automap. (Multiple locations need manual drag-and-drop.)

The figure below shows an example of a Deployment Profile for an eTL Project. The eTL Collaboration runs on the Logical Host; the scheduler service runs on the Scheduler external system, and the flat file services run on the flat file external system.

If the flat files reside in different locations, create separate flat file database external applications for each location.

Figure 21 Creating Deployment Profiles



For instructions on activating deployments and running Projects, refer to the *eGate Integrator User's Guide.*

Chapter 5

Building Business Logic for eTL Projects

This chapter describes how to build business logic for eTL Projects. Project business logic is contained in Business Processes for eInsight, and in Collaborations for eTL Integrator used without eInsight.

To build data transformations, you use the eTL Collaboration Editor. For information about creating flat-file database definitions and database OTDs, refer to the previous chapter.

What's in This Chapter

5.1 Opening eTL Collaborations for Editing

The procedure below describes how to open an eTL Collaboration for editing. For information about creating the Collaboration, refer to **Creating eTL Collaboration Definitions** on page 31.

To open eTL Collaborations for editing

1. In the **Project Explorer** tab of Enterprise Designer, expand the Repository and the eTL Project.

   If the Collaboration has a red lock icon, it is checked in. Before you can edit the Collaboration, check it out as described in the next step.

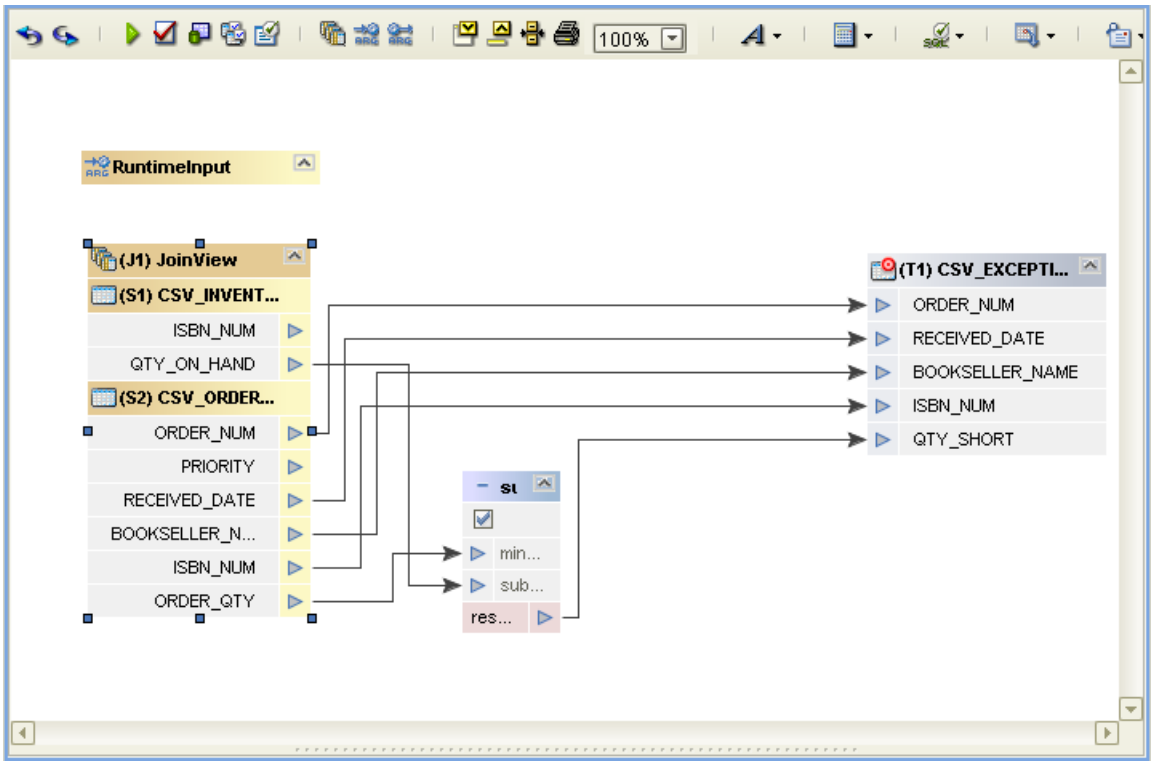
   If the Collaboration does not have the red lock icon, continue with step 3.

2. Right-click the Collaboration to be edited, click **Version Control**, click **Check Out** if the option is not greyed out, and click **Check Out**.

If the **Check Out** option is unavailable, the Collaboration is already checked out and available for editing. See "Version Control" in the *eGate Integrator System Administration Guide*.

3 Right-click the Collaboration and click **Open**. The Collaboration appears in the eTL Collaboration Editor similar to the following figure.

Figure 22 Opening Collaborations for Editing



5.2 Using the eTL Collaboration Editor

You use the eTL Collaboration Editor to create the business logic for eTL Collaborations. The table below describes the eTL Collaboration Editor toolbar

Table 5 Using the eTL Collaboration Editor Toolbar

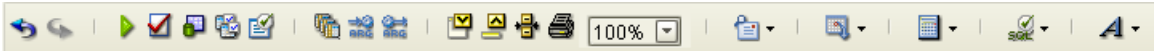
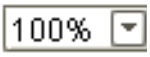
	
	**Undo/Redo** Undo or redo a previous action. For example, if you delete an operator, click Undo to restore it.

Table 5 Using the eTL Collaboration Editor Toolbar

	**Test Run Collaboration** Executes the Project and generates a message log. The log shows messages and errors if the execution fails.
	**Validate Collaboration** Validates the mapping logic without executing the Project.
	**Edit Database Properties** Enables you to configure database OTDs to point the database URL to a different location for design time. This is a temporary setting for design time only; the setting is not saved with the OTD for runtime. See **Changing the Database URL for Design Time** on page 51.
	**Select Source and Target Tables** Enables you to select source and target tables to be used in the Collaboration.
	**Toggle Output View** Toggles between a full- screen pane and a divided pane which shows output messages: data from logs, validations, source/target table data, SQL code, and rejected rows table.
	**Create New Join** Creates a new join view that permits the direct entry of source table relationships (joins), including the ability to view multiple joins. The New Join dialog box automatically appears when: 1. You map a specific source table column (s1) to the target table (t) and the system detects that the target table (t) was already mapped to a different source table (s2) which was not joined to the first source table (s1). 2. You connects a specific source table column (s1) to an operator (o), such as **concatenate**, and eTL detects that this operator (o) is already connected to a different source table (s2) which is not joined to the first source table (s1).
	**Add/Edit Runtime Inputs** Input variables that are assigned by an external system, such as eInsight, are called runtime inputs. See **Filtering Source Data with Runtime Inputs** on page 54.
	**Add/Edit Runtime Outputs** Enables you to add runtime output variables to the Collaboration. See **Filtering Source Data with Runtime Inputs** on page 54.
	**Expand all Graph Object** Expands the operators display to show all mapping elements and fields. This is the default view.
	**Collapse all Graph Object** Collapses the operators to declutter the eTL Collaboration Editor window. For example, if the view is collapsed when tables are displayed, only the headers appear with the mapping arrows.
	**Auto Layout all Graph Object** Automatically arranges all eTL Collaboration Editor window components.

Table 5 Using the eTL Collaboration Editor Toolbar

	**Print Graph** Prints the contents in the eTL Collaboration Editor window. You can select **Print at standard scale** or **Scale to fit page**.
	**Scale** Choose the zoom percentage for the window.

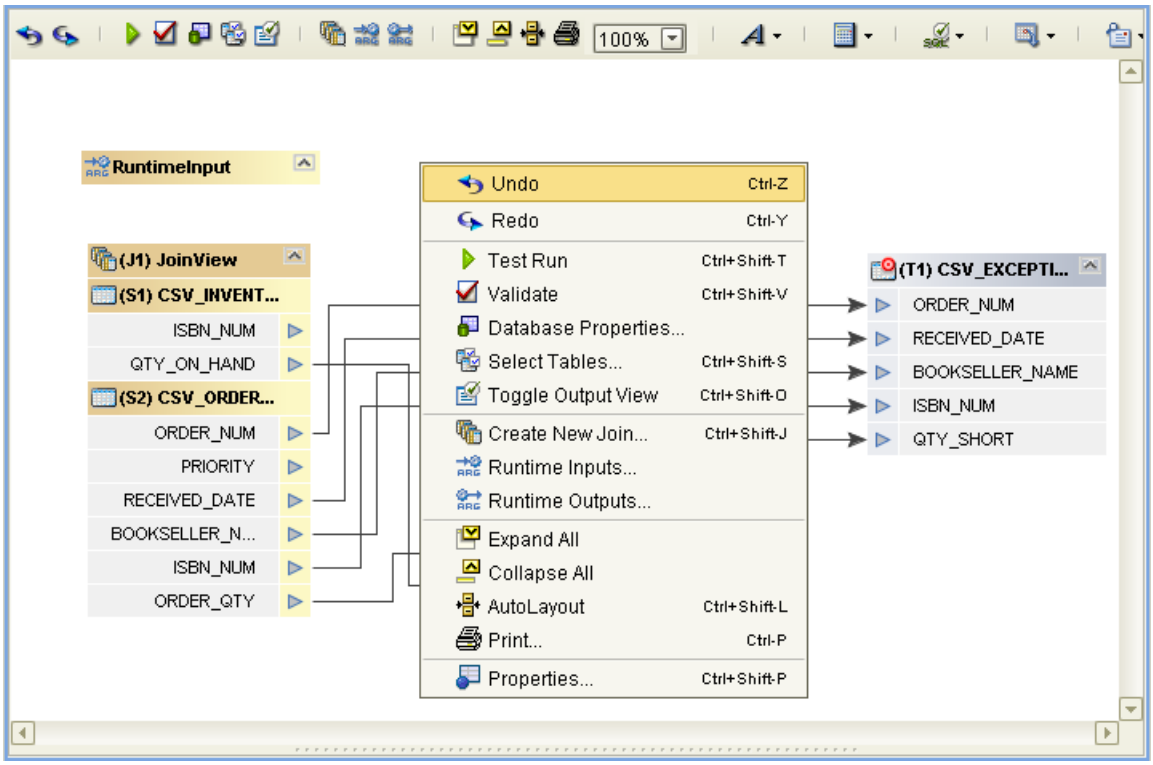
5.2.1 Selecting Tables for Existing Collaborations

The procedure below describes an optional method of selecting tables for existing Collaborations in the **eTL Collaboration Editor** window.

To select tables for existing Collaborations

1 In the eTL Collaboration Editor, right-click the canvas and click **Select Tables**.

Figure 23 Using the Collaboration Editor Context Menu



The **Source and Target Tables** dialog appears.

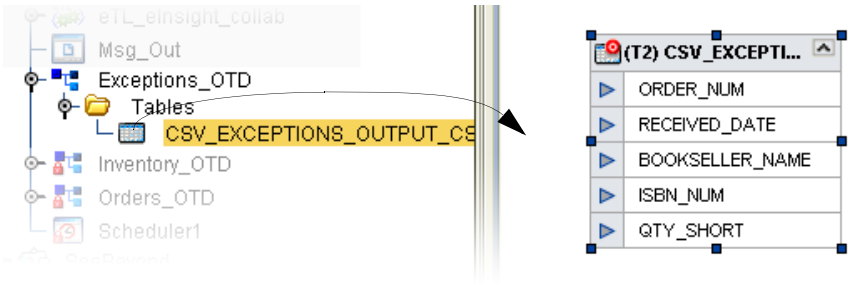
2 Click **Finish** in the **Source and Target Tables** dialog box.

The table appears in the designer pane.

To drag and drop from Project Explorer

1 Click the table in the Project Explorer pane and drag to the designer pane.

Figure 24 Dragging and Dropping Tables



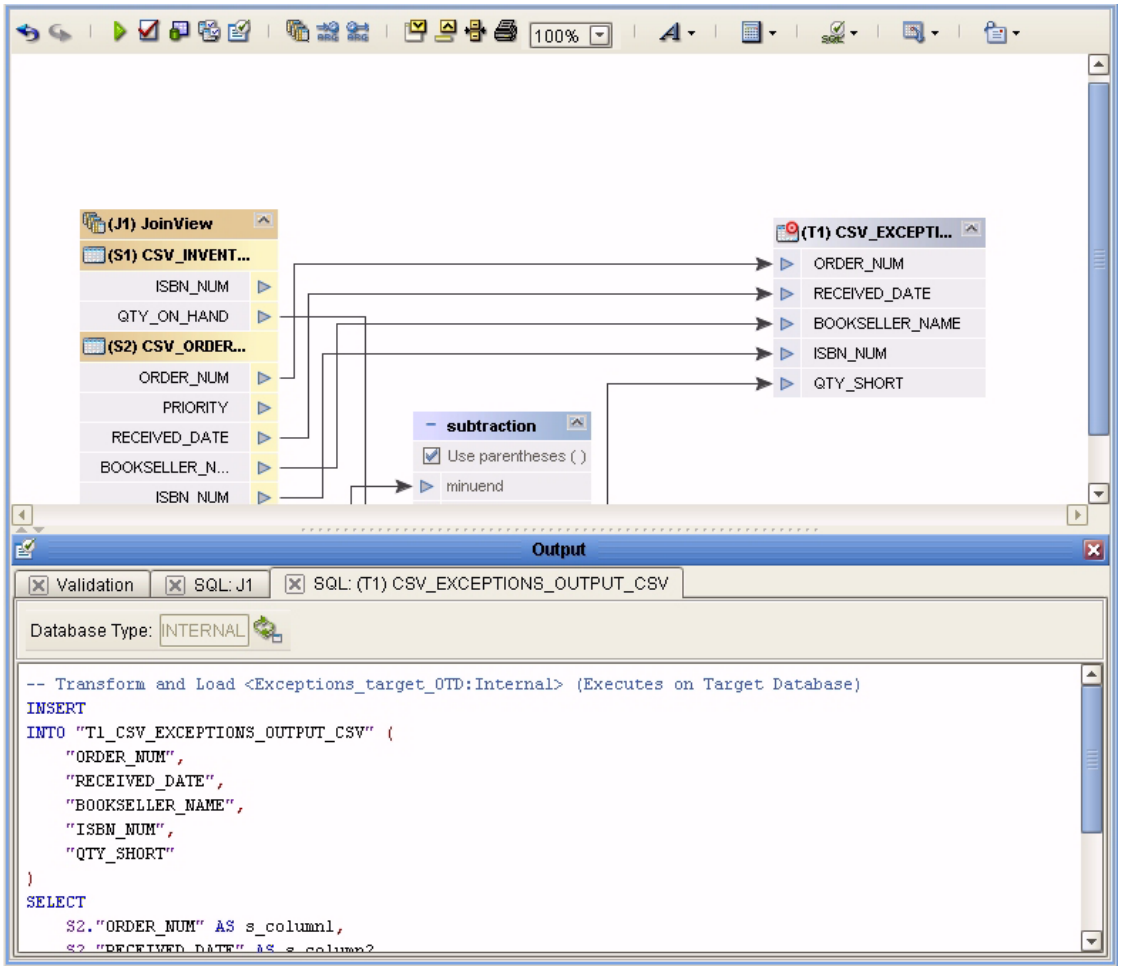
5.2.2 Displaying SQL Code

The procedure below describes how to view SQL code for a table or operator in the eTL Collaboration canvas.

To display SQL code

1. Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.
2. Right-click the table or operator on the canvas and click **Show SQL**. The **Output** section of the eTL Collaboration Editor shows the SQL code. The following figure shows an example of the SQL code for the target table shown under **Output**.

Figure 25 Viewing SQL Code



5.3 Collaboration Execution Strategies

eTL Integrator automatically employs the most optimum execution strategy for Collaborations. Which strategy is employed depends on the specific nature of a Collaboration.

If you do not want eTL to determine the best execution strategy, you can configure a Collaboration for either the staging or the pipeline execution strategy. Which strategy you can use depends on what specifically a Collaboration is set up to do; for example, if your Collaboration business logic contains Java operators, you can only use the pipeline strategy. The section below describes what criteria eTL Integrator uses to determine the best execution strategy, and these are also the criteria to use when you force an execution strategy.

5.3.1 Execution Strategy Selection

By default, eTL Integrator determines the execution strategy for Collaborations. As such, the execution strategy configuration for Collaborations is set to the **Best Fit** option. When eTL Integrator determines what execution strategy to use for a Collaboration, it evaluates the Collaboration for specific attributes. For example, eTL takes tables and columns in consideration.

In addition, the selected execution strategy depends on whether a Collaboration contains Java operators, which are operators that are not available across all supported databases. Examples of Java operators are date transformation operators, operators that parse business names and addresses, and normalize names. For more information about the operators, refer to **"Operator/Database Support" on page 110**.

Projects with Java operators must be executed with a pipeline strategy.

The sections below list the different execution strategies, and when eTL Integrator employs them. Should you wish to force the execution strategy for a Collaboration by changing its setting from **Best Fit** to **Staging** or **Pipeline**, as described in **Forcing Execution Strategies for Collaborations** on page 50, read this section carefully to understand which strategy to select.

eTL Integrator uses the following execution methods depending on a Collaboration's attributes:

- direct/simple
- one pass
- staging
- pipeline

Direct/Simple

With direct/simple execution strategy, all extraction, transformation and loading happens in a single database. When the **Best Fit** option is enabled, this execution strategy is used when:

- All source tables and the target table reside in the same database.
- No Java operators are used.
- The data validation condition is not used.

One Pass

With one pass execution, extraction and transformation occurs in the source database. When the **Best Fit** option is enabled, this execution strategy is used when:

- All source tables are in the same database.
- No Java operators are used.
- The data validation condition is not used.

Staging

When you set Project execution strategy to **Staging**, all source tables are extracted from source databases and staged in temporary tables in the target database. Join and Transformation happens in the Target database. This setting is used automatically when the **Best Fit** option is enabled and the conditions below occur. You can also select this option manually to force its use, in which case this execution strategy is recommended when:

- Source tables are scattered across different databases.
- No Java operators are used.
- The data validation condition is not used.

Pipeline

When you set Project execution strategy to **Pipeline**, transformation and loading (indirectly to target database table) occurs in the internal database engine. This setting is used automatically when the **Best Fit** option is enabled and the conditions below occur. You can also select this option manually to force its use, in which case this execution strategy is recommended when:

- All tables are flat file database tables.
- Java operators are used.
- The data validation condition is used.

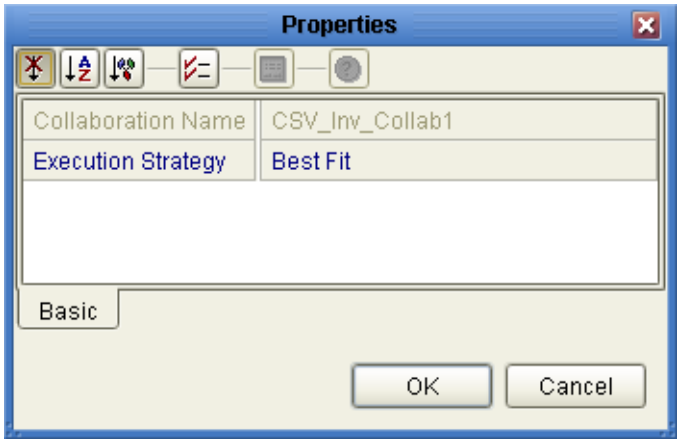
5.3.2 Forcing Execution Strategies for Collaborations

The procedure below describes how to force an execution strategy for eTL Collaborations. Before you do so, refer to **"Execution Strategy Selection" on page 49** for information. If you are using Java operators, you must select the **Pipeline** option.

To force execution strategies for Collaborations

1. Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.
2. Right-click the eTL Collaboration Editor window and click **Properties**. The **Properties** sheet appears as shown below.

Figure 26 Configuring Collaboration Execution Strategy



3 Select **Pipeline** or **Staging** and click **OK**.

5.4 Explicit and Implicit Joins

The condition specified on source tables by the **Edit JoinView** is called 'explicit join.' The condition specified on target tables by the **Target Join Condition** is called 'implicit join.'

The target condition is used differently in `insert` and `update` statements.

If the statement is `update`, the condition from the target table is used to identify the proper rules to update and match the rules to the target table.

If the statement is `insert`, the condition from the target table is used to verify that no duplicate rules are inserted.

5.5 Changing the Database URL for Design Time

For database eTL Collaborations, the design-time test run uses the same URL, catalog, or schema name to connect to the database table as when the database definition OTD was created.

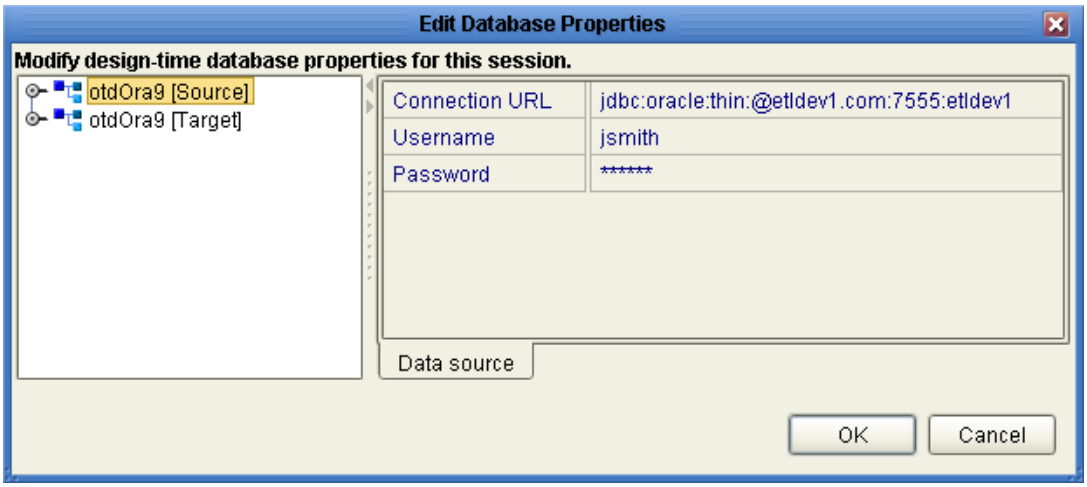
You can change the database URL to point to a different location or even a different table name as long as the content structure is the same. The procedure below describes how to change the URL. Restarting the Enterprise Designer, or refreshing from the Repository, will cause a revert to original values. Change the table name to be used by modifying the table properties and adding a User Defined Table Name in the Expert tab. Changes in the Properties sheet are saved with the eTL Collaboration.

Note: *To change DB2 catalog and schema names, modify the table properties by adding user-defined information in the Expert tab.*

To change the database URL for design time

1. Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.
2. Right-click the eTL Collaboration Editor window and click **Database Properties**. The **Edit Database Properties** dialog box appears as shown below.

Figure 27 Setting Design-Time Database URL



3. Enter the URL for the database to connect to during design time.
4. Enter or verify **Username** and **Password**.
5. click **OK**.

5.6 Using Pre-created Temporary Tables

This feature provides the ability to manage temporary tables.

Staging Execution Strategy

When all the source tables in an eTL Collaboration are configured with a valid table name for the **Staging Table Name** property, you do not require any create or drop privileges for the target environment.

Design Time

Staging Table Name Property

1. Click any source table regardless of the OTD type.
2. Provide a valid table name.

   Ensure that the source and staging/temporary table structures are the same, including column names and data types. If the staging table structure does not

match the corresponding source table, the eTL Collaboration will fail with an appropriate error message.

3 Use the **Drop Staging Table** property to specify whether or not to drop the staging temporary table after the eTL process completes.

4 Use the **Truncate Staging Table** property to specify whether or not to truncate the staging temporary table before each run.

Run Time

The **Staging Table Name** property is used if the execution strategy is 'staging. ' When the execution strategy is staging and the property is set, the eTL Collaboration will not create a temporary/staging table for the source tables. Instead the table with the name specified in the property that is available in the default table space for the target database will be used for staging.

When all the source tables in an eTL Collaboration are configured with a valid table name for the **Staging Table Name** property, the eTL process at runtime will not create or delete any temporary tables. Also, the target tables should not be modified or altered other than for updating records as per the eTL Collaboration.

Be careful about changing the default settings. By default, the staging table will be dropped after each run, unless the **Drop Staging Table** property is set to False.

By default, the data in a temporary table will be truncated before each run, unless the **Truncate Staging Table** property is set to False.

Figure 28 Source Properties - Expert tab - Database

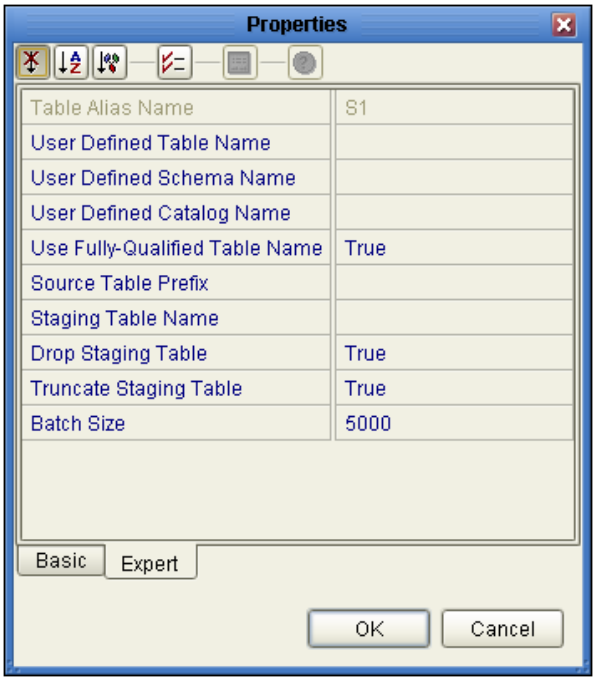
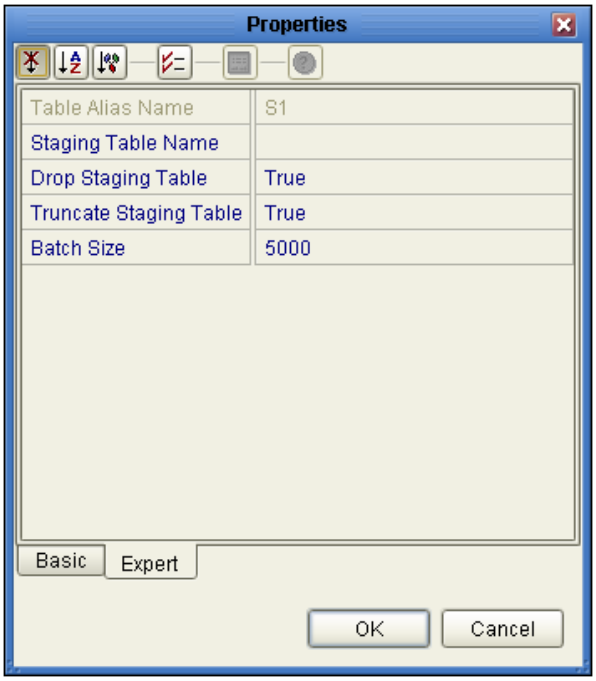


Figure 29 Source Properties - Expert tab - Flat File



5.7 Extracting Data

eTL Collaborations can extract data without filtering or with filtering. This section describes the various methods of data extraction:

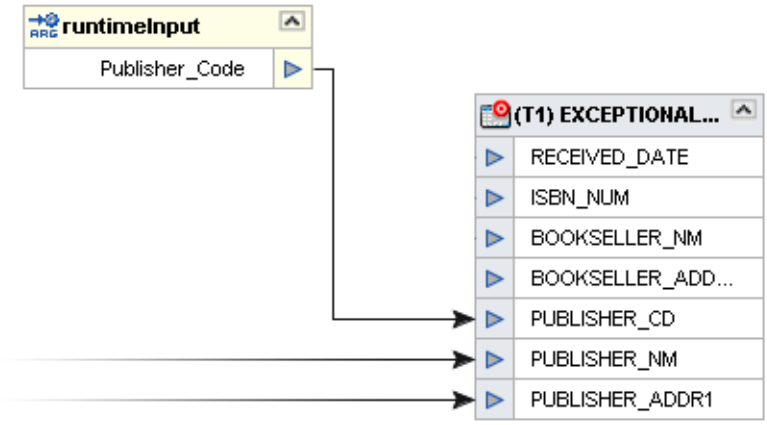
5.7.1 Filtering Source Data with Runtime Inputs

eTL Integrator allows you to pass values to eTL Collaborations at runtime. You can use these values in extraction conditions. However, the use of such dynamic values are not limited extraction; you can also pass values from eInsight Business Processes.

For example, you could specify to have the Collaboration to filter the publisher code record for one specific entry, such as Adams Publishing. To do this, you would specify

the name of the publisher code (PUBLISHER_CODE) as the argument name, and Adams Publishing as the default value.

Figure 30 Map Runtime Input Argument

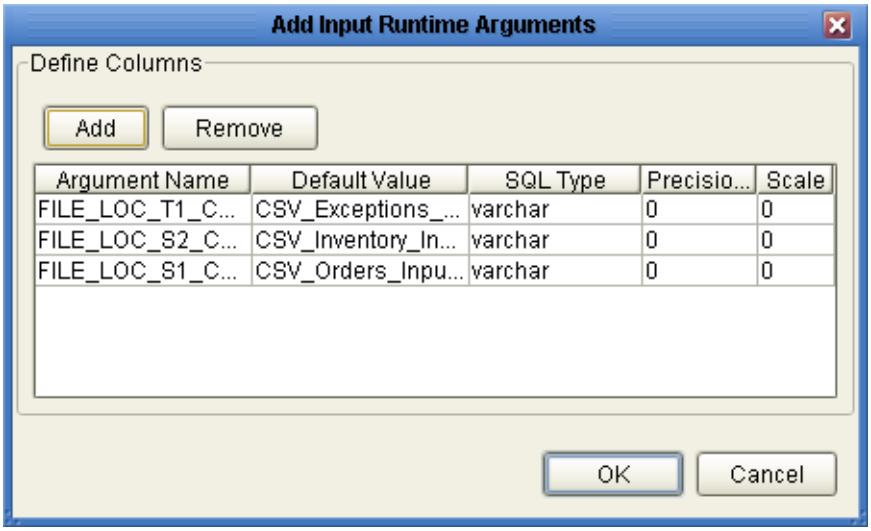


The procedure below describes how to add input runtime arguments to a Collaboration.

To filter source data with runtime inputs

1. Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.
2. Right-click the eTL Collaboration Editor window and click **Runtime Inputs**. The **Add Input Runtime Arguments** dialog box appears as shown below.

Figure 31 Filtering Source Data with Runtime Inputs



3 Click **Add**. An empty row appears.

4 Double-click the empty row under **Argument Name** and enter the name for source record to be filtered.

5 Press TAB and enter the content that the record must contain to be selected.

6 Press TAB and select the SQL type for the record.

7 Press TAB and enter the number indicating the maximum length of the record.

8 Press TAB and enter the number indicating the scale for the record.

9 Click **OK**.

5.7.2 Passing Flat Filenames at Runtime Inputs

eTL Integrator enables you to dynamically pass the name of the flat files at runtime. This feature is useful for Projects that extract/load data from/to flat files and where the name of the flat files is different from the one used when the flat file database definition was created. These values are automatically made available when a flat file database definition is used in the eTL Collaboration. You can also pass these values from eInsight Business Processes.

The configuration for this setting is described in **"Configuring the External Applications" on page 39**.

If the **Dynamic Flat File** property is disabled, the **Default Value** setting for the runtime argument is used for the table name, and the **Directory** property is used as the path. If the filename is passed in the Business Process, the **Directory** property is used and the Business Process information is appended.

If the **Dynamic Flat File** property is enabled, the **Directory** property setting is ignored. eTL looks for the fully qualified path and filename in the Business Process. If it cannot find those, eTL searches for the path and filename in the **Default Value** setting for the runtime argument.

5.7.3 Extracting Source Data with Conditions and Validations

You can set up Collaborations to filter data from source tables using extraction conditions and validations. The procedure below describes how to configure source tables for extraction conditions and validations.

To extract source data with conditions and validations

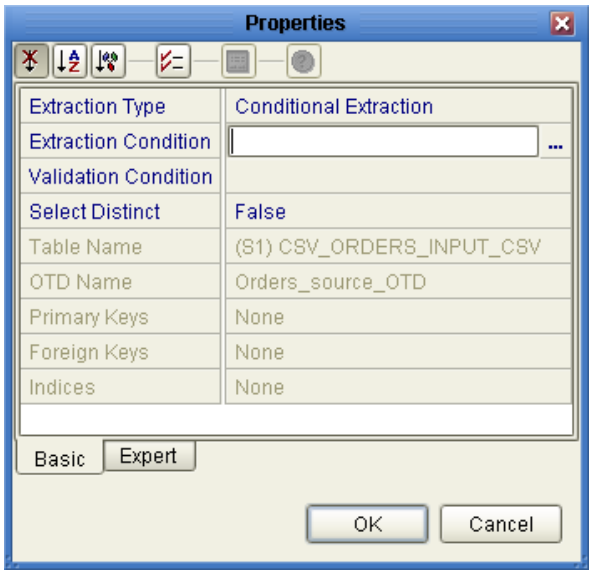
1 Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.

2 Right-click the source table and click **Properties**. The **Properties** sheet appears.

By default, the condition type is configured for conditional extraction. To leave the source data unfiltered, set the **Condition** field to **Full Extraction**.

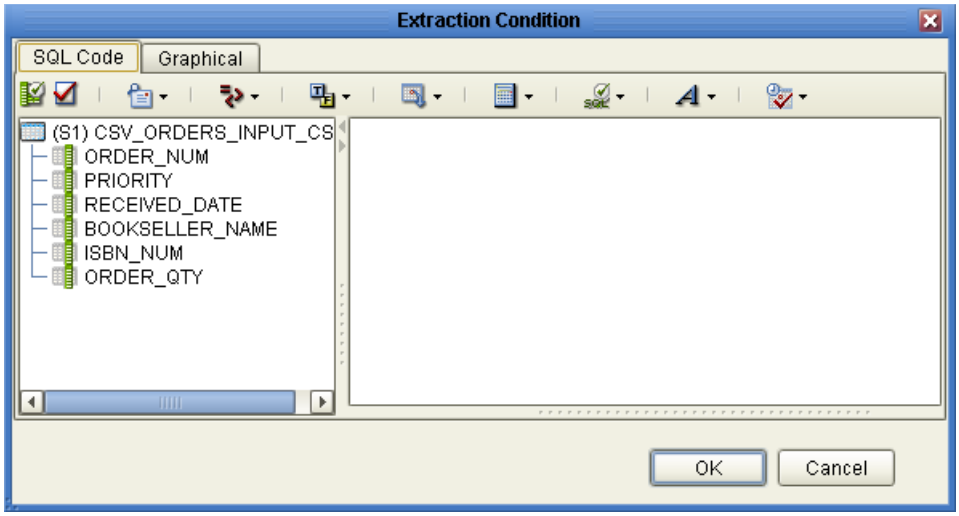
3 Click the **Extraction Condition** field. The ellipsis appears.

Figure 32 Extracting Source Data with Conditions



4 Click the ellipsis (...). The **Extraction Condition** window appears.

Figure 33 Specifying Extraction Conditions



5 Specify the condition and click **OK**.

6 Click the **Validation Condition** field and click the ellipsis that appears. The **Validation Condition** window appears.

7 Specify the validation condition and click **OK**.

8 Specify **True** or **False** for the **Select Distinct** field.

9 Click **OK**.

5.7.4 Setting the Batch Size for Joined Tables

To increase Collaboration execution performance, you can configure the batch size for temporary tables created for joined source tables. By tuning the batch size you can load data more efficiently into source tables.

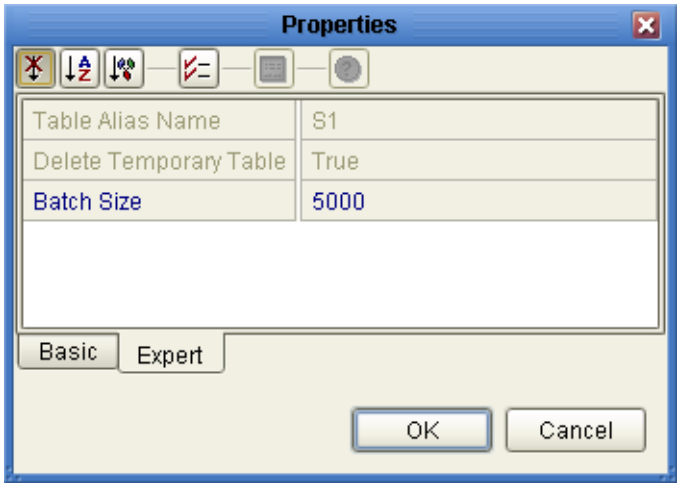
By default, 5000 rows are populated at the same time into a source table. There is no upper limit. The limit is determined by the amount of internal memory of the Local Host running the Collaboration. Generally, the lower the number the better, but adjust the value to determine the optimum performance.

Note: *The source table batch size only affects temporary source tables. To limit the number of rows fetched at a time, specify the batch size for the target table as described in* **Configuring Loading Options for Target Tables** *on page 73.*

To set the batch size for joined tables

1. Open the Collaboration for editing as described in **"Opening eTL Collaborations for Editing" on page 43**.
2. Right-click the source table and click **Properties**. The **Properties** sheet appears.
3. Click **Expert**. The **Expert** tab appears as follows:

Figure 34 Setting the Batch Size for Joined Tables



4. In the **Batch Size** field, enter the number of rows populated at the same time into the temporary source table.
5. Click **OK**.

5.7.5 Using Subqueries

To include a subquery in one of your source tables, you can create a view including one or more subqueries in one of your source environments.

See the subquery below.

```
select *
from Orders s2
where not exists
   (select s1.qty_on_hand
   from Inventory s1
   where s1.qty_on_hand < 100
   and s1.isbn_num = s2.isbn_num)
```

To use subqueries

1. Implement the subquery inside your SQL statement as a database view.

   After you create this view, create an OTD and use that OTD in an eTL Collaboration.

2. Create an OTD using the **Object Type Definition Wizard** to select your view to use in the eTL Collaboration.

3. Create an eTL Collaboration, using the OTD you created, and select your view as source table.

   The view you create is used as your source and can be used the same as any source table.

5.7.6 Setting Join Conditions

eTL Integrator enables you to join data from multiple sources before extraction. You can create join views by creating a join condition that joins source tables. You can define the join condition by clicking the **Edit Join View** icon.

Note: *For optimal performance, join the most unique columns in the first join and the least unique columns in the second join.*

When you attempt to link a source table to a target table, that is already joined to a different source table, you are prompted to create a new join view.

This feature serves two purposes:

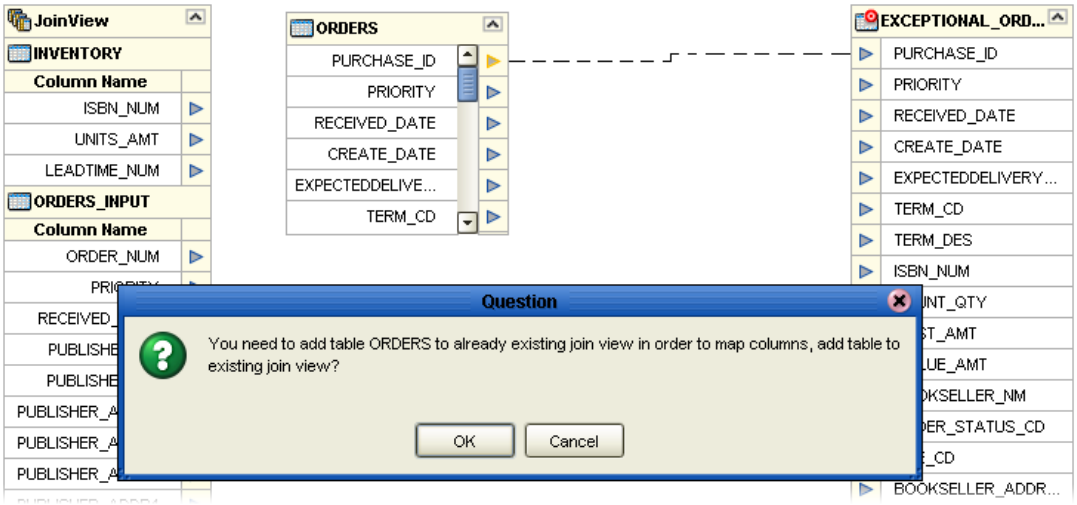
- It affords quick access to the set join condition interfaces
- It serves as a warning message in case the join is not necessary

The following steps are an example only.

To create a join view using the Link Time Join

1. Attempt to connect a node from source table **ORDERS.PURCHASE_ID** to the target **EXCEPTIONAL_ORDERS.PURCHASE_ID**. The following message appears.

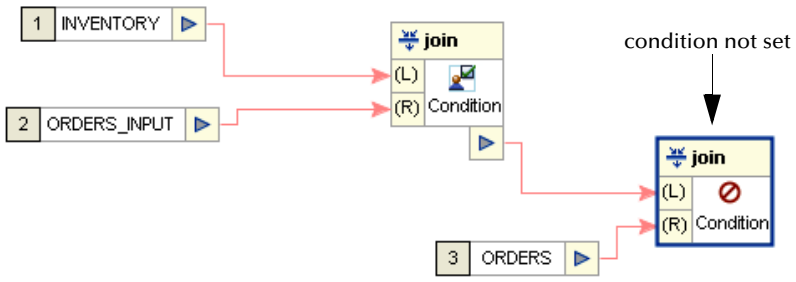
Figure 35 Adding Tables to Existing Views



2 Click **OK** to indicate your intention to add the **ORDERS** table to the existing join view.

A graphical view of the join appears. You can set the conditions based on the establishment of relationships between sources explained below.

Figure 36 Joined View



3 Click the icon/button shown in the previous figure.

4 Add the join conditions.

5.7.7 Using Table Aliases with Multiple Source Table Views

eTL Integrator enables you to use the same table as multiple instances of source data with different aliases. For example, a Project has the following source tables: **EMP_TBL** and **CODES_TBL**. You can create a join view with these tables. In addition, you can drag another view of the **CODES_TBL** to your Designer Pane to do a third join which is used in the code lookup.

The following table displays the sample data for the **EMP_TBL** source table:

Table 6 Employee Table

NAME	ID	JOB CODE	DEPT CODE
Dave	1	P	D1
Judy	2	C	D2

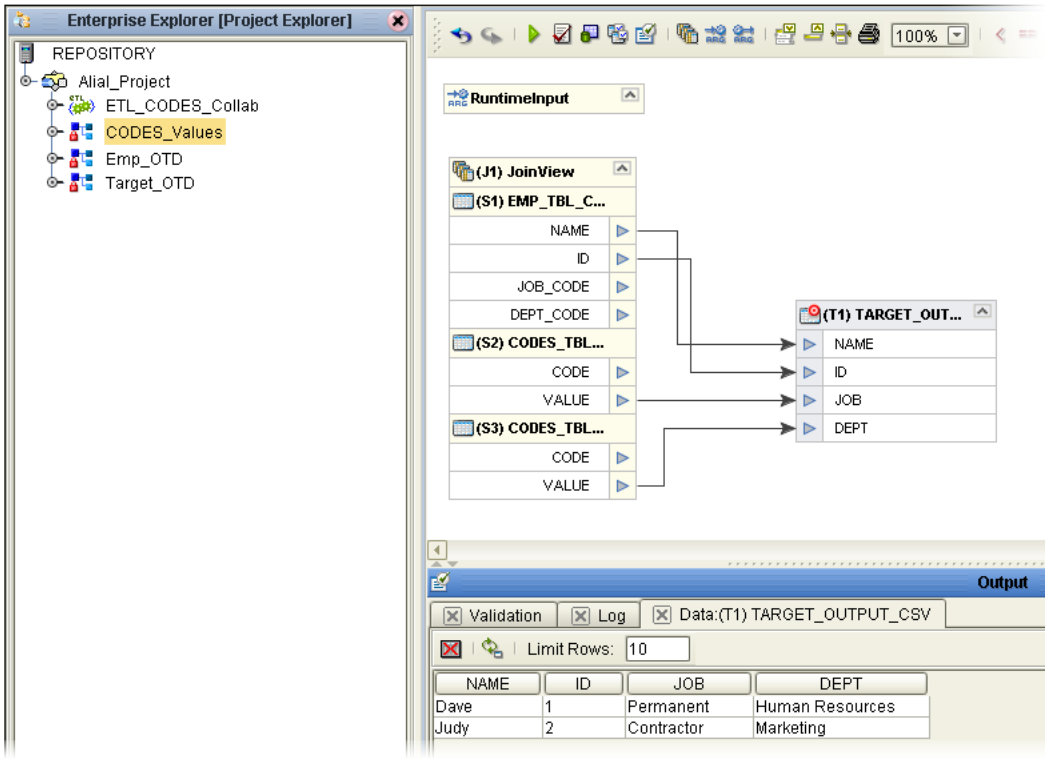
The following table displays the sample data for the **CODES_TBL** source table:

Table 7 Company Codes

CODE	VALUE
D1	Human Resources
D2	Marketing
P	Permanent
C	Contractor

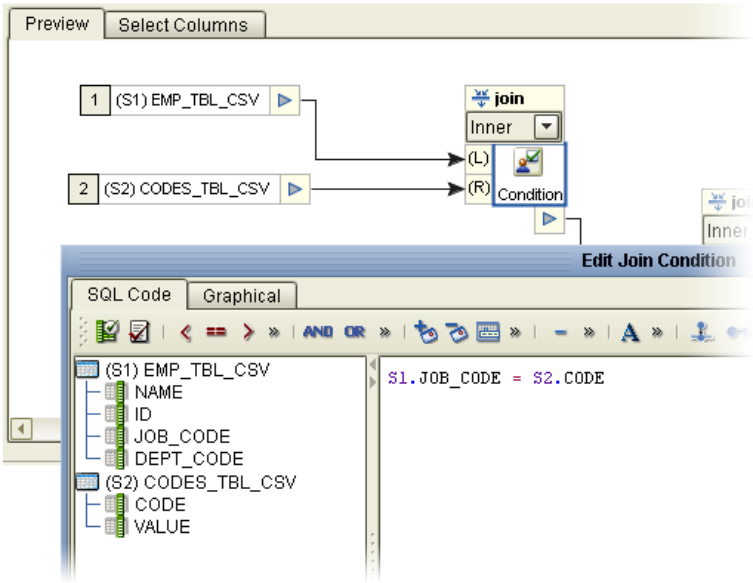
The following figure shows the Collaboration and mapping with the correct data from a test run. The lookup loads the description for both jobs and departments from the **CODES_TBL** table. In this example, the table CODES_TBL is used twice in the join condition with aliases S2 and S3. In the join condition S2.Code is joined with S1.JOB_CODE and S3.Code is joined with S1.DEPT_CODE.

Figure 37 Employee and Codes Example



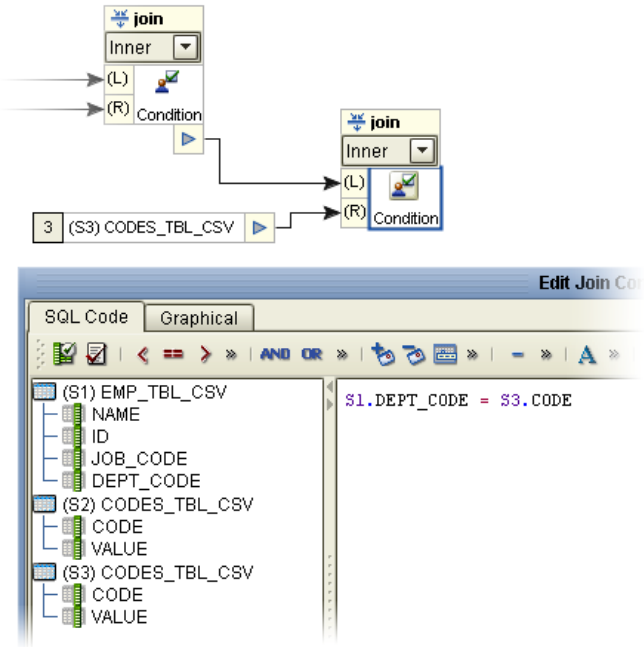
Refer to the first join shown in the following figure. This join view shows the condition **S1.JOB_CODE = S2.CODE**. This will load the job descriptions from the **CODES_TBL** to the target table column **JOB**.

Figure 38 Employee and Codes Example—First Join



Refer to the second join shown in the following figure. This join view shows the condition **S1.DEPT_CODE = S3.CODE**. This loads the department descriptions from the **CODES_TBL** to the target table column **DEPT**.

Figure 39 Employee and Codes Example—Second Join



5.8 Transforming Data

This section describes the transformation options available in the eTL Integrator for transforming data in eTL Collaborations. The transformation operators are available in the eTL Collaborations Editor regardless of the type of source and the target data store that the Project will be connected to. Some operators are in the validation dialog only.

5.8.1 Transformation Operators

The transformation operators can be broadly classified into the following categories.

This classification is made solely for organizational purposes; you can cascade multiple function and model complex expressions. This section lists all the operators available in eTL Integrator and briefly describes the functionality of these operators.

You can use the operators and model expressions completely in the graphical mode by merely dragging and dropping the operators to the canvas and connecting them to the appropriate columns.

You can also verify the resulting SQL code in the eTL Collaboration Editor by right-clicking the source or target table and selecting **Show SQL**.

Data Cleansing Operators

These operators are used to cleanse the extracted data and normalize the values before performing further transformations or loading the target tables.

Normalize Name

This operator is used to normalize the Name of a person. The user shall normalize either the First Name or the Last Name to United States or to United Kingdom formats. For example, if the value "Bob" was given as the input to Normalize Name operator with the option to normalize the ‘First Name’ in ‘United States’ format, then the result would be "Robert". Similarly all the commonly used acronyms for the person names can be normalized.

Parse Address

This operator is used to parse multiple parts of an address coming in a single string and return parts of the address in a normalized form. The operator provides the flexibility to parse, normalize and return different parts of the address to match the target data

structure. Currently, the parts can be normalized in two formats - United States and United Kingdom respectively.

Parse Business Name

This operator is used to parse Business Name and Business Type from a single string and normalize Business Type before returning the value. For instance, if the incoming value has “Bob Smith Co.” or “Bob Smith Corp.”, the resultant value is “Bob Smith” for the business name and “Corporation” for the business type. This operator currently provides the flexibility to normalize the business type in two formats: United States and United Kingdom.

SQL-specific Operators

eTL Integrator provides a list of ANSI 92 SQL-specific operators. These operators can be used for performing transformation on extracted data for further processing or loading the target data structures.

Case

The case operator filters or selects data using If (WHEN), THEN and ELSE logic. The case operator consists of multiple WHEN statements and one default statement. Each WHEN statement has one condition and one return input.

The result value depends on the following:

WHEN statements are evaluated in sequence until a true condition is met. When a condition is true, the return input is entered as the result. If none of the statements are true, the default value is the result.

Castas

The castas operator is used to change the data type of data. For example, you can change numeric data to variable characters. The “Input to” column is directly from a data value (for example a table column) and not a condition. The result is converted to values such as char, double, float, integer, timestamp or varchar, based on the option selected. Note that data may be truncated depending on how this operator is used.

Coalesce

You can use the Coalesce operator to fill an input column that is null. This is useful for target tables that do not accept null values.

Count

The Count operator returns the number of rows in a query.

Nullif

You can use the Nullif operator to set an input column to null if the column contains a the value you specified.

userFX

See **Support for User Defined Functions** on page 69.

Date Operators

eTL Integrator provides an array of operators to process and manipulate date values. These operators can be used on extracted data for further processing or loading target data structures.

String to Date

This operator is used to take a VARCHAR string that contains a date value and return the data of type TimeStamp. The operator has the flexibility to process incoming date in an array of formats.

Date to String

This operator is used to take a value of type Date and return a VARCHAR String. This operator has the flexibility to convert the incoming date value into an array of date formats.

Add to Date

The add to date operator adds a user-selected date interval in days, weeks, months, etc. to the date field and returns a new date. The resulting date format is determined by the date format of the target table's database. Click the drop-down arrow to select an option, such as days, weeks, months.

Date Difference

The date difference operator calculates the difference between two dates. The resulting format is numeric. Click the drop-down arrow to select an option, such as days, weeks, months, etc.

Date Part

The Date Part operator takes in a timestamp value and allows you to extract different parts of the timestamp (day, month, year, hour, minute, etc.) in different formats (DD, DDD, MM, MMM, YY, YYYY, etc.). The resulting format is varchar.

Now

The Now operator returns a date plus time in timestamp format. It is the current date and time. The format depends on the database. For example, in DB2 the format is: yyyy-mm-dd hh:mm:ss

Number Operators

eTL Integrator provides a set of Number Operators to model simple and complex expressions to manipulate numerical data.

Addition

The addition operator adds the value of multiple input nodes which both map to the addend node. The result (numeric) is the sum.

Subtraction

The subtraction operator subtracts the numerical value subtrahend from the numerical value of minuend and returns the difference result (numeric). Minuend is the number

from which another number is subtracted. Subtrahend is the number to be subtracted from another number.

Multiplication

The multiplication operator multiplies the value of multiplicand1 by the value of multiplicand2 and returns the product result (numeric).

Division

The division operator divides the value of dividend by the value of divisor and returns the quotient result (numeric).

Mod

The modulus operator divides the numerical value of dividend by the numerical value of divisor and returns the remainder result (numeric).

Sum

The sum operator adds all the values in numeric column. The result (numeric) is the sum.

Minimum

The minimum operator determines the smallest value in a column and returns the result (numeric).

Maximum

The maximum operator determines the largest value in a column and returns the result (numeric).

Average

The Average operator calculates the average (or the mean) of numeric data for a mapped data value (column) and returns the result (numeric) either to another operator or to a column in a target table.

Absolute Value

The Absolute Value operator returns the value of a number without regard to its algebraic sign. The Input is a number that is positive or negative; the result (numeric) is the absolute value without regard to sign.

Sign

The Sign operator takes numeric datatype as input and returns 1, 0, or -1: if input is positive numeric value then operator returns "1"; if negative numeric value, then operator returns "-1"; if zero value, then operator returns "0".

String Operators

To Upper Case

The ToUpperCase operator converts the characters in input string to upper case characters. The result string (varchar) is all upper case.

To Lower Case

The ToLowerCase operator converts the characters in input string to lower case characters. The result string (varchar) is all lower case.

SubString

The substring operator returns the substring of the first argument string starting at the position specified in the second argument startpos (starting position) which is a literal integer. The length specified in the third argument length is also a literal integer and represents the character length of the first argument. The result (varchar) is the resulting, shortened output string.

Concat

This operator is used to link or concatenate two or more fields. Values are concatenated in the order they are connected to the string. Multiple nodes may be connected to the string.

Replace

The replace operator replaces (or substitutes) a string of characters with a string of your choosing. The source string is the input string. The to be replaced string is the entire string or part of the string that is replaced. The to substitute string is the actual new value replaces the to be replaced string. The result (varchar) is the output string.

String Length

This operator counts a string length.

Left Trim

The Lefttrim operator removes leading spaces from the left end of a string.

Right Trim

The righttrim operator removes trailing spaces from the right end of a string.

Number to Hex

(for use with DB2 and flat files only)

This operator is used for converting an integer value to its corresponding Hexadecimal representation.

String to Hex

(for use with DB2 and flat files only)

This operator is used for converting a String value to its corresponding Hexadecimal representation.

Comparison Operators

Not Equal

The Not Equal operator returns true if the data column mapped to the left property is not equal to the right property; otherwise, returns false.

Lesser than

The Lesser Than operator returns true if the data value mapped to the left property is less than the data value mapped to the right property; otherwise, returns false.

Lesser or equal

The Lesser or Equal operator returns true if the data value mapped to the left property is less than or equal to the data value mapped to the right property; otherwise, returns false.

Equal

The Equal operator returns true if the data value mapped to the left property is equal to the data value mapped to the right property; otherwise, returns false.

Greater than

The Greater Than operator returns true if the data value mapped to the left property is greater than the data value mapped to the right property; otherwise, returns false.

Greater or equal

The Greater or equal operator returns true if the data value mapped to the left property is greater than or equal to the data value mapped to the right property; otherwise, returns false.

Is Null?

The Is null operator returns true if the input column value is null; otherwise, returns false.

Is not Null?

The Is not Null operator returns true if the input column value is not null; otherwise, returns false.

LIKE

The LIKE operator returns true if the input (varchar) value matches the mapped pattern (varchar) value within a string; otherwise, returns false.

AND

The AND operator returns true if both the data value mapped to the left property and the data value mapped to the right property are true; otherwise, returns false.

OR

The OR operator returns true if either the data value mapped to the left property or the data value mapped to the right property is true; otherwise, returns false.

NOT

The NOT operator takes boolean expression as input and returns the opposite. If input value is True, then operator returns False, and vice versa.

Validation Operators

eTL Integrator provides operators to validate extracted data. You can validate multiple columns in a record through Data Validation Condition. If the validation fails for at

least one of the columns then the record is rejected thereby preventing it from being loaded into target tables. All windows that show conditions (for example, the Data Validation Condition window and the Extraction Condition window) provide the operators (above) to enable you to model complex validation conditions.

You can view rejected rows at design time. If a data validation condition is set, click **Test Run**. If rejected rows exist, right-click the target table and select **Show Rejected Data**. The rejected data displays in the **Output** pane.

Is Valid DateTime

This operator is used to validate if the input value is a valid date. This operator takes in a VARCHAR String as a parameter. You can choose within an array of date formats that can be applied on the input string for validation. If the value is not a valid date in the specified format then the record is rejected.

Matches

This operator validates if the input value is valid as per the regular expression. This operator is preconfigured to validate US zip codes and social security numbers. If the validation fails for a column then the entire record is rejected.

You can modify existing regular expression, or enter your own, as long as the expressions are valid.

5.9 Support for User Defined Functions

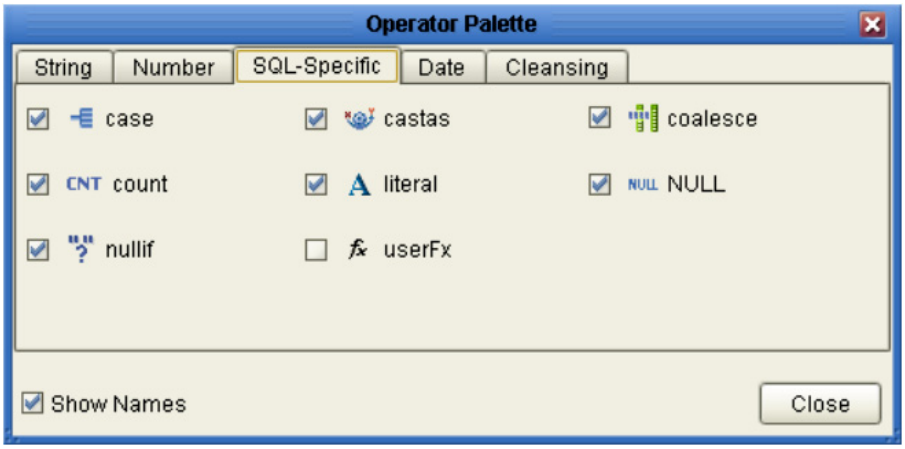
This feature provides for the ability to invoke native functions in the database and user defined functions/stored procedures that are available in the database. Such functions can be used for data transformations, data extraction conditions from the source tables, and data update conditions to the target tables. This feature allows you to create custom transformation and data filtering functions when a suitable function is not available natively in eTL Integrator.

Currently eTL Integrator supports User Functions with one output parameter. The data types for input and output parameters are of type any, and data type cannot be changed. To perform data type validations:

Steps for Using User Defined Functions:

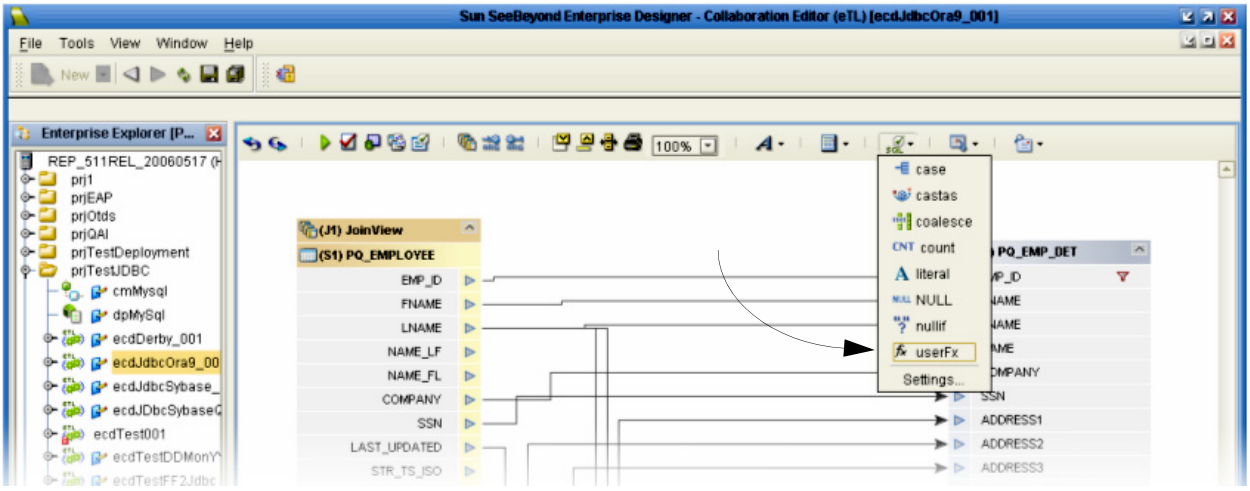
1 Select the `userFx` function from Operator Palette.

Figure 40 userFX Function



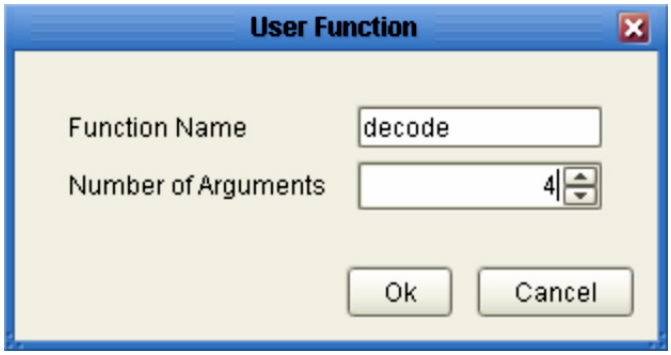
Once selected, the UserFx will be available in the list of SQL Specific functions:

Figure 41 userFX is Available



2 When the UserFx function is dragged into the canvas, the following window appears.

Figure 42 User Function

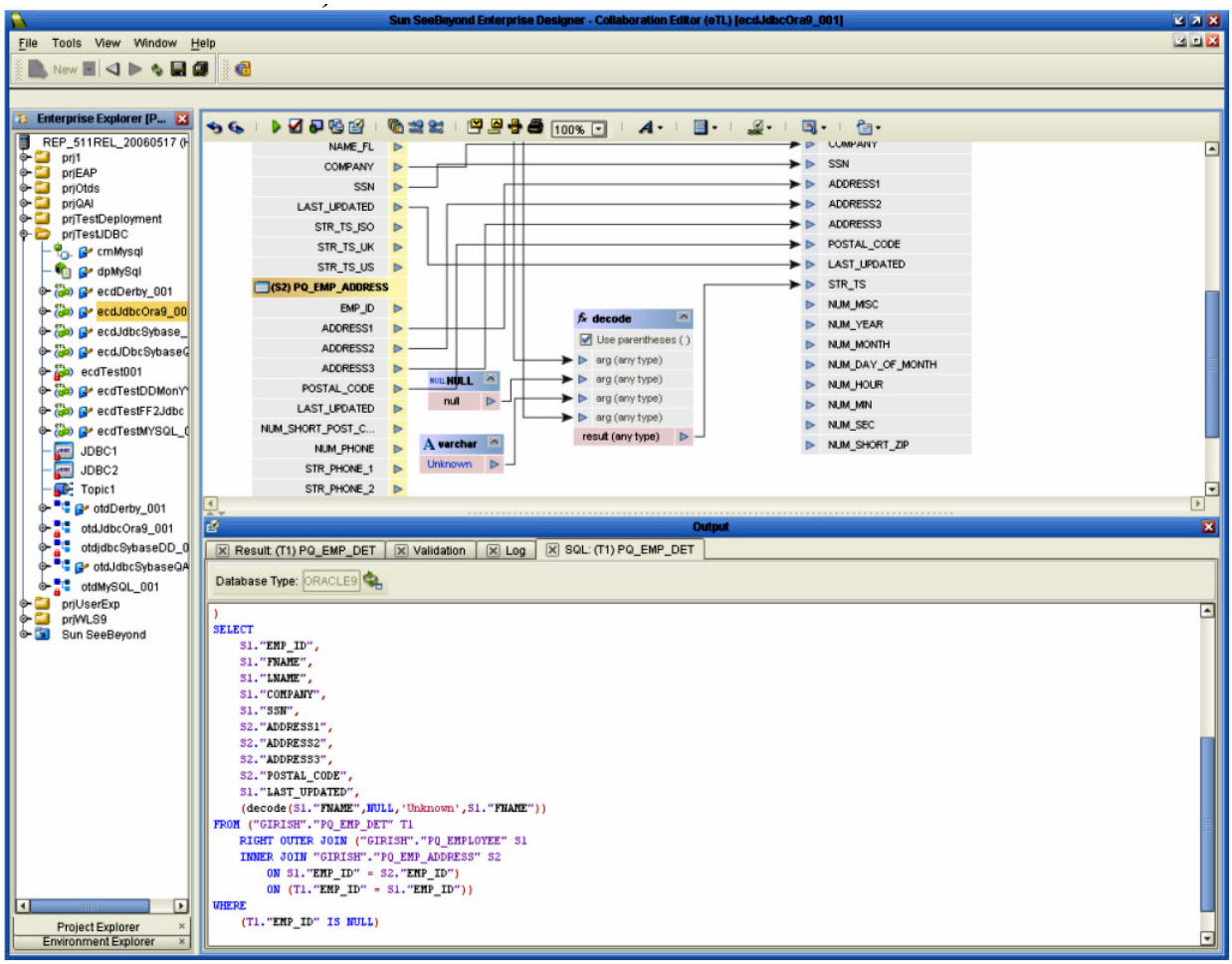


Configure the Function Name and Number of (input) Arguments.

Note: *The name and the number of input parameters provided here should exactly match the signature of the function in the database. The argument names cannot be modified, and at runtime the arguments will be passed in the order specified.*

Once configured, the operator will look like the attached: (decode() is a User Defined function.)

Figure 43 UserFX - Decode Used for Data Transformation



Please note the following rules when using User Defined Functions:

1. When the Execution Strategy is Pipeline, User Defined Functions can't be used.
2. When the Execution Strategy is Staging, and
   - A if the User Defined Functions are used in Extraction Conditions, then the function should be available in both Source and Target databases.
   - B if the User Defined Functions are used in Data Transformations or Update Conditions, then the function should be available in Target databases.

3 When the Execution Strategy is Simple, then User Defined Functions should be available in the Target database.

4 When the user Execution Strategy is One Pass, and

A if the User Defined Functions are used in Extraction Conditions, then the function should be available in both Source and Target databases.

B if the User Defined Functions are used in Data Transformations or Update Conditions, then the function should be available in Target databases.

For details about Execution Strategies refer to **Collaboration Execution Strategies** on page 48.

5.9.1 Modeling Complex Transformations

Applying Parenthesis to Expressions

Order of precedence is dependent on operators and the use of parenthesis. This is especially important for the proper execution of mathematical calculations. You can manage the precedence of evaluation of expressions by using parenthesis in eTL Collaborations.

Cascading Operators

eTL Integrator allows you to cascade operators to create complex expressions. For example, you can validate a VARCHAR String by trimming the string and validating that it is a valid date. If it is a valid date, you can parse it to extract the month, day, year and hour separately and load the target table.

Joining Heterogeneous Source Data

You can use the Join operator to join two tables at one time. The tables that are being joined could be from heterogeneous sources and eTL Integrator can join those tables seamlessly. The join conditions can be configured in the Join Conditions editor.

Example: Two tables may be joined unconditionally by connecting a node from the table header arrows to the left and right input links of the join operation; no condition is required. When two tables are joined based on a condition, there is an input node from a conditional expression to condition. If you want to link multiple tables, result (join) can be input to another join operator.

5.9.2 Dynamic Values in eTL Collaborations

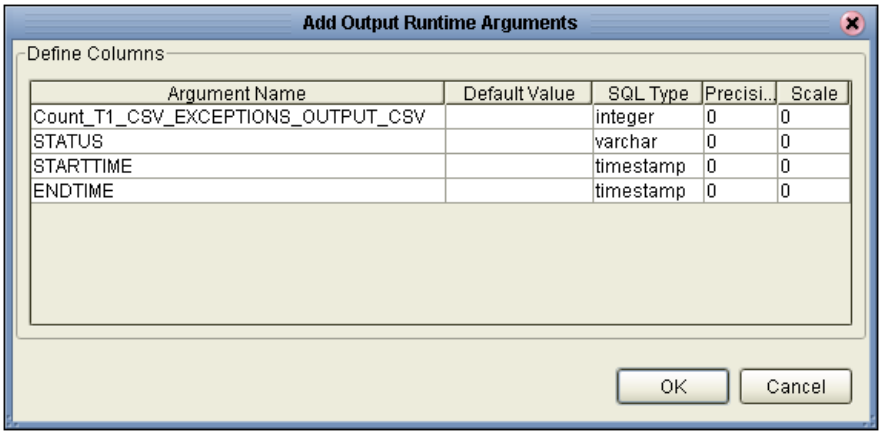
eTL Integrators allows for values to be dynamically passed from eInsight Business Processes. These values can be used in the transformation operators in the eTL Collaborations.

5.10 Loading Data

5.10.1 Runtime Output Arguments

eTL Integrator provides a constant list of output arguments for all eTL Collaborations. These are the total number of records affected at target, the status, the start time and the end time of the Collaboration.

Figure 44 Runtime Output Arguments



Runtime output arguments can be captured and displayed or written to a file. These "messages" are made available automatically by the system.

- Count shows the row count for the Insert, Update, or Delete statement.
- STATUS shows whether the Insert, Update, or Delete operation was successful or failed.
- STARTTIME shows the start time of the runtime eTL Collaboration
- ENDTIME - is the end time of the eTL process

5.10.2 Configuring Loading Options for Target Tables

The procedure below describes how to configure the loading options for target tables. You can figure the target tables for the following:

- Perform a full load or conditional load
- Specify the condition for conditional loading
- Group input data by a specified expression
- Specify the number of rows fetched at a time for loading (batch size)

Grouping input data using the Group By Expression option

eTL Integrator supports extracting aggregated data, applying special transformations, and loading them to a target table. Specific transformations are supported for

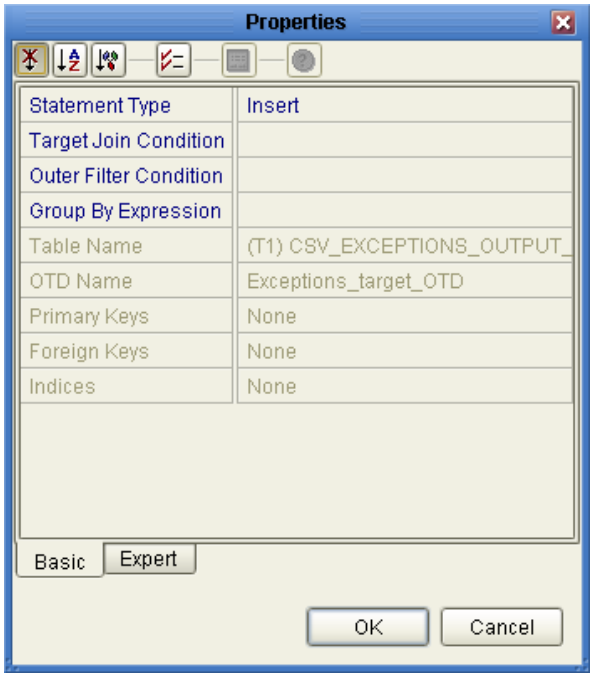
aggregated values such as Minimum, Maximum, Count, Sum and Average. You can aggregate column(s) based on a selection specified using the **Group By Expression** option described in the following procedure. The **Group By Expression** option can only be used with Insert/Update statements.

To configure target table loading options

1 Open the Collaboration for editing as described in **Opening eTL Collaborations for Editing** on page 43.

2 In the eTL Collaboration Editor, right-click the target table and click **Properties**. The **Properties** sheet appears.

Figure 45 Configuring Target Table Loading Options



3 In the **Statement Type** field, click one of the following:

Target Table Load Option	Description
Insert	Always append new rows (full load)
Insert/Update	Update an existing row or append a new row, depending on the evaluation of a condition (upsert)
Update	Update existing rows only
Delete	Delete rows

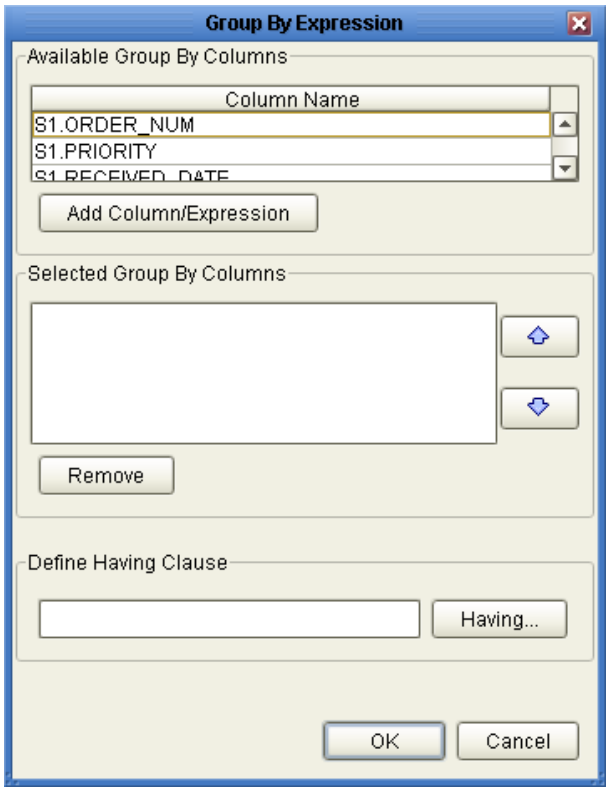
4 To specify a condition for loading, click the **Condition** field and click the ellipsis. The Condition window appears.

5 Enter the condition expression and click **OK**.

6 To group input data by a specified expression for an Insert statement, click in the **Group By Expression** field, and click the ellipsis. The **Group By Expression** dialog box appears.

Note: *The* **Group By Expression** *option does not have an effect for Upsert or Delete statements.*

Figure 46 Group by Column



7 Click **Add Column/Expression** to add the column or expression for grouping.

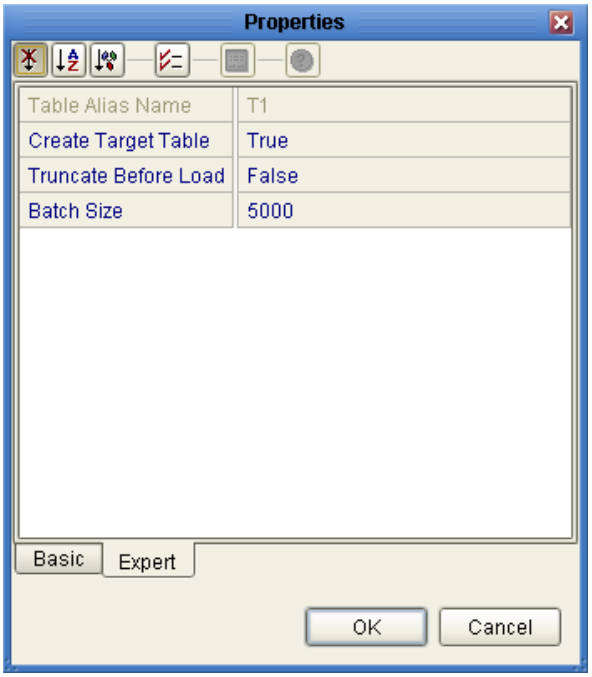
8 Click **Having**. The **Having Condition** window appears.

9 Define the expression that a column must include to be grouped and click **OK**.

10 Click **OK**.

11 Click **Expert**. The Expert tab appears as shown below.

Figure 47 Configuring Target Tables



12 For the **Create Target Table** field, click **True** to create the target table at runtime, and **False** if the table already exists.

13 For the **Truncate Before Load** field, click **True** to truncate the contents before loading, or False to leave the content as is.

14 In the **Batch Size** field, enter the number of rows to be fetched at a time for loading to the target table.

15 Click **OK**.

5.10.3 Using Auto Mapping

Auto mapping allows you to map all the matching columns of a source table to a single target table with two clicks of the mouse.

- Right-click the source table and click **Auto Map**.

All the columns from the selected source are automatically mapped to the matching columns in the target table. If two source tables have the same column name, only one will automap.

When multiple target tables have the same column name, a dialog box appears, enabling you to select the correct target table.

5.10.4 Mapping Multiple Targets

You can have multiple targets in one Collaboration. You can map each target table to a single source table or single joined table view.

Note: *One source column maps to a single target column. You cannot map one source column to two or more target tables, but you can drag and drop another instance of the table.*

5.11 Support for JDBC eWay

This feature provides the ability to access databases that are not natively supported by eTL Integrator. This feature requires a compatible JDBC Driver for the Database to be connected – the JDBC drivers are not available with this product.

Steps for Connecting Using the JDBC eWay:

1. Create DB OTDs using the JDBC eWay – refer to *JDBC eWay User's Guide*.
   - eTL Integrator currently supports Table or View based OTDs.
   - Install the JDBC Drivers as specified in the *JDBC eWay User's Guide.*
2. Once the OTDs are created, follow the steps in the eTL Collaboration wizard and select the Tables from JDBC OTDs as input and/or output.
3. All the operators in the eTL Collaboration canvas can be used for processing data. However, based on the nature of the database being connected to and the driver used, some functions may not be available or may not function as expected. So, eTL Collaborations using JDBC OTDs should be adequately tested at design time.
4. For configuration of the JDBC External Application and External Systems, please refer to *JDBC eWay User's Guide.*

Chapter 6

Tutorial: Building eTL Projects

This chapter provides step-by-step instructions for building an eTL Project. By following the instructions in this chapter you will quickly learn the steps necessary to build eTL Projects.

What's in This Chapter

6.1 Installation Requirements

You must have the File eWay installed to build the eTL Project in this tutorial. For information, refer to **"Installing eTL Integrator and Sample Projects" on page 18**.

6.2 Creating Source and Target Sample Files

When you create the flat file database definitions for the Project, you use source and target files to set up the structure for the definitions. When you create the definition at design time, these files must be smaller than 5 MB. At runtime, the files can be any size.

This section describes how to create the sample CSV files for the Project — or you can download them as described in **Chapter 3 "Installing eTL Integrator" on page 17**.

The Project uses two source files and creates one target file. The source files contain customer order and inventory records.

The target file defines the information for orders that could not be processed due to insufficient inventory, indicating the inventory shortage for the order.

6.2.1 Creating the Customer Order Source File

The customer order file defines the source table for the orders received.

To create the customer order source file

1. Create a CSV file called **CSV_Orders_Input.csv**.
2. In the file, enter the values as shows below, and save the file.

Figure 48 Creating the Customer Order Source File

```
ORDER_NUM,PRIORITY,RECEIVED_DATE,BOOKSELLER_NAME,ISBN_NUM,ORDER_QTY
10001,5,09-26-2006,Adamson Publishing,0-4545-2110-2,250
10001,3,09-26-2006,Adamson Publishing,0-4545-2210-1,400
10002,7,04-04-2006,Firestone Livres,0-4545-2310-7,120
10002,5,04-04-2006,Firestone Livres,0-4545-3221-2,20
10003,4,05-11-2006,Hardcourt Publishing,0-4545-3366-5,210
10003,9,05-11-2006,Hardcourt Publishing,0-4545-3413-2,20
10004,2,06-10-2006,Moonves Books,0-4545-3421-6,400
10004,5,06-10-2006,Moonves Books,0-4545-3535-3,100
10004,7,06-10-2006,Moonves Books,0-4545-4369-1,20
10005,8,10-17-2006,Ural Russian Literary,0-4545-5413-8,70
```

6.2.2 Creating the Inventory Source File

The inventory file defines the source table for the inventory available for all products.

To create the inventory source file

1. Create a CSV file called **CSV_Inventory_Input.csv**.
2. In the file, enter the following information and save the file.

Figure 49 Creating the Inventory Source File

```
ISBN_NUM,QTY_ON_HAND
0-4545-2110-2,300
0-4545-2210-1,200
0-4545-2310-7,80
0-4545-3221-2,300
0-4545-3366-5,50
0-4545-3413-2,20
0-4545-3421-6,400
0-4545-3535-3,80
0-4545-4369-1,30
0-4545-5413-8,40
```

6.2.3 Creating the Exceptions Target File

The target file defines a sample target table to store information about orders which cannot be filled due to insufficient inventory.

To create the exceptions target file

1. Create a CSV file called **CSV_Exceptions_Output.csv**.
2. In the file, enter the following information:

```
ORDER_NUM, RECEIVED_DATE, BOOKSELLER_NAME, ISBN_NUM, QTY_SHORT
```

3. Save the file.

6.3 Creating the Project and Database Definitions

This section leads you through creating the flat file Project and its database definitions. You will create the following flat file database definitions:

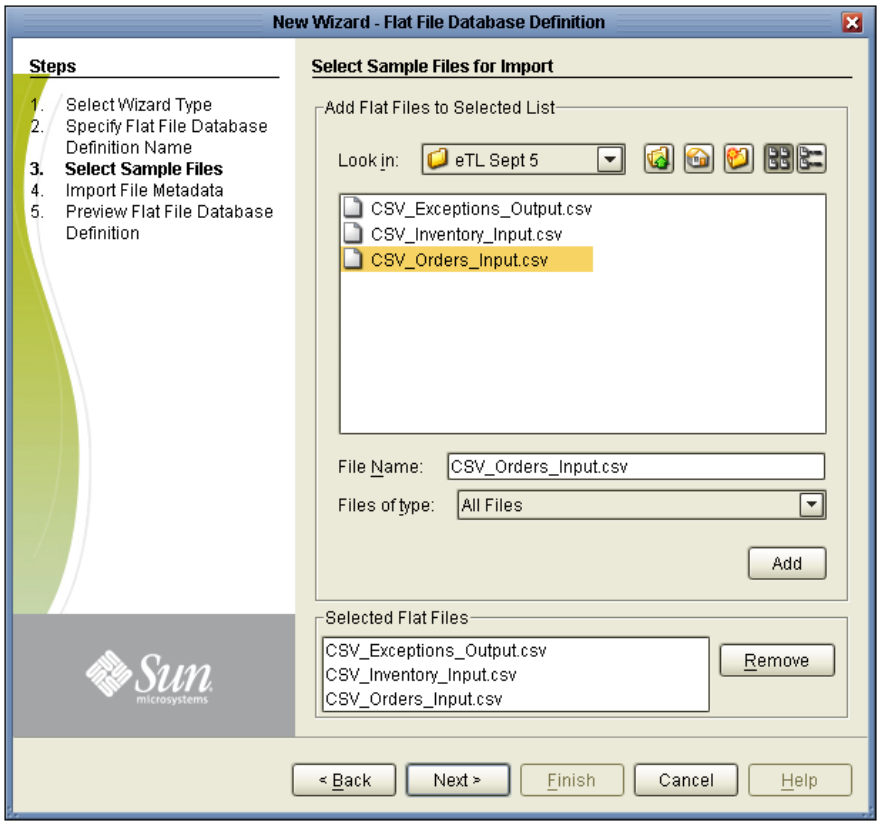
- Customer order (**Orders_Source_OTD**)
- Inventory source (**Inventory_Source_OTD**)
- Exceptions target (**Exceptions_Target_OTD**)

6.3.1 Creating the Project and Customer Order Definition

To create the Project and customer order definition

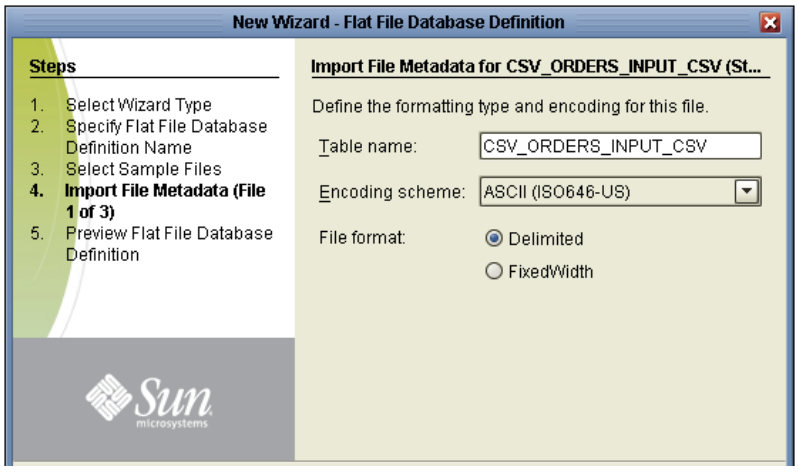
1. In the **Project Explorer** tab of the Enterprise Designer, right-click the Repository, click **New Project**. This adds a new Project under the Repository.
2. Right-click **Project1**, click **Rename** and change the name to **ProcessOrdersProject**.
3. Right-click **ProcessOrdersProject**, click **New**, and click **Object Type Definition**. The **New Object Type Definition** page appears.
4. Click **Flat File Database Definition** and click **Next**.
5. In the **New Flat File Database Name** box, enter **Orders_source_OTD** and click **Next**. The **Select Sample Files for Import** page appears.
6. In the **Look In** box, navigate to the folder where the sample files for database definitions reside. You must have write permission to the files. For information about creating the files, refer to **Creating Source and Target Sample Files** on page 78.
7. Double-click **CSV_Order_Input.csv**. This adds the files under **Selected Flat Files**.

Figure 50 Selecting the Customer Order Sample File



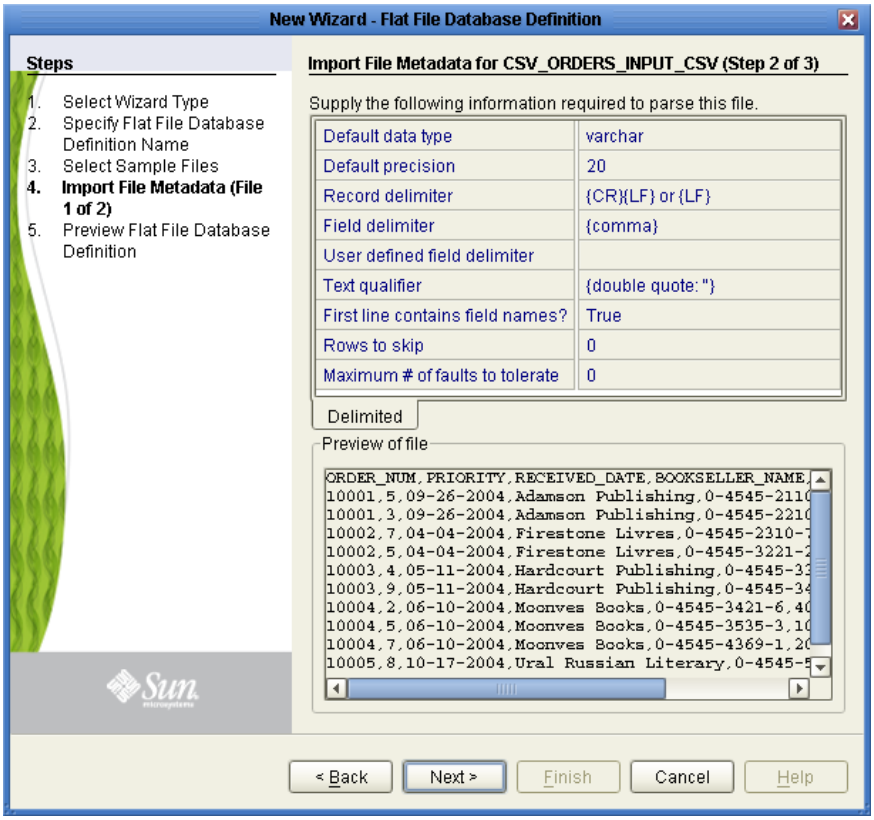
8 Click **Next**. The following page appears.

Figure 51 Importing Customer Order Source File



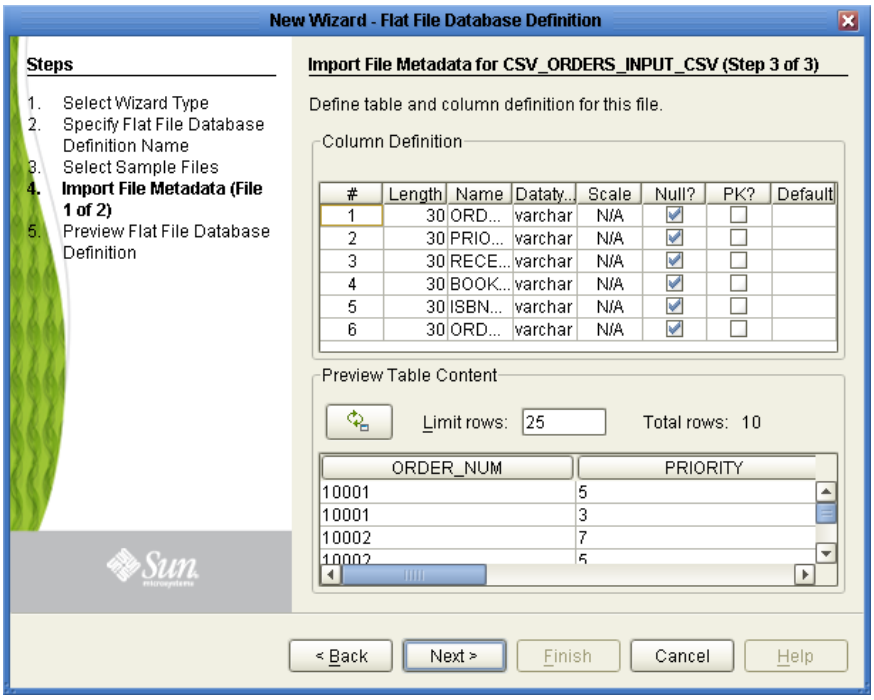
9 Click **Next**. The following page appears.

Figure 52 Setting Customer Order Table Properties



10 Click **Next**. The following page appears.

Figure 53 Defining Customer Order Record Properties



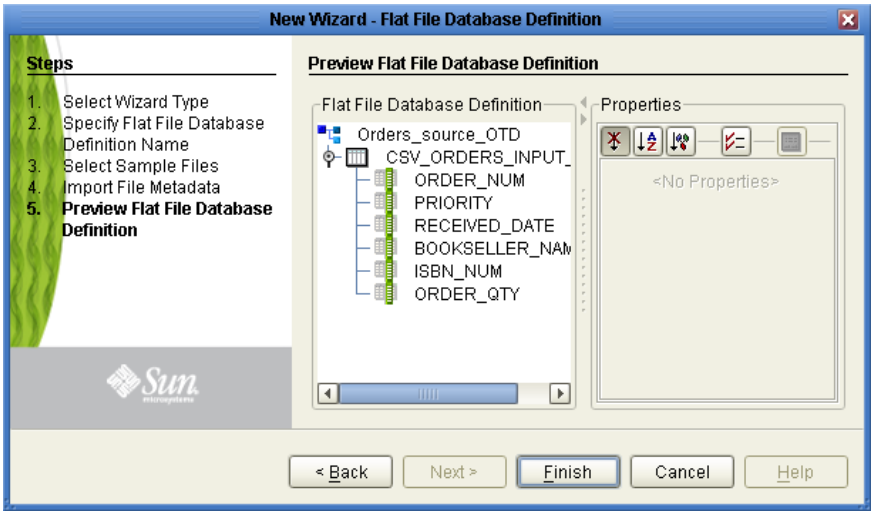
11 In row 6, ORDERS_QTY, change the **Datatype** column entry to **numeric**.

Figure 54 Setting the ORDERS_QTY to Numeric

#	Length	Name	Datat...	Scale	Null?	PK?	Def...
1	30	ORDER_NUM	varc...	N/A	☑	☐	
2	30	PRIORITY	varc...	N/A	☑	☐	
3	30	RECEIVED_...	varc...	N/A	☑	☐	
4	30	BOOKSELLE...	varc...	N/A	☑	☐	
5	30	ISBN_NUM	varc...	N/A	☑	☐	
6	6	ORDER_QTY	num...	0	☑	☐	

12 Click **Next**. You can now preview the customer order source database definition.

Figure 55 Previewing the Customer Order Database Definition



13 Click **Finish**. The **Orders_source_OTD** displays in the **Project Explorer** tab in the Enterprise Designer.

6.3.2 Creating the Inventory Database Definition

To create the inventory database definition

1 In the **Project Explorer** tab of the Enterprise Designer, right-click **ProcessOrdersProject**, click **New**, and click **Object Type Definition**. The following page appears.

2 Click **Flat File Database Definition** and click **Next**.

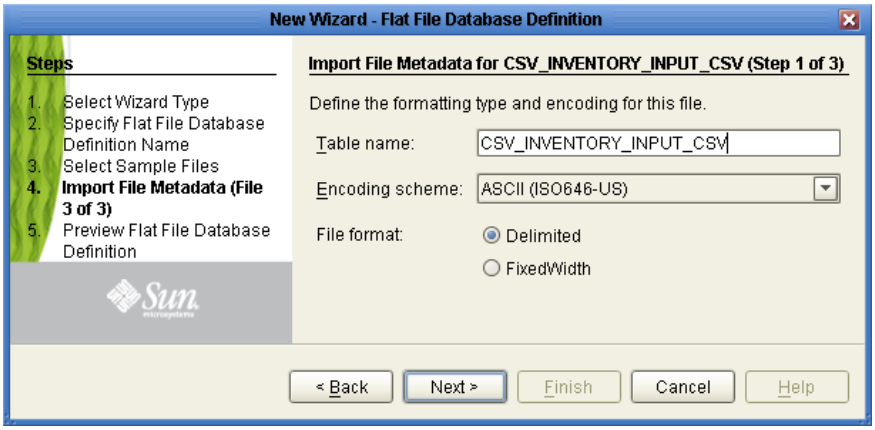
3 In the **New Flat File Database Name** box, enter **Inventory_source_OTD** and click **Next**. The **Select Sample Files for Import** page appears.

4 In the **Look In** box, navigate to the folder where the sample file for the database definitions reside.

5 Double-click **CSV_Inventory_Input.csv**. This adds the files under **Selected Flat Files**.

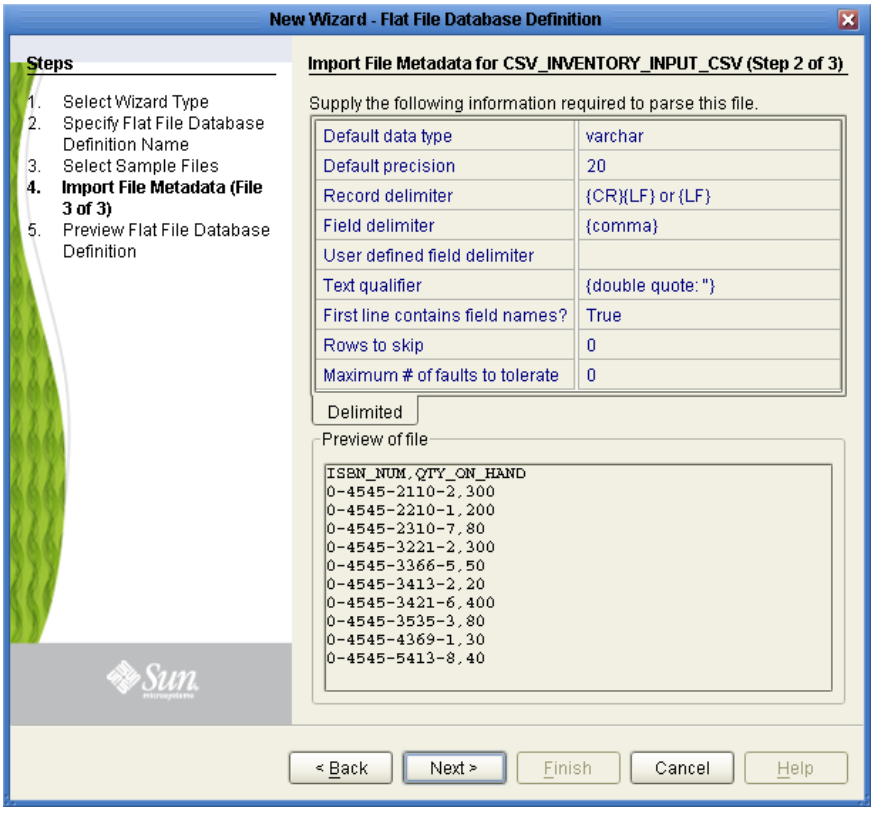
6 Click **Next**. The following page appears.

Figure 56 Importing the Inventory Source File



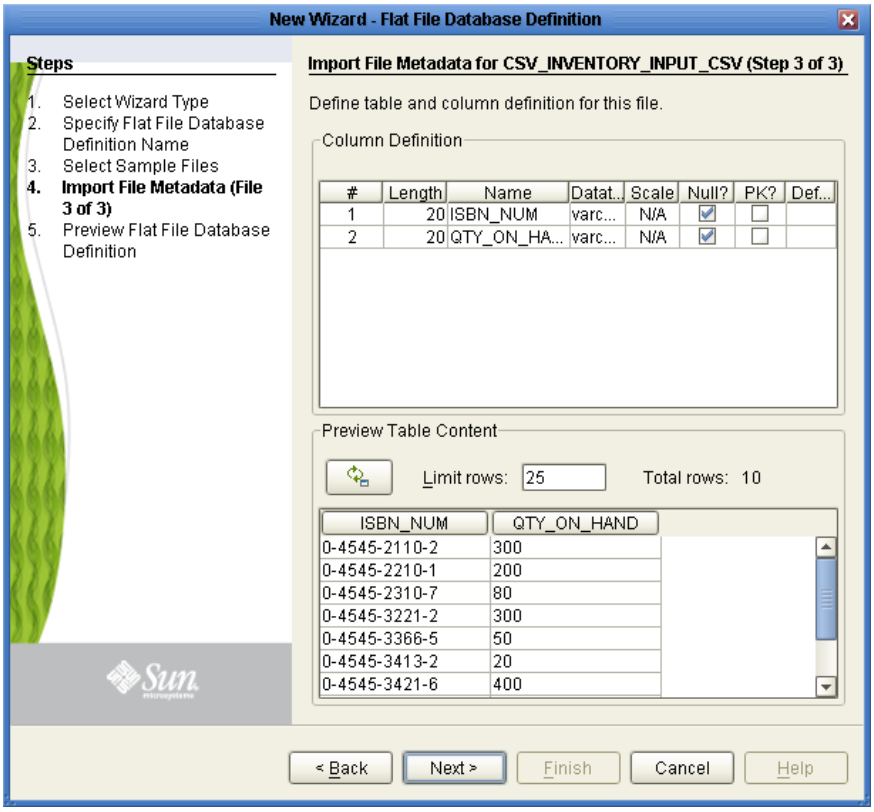
7 Click **Next**. The following page appears.

Figure 57 Setting Inventory Table Properties



8 Click **Next**. The following page appears.

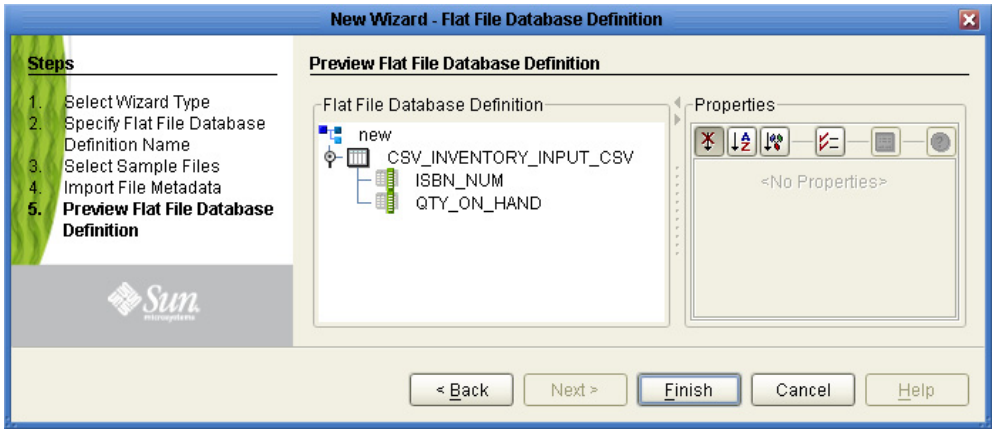
Figure 58 Defining Inventory Record Properties



9 In row 2, QTY_ON_HAND, change the **Datatype** column entry to **numeric**.

10 Click **Next**. The following page appears.

Figure 59 Previewing the Inventory Database Definition



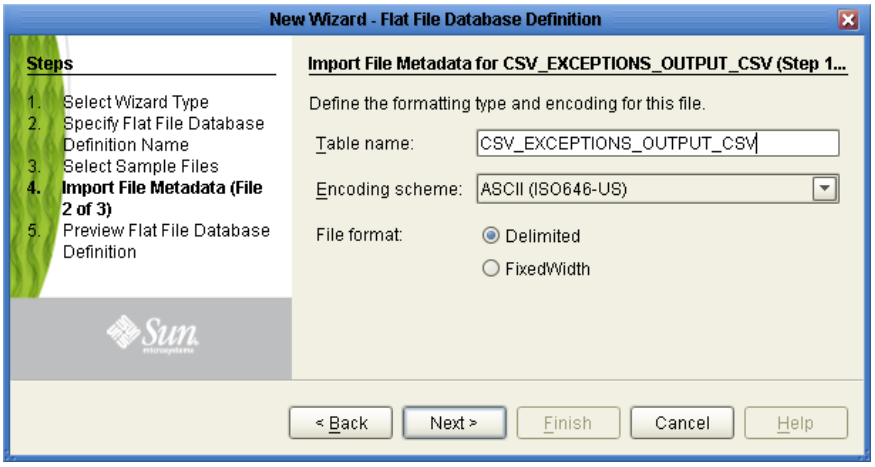
11 Click **Finish**. The **Inventory_source_OTD** displays in the **Project Explorer** tab in the Enterprise Designer.

6.3.3 Creating the Target Database Definition

To create the target database definition

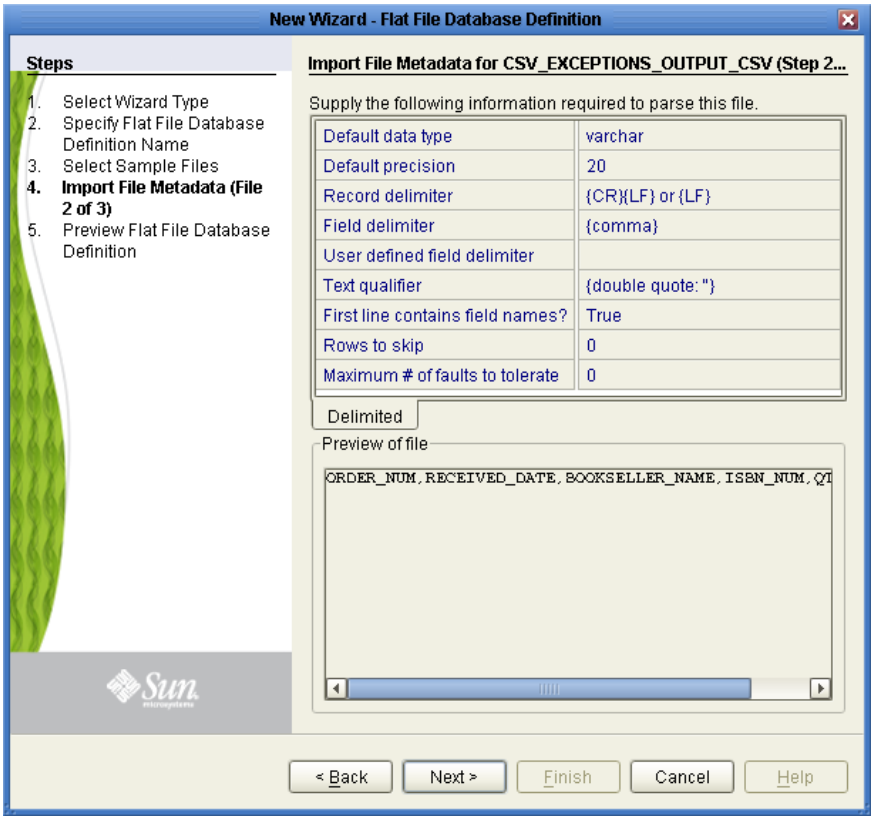
1. In the **Project Explorer** tab of the Enterprise Designer, right-click **ProcesOrdersProject**, click **New**, and click **Object Type Definition**. The following page appears.
2. Click **Flat File Database Definition** and click **Next**.
3. In the **New Flat File Database Name** box, enter **Exceptions_target_OTD** and click **Next**. The **Select Sample Files for Import** page appears.
4. In the **Look In** box, navigate to the folder where the sample input and output flat files reside.
5. Double-click **CSV_Exceptions_Output.csv**. This adds the files under **Selected Flat Files**.
6. Click **Next**. The following page appears.

Figure 60 Importing Exceptions Target File



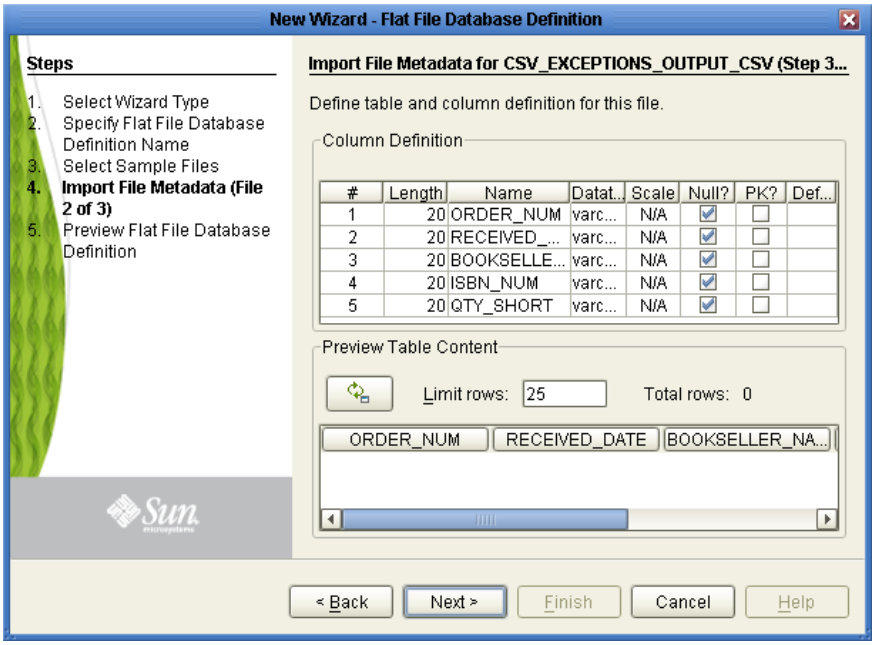
7. Click **Next**. The following page appears.

Figure 61 Setting Target Table Properties



8 Click **Next**. The following page appears.

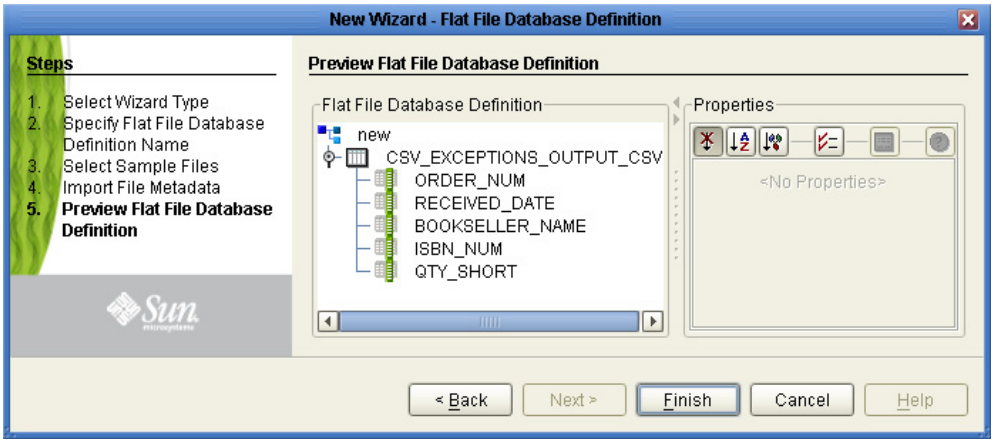
Figure 62 Defining Target Record Properties



9 In row 5, QTY_SHORT, change the **Datatype** column entry to **numeric**.

10 Click **Next**. The following page appears.

Figure 63 Previewing the Target Database Definition



11 Click **Finish**. The **Exceptions_target_OTD** displays in the **Project Explorer** tab in the Enterprise Designer.

6.4 Creating the eTL Collaboration

After you have created the database definitions for the eTL Project, create the eTL Collaboration as described in the procedure below. When you create an eTL Collaboration, you define which database definitions, source tables, and target table the Collaboration uses. You also define which (if any) source tables are to be joined.

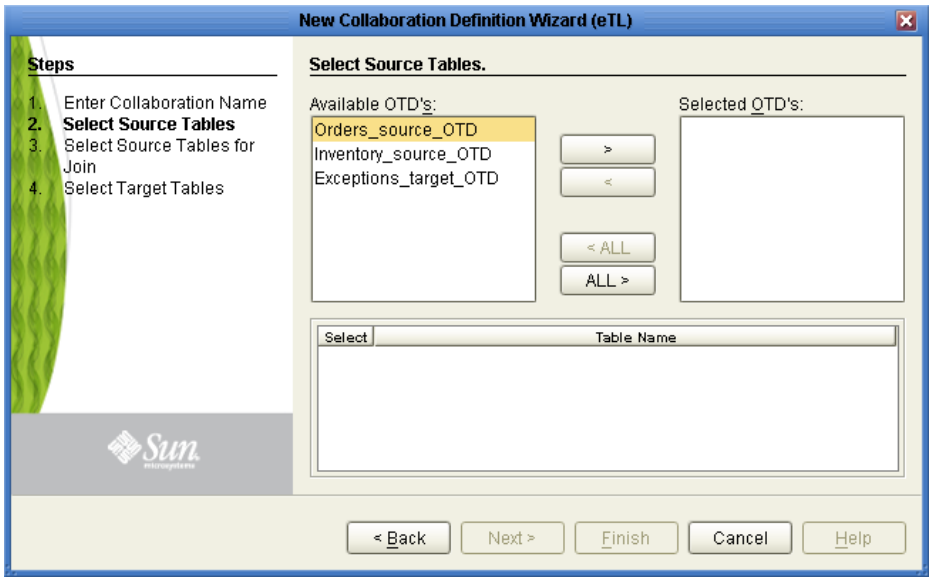
To create the eTL Collaboration

1 In the **Project Explorer** tab of the Enterprise Designer, expand **ProcessOrdersProject**, right-click the eTL Project, click **New**, and click **Collaboration Definition (eTL)**.

The **New Collaboration Definition Wizard (eTL)** dialog box appears.

2 Enter **CSV_Inv_Collab** and click **Next**. The following page appears.

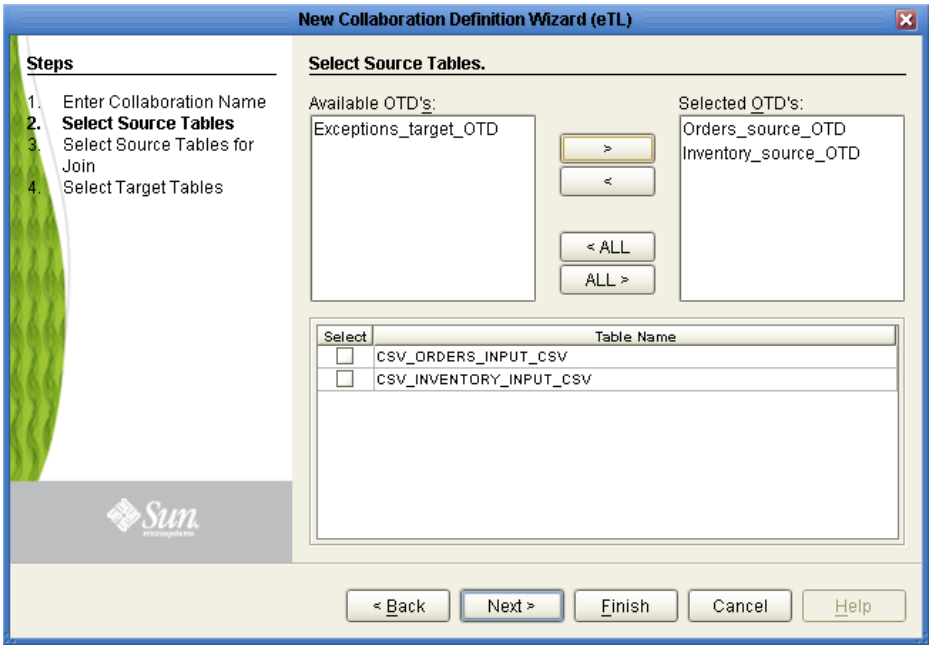
Figure 64 Selecting Source Database Definitions



3 Click **Orders_source_OTD** and **Inventory_source OTD** and click the right arrow. This moves the flat file database definitions to under **Selected OTDs**.

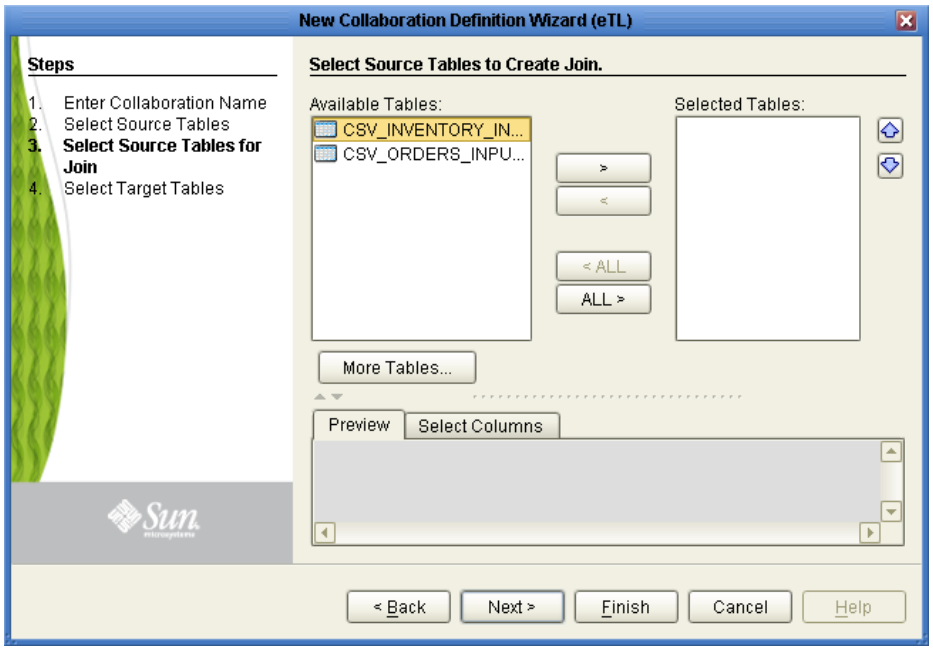
4 Under **Select**, select the available tables.

Figure 65 Selecting Source Tables



5 Click **Next**. The following page appears.

Figure 66 Selecting Tables for Joining



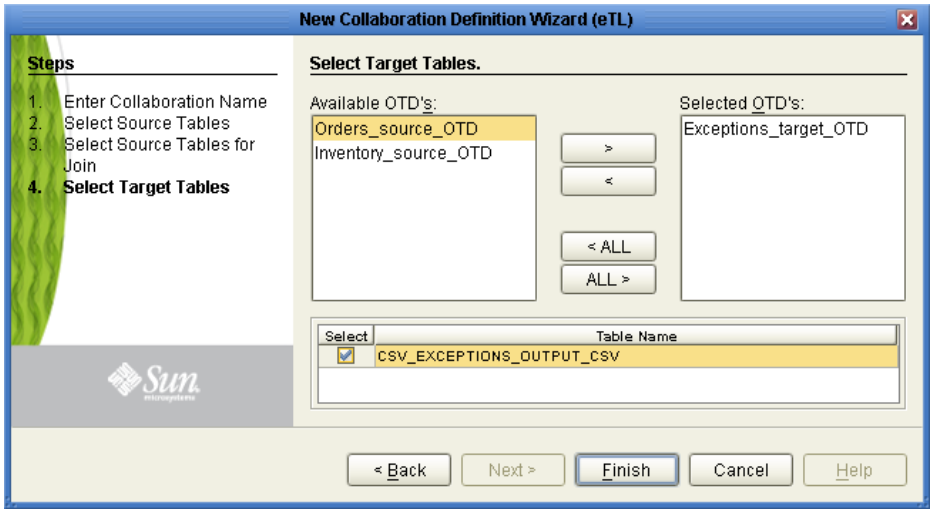
6 Under **Available Tables**, click **CSV_INVENTORY_INPUT_CSV** and **CSV_ORDERS_INPUT_CSV**, and click the right arrow. This moves the tables under **Selected Tables**.

7 Click **Next**. The **Select Target Tables** page appears.

8 Click **Exceptions_target_OTD** and click the right arrow. This moves the database definition under **Selected OTDs**.

9 Under **Select**, click **CSV_EXCEPTION_OUTPUT_CSV**.

Figure 67 Selecting the Target Table



10 Click **Finish**. This adds the Collaboration under the eTL Project in the **Project Explorer** tab. The eTL Collaboration Editor window opens and displays the Collaboration.

6.5 Creating the eTL Collaboration Logic

Up to this point, you have created a Project, flat file database definitions, and an eTL Collaboration. Now that the Project has its basic components, it is time to create the code for the eTL Collaboration as described in the procedures below.

The procedure below describes how to specify the following for the Collaboration:

- **Joining the Source Tables** on page 92
- **Mapping Source Records to Target Records** on page 96

6.5.1 Joining the Source Tables

The first step for the eTL Collaboration is to complete the join of the two source tables, **CSV_ORDERS_INPUT** and **CSV_INVENTORY_INPUT**. During the Collaboration creation, you defined that the two tables are to be joined, but did not set the join condition. In the procedure below, you define the following join condition:

- The ISBN number in the customer order table is the same as the ISBN number in the inventory table.
- The order quantity in the customer order table is larger than the inventory in stock.

The complete SQL code is as follows:

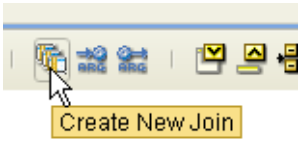
```
S1.ISBN_NUM = S2.ISBN_NUM and S1.ORDER_QTY > S2.QTY_ON_HAND
```

To join the source tables

1 If the tables are not yet joined, click the **Create New Join** icon in the eTL Collaboration Editor.

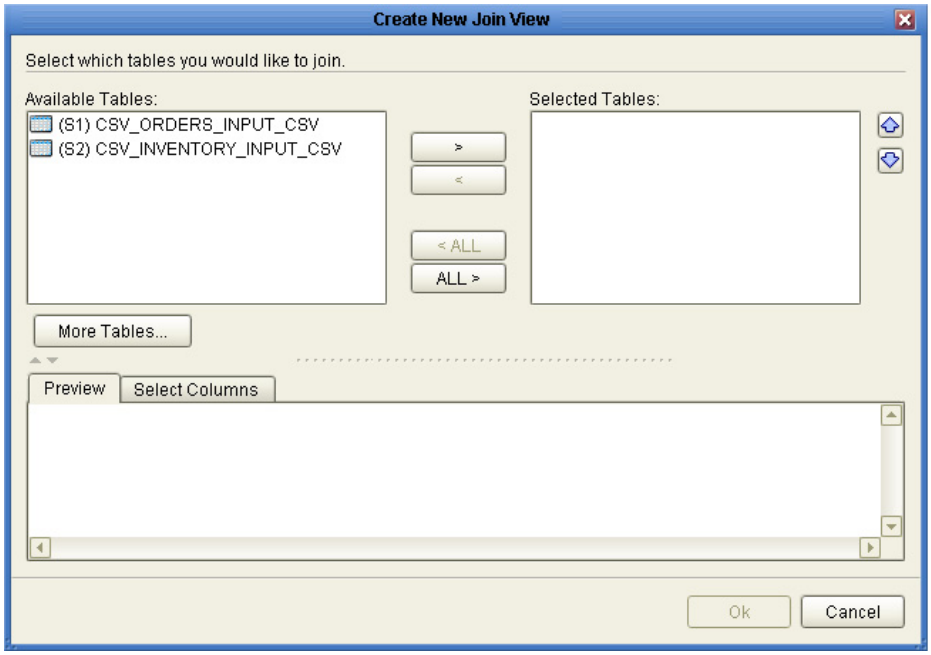
If the tables are already joined, right-click the joined tables, and click Edit Joint View and go to step 3.

Figure 68 Creating the New Join Condition



The following page appears.

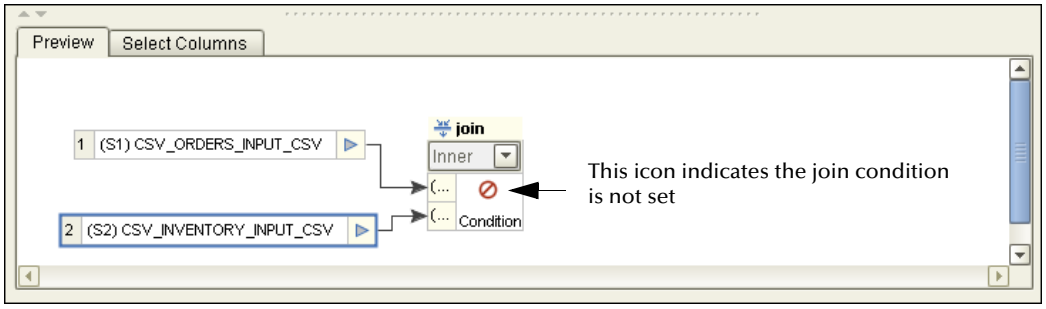
Figure 69 Selecting Tables to Be Joined



2 Click **CSV_ORDERS_INPUT_CSV** and **CSV_INVENTORY_INPUT_CSV** and click the right arrow. This moves the tables under **Selected Tables**.

The **Preview** box shows the join between the tables. The red **Join** icon indicates that the Join condition is not set.

Figure 70 Previewing Tables To Be Joined

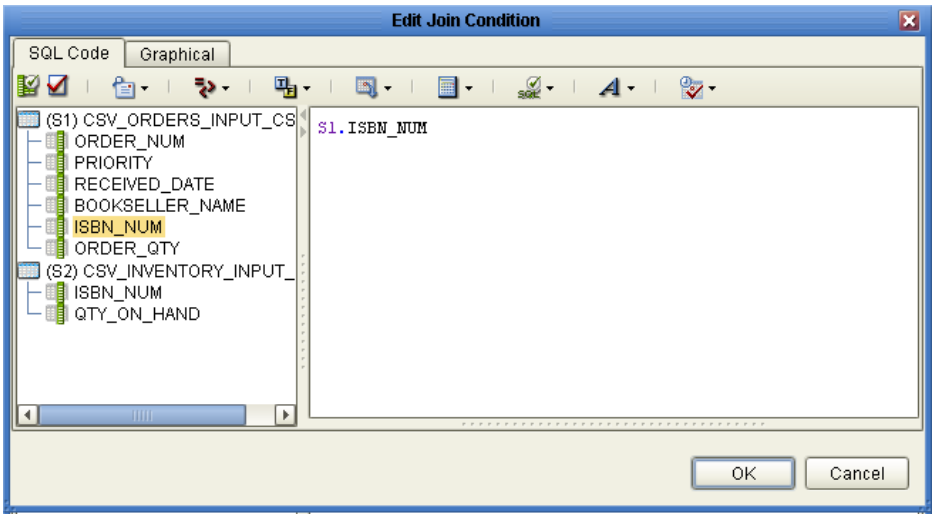


Note: *With reference to Figure 70, Inner (join) is the default. You can also select the options left outer, right outer or full outer.*

3 Double-click the Join condition. The **Edit Join Condition** window appears.

4 Drag **ISBN_NUM** from **CSV_ORDERS_INPUT_CSV** to the pane on the right.

Figure 71 Creating the Join Condition

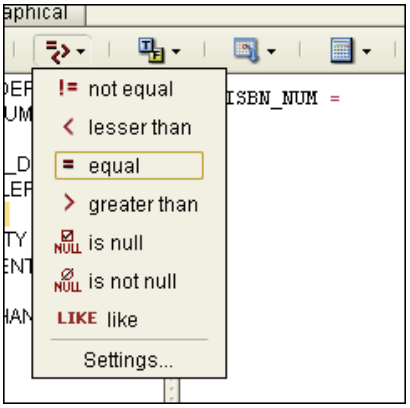


Note: *With reference to Figure 71, the condition can be written by using the **SQL Code** tab or the **Graphical** tab; you can toggle between these views. Use the **Validate** icon to verify the code. You can edit and enter SQL code using the following conditions (GUIs): Join, Extraction, Validation, Join/filter.*

5 Click **Comparison Operators** icon, and drag **= equal** onto the pane after SI.IBSN_NUM. The line now reads:

```
S1.ISBN_NUM =
```

Figure 72 Selecting the Equal Operator



6 Drag **ISBN_NUM** from **CSV_INVENTORY_INPUT_CSV** to the pane on the right. after the equal sign to form the following line:

```
S1.ISBN_NUM = S2.ISBN_NUM
```

7 Click **Boolean Operators** and drag **AND** onto the pane to form the following line:

```
S1.ISBN_NUM = S2.ISBN_NUM and
```

8 Drag **ORDER_QTY** from **CSV_ORDERS_INPUT_CSV** after **and**. The line now reads:

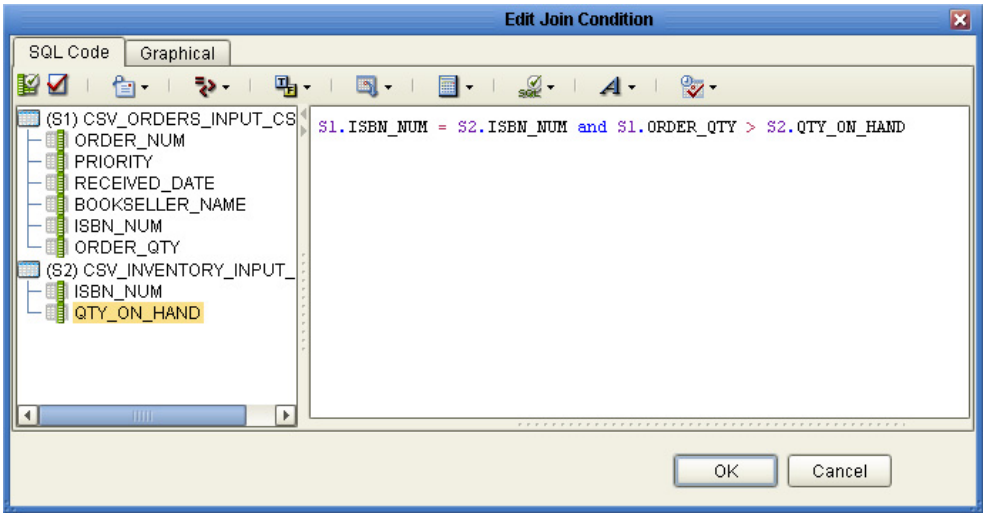
```
S1.ISBN_NUM = S2.ISBN_NUM and S1.ORDER_QTY
```

9 Click **Comparison Operators** icon, and type > in the pane after **S2.ORDER_QTY.** The line now reads:

```
S1.ISBN_NUM = S2.ISBN_NUM and S1.ORDER_QTY >
```

10 Drag **QTY_ON_HAND** from **CSV_INVENTORY_INPUT_CSV** after **S1.ORDER_QTY**. The condition is now complete.

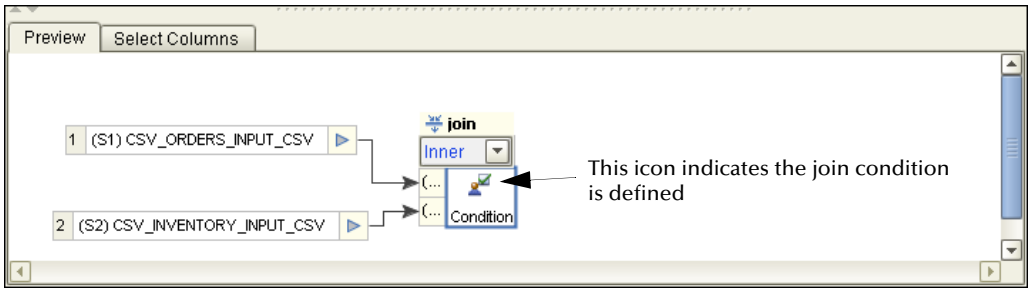
Figure 73 Completed Join Condition



11 Click the **SQL Validate Condition** icon. The message "Condition is valid" should appear.

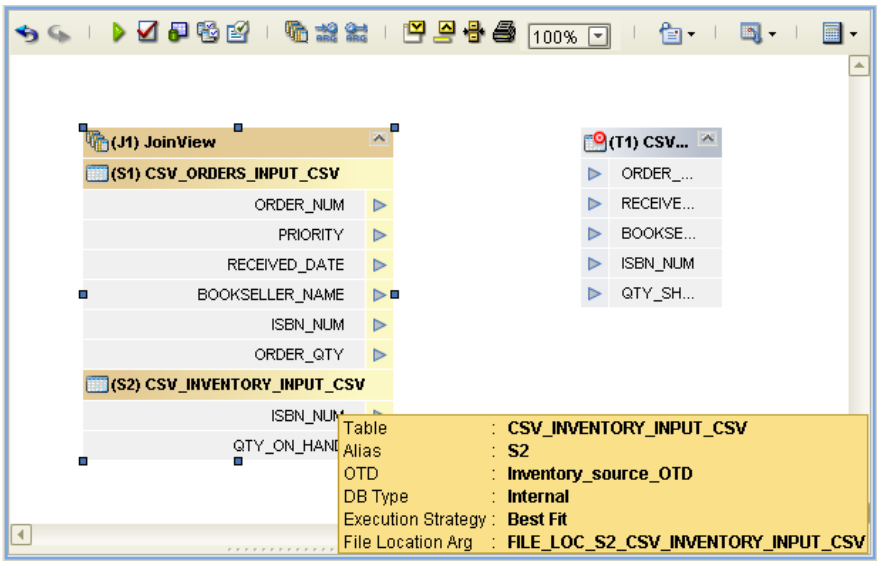
12 Click **OK**. The **Edit Join Condition** window closes. The Preview box shows that the join condition is set.

Figure 74 Defined Join Condition



13 Click **OK**. The eTL Collaboration Editor shows the Join View.

Figure 75 Join View in Collaboration Editor

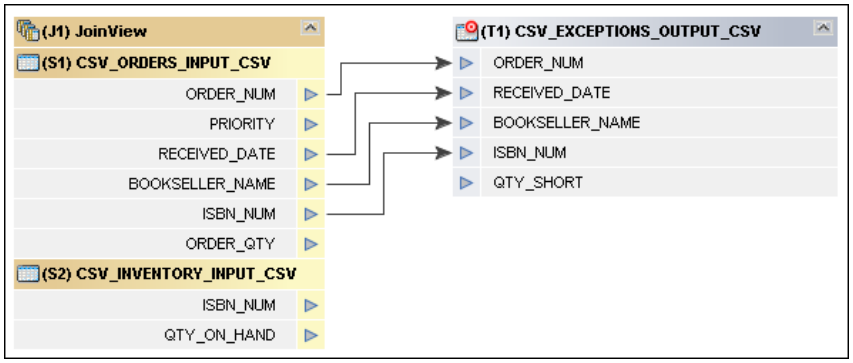


6.5.2 Mapping Source Records to Target Records

To map source records to target records

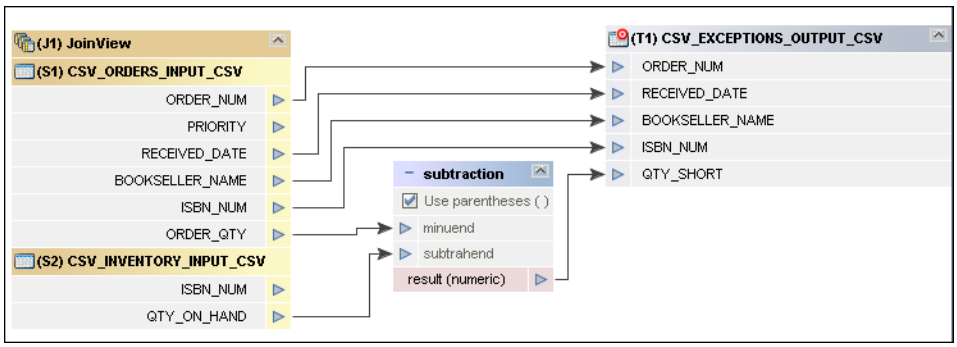
1. In the Collaboration Editor, map the source records from the **CSV_ORDERS_INPUT_CSV** table to the target records in the **CSV_EXCEPTIONS_OUTPUT_CSV** as shown in the figure below.

Figure 76 Mapping Source Records to Target Records



2. Click the **Number Operators** icon and drag the **- subtraction** operator to the canvas. This adds a subtraction operation to the canvas.
3. Define the subtraction operation as shown in the figure below:

Figure 77 Defining the Subtraction Operation

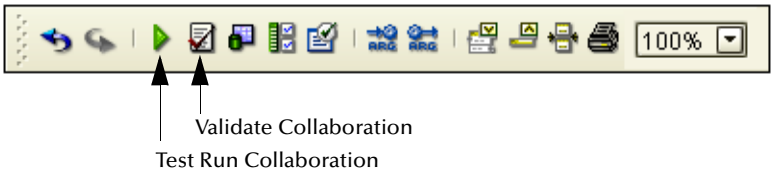


4 Click the **Save** icon.

6.6 Validating and Testing the Collaboration

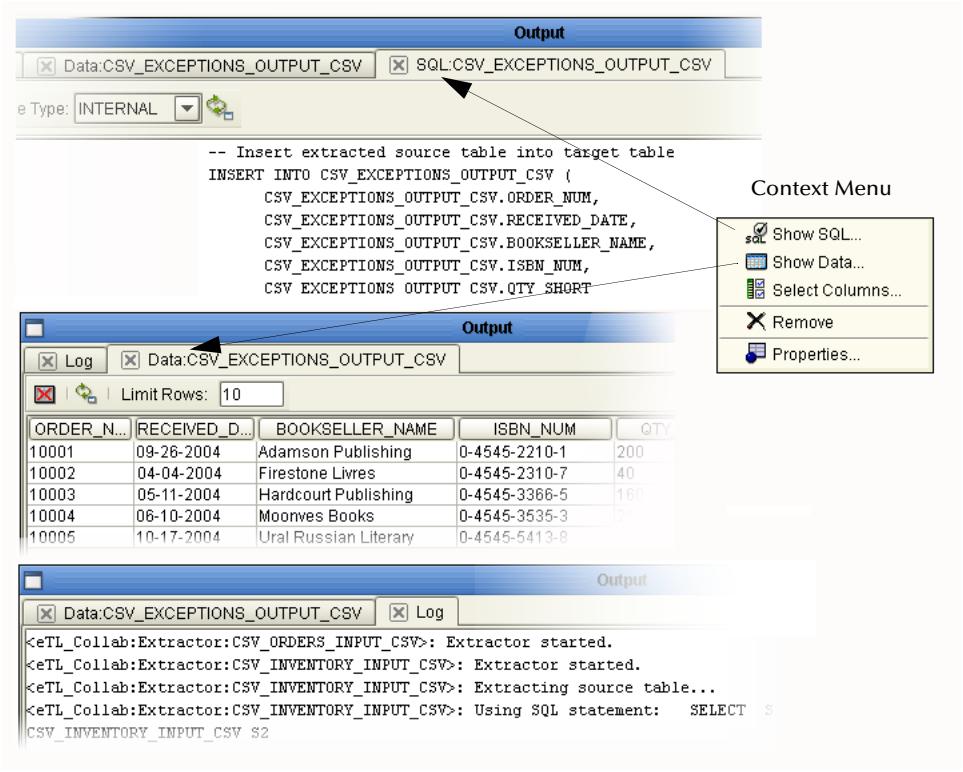
You can validate your Collaboration to verify the generated Collaboration source code syntax at design time. When you execute a test, you can check the SQL, the result data, or the log. (The log lists execution, success or failure messages, and other messages. See *eGate Integrator System Administration Guide* for more information about log files.) See the following figure.

Figure 78 Validating and Testing Collaborations



- Click the **Validate Collaboration** icon to verify that there are no violations of the Collaboration rules. The **Output** pane display the **Validation** tab with the results.
- Click the **Test Run Collaboration** icon to run the Collaboration. The **Output** pane displays the **Log** tab with the results.
- Right-click the **Target** table. The Context menu appears. See the following figure.

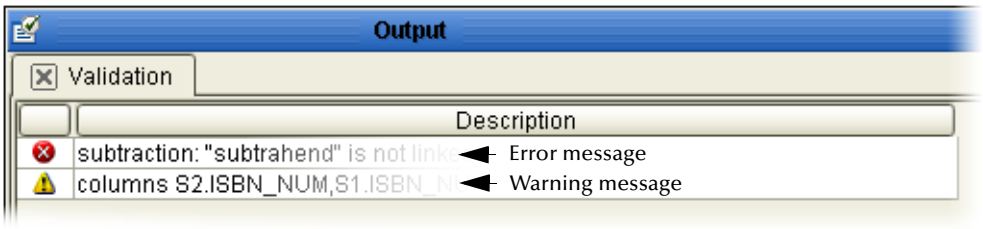
Figure 79 Show SQL, Data, and Log



- Verify that the logic has produced the correct output data.

  See the following example of warning and error messages. These message are the result of clicking **Validate Collaboration**; if any issues existed when you clicked **Test Run Collaboration**, an error will appear.

Figure 80 Warning and Error Message Example

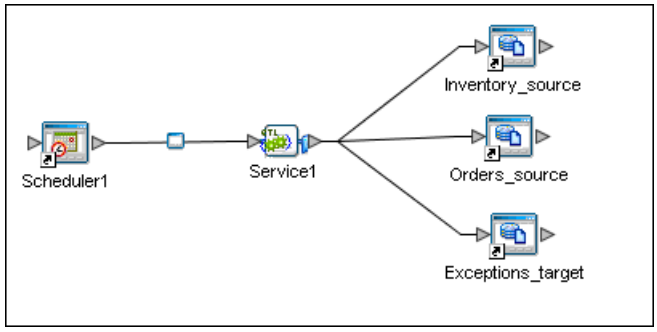


- Error message - not a valid eTL Collaboration
- Warning message - warning only

6.7 Creating the Connectivity Map

After having created the eTL Collaboration, you can create the Connectivity Map for the eTL Project. Once you have completed the procedure below, the Connectivity Map looks similar to the following:

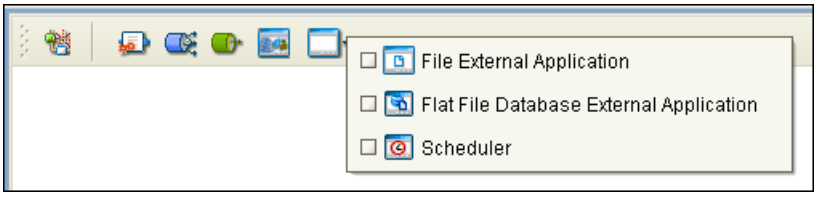
Figure 81 Completed Connectivity Map



To create the Connectivity Map

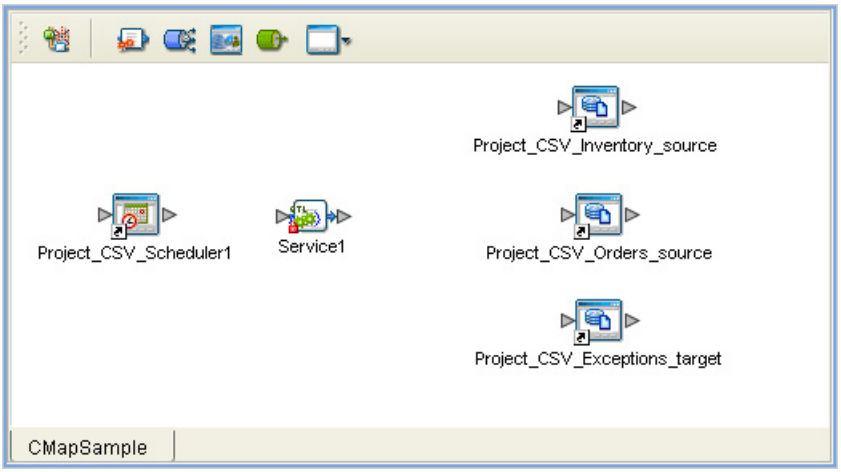
1. In the **Project Explorer** tab of the Enterprise Designer, right-click the eTL Project, click **New**, and click **Connectivity Map**. This adds a new Connectivity Map under the eTL Project.
2. Click the **External Applications** icon and click **Flat File Database External Application**.

Figure 82 Selecting the Flat File Application



3. Click the **External Applications** icon and click **Scheduler**.
4. Drag the **Scheduler** icon to the Connectivity Map.
5. Drag three **Flat File Database** icons to the Connectivity Map.
6. From the **Project Explorer** tab, drag **CSV_Inv_Collab** to the Connectivity Map.
7. Arrange the icons as shown in the figure below.

Figure 83 Arranging Connectivity Map Components



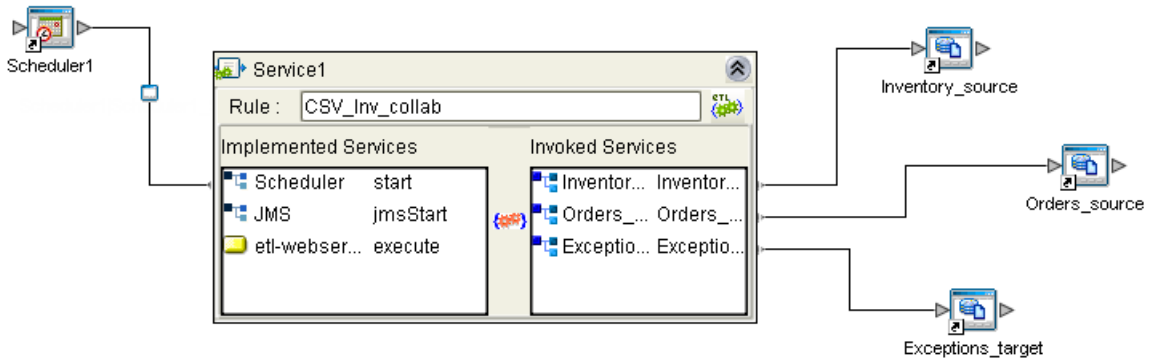
8 Double-click the **CSV_Inv_Collab** icon in the Connectivity Map. This opens the **Service** window.

9 Link the following components to each other by dragging the components from the Linking window to the Connectivity Map icons:

- Scheduler OTD to the **Scheduler** icon; double-click the Scheduler link and configure the Scheduler.
- **Inventory_source_OTD**, **Orders_source_OTD**, and **Exceptions_target_OTD** to each of the three **Flat File Database External Applications** icons.

The figure below shows the linking details.

Figure 84 Linking Connectivity Map Components



10 Click the double chevron in the upper right-hand corner to close the **Service** window.

11 Click the **Save** icon.

6.8 Creating the eTL Environment

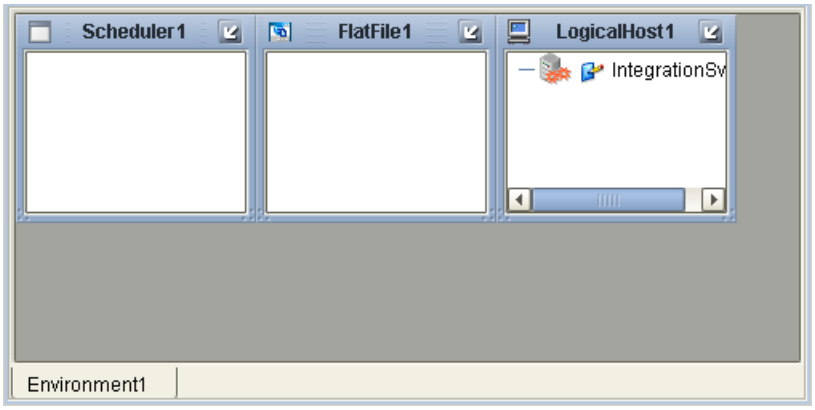
The procedure below describes how you create the eTL Environment. The Environment defines the systems on which each service runs. For this Project, the Environment includes Logical Host, Flat File Database, and Scheduler systems.

To create the eTL Environment

1. In the **Environment Explorer** tab of the Enterprise Designer, right-click the Repository and click **New Environment**.
2. Right-click the Environment, click **New**, and click **Logical Host**.
3. Right-click the Logical Host, click **New**, and click **Integration Server** or **Sun Java System Application Server**.
4. For the integration/application server, configure the following properties and click **OK**:
   - In the **Application Workspace Directory** property, specify the workspace directory.
   - Set the Integration Server URL, username, and password.
5. Right-click **Environment1**, click **New**, **Flat File Database External System**, enter **FlatFile1**, and click **OK**.
6. Right-click **Environment1**, click **New**, **Scheduler**, enter **Scheduler1**, and click **OK**.

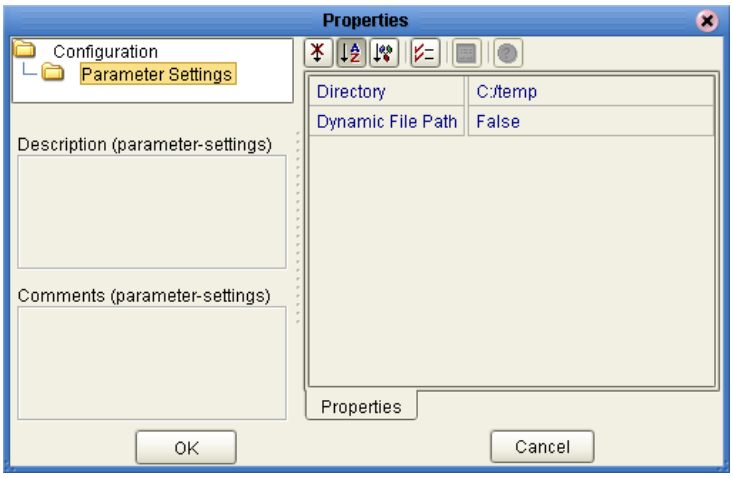
   The figure below shows the Environment.

Figure 85 The eTL Environment



7. Right-click **FlatFile1** and click **Properties**. The **Properties** sheet appears.

Figure 86 Configuring Flat File Database External Applications



8 In the **Directory** box, enter the folder where the flat files reside. If the flat files are in multiple locations, create multiple flat file database external applications for each location.

9 Click **OK**.

6.9 Creating the Deployment Profile

After creating the eTL Environment, you are now ready to create the Deployment Profile for the eTL Project. Deployment Profiles specify which service runs on which system in the eTL Environment you created.

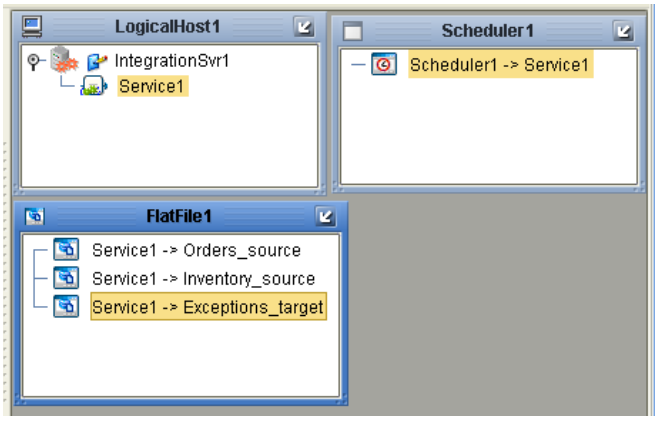
To create the Deployment Profile

1 In the **Project Explorer** tab of the Enterprise Designer, right-click the eTL Project and click **New Deployment Profile**.

2 Enter the name for the Deployment Profile, click **Environment1**, and click **OK**.

3 Click the **Automap** icon and click **OK**. This creates the following Deployment Profile:

Figure 87 Completed Deployment Profile



4 Click the **Save** icon.

For instructions on activating deployments and running Projects, refer to the *eGate Integrator User's Guide.*

Chapter 7

Managing Runtime eTL Projects

This chapter describes how to manage runtime eTL Projects. Once you have built and deployed an eTL Project, you can monitor it using the Enterprise Manager. This chapter includes information about monitoring eTL Projects as well as reconfiguring deployed Projects.

What's in This Chapter

7.1 Reconfiguring Runtime Projects

This section describes how you reconfigure the properties of runtime Projects. For Project connecting to a DB2 database, you can change the schema and catalog at runtime.

7.1.1 Reconfiguring Project Properties

To reconfigure a currently deployed Project, you change the configuration and then build and redeploy the Deployment Profile. If you also made changes to the logical properties in the Environment, apply the changes to the Logical Host as described in the next section.

The procedure below describes how you reconfigure the logical Project properties.

To reconfigure runtime Project properties

1. In the **Environment Explorer** tab of Enterprise Designer, expand the Repository, right-click the Environment for the Project to be reconfigured and click **Properties**.
2. Configure the logical properties in the Java CAPS Environment as described the *eGate Integrator User's Guide* and if appropriate, the eWay User's Guide for the eWay used in the Project.
3. Build and redeploy the Project.

7.1.2 Modifying DB2 Catalog and Schema Configurations at Runtime

The procedure below describes how to change the DB2 catalog and schema configurations for runtime Projects in the Java CAPS Environment.

Note: *Changing the DB2 configurations in the Java CAPS Environment provides a quick way to change these settings; the alternate method for changing these configurations would be to modify the DB2 table's catalog and schema name inside the eTL Collaboration and modifying the DB2 Properties' JDBC Connector Setting.*

To modify DB2 catalog and schema configurations at runtime

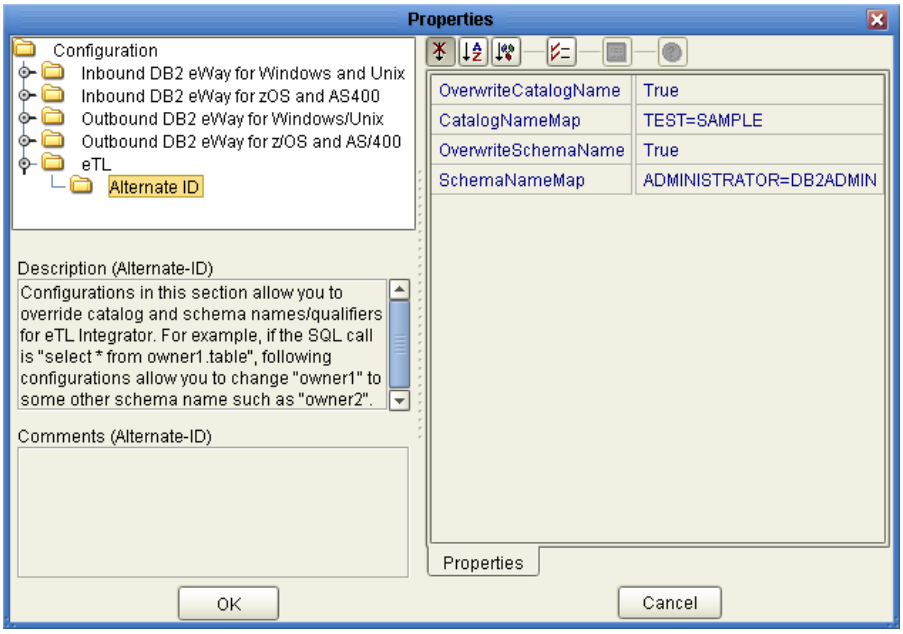
1. In the **Environment Explorer** tab of Enterprise Designer, expand the Repository, right-click the DB External Application to be reconfigured and click **Properties**. The **Properties** sheet appears.
2. Expand **eTL** and click **Alternate ID**.
3. Enter the following items and click **OK**:

Table 8 Runtime DB2 Schema and Catalog Configurations

Options	Description
OverwriteCatalogName	Set this property to **True** to overwrite the current runtime setting and to use the catalog information entered in the **CatalogNameMap** box.
CatalogNameMap	Enter a string mapping the current runtime catalog setting to the new the DB2 catalog that you want the runtime Project to use. Use the following format: *oldname1=newname1;oldname2=newname2*
OverwriteSchemaName	Set this property to **True** to overwrite the current runtime setting and to use the schema information entered in the **SchemaNameMap** box.
SchemaNameMap	Enter a string mapping the current runtime schema setting to the new the DB2 schema that you want the runtime Project to use. Use the following format: *oldname1=newname1;oldname2=newname2*

The following figure shows an example of the runtime DB2 schema and catalog configurations.

Figure 88 Modifying DB2 Schema and Catalog Configurations at Runtime



4 Build and redeploy the Project.

7.2 Monitoring eTL Collaborations

You monitor eTL Collaborations with the Enterprise Manager. The procedure below describes how to do this.

The eTL plugin for Enterprise Manager must be installed to view eTL Collaborations. For information, refer to **"Deploying the eTL Enterprise Manager Plug-in" on page 19**.

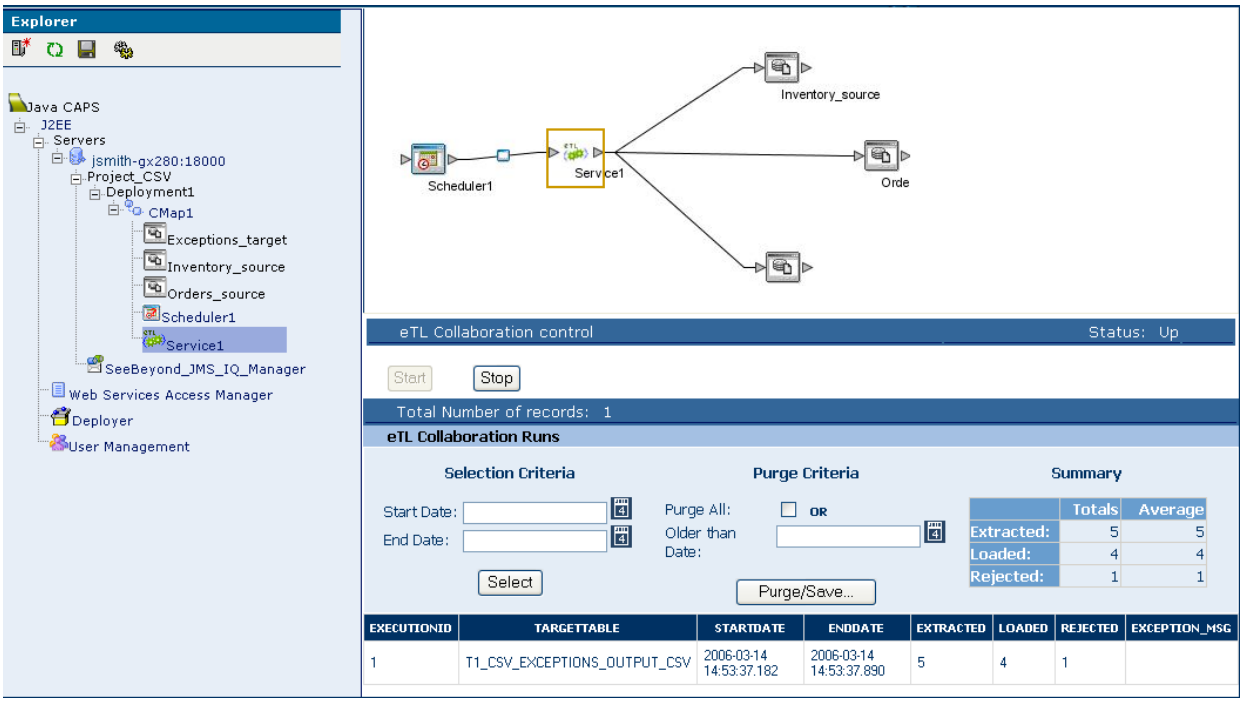
To monitor eTL Collaborations

1 In Enterprise Manager, open the Connectivity Map for the runtime Project.

2 In the Connectivity Map, click the eTL Collaboration. The pane below the Connectivity Map displays the events for the runtime Collaboration as shown in the following figure.

Note: *To view a full-screen pane for the eTL Collaboration information, click the eTL Collaboration in the Explorer pane.*

Figure 89 Monitoring the eTL Collaboration

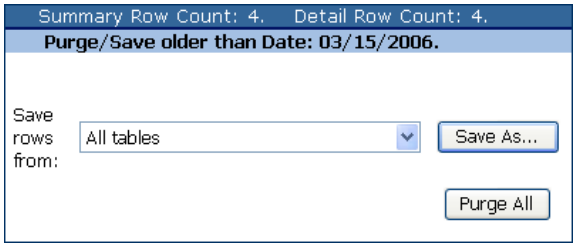


3 To view events for a particular date range, enter a start and end date in the **Start Date** and **End Date** fields and click **Select**.

4 To purge or save all events from the eTL Collaboration Control window, select the **Purge All** option, click **Purge/Save**, and click **OK** when the "Do you want to purge all records?" message appears.

5 To purge events up to a specific date, enter a date in the **Older than Date** field and click **Purge/Save**. The following window appears.

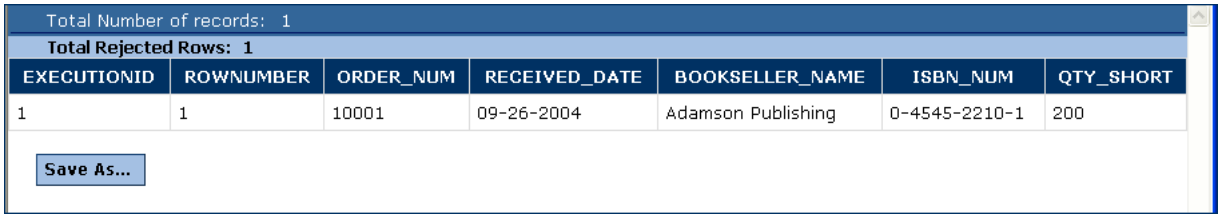
Figure 90 Purging Rows from the eTL Collaboration Monitor



6 Click **Purge All**, and then **Close** to purge the information from the eTL Collaboration Control pane or Click **Save As** and follow the on-screen instructions to save the information. The information is saved as a text file.

7 To view rejected rows, click the number under the **Rejected** column. To save the rejected row information, click **Save As** and follow the on-screen instructions to save the information. The information is saved in a **RejectedData*.txt** file.

Figure 91 Viewing Rejected Rows



7.3 Log Files and Alerts

eTL alerts are logged in the Logical Host log file, **server.log**. For information about this log file, and how to change the logging level in Enterprise Manager, refer to the *Sun SeeBeyond eGate Integrator System Administrator Guide*.

Runtime eTL Integrator Projects include the following predefined alerts:

Table 9 eTL Integrator Project Alerts

Code	Description	Severity Level
ETL-00001	Execution of eTL Collaboration *name* has started.	Info
ETL-00002	Execution of eTL Collaboration *name* has completed.	Info
ETL-00003	A critical error occurred during Collaboration *name* execution.	Error
ETL-00004	Transformation of target table *name* has started.	Info
ETL-00005	Transformation of target table *name* has completed.	Info
ETL-00006	A critical error occurred during target table *name* transformation.	Error

7.4 Log Levels

The following are eTL server modules. By default, each of these have log levels set to WARNING.

- `com.stc.flatfile.db`
- `org.axiondb`
- `com.stc.sql.framework`
- `com.stc.etl`

You can change the severity level. Log levels are explained in various locations in the *Sun SeeBeyond eGate Integrator System Administrator Guide.*

Appendix A

Data Type and Operator Support

This appendix provides information about the data types and operators that eTL Integrator supports.

What's in This Appendix

8.5 Supported Data Types

Table 10 Supported Data Types

JDBC Data Type	Oracle 9i	DB2	Flat File DB	SQL Server	Sybase
array	varray	array	array	array	array
bigint	numeber	bigint	bigint	bigint	bigint
binary	raw	char	binary	binary	binary
bit	numeber	numeric	bit	bit	bit
blob	blob	blob	blob	blob	blob
boolean	boolean	boolean	boolean	boolean	boolean
char	char	char	char	char	char
clob	clob	clob	clob	clob	clob
datalink	datalink	datalink	datalink	datalink	datalink
date	date	date	date	datetime	datetime
decimal	decimal	decimal	decimal	decimal	decimal
distinct	distinct	distinct	distinct	distinct	distinct
double	double precision	float	double	double precision	double precision
float	float	float	float	float	float
integer	integer	integer	integer	integer	integer
java_object	java_object	java_object	java_object	java_object	java_object
longvarbinary	long raw	long varchar	longvarbinary	image	image

Table 10 Supported Data Types (Continued)

JDBC Data Type	Oracle 9i	DB2	Flat File DB	SQL Server	Sybase
longvarchar	long	long varchar	longvarchar	text	text
null	null	null	null	null	null
numeric	numeber	numeric	numeric	numeric	numeric
other	varchar	varchar	varchar	varchar	varchar
real	real	real	decimal	real	real
ref	ref	ref	ref	ref	ref
smallint	smallint	smallint	integer	smallint	smallint
struct	object	struct	struct	struct	struct
time	date	time	time	datetime	datetime
timestamp	timestamp	timestamp	timestamp	datetime	datetime
tinyint	numeber	numeric	tinyint	tinyint	tinyint
varbinary	raw	varchar	varbinary	varbinary	varbinary
varchar	varchar	varchar	varchar	varchar	varchar

8.6 Operator/Database Support

The table below shows what operators are supported for what supported databases. The list below provides a legend:

- **Y**: Supported using database native functions
- **YD**: Supported natively using non ANSI standard function or using multiple native functions
- **S:** Supported through Java operators (use forces pipeline execution strategy)

Table 11 Operator/Database Support

Operator	Type	Oracle 9i	DB2	Flat Files	SQL Server 2000	Sybase 11
NOT EQUAL !=	Comparison	Y	Y	Y	Y	Y
EQUAL =	Comparison	Y	Y	Y	Y	Y
GREATER THAN >	Comparison	Y	Y	Y	Y	Y
LESS THAN <	Comparison	Y	Y	Y	Y	Y
GREATER THAN or EQUAL TO >=	Comparison	Y	Y	Y	Y	Y
LESS THAN or EQUAL TO <=	Comparison	Y	Y	Y	Y	Y

Table 11 Operator/Database Support (Continued)

Operator	Type	Oracle 9i	DB2	Flat Files	SQL Server 2000	Sybase 11
IS NULL	Comparison	Y	Y	Y	Y	Y
IS NOT NULL	Comparison	Y	Y	Y	Y	Y
LIKE	Comparison	Y	Y	Y	Y	Y
AND	Boolean	Y	Y	Y	Y	Y
OR	Boolean	Y	Y	Y	Y	Y
Absolute value	Number	Y	Y	Y	Y	Y
Addition	Number	Y	Y	Y	Y	Y
Multiplication	Number	Y	Y	Y	Y	Y
Subtraction	Number	Y	Y	Y	Y	Y
Division	Number	Y	Y	Y	Y	Y
Mod / Remainder	Number	Y	Y	Y	Y	Y
Sum	Number/ Aggregate	Y	Y	Y	Y	Y
Minimum	Number/ Aggregate	Y	Y	Y	Y	Y
Maximum	Number/ Aggregate	Y	Y	Y	Y	Y
Average	Number/ Aggregate	Y	Y	Y	Y	Y
SQL Literals	SQL Specific	Y	Y	Y	Y	Y
SQL NULL	SQL Specific	Y	Y	Y	Y	Y
CASE	SQL Specific	Y	Y	Y	Y	Y
CAST AS	SQL Specific	Y	Y	Y	Y	Y
COUNT	SQL Specific	Y	Y	Y	Y	Y
To lower case	String	Y	Y	Y	Y	Y
To upper case	String	Y	Y	Y	Y	Y
Left trim	String	Y	Y	Y	Y	Y
Right trim	String	Y	Y	Y	Y	Y
String length	String	Y	Y	Y	Y	Y
Concatenate	String	Y	Y	Y	Y	Y

Table 11 Operator/Database Support (Continued)

Operator	Type	Oracle 9i	DB2	Flat Files	SQL Server 2000	Sybase 11
Substring	String	Y	Y	Y	Y	Y
Replace	String	Y	Y	Y	Y	YD
String to Date from various formats	Date	S	S	Y	S	S
Extract various Date and time part from Timestamp	Date	S	S	Y	S	S
Add to Date various intervals	Date	Y	Y	Y	Y	Y
Convert Date to various String formats	Date	S	S	Y	S	S
Date difference in various time units	Date	Y	Y	Y	Y	Y
"""NOW"" current timestamp"	Date	Y	Y	Y	Y	Y
Cleanse Person Name						
Cleanse Person Name - First Name - US & UK	Cleansing	S	S	Y	S	S
Cleanse Person Name - Last Name - US & UK	Cleansing	S	S	Y	S	S
Parse and Cleanse Address						
Unit or Apartment Desc - US & UK	Cleansing	S	S	Y	S	S
Unit or Apartment # - US & UK	Cleansing	S	S	Y	S	S

Table 11 Operator/Database Support (Continued)

Operator	Type	Oracle 9i	DB2	Flat Files	SQL Server 2000	Sybase 11
Building or House # - US & UK	Cleansing	S	S	Y	S	S
Matched Street Name - US & UK	Cleansing	S	S	Y	S	S
Original Street - US & UK	Cleansing	S	S	Y	S	S
Street Type - US & UK	Cleansing	S	S	Y	S	S
Parse and Cleanse Business Name						
Firm Name	Cleansing	S	S	Y	S	S
Firm Type	Cleansing	S	S	Y	S	S
String has valid date and time	Validation	S	S	Y	S	S
Regular expression validation	Validation	S	S	Y	S	S
NULLIF	SQL Specific	Y	Y	Y	Y	?Y
NOT	Boolean	Y	Y	Y	Y	?Y
CharIndex/INSTR	String	N	N	N	N	N
COALACE	SQL Specific	Y	Y	Y	Y	?Y
SIGN	Number	Y	Y	Y	Y	?Y
CHAR	String	N	N	N	N	N

Index