SeeBeyond ICAN Suite

Oracle eWay Intelligent Adapter User's Guide

Release 5.0.1

Version 20031113135255.

Contents

Chapter 1

Introducing the Oracle eWay 6

Overview 6

Supported Operating Systems 6

System Requirements 7

External System Requirements 7

Chapter 2

Installing the Oracle eWay 8

Before Installing the eWay 8

Installing the Oracle eWay 8

After Installation 9

Chapter 3

Properties of the Oracle eWay 10

Working with eWay Property Sheets 10

Setting the Properties in the Outbound eWay 10

ClassName 11

Description 11

InitialPoolSize 12

LoginTimeOut 12

MaxIdleTime 12

MaxPoolSize 12

MaxStatements 12

MinPoolSize 13

NetworkProtocol 13

PropertyCycle 13

RoleName 13

Setting the Properties in the Inbound eWay 14

Pollmilliseconds 14

PreparedStatement 14

Setting the Properties in the Outbound eWay Environment 15

DatabaseName 15

DataSourceName 15
Delimiter 16
Description 16
DriverProperties 16
Password 16
PortNumber 16
ServerName 17
User 17
Setting the Properties in the Inbound eWay Environment 17
DatabaseName 18
Password 18
PortNumber 18
ServerName 18
User 18
Setting the Properties in the Outbound eWay with XA Support 19
ClassName 19
Description 19
InitialPoolSize 20
LoginTimeOut 20
MaxIdleTime 20
MaxPoolSize 20
MaxStatements 20
MinPoolSize 21
NetworkProtocol 21
PropertyCycle 21
RoleName 21
Setting the Properties in the eWay XA Environment 22
DatabaseName 22
DataSourceName 22
Delimiter 23
Description 23
DriverProperties 23
Password 23
PortNumber 23
ServerName 24
User 24

Chapter 4

Using the Oracle eWay Database Wizard 25

Using the Database OTD Wizard 25

Chapter 5

Importing the Project 36

eInsight Engine and eGate Components 36

Using the Sample Project in eInsight 36
The Business Process 37
Working with the BPEL Operations 38

SelectAll 39
SelectMultiple 40
SelectOne 42
Insert 43
Update 45
Delete 46

Using the Sample Project in eGate 48
Working with the Sample Project in eGate 48
Creating a Connectivity Map 50
Creating the Object Type Definition 51
The Database OTD Wizard 53
Creating the Collaboration 54
Creating Data Connections on the Connectivity Map 56
Configuring the eWays 56
Creating a Rules within the Collaboration 57
Creating a Deployment Profile 63
Deploying the DatabaseSelect Project 64
Running the Sample 65

Supported Data Types 65

Converting Data Types in the Oracle eWay 67

Using OTDs with Tables, Views, and Stored Procedures 68
Data Types 68
The Table 69
The Query Operation 69
The Insert Operation 70
The Update Operation 71
The Delete Operation 71
The Stored Procedure 72
Executing Stored Procedures 72

Creating or Editing an OTD from an Exhisting OTD 73
Editing and Exhisting OTD 73
Creating an OTD from an Exhisting OTD 74

Using XA 74

Alerting and Logging 74

Appendix A

Using CLOB in the Oracle eWay 75

Index 81

Chapter 1

Introducing the Oracle eWay

This document describes how to install and configure the eWay Intelligent Adapter for Oracle.

This Chapter Includes:

- **Overview** on page 6
- **Supported Operating Systems** on page 6
- **System Requirements** on page 7
- **External System Requirements** on page 7

1.1 Overview

The Oracle eWay enables eGate Integrator Projects to exchange data with external Oracle databases. This user's guide describes how to install and configure the Oracle eWay.

1.2 Supported Operating Systems

The Oracle eWay is available on the following operating systems:

- Windows Server 2003, Windows XP SP1a, and Windows 2000 SP3
- Solaris 8 and 9
- HP Tru64 5.1A
- HP-UX 11.0 and 11i

Note: *SeeBeyond only supports HPUX running on 9000/8xx machines. 9000/800 is 64 bits, but can also run in 32 bits mode. To determine if the system is 32 or 64 bits, type:* ***getconf KERNEL_BITS****. This returns either 32 or 64.*

- IBM AIX 5.1 and 5.2
- Red Hat Enterprise Linux AS 2.1
- Red Hat Linux 8 (Intel Version)

Although the Oracle Universal Database eWay, the Repository, and Logical Hosts run on the platforms listed above, the Enterprise Designer requires the Windows operating system. Enterprise Manager can run on any platform that supports Internet Explorer 6.0.

1.3 System Requirements

The system requirements for the Oracle eWay are the same as for eGate Integrator. For information, refer to the *eGate Integrator Installation Guide*. It is also helpful to review the **Readme.txt** for any additional requirements prior to installation. The **Readme.txt** is located on the installation CD-ROM.

The Oracle eWay installation installs the type 4 Oracle JDBC 9.0.1.3.0 driver required to connect to the external Oracle database.

Note: *To enable Web Services, you must install and configure the SeeBeyond ICAN Suite eInsight Business Process Manager.*

1.4 External System Requirements

The Oracle eWay supports the following software for external systems:

- Oracle version 8i with patch 8.1.7
- Oracle version 9i with patch 9.0.1.3
- Oracle version 9i release 9.2.0

Chapter 2

Installing the Oracle eWay

This chapter describes how to install the Oracle eWay.

This Chapter Includes:

- **Before Installing the eWay** on page 8
- **Installing the Oracle eWay** on page 8
- **After Installation** on page 9

2.1 Before Installing the eWay

Open and review the **Readme.txt** for the Oracle eWay for any additional information or requirements, prior to installation. The **Readme.txt** is located on the installation CD-ROM.

2.2 Installing the Oracle eWay

During the eGate Integrator installation process, the Enterprise Manager, a web-based application, is used to select and upload eWays (eWay.sar files) from the eGate installation CD-ROM to the Repository.

The installation process includes installing the following components:

- Installing the Repository
- Uploading products to the Repository
- Downloading components (such as Enterprise Designer and Logical Host)
- Viewing product information home pages

Follow the instructions for installing the eGate Integrator in the *ICAN Installation Guide*, and include the following steps:

- On the Enterprise Manager, select the **OracleeWay.sar** (to install the Oracle eWay) file to upload.
- On the Enterprise Manager, select the **FileeWay.sar** (to install the File eWay, used in the sample Project) file to upload.

- On the Enterprise Manager, install the **OracleeWayDocs.sar** (to install the documentation, the Javadoc, and the sample) file to upload.
- On the Enterprise Manager under the Documentation tab, click on the document link, the Javadoc link, or the sample file link. For the sample project, it is recommended that you extract the file to another file location prior to importing it using the Enterprise Explorer's Import Project tool.

For additional information on how to use eGate, please see the *eGate Tutorial*.

Continue installing the eGate Enterprise Designer as instructed.

2.3 After Installation

Once the eWay is installed and configured it must then be incorporated into a Project before it can perform its intended functions. See the *eGate Integrator User's Guide* for more information on incorporating the eWay into an eGate Project.

Chapter 3

Properties of the Oracle eWay

This chapter describes how to set the properties of the Oracle eWay.

This Chapter Includes:

- **Setting the Properties in the Outbound eWay** on page 10
- **Setting the Properties in the Inbound eWay** on page 14
- **Setting the Properties in the Outbound eWay Environment** on page 15
- **Setting the Properties in the Inbound eWay Environment** on page 17
- **Setting the Properties in the Outbound eWay with XA Support** on page 19
- **Setting the Properties in the eWay XA Environment** on page 22

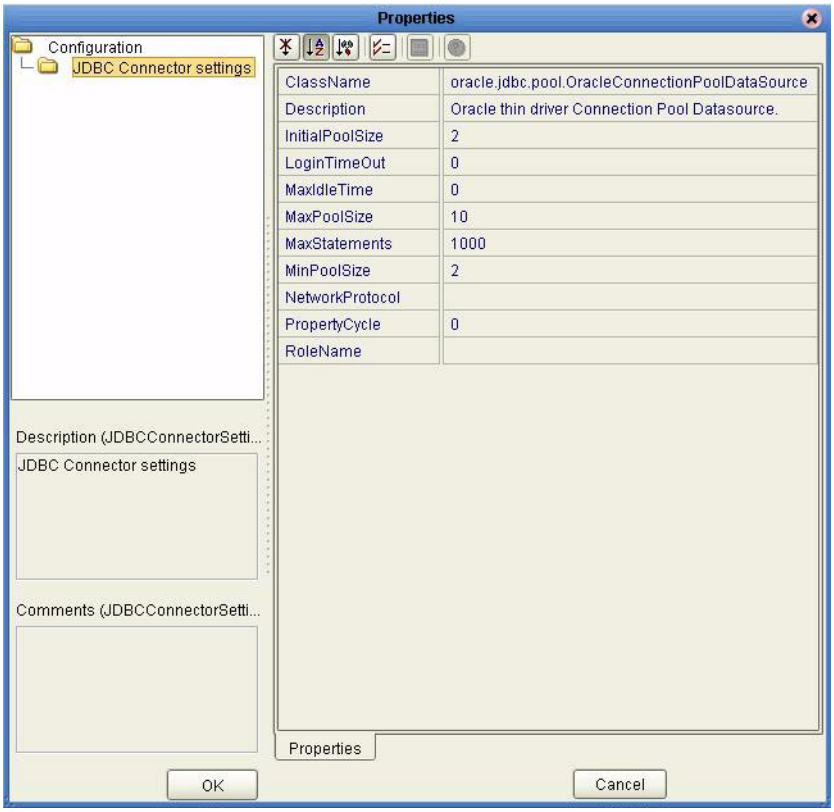
3.1 Working with eWay Property Sheets

On the Properties sheet window and using the descriptions below, enter the information necessary for the eWay to establish a connection to the external application.

3.1.1. Setting the Properties in the Outbound eWay

The DataSource settings define the properties used to interact with the external database.

Figure 1 The Outbound eWay Properties



The DataSource settings define the properties used to interact with the external database.

ClassName

Description

Specifies the Java class in the JDBC driver that is used to implement the ConnectionPoolDataSource interface.

Required Values

A valid class name.

The default is **oracle.jdbc.pool.OracleConnectionPoolDataSource**.

Description

Description

Enter a description for the database.

Required Value

A valid string.

InitialPoolSize

Description

Enter a number for the physical connections the pool should contain when it is created.

Required Value

A valid numeric value.

LoginTimeOut

Description

The number of seconds driver will wait before attempting to log in to the database before timing out.

Required Value

A valid numeric value.

MaxIdleTime

Description

The maximum number of seconds that a physical connection may remain unused before it is closed. 0 (zero) indicates that there is no limit.

Required Value

A valid numeric value.

MaxPoolSize

Description

The maximum number of physical connections the pool should keep available at all times. 0 (zero) indicates that there is no maximum.

Required Value

A valid numeric value.

MaxStatements

Description

The maximum total number of statements that the pool should keep open. 0 (zero) indicates that the caching of statements is disabled.

Required Value

A valid numeric value.

MinPoolSize

The minimum number of physical connections the pool should keep available at all times. 0 (zero) indicates that there should be no physical connections in the pool and the new connections should be created as needed.

Required Value

A valid numeric value.

NetworkProtocol

Description

The network protocol used to communicate with the server.

Required Values

Any valid string.

PropertyCycle

Description

The interval, in seconds, that the pool should wait before enforcing the current policy defined by the values of the other connection pool properties in this deployment descriptor.

Required Values

A valid numeric value.

RoleName

Description

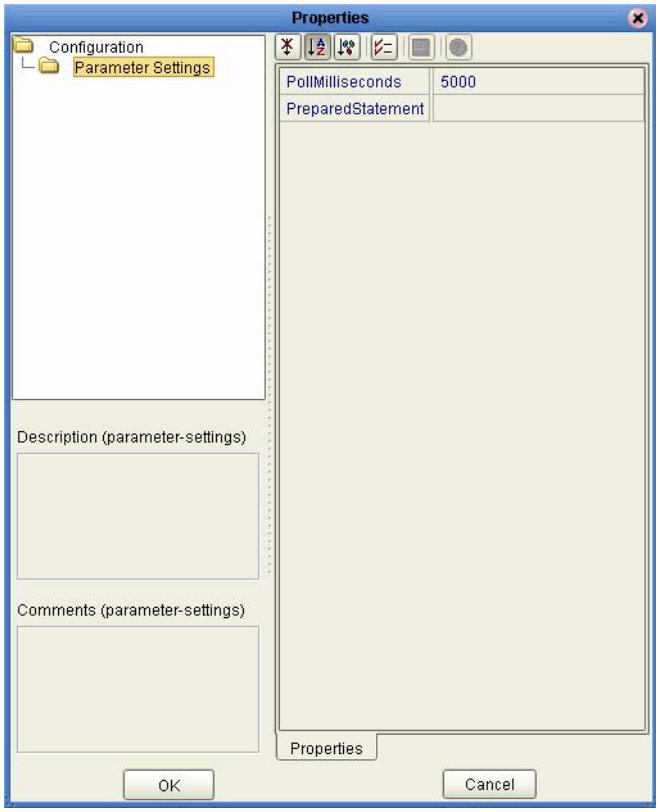
An initial SQL role name.

Required Values

Any valid string.

3.1.2. Setting the Properties in the Inbound eWay

Figure 2 Properties of the Inbound eWay



Pollmilliseconds

Description

Polling interval in milliseconds.

Required Value

A valid numeric value. The default is 5000.

PreparedStatement

Description

Prepared Statement used for polling against the database.

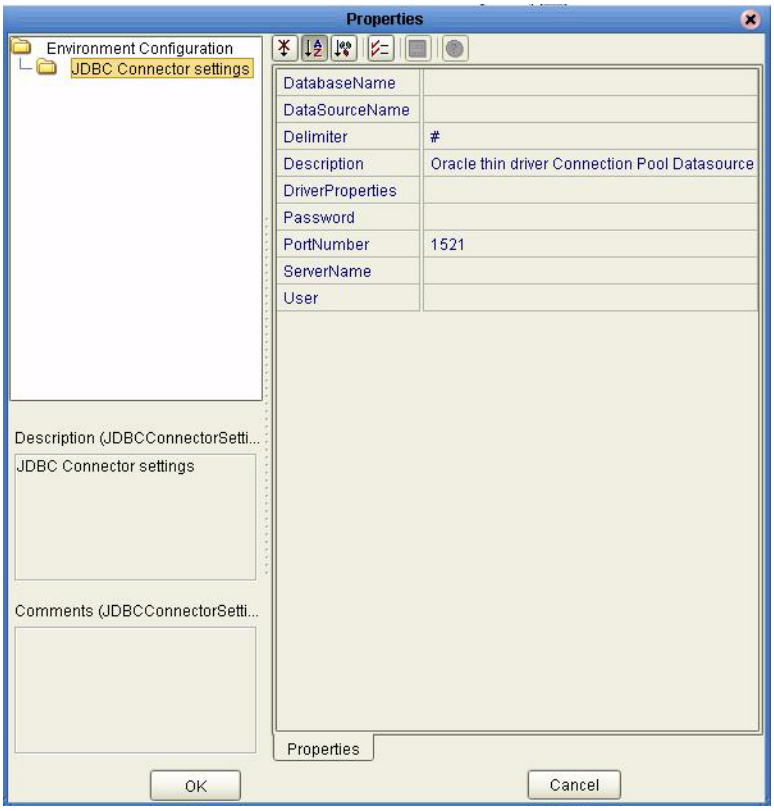
Required Value

The Prepared Statement must be the same Prepared Statement you created using the Database OTD Wizard. Only SELECT Statement is allowed. Additionally, no place holders should be specified. There should not be any "?" in the Prepared Query.

3.1.3. Setting the Properties in the Outbound eWay Environment

Before deploying your eWay, you will need to set the properties of the eWay environment using the following descriptions.

Figure 3 eWay Environment Configuration



DatabaseName

Description

Specifies the name of the database instance.

Required Values

Any valid string.

DataSourceName

Description

Provide the name of the ConnectionPoolDataSource object that the DataSource object delegates behind the scenes when connection pooling or distributed transaction management is being done.

Required Value

Optional. In most cases, leave this box empty.

Delimiter

Description

This is the delimiter character to be used in the DriverProperties prompt.

Required Value

The default is #

Description

Description

Enter a description for the database.

Required Value

A valid string.

DriverProperties

Description

If you choose to not use the JDBC driver that is shipped with this eWay, you will need to add the drivers properties to the eWay. Often times the DataSource implementation will need to execute additional properties to assure a connection. The additional methods will need to be identified in the Driver Properties.

Required Value

Any valid delimiter.

Valid delimiters are: "<method-name-1>#<param-1>#<param-2>##.....<param-n>##<method-name-2>#<param-1>#<param-2>#.....<param-n>##.....##".

For example: to execute the method setURL, give the method a String for the URL "setURL#<url>##".

Password

Description

Specifies the password used to access the database.

Required Values

Any valid string.

PortNumber

Description

Specifies the I/O port number on which the server is listening for connection requests.

Required Values

A valid port number. The default is 1521.

ServerName

Description

Specifies the host name of the external database server.

Required Values

Any valid string.

User

Description

Specifies the user name the eWay uses to connect to the database.

Required Values

Any valid string.

3.1.4. Setting the Properties in the Inbound eWay Environment

Figure 4 Inbound eWay Environment

DatabaseName

Description

Specifies the name of the database instance.

Required Values

Any valid string.

Password

Description

Specifies the password used to access the database.

Required Values

Any valid string.

PortNumber

Description

Specifies the I/O port number on which the server is listening for connection requests.

Required Values

A valid port number. The default is 1433.

ServerName

Description

Specifies the host name of the external database server.

Required Values

Any valid string.

User

Description

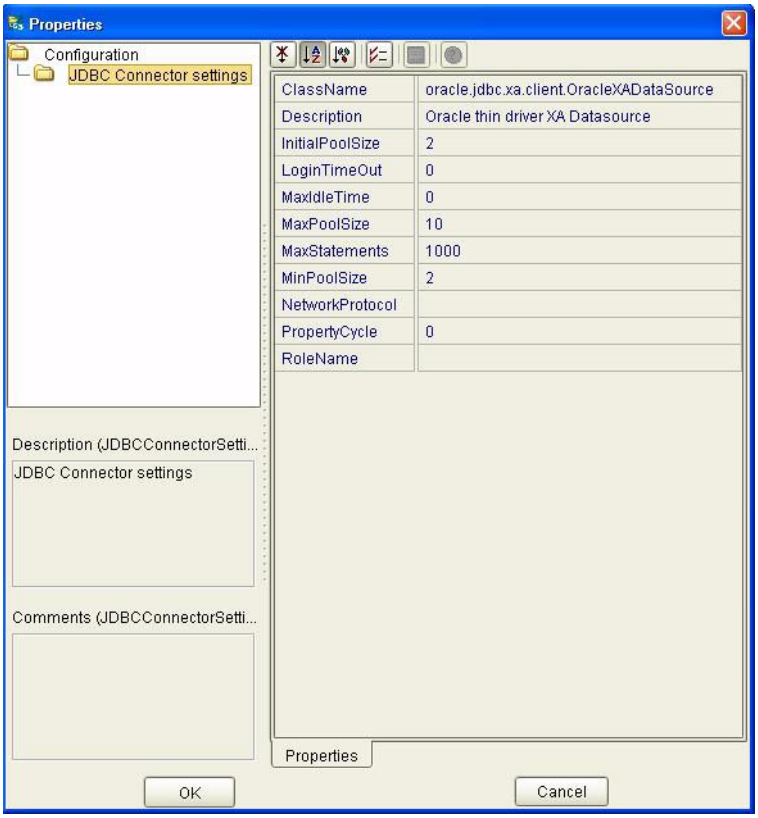
Specifies the user name the eWay uses to connect to the database.

Required Values

Any valid string.

3.1.5. Setting the Properties in the Outbound eWay with XA Support

Figure 5 Outbound Oracle eWay with XA Support



ClassName

Description

Specifies the Java class in the JDBC driver that is used to implement the ConnectionPoolDataSource interface.

Required Values

A valid class name.

The default is **oracle.jdbc.xa.client.OracleXADataSource**.

Description

Description

Enter a description for the database.

Required Value

A valid string.

InitialPoolSize

Description

Enter a number for the physical connections the pool should contain when it is created.

Required Value

A valid numeric value. The default is 2.

LoginTimeOut

Description

The number of seconds driver will wait before attempting to log in to the database before timing out.

Required Value

A valid numeric value.

MaxIdleTime

Description

The maximum number of seconds that a physical connection may remain unused before it is closed. 0 (zero) indicates that there is no limit.

Required Value

A valid numeric value.

MaxPoolSize

Description

The maximum number of physical connections the pool should keep available at all times. 0 (zero) indicates that there is no maximum.

Required Value

A valid numeric value. The default is 10.

MaxStatements

Description

The maximum total number of statements that the pool should keep open. 0 (zero) indicates that the caching of statements is disabled.

Required Value

A valid numeric value. The default is 1000.

MinPoolSize

The minimum number of physical connections the pool should keep available at all times. 0 (zero) indicates that there should be no physical connections in the pool and the new connections should be created as needed.

Required Value

A valid numeric value.

NetworkProtocol

Description

The network protocol used to communicate with the server.

Required Values

Any valid string. The default is 2.

PropertyCycle

Description

The interval, in seconds, that the pool should wait before enforcing the current policy defined by the values of the other connection pool properties in this deployment descriptor.

Required Values

A valid numeric value.

RoleName

Description

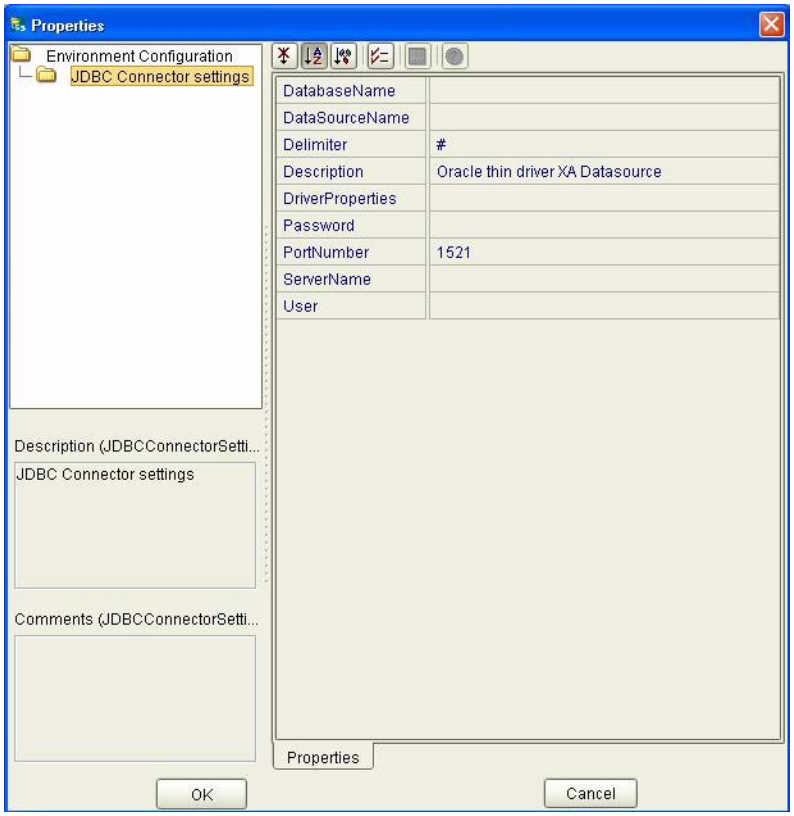
An initial SQL role name.

Required Values

Any valid string.

3.1.6. Setting the Properties in the eWay XA Environment

Figure 6 Environment Properties of the eWay with XA



DatabaseName

Description

Specifies the name of the database instance.

Required Values

Any valid string.

DataSourceName

Description

Provide the name of the ConnectionPoolDataSource object that the DataSource object delegates behind the scenes when connection pooling or distributed transaction management is being done.

Required Value

Optional. In most cases, leave this box empty.

Delimiter

Description

This is the delimiter character to be used in the DriverProperties prompt.

Required Value

The default is #

Description

Description

Enter a description for the database.

Required Value

A valid string.

DriverProperties

Description

If you choose to not to use the JDBC driver that is shipped with this eWay, you will need to add the drivers properties to the eWay. Often times the DataSource implementation will need to execute additional methods to assure a connection. The additional methods will need to be identified in the Driver Properties.

Required Value

Any valid delimiter.

Valid delimiters are: "<method-name-1>#<param-1>#<param-2>##.....<param-n>##<method-name-2>#<param-1>#<param-2>#.....<param-n>##.....##".

For example: to execute the method setURL, give the method a String for the URL "setURL#<url>##".

If you are using Spy Log. Optional: "setURL#jdbc:Seebeyond:db2://<server>:50000;DatabaseName=<database>##setSpyAttributes#log=(file)c:/temp/spy.log;logTName=yes##".

Password

Description

Specifies the password used to access the database.

Required Values

Any valid string.

PortNumber

Description

Specifies the I/O port number on which the server is listening for connection requests.

Required Values

A valid port number. The default is 1521.

ServerName

Description

Specifies the host name of the external database server.

Required Values

Any valid string.

User

Description

Specifies the user name the eWay uses to connect to the database.

Required Values

Any valid string.

Chapter 4

Using the Oracle eWay Database Wizard

This chapter describes how to use the Oracle eWay Database Wizard to build OTD's.

This Chapter Includes:

- **Select Wizard Type** on page 25
- **Connect to Database** on page 26
- **Select Database Objects** on page 27
- **Select Table/Views** on page 27
- **Select Procedures** on page 31
- **Add Prepared Statements** on page 32
- **Specify the OTD Name** on page 34

4.1 Using the Database OTD Wizard

The Database OTD Wizard generates OTDs by connecting to external data sources and creating corresponding Object Type Definitions. The OTD Wizard can create OTDs based on any combination of Tables and Stored Procedures or Prepared SQL Statements.

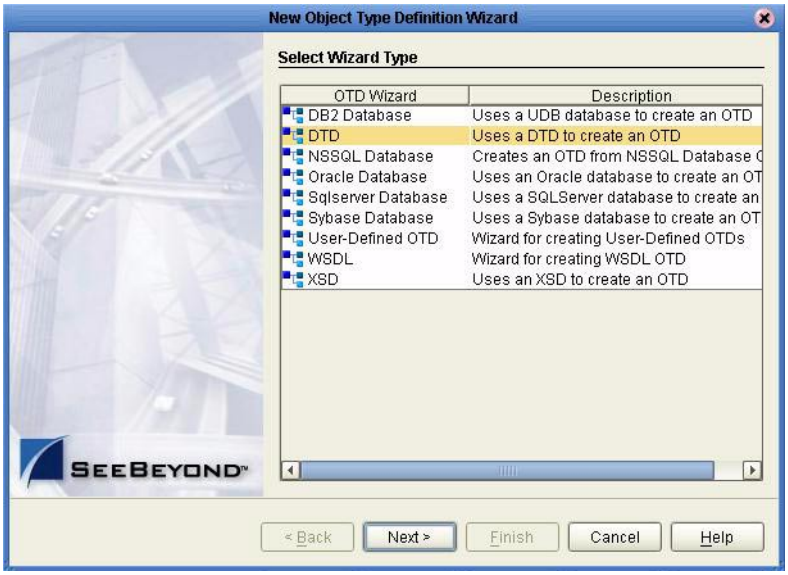
Field nodes are added to the OTD based on the Tables in the external data source. Java method and parameter nodes are added to provide the appropriate JDBC functionality. For more information about the Java methods, refer to your JDBC developer's reference.

Note: *Database OTDs are not messagable. For more information on messagable OTDs, see the eGate Integrator User's Guide.*

Select Wizard Type

1. On the Enterprise Explorer, right click on the project and select **Create an Object Type Definition** from the shortcut menu.
2. From the OTD Wizard Selection window, select the **Oracle Database** and click **Next**. See **Figure 7**.

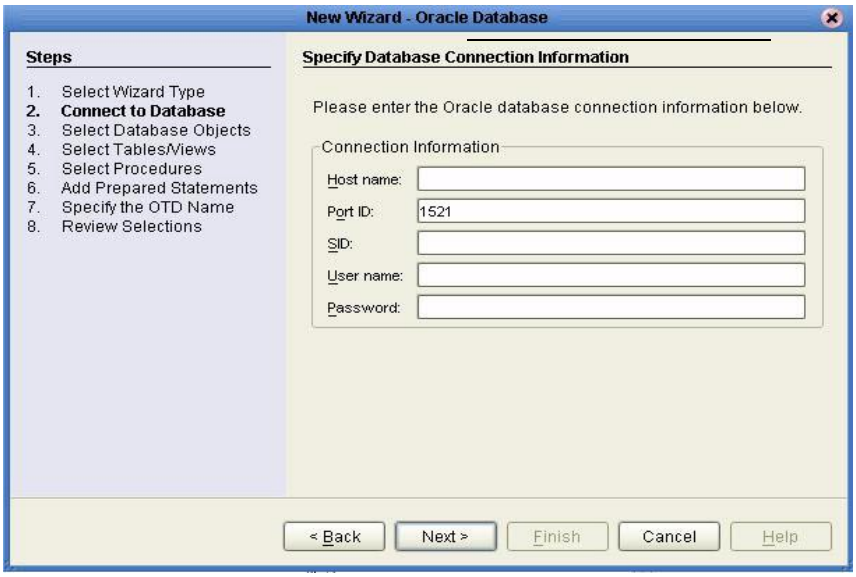
Figure 7 OTD Wizard Selection



Connect to Database

3 Specify the connection information for your database including your **UserName** and **Password** and click **Next**. See **Figure 8**.

Figure 8 Database Connection Information

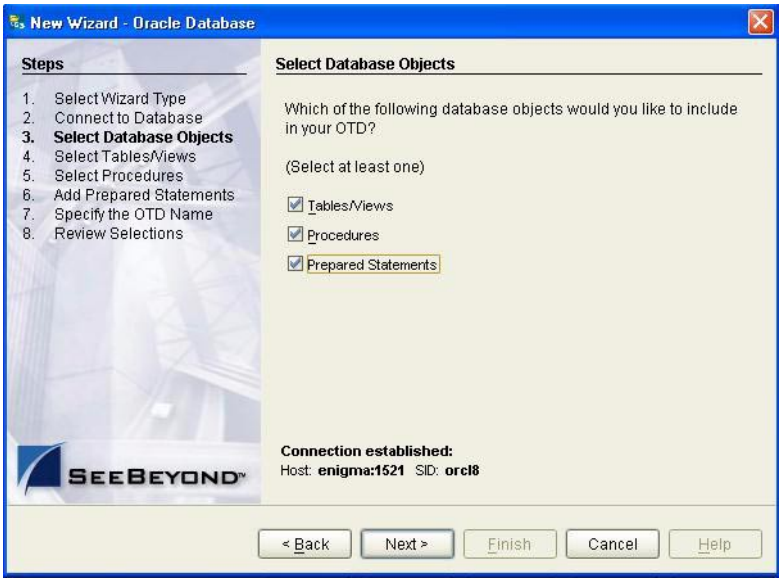


Select Database Objects

1 In the **Select Database Objects** window, see **Figure 9**, select **Tables/Views** for this sample. When selecting Database Objects, you can select any combination of **Tables**, **Views**, **Procedures**, or **Prepared Statements** you would like to include in the .otd file. Click **Next** to continue.

Note: *Views are read-only and are for informational purposes only.*

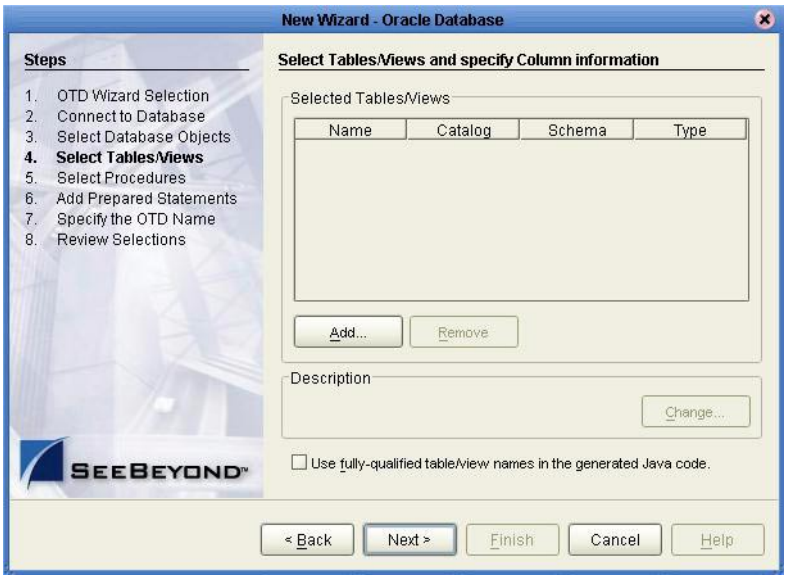
Figure 9 Select Database Objects



Select Table/Views

1 In the **Select Tables/Views** window, click **Add**. See **Figure 10**.

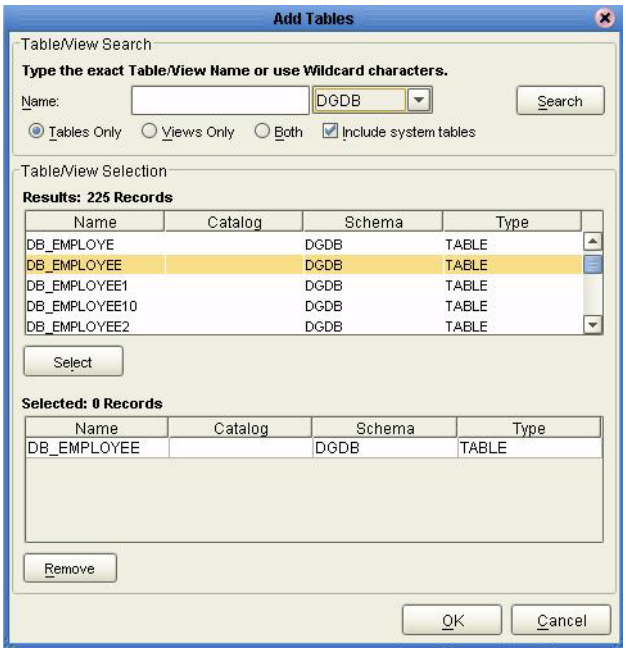
Figure 10 Select Tables/Views



2 In the **Add Tables** window, select if your selection criteria will include table data, view only data, both, and/or system tables.

3 From the **Table/View Name** drop down list, select the location of your database table and click **Search**. See **Figure 11**. You can search for **Table/View Names** by entering a table name.

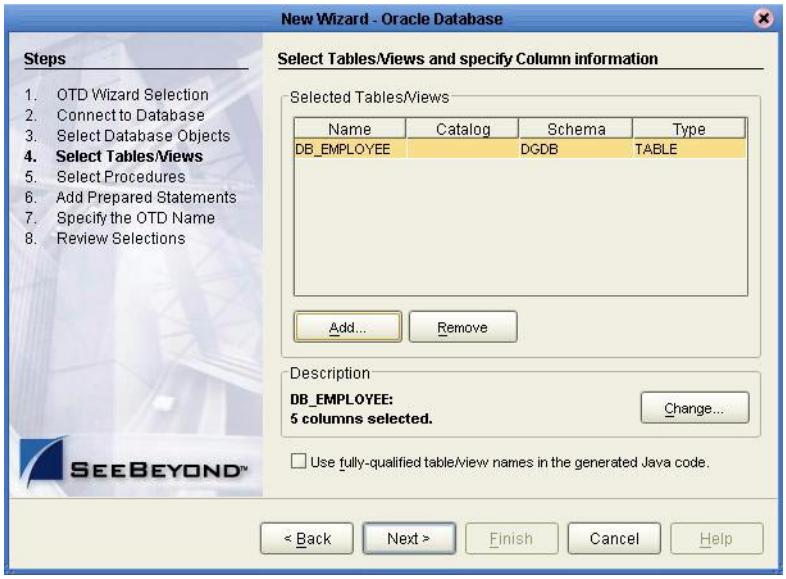
Figure 11 Database Wizard - All Schemes



4 Select the table of choice and click **OK**.

The table selected is added to the **Selected Tables/Views** window. See **Figure 12**.

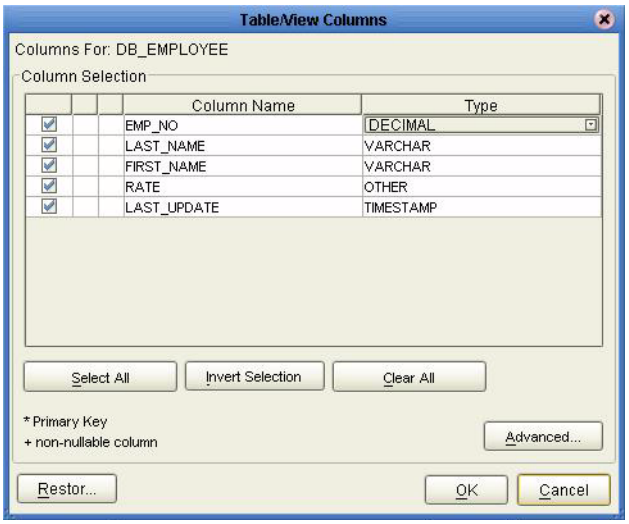
Figure 12 Selected Tables/Views window with a table selected



5 In the **Selected Tables/Views** window, review the table(s) you have selected. To make changes to the selected Table or View, click Change. If you do not wish to make any additional changes, click **Next** to continue.

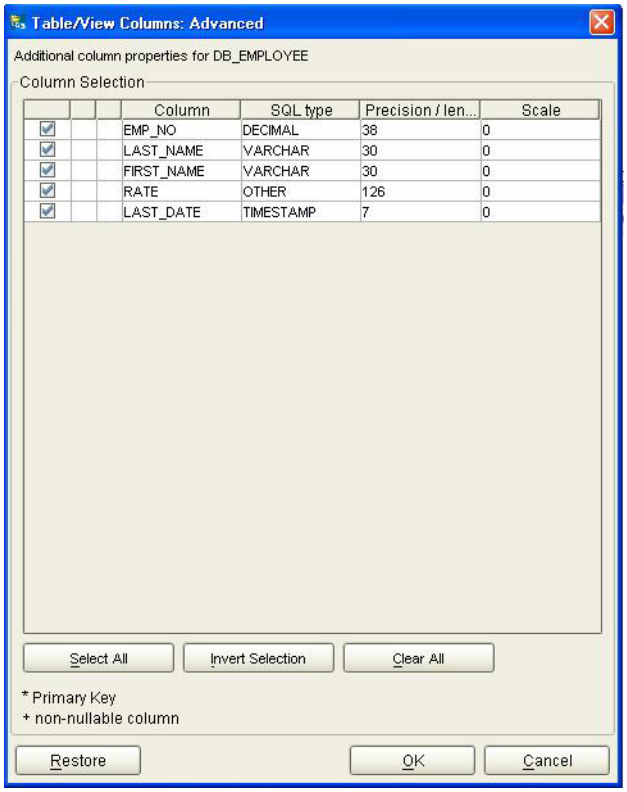
6 In the **Table/View Columns** window, you can select or deselect your table columns. You can also change the data type for each table by highlighting the data type and selecting a different one from the drop down list. If you would like to change any of the tables columns, click **Change**. See **Figure 13**.

Figure 13 Table/View Columns



7 Click **Advanced** to change the data type, percision/length, or scale. Once you have finished your table choices, click **OK**. See **Figure 14**. In general, you do not need to change the percision or scale.

Figure 14 Table/View Columns — Advanced

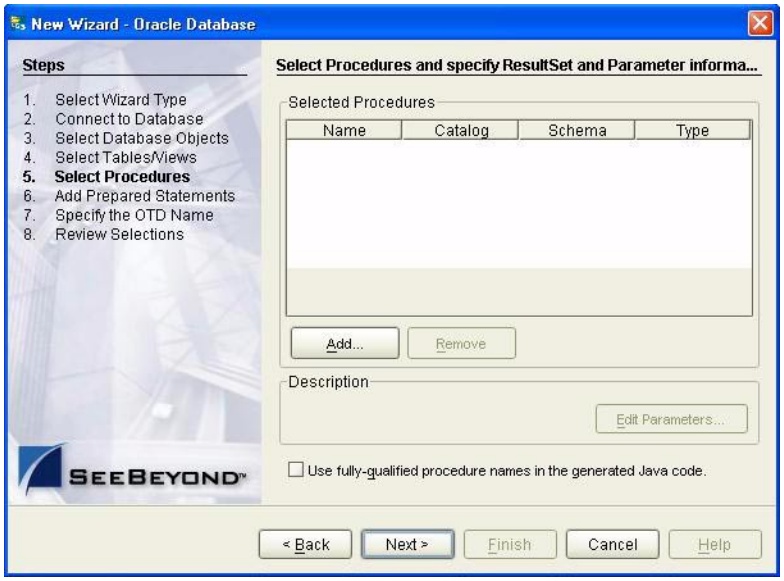


8 When using Oracle packages, select **Use fully qualified table/view names in the generated Java code**. See Figure 12.

Select Procedures

1 On the **Select Procedures and specify Resultset and Parameter Information** window, click **Add.**

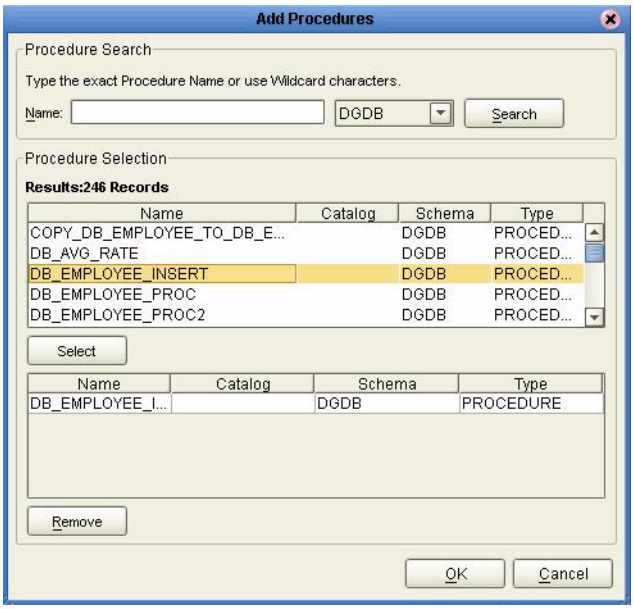
Figure 15 Select Procedures and specify Resultset and Parameter Information



2 On the **Select Procedures** window, enter the name of a Procedure or select a table from the drop down list. Click **Search**. Wildcard characters can also be used.

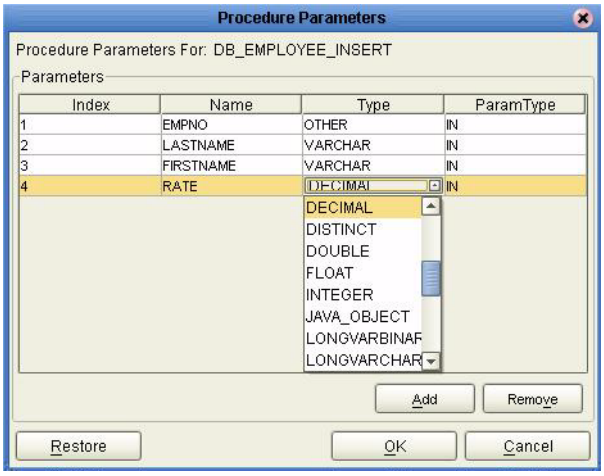
3 In the resulting **Procedure Selection** list box, select a Procedure. Click **OK**.

Figure 16 Add Procedures



4 On the **Select Procedures and specify Resultset and Parameter Information** window click **Edit Parameters** to make any changes to the selected Procedure. See **Figure 17**.

Figure 17 Procedure Parameters



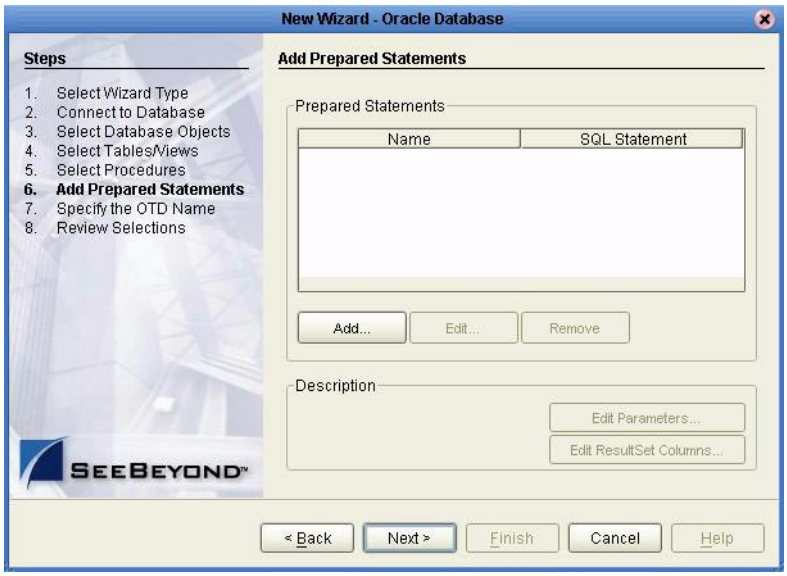
5 To restore the data type, click **Restore**. When finished, click **OK**.

6 On the **Select Procedures and specify Resultset and Parameter Information** window click **Next** to continue.

Add Prepared Statements

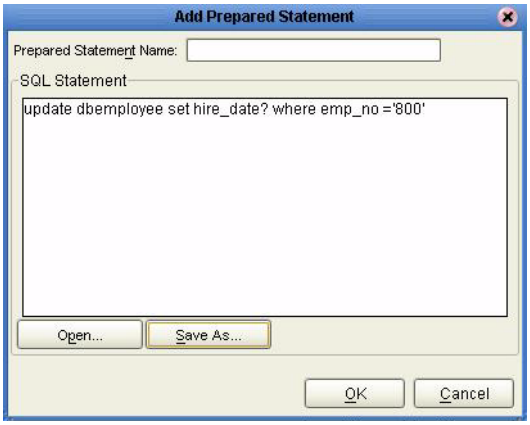
1 On the **Add Prepared Statements** window, click **Add**.

Figure 18 Prepared Statement



2 Enter the name of a Prepared Statement or create a SQL statement by clicking in the SQL Statement window. When finished creating the statement, click **Save As** giving the statement a name. This name will appear as a node in the OTD. Click **OK**. See **Figure 19**.

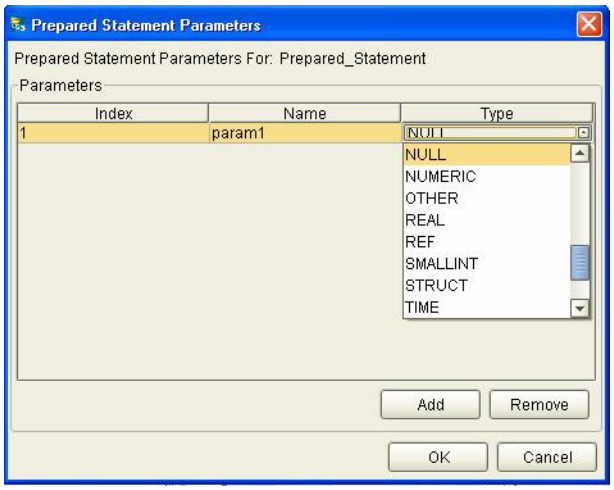
Figure 19 Prepared SQL Statement



3 On the **Add Prepared Statement** window, the name you assigned to the Prepared Statement appears. To edit the parameters, click **Edit Parameters**. You can change the datatype by clicking in the **Type** field and selecting a different type from the list.

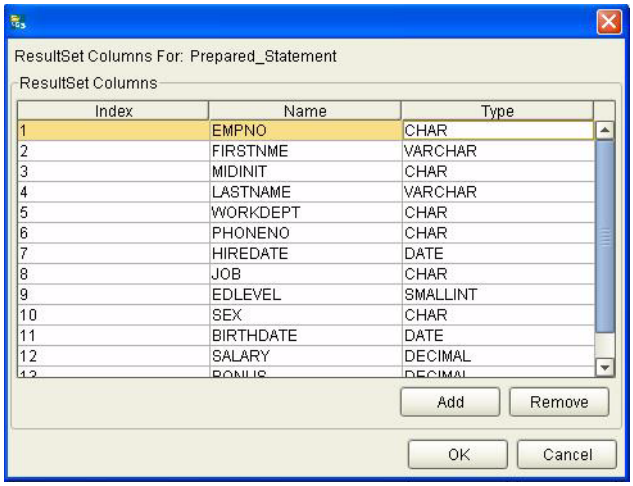
4 Click **Add** if you want to add additional parameters to the Statement or highlight a row and click **Remove** to remove it. Click **OK**. See **Figure 20**.

Figure 20 Edit the Prepared Statement Parameters



1 To edit the Resultset Columns, click **Edit Resultset Columns**. Both the Name and Type are editable but it is recommend you do not changed the Name. Doing so will cause a loss of integrity between the Resultset and the Database. Click **OK**. See **Figure 21**.

Figure 21 ResultSet Columns

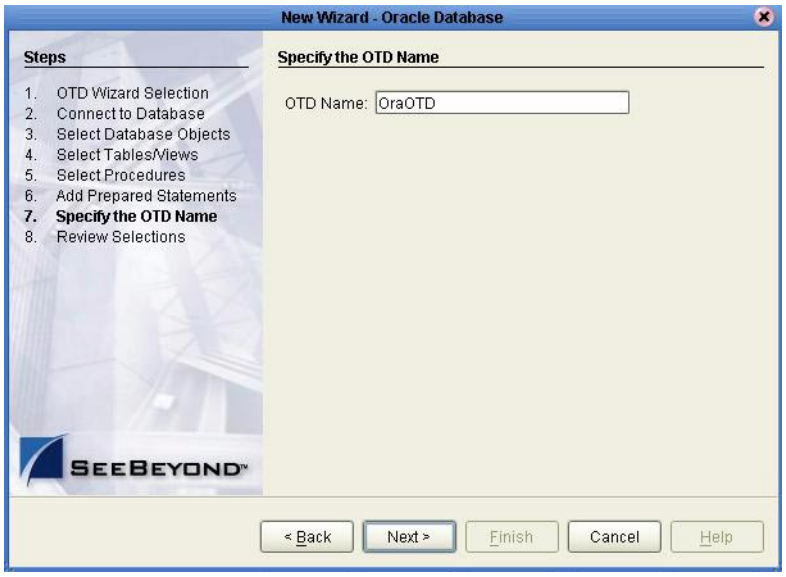


2 On the Add Prepared Statements window, click **OK**.

Specify the OTD Name

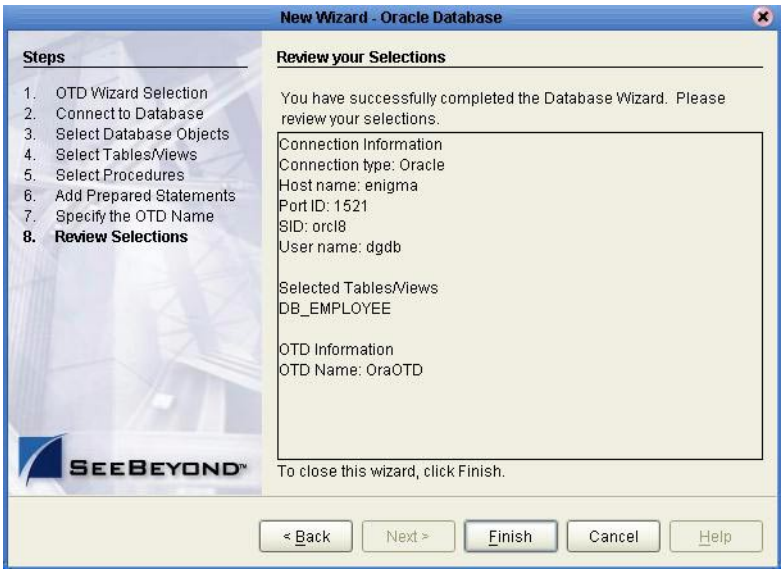
1 Enter a name for the OTD. The OTD contains the selected tables and the package name of the generated classes. See **Figure 22**.

Figure 22 Naming an OTD



2 View the summary of the OTD. If you find you have made a mistake, click **Back** and correct the information. If you are satisfied with the OTD information, click **Finish** to begin generating the OTD. See **Figure 23**.

Figure 23 Database Wizard - Summary



The resulting **OTD** will appear on the Enterprise Designer's canvas.

Chapter 5

Importing the Project

This chapter discusses an overview of the Sample Projects provided with this eWay.

This Chapter Includes:

- **Using the Sample Project in eInsight** on page 36
- **Working with the BPEL Operations** on page 38
- **Working with the Sample Project in eGate** on page 48

5.1 eInsight Engine and eGate Components

You can deploy an eGate component as an Activity in an eInsight Business Process. Once you have associated the desired component with an Activity, the eInsight engine can invoke it using a Web Services interface. Examples of eGate components that can interface with eInsight in this way are:

- Java Messaging Service (JMS)
- Object Type Definitions (OTDs)
- An eWay
- Collaborations

Using the eGate Enterprise Designer and eInsight, you can add an Activity to a Business Process, then associate that Activity with an eGate component, for example, an eWay. When eInsight runs the Business Process, it automatically invokes that component via its Web Services interface.

5.2 Using the Sample Project in eInsight

To begin using the sample eInsight Business Process project, you will need to import the project and view it from within the Enterprise Designer using the Enterprise Designer Project Import utility. Import the **Ora_BPEL_Sample.zip** file contained in the eWay sample folder on the installation CD-ROM.

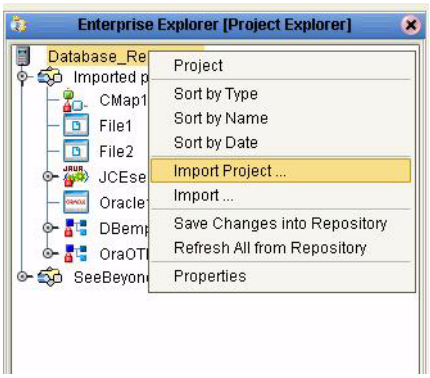
Note: *eInsight is a Business Process modeling tool. If you have not purchased eInsight, contact your sales representative for information on how to do so.*

Before recreating the sample Business Process, review the *eInsight Business Process Manager User's Guide* and the *eGate Tutorial*.

Importing the Sample Project

1 On the Enterprise Explorer highlight the repository and right click. Select **Import Project**. See **Figure 24**.

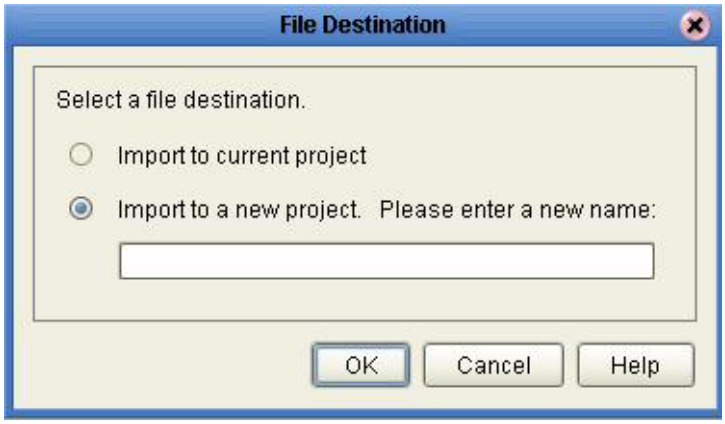
Figure 24 Importing the sample project



1 In the **Select File to Import** window, browse to the location of the sample folder and select the following .zip file **Ora_BPEL_Sample.zip** and click **Open**.

2 On the **File Destination** window, select **Import a new project. Please enter a new name**. Enter a name for the sample project and click OK. See **Figure 25**.

Figure 25 Select the project file destination



3 Click the **Refresh All From Repository** icon located on the **Enterprise Explorer** toolbar.

The Business Process

The data used for this sample project is contained within a table called DBEmployee. The table has the following columns:

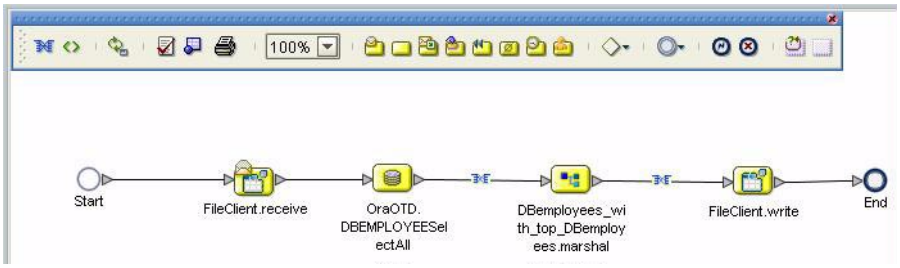
Table 1 Sample project data
Table 2

Column Name	Mapping	Data Type	Data Length
EMP_NO	empno	varchar2	10
LAST_NAME	lastname	varchar2	30
FIRST_NAME	firstname	varchar2	30
SS_NUMBER	ssnumber	varchar2	20
HIRE_DATE	hiredate	varchar2	12

The sample project consists of an input file containing data that is passed into a database collaboration, marshalled, and then written out to an output file

Refer to the *eInsight Business Process Manager User's Guide* for specific information on how to create and use a Business Process.

Figure 26



5.2.1. Working with the BPEL Operations

You can associate an eInsight Business Process Activity with the Oracle eWay, both during the system design phase. To make this association, select the desired **receive** or **write** operation under the eWay in the Enterprise Explorer and drag it onto the eInsight Business Process canvas. For the Oracle eWay, the following operations are available:

- SelectAll
- SelectMultiple
- SelectOne
- Insert
- Update
- Delete

The operation automatically changes to an Activity with an icon identifying the component that is the basis for the Activity.

At run time, the eInsight engine invokes each step in the order that you defined in the Business Process. Using the engine's Web Services interface, the Activity in turn invokes the Oracle eWay. You can open a file specified in the eWay and view its contents before and after the Business Process is executed.

The table below shows the inputs and outputs to each of these eInsight operations:

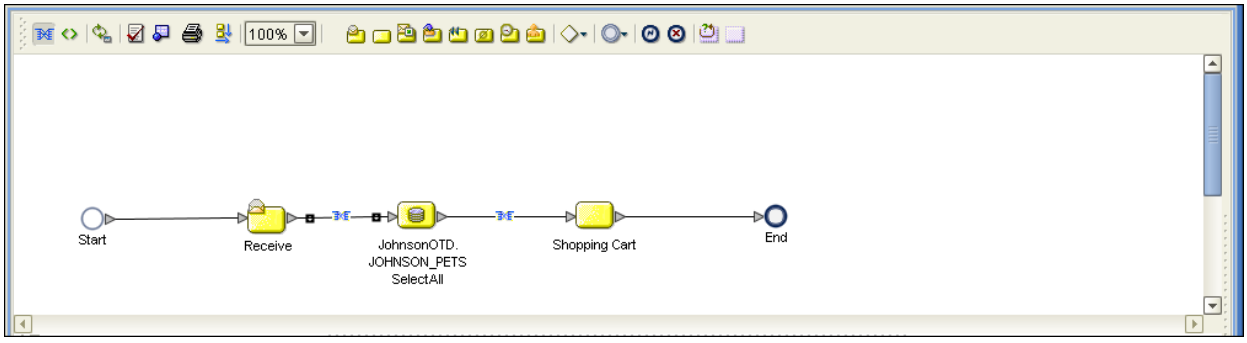
eInsight Operation	Input	Output
SelectAll	where() clause (optional)	Returns all rows that fit the condition of the where() clause
SelectMultiple	number of rows where() clause. Optional	Returns the number of rows specified that fit the condition of the where() clause
SelectOne	where() clause. Optional	Returns the first row that fits the condition of the where() clause
Insert	definition of new item to be inserted	Returns status.
Update	where() clause	Returns status.
Delete	where() clause	Returns status.

SelectAll

The input to a SelectAll operation is an optional where() clause. The where() clause defines to which criteria rows must adhere to be returned. In the SelectAll operation, all items that fit the criteria are returned. If the where() clause is not specified, all rows all returned.

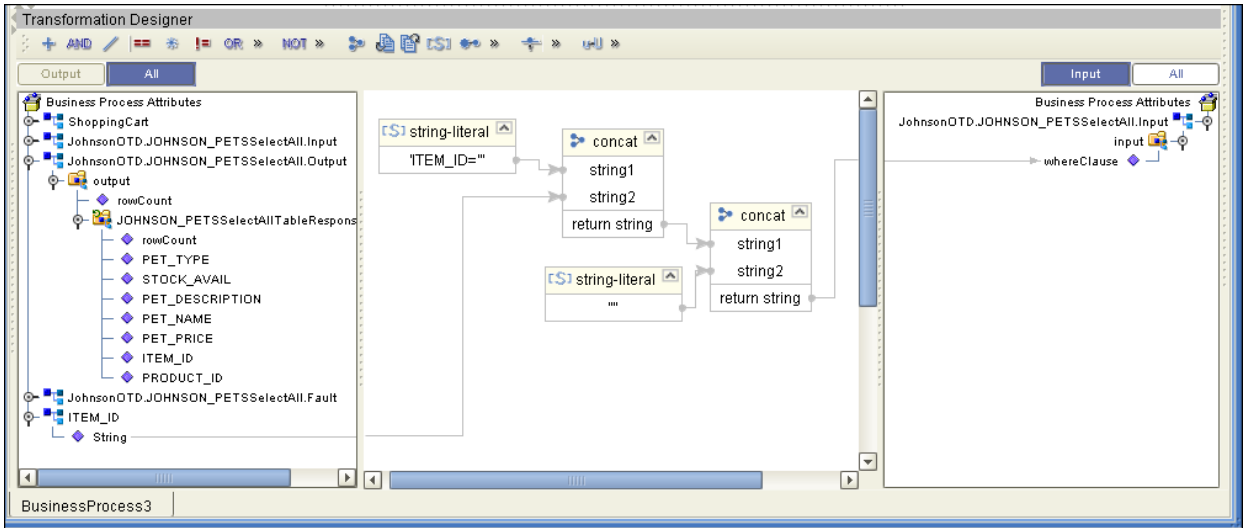
The figure below shows a sample eInsight Business Process using the SelectAll operation. In this process, the SelectAll operation returns all rows where the ITEM_ID matches the selected ITEM_ID to the shopping cart.

Figure 27 SelectAll Sample Business Process



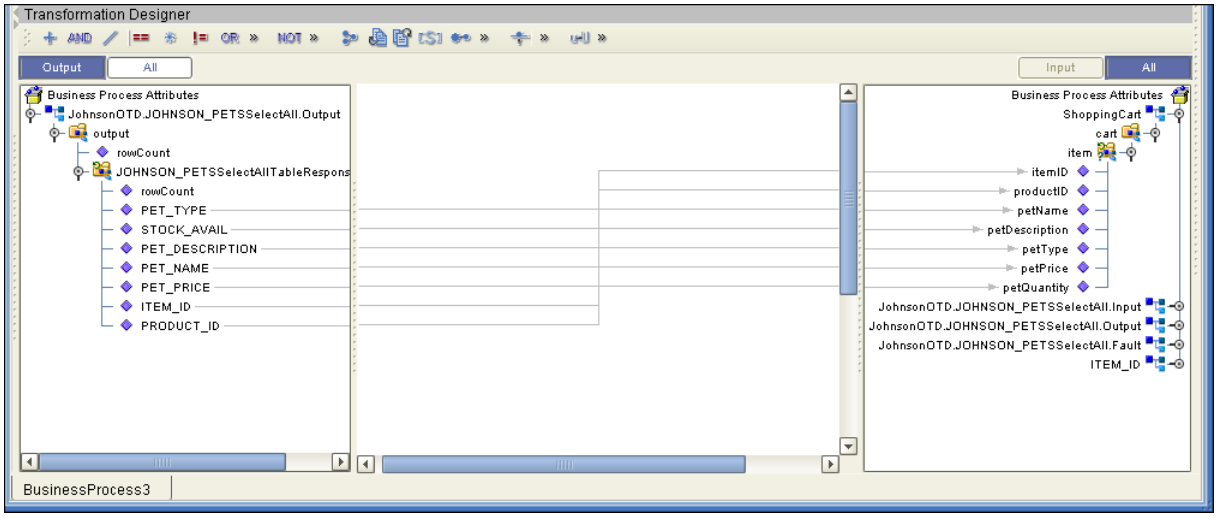
The figure below shows the definition of the where() clause for the SelectAll operation.

Figure 28 SelectAll Input



The figure below shows the definition of the output for the SelectAll operation. For each row selected during the operation, the shopping cart shows the columns of those rows as defined here.

Figure 29 SelectAll Output

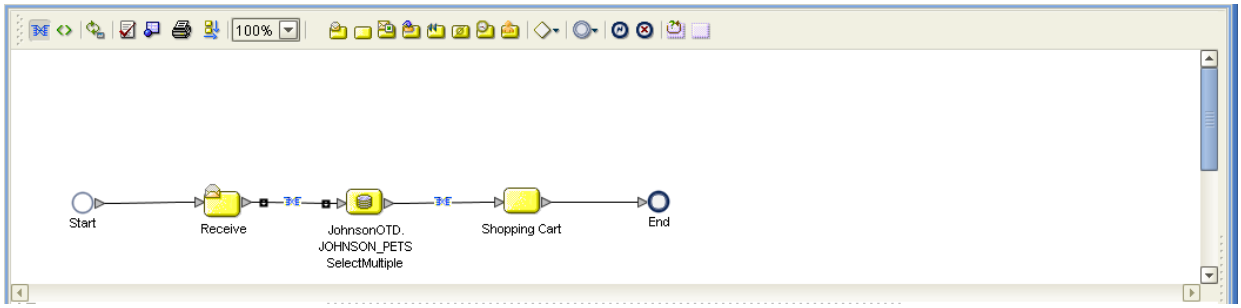


SelectMultiple

The input to a SelectMultiple operation is the number of rows to be selected and a where() clause. The number of rows indicates how many rows the SelectMultiple operation returns. The where() clause defines to which criteria rows must adhere to be returned.

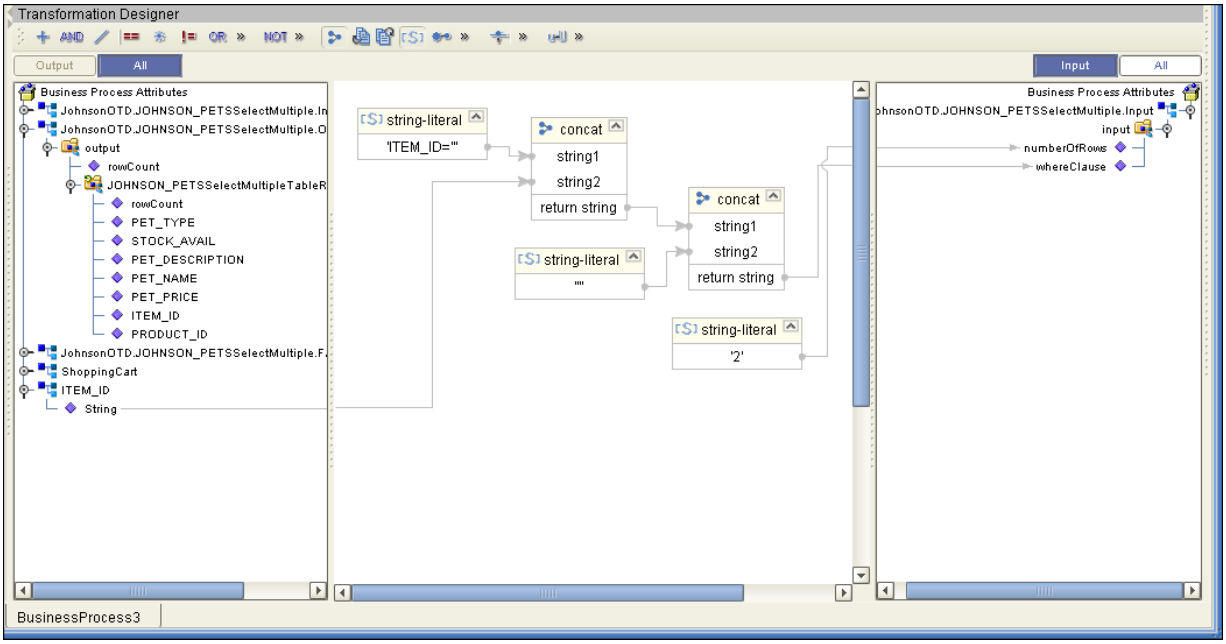
The figure below shows a sample eInsight Business Process using the SelectMultiple operation. In this process, the SelectMultiple operation returns the first two rows where the ITEM_ID matches the selected ITEM_ID to the shopping cart.

Figure 30 SelectMultiple Sample Business Process



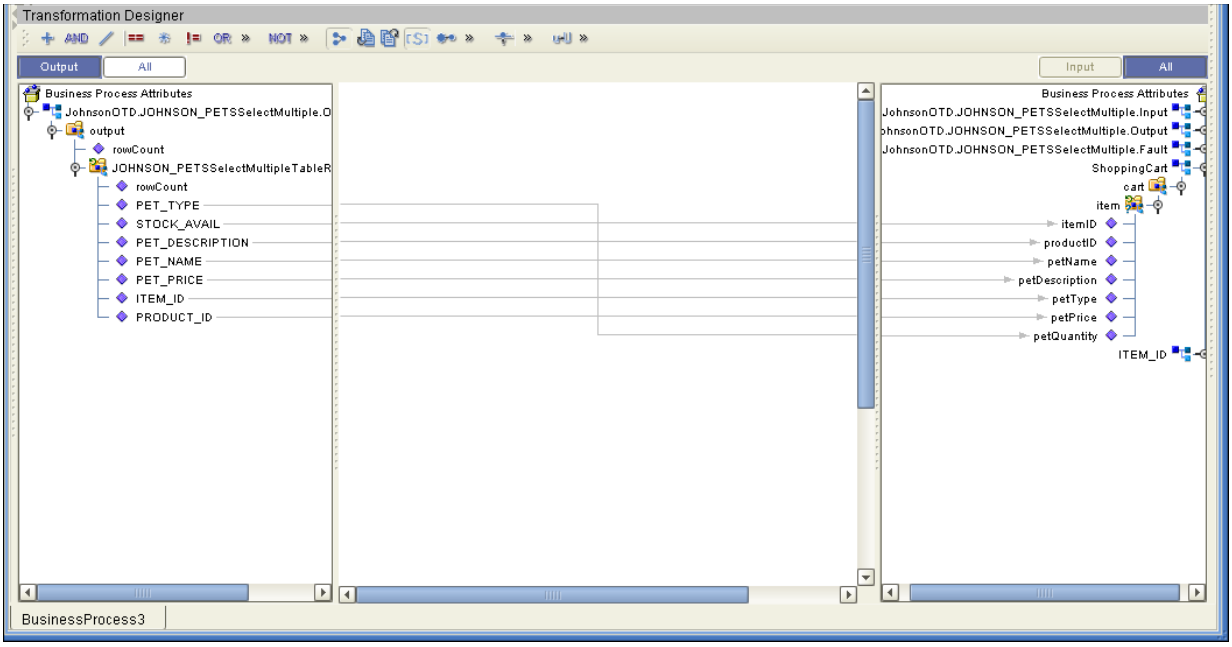
The figure below shows the definition of the number of rows and where() clause into the input for the SelectMultiple operation. You could also use an empty string or Item_ID='123'.

Figure 31 SelectMultiple Input



The figure below shows the definition of the output for the SelectMultiple operation. For each row selected during the operation, the shopping cart shows the columns of those rows as defined here.

Figure 32 SelectMultiple Output

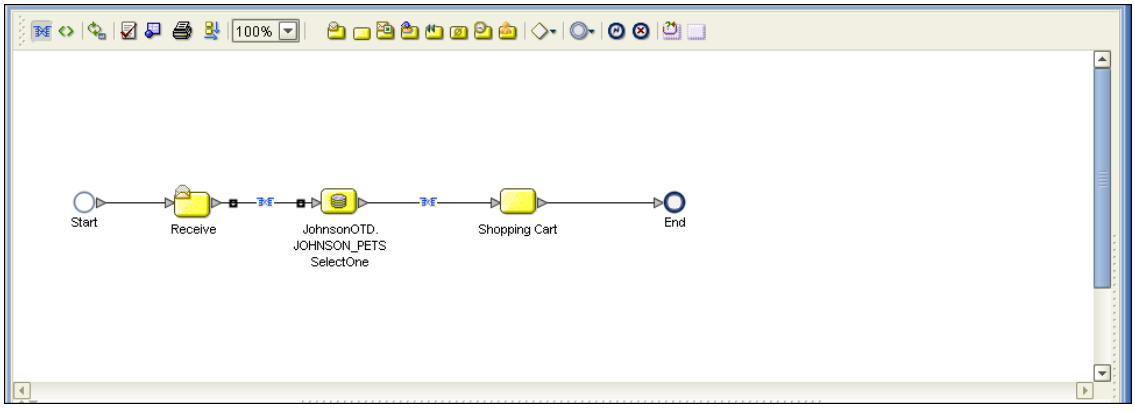


SelectOne

The input to a SelectOne operation is a where() clause. The where() clause defines to which criteria rows must adhere to be selected for the operation. In the SelectOne operation, the first row that fits the criteria is returned.

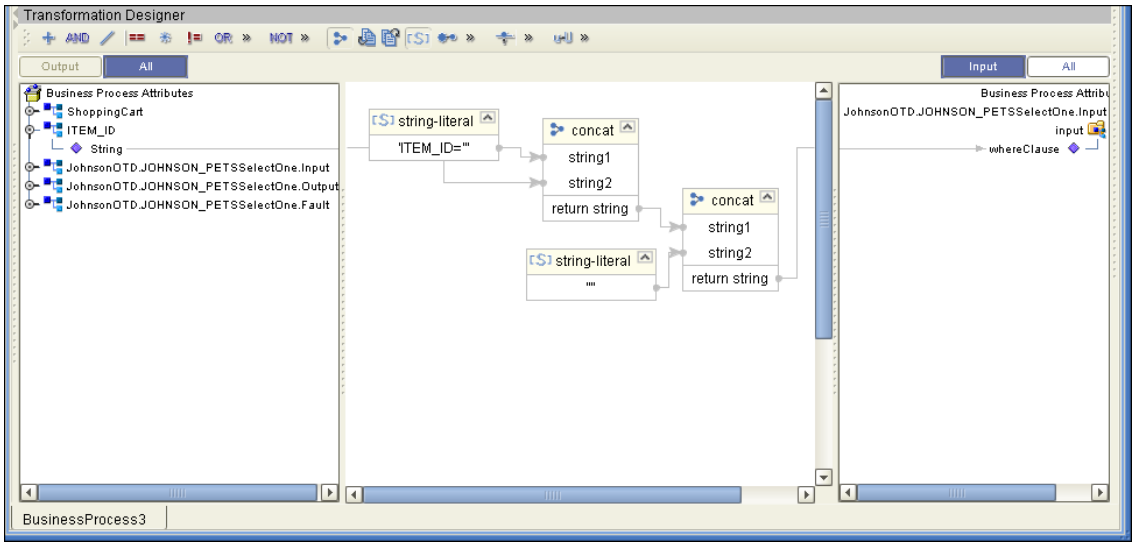
The figure below shows a sample eInsight Business Process using the SelectOne operation. In this process, the SelectOne operation returns the first row where the ITEM_ID matches the specified ITEM_ID to the shopping cart.

Figure 33 SelectOne Sample Business Process



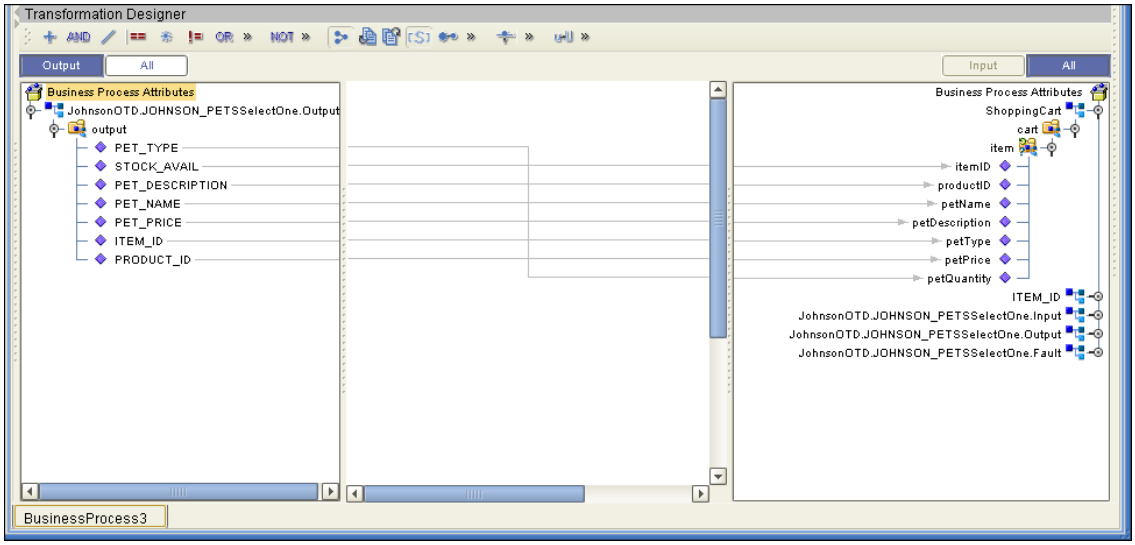
The figure below shows the definition of the where() clause for the SelectOne operation.

Figure 34 SelectOne Input



The figure below shows the definition of the output for the SelectOne operation. For the first row selected during the operation, the shopping cart shows the columns of that row as defined here.

Figure 35 SelectOne Output

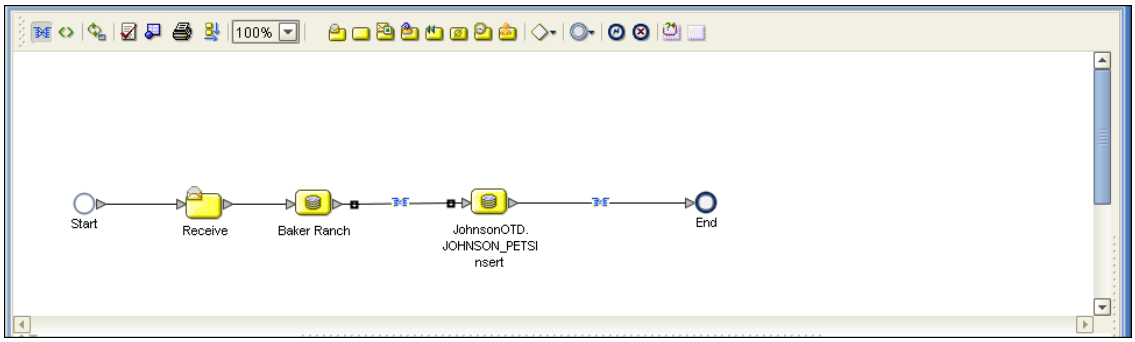


Insert

The Insert operation inserts a row. The input to an Insert operation is a where() clause. The where() clause defines to which criteria rows must adhere to be selected for the operation. In the Insert operation, the first row that fits the criteria is returned.

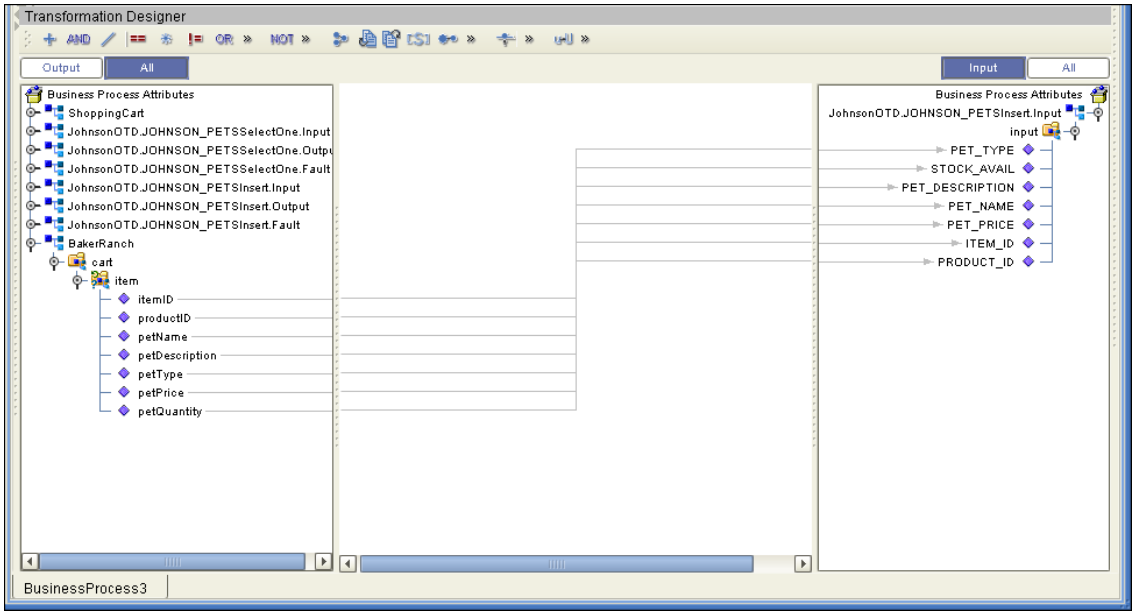
The figure below shows a sample eInsight Business Process using the Insert operation. In this process, the operation inserts a new row into the database to accommodate a new item provided by a vendor.

Figure 36 Insert Sample Business Process



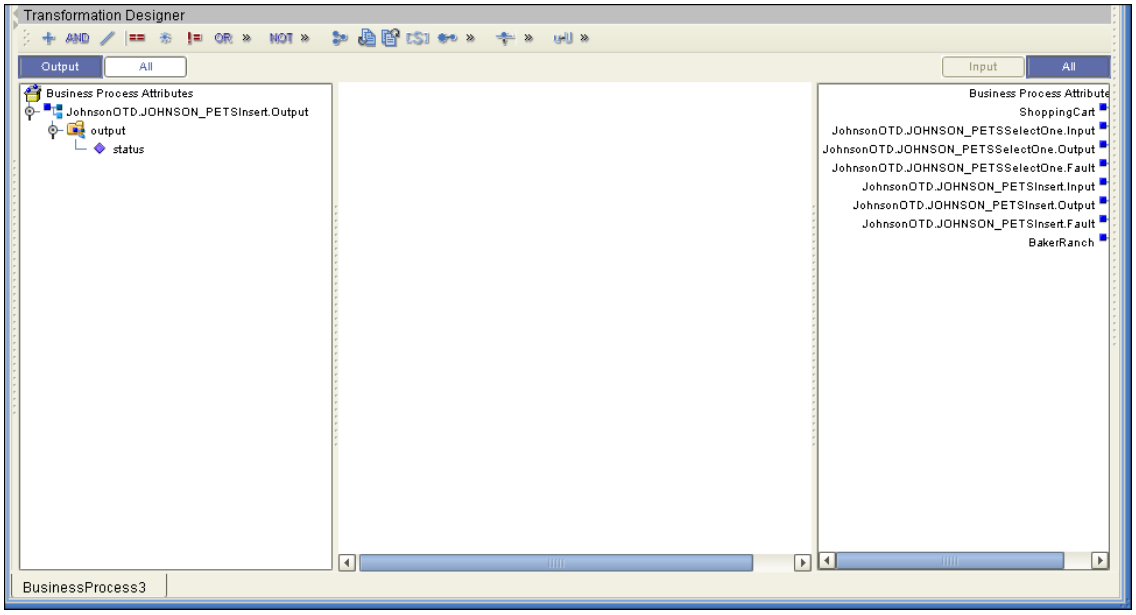
The figure below shows the definition of the input for the Insert operation.

Figure 37 Insert Input



The figure below shows the output of the Insert operation, which is a status indicating the number of rows created.

Figure 38 Insert Output

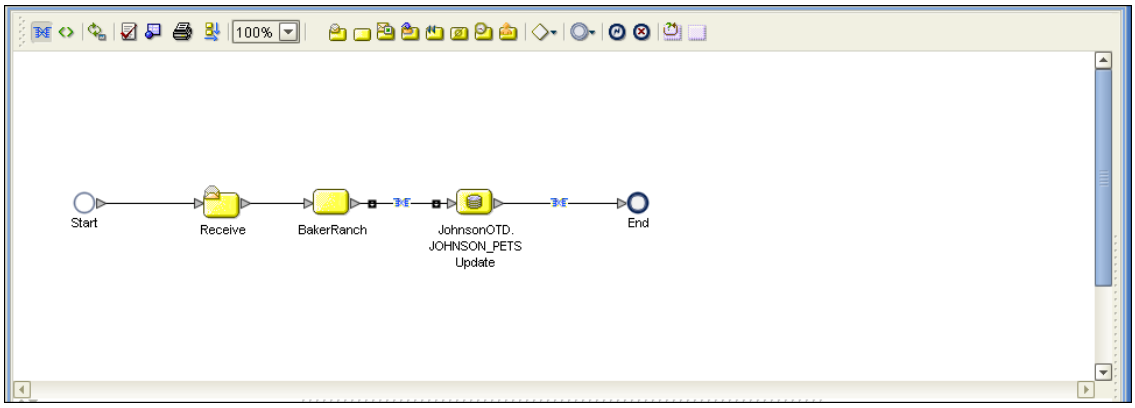


Update

The Update operation updates rows that fit certain criteria defined in a where() clause.

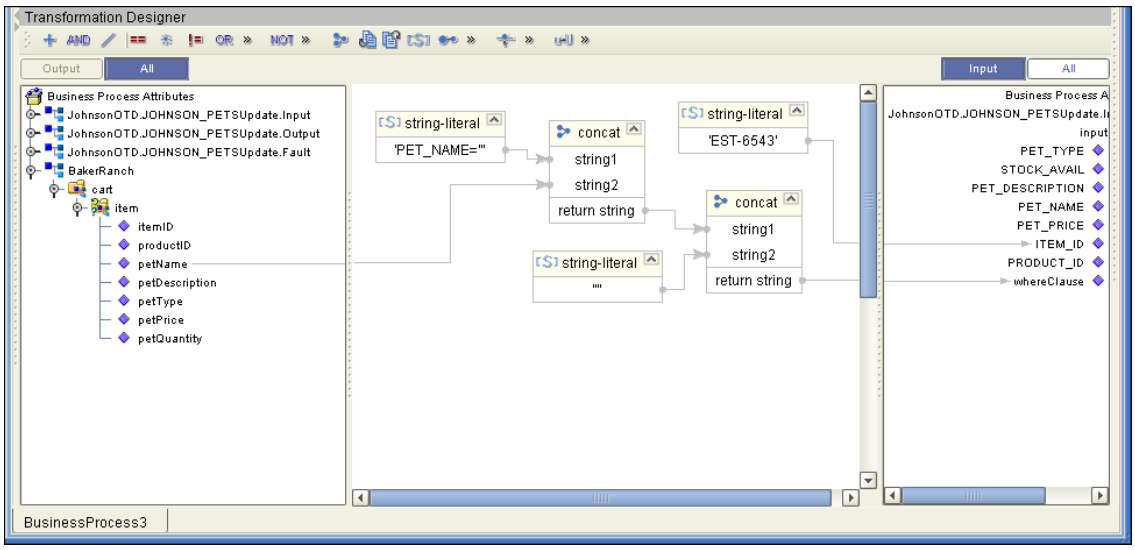
The figure below shows a sample eInsight Business Process using the Update operation. In this process, the operation updates the ITEM_ID for all items with a certain name to ESR_6543.

Figure 39 Update Sample Business Process



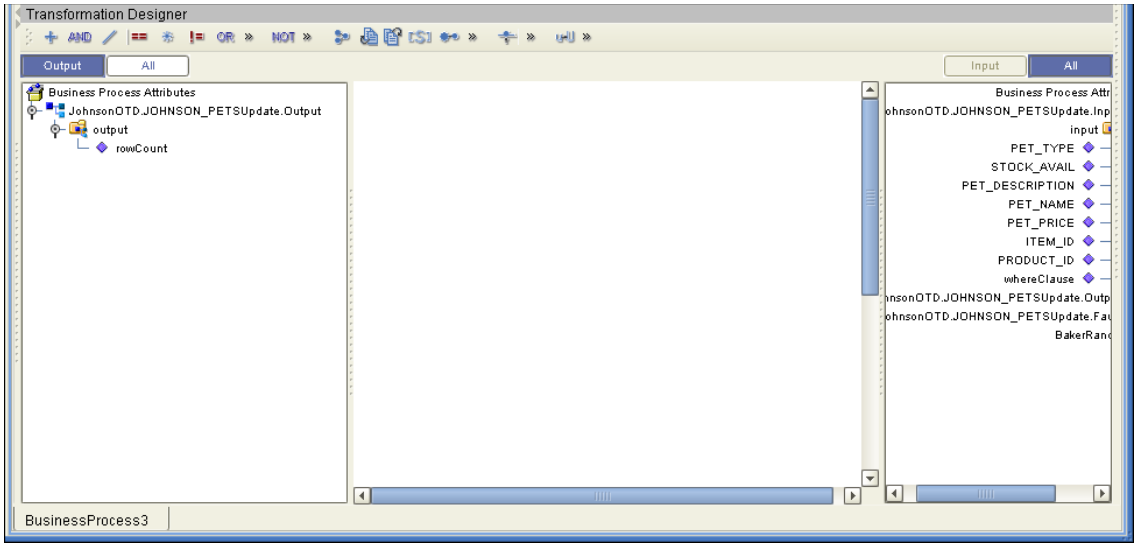
The figure below shows the definition of the where() clause for the Update operation.

Figure 40 Update Input



The figure below shows the output of the Update operation, which is a status indicating the number of rows updated.

Figure 41 Update Output

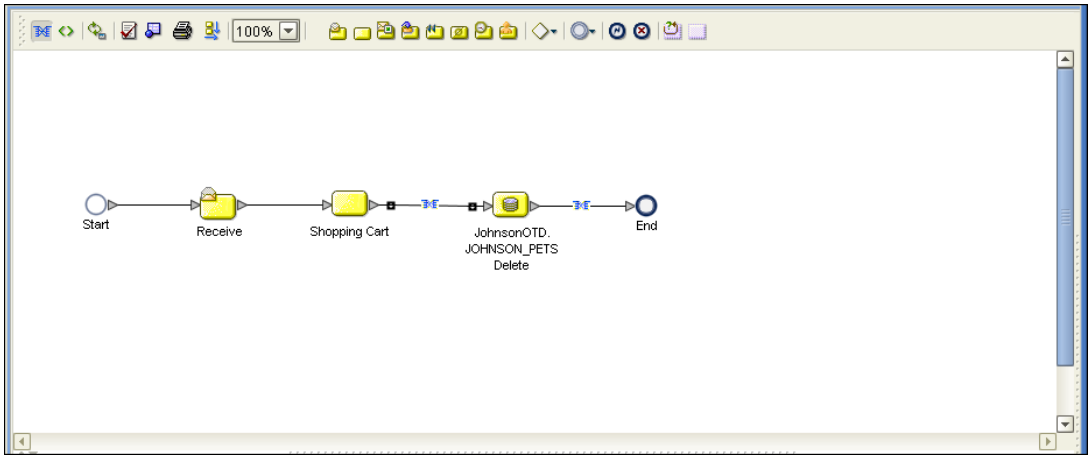


Delete

The Delete operation deletes rows that match the criteria defined in a where() clause. The output is a status of how many rows where deleted.

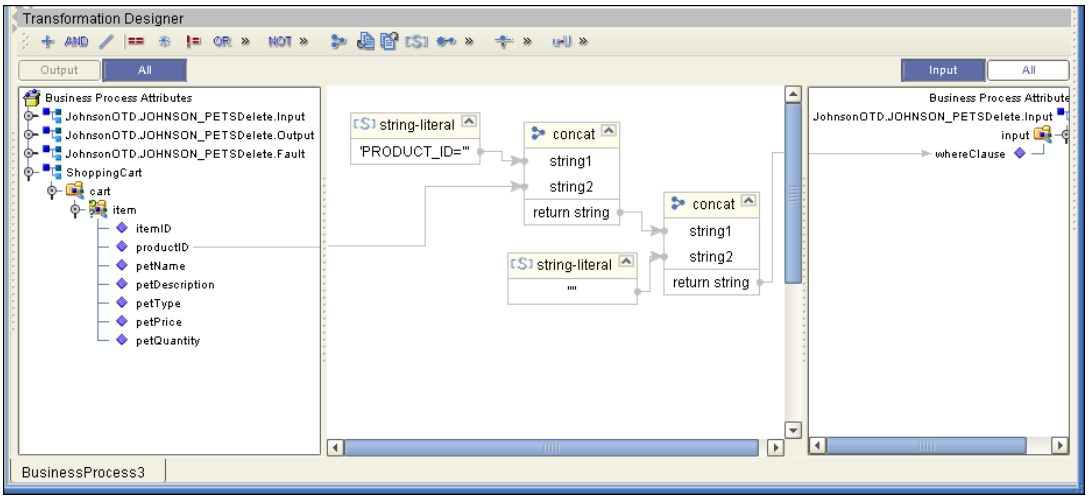
The figure below shows a sample eInsight Business Process using the Delete operation. In this process, the operation deletes rows with a certain product ID from the shopping cart.

Figure 42 Delete Sample Business Process



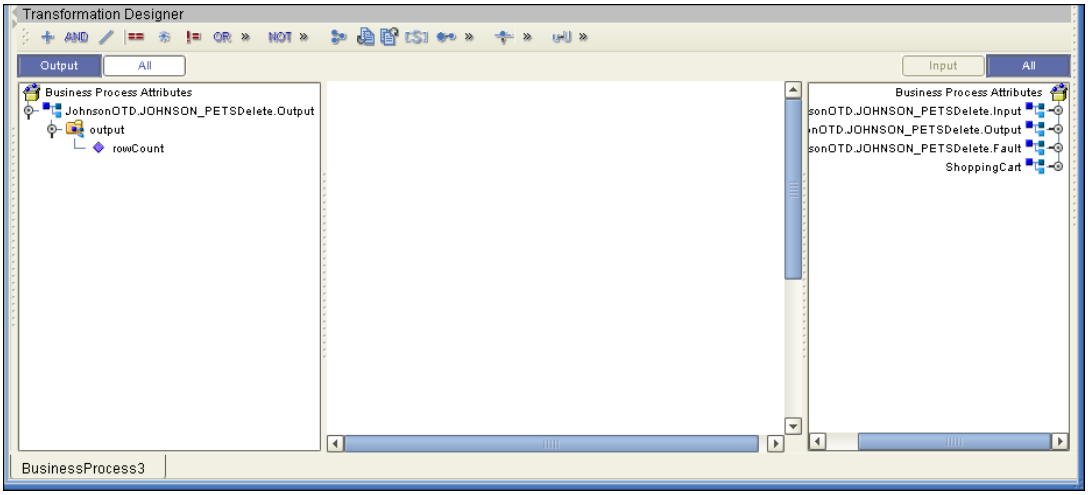
The figure below shows the definition of the where() clause for the Delete operation.

Figure 43 Delete Input



The figure below shows the output of the Delete operation, which is a status indicating the number of rows deleted.

Figure 44 Delete Output



5.3 Using the Sample Project in eGate

Follow the instructions given in **Importing the Sample Project** on page 37 importing the **Ora_JCE_Sample.zip** file.

5.3.1. Working with the Sample Project in eGate

This sample project updates the hire_date column within the DBEmployee table by selecting the employee whose employee number is equal to 800 and then updates the record.

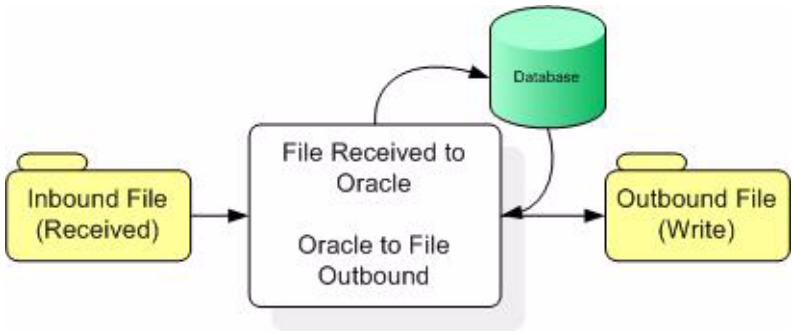
The data used for this projects is within a table called DBEmployee. The table contains the following columns with the following columns:

Table 3 Sample project data
Table 4

Column Name	Mapping	Data Type	Data Length
EMP_NO	Empno	varchar2	10
LAST_NAME	Lastname	varchar2	30
FIRST_NAME	Firstname	varchar2	30
SS_NUMBER	SSnumber	varchar2	20
HIRE_DATE	HireDate	varchar2	12

The sample project consists of an input file containing data that is passed into a collaboration and out to the Oracle database from which data is retrieved and passed back into the collaboration and then to an output file.

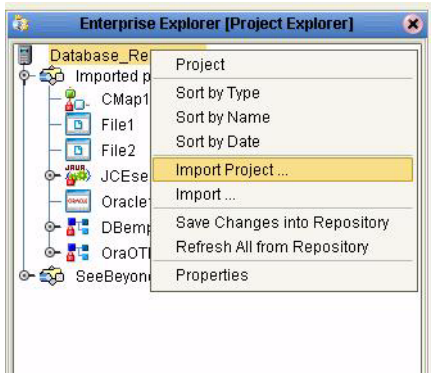
Figure 45 Database project flow



Building the Sample Project

1 On the Enterprise Explorer highlight the repository and right click. Select **Import Project**. See **Figure 46**.

Figure 46 Importing the sample project

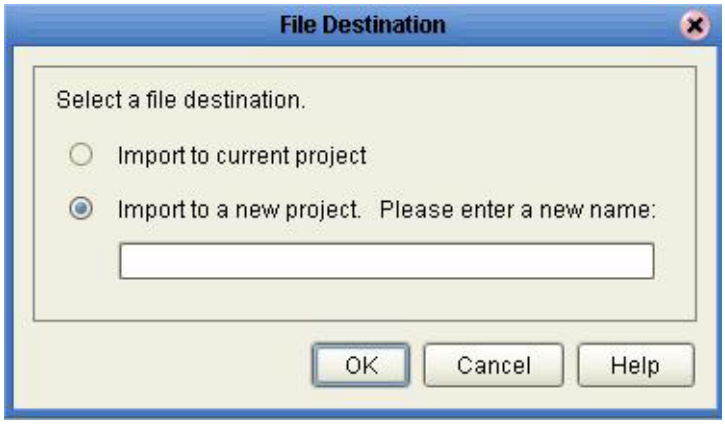


In the **Select File to Import** window, browse to the location of the sample folder and select the following .zip file **Ora_JCE_Sample.zip** and click **Open**. See **Figure 47**.

Figure 47 Select File to Import

On the **File Destination** window, select **Import a new project. Please enter a new name**. Enter a name for the sample project and click OK. See **Figure 48**.

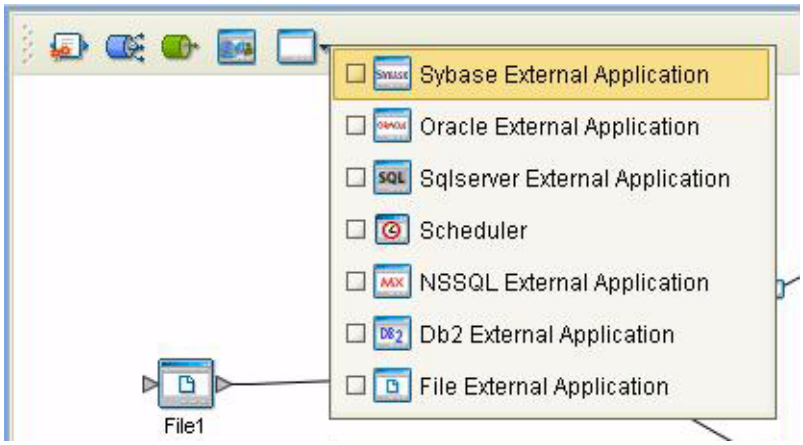
Figure 48 Select the project file destination



Creating a Connectivity Map

1. Create a Connectivity Map. On the Enterprise Explorer right-click on the sample project. Select New followed by Connectivity Map.
2. On the Connection Map toolbar, select the eWay's external application from the submenu of the External Application icon. See **Figure 49**.

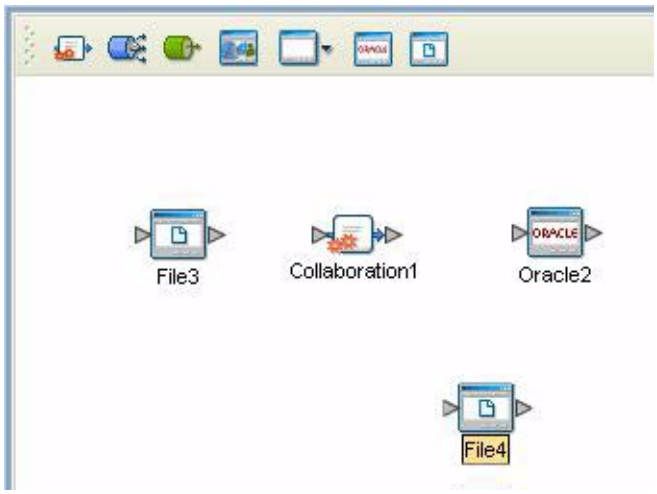
Figure 49 External Application



3. An icon representing the external application will appear on the tab's toolbar. Drag the external application icon onto the Project Canvas.

   For this project, drag and drop onto the canvas two File eWays, a Collaboration, and an Oracle eWay. See **Figure 50**.

Figure 50 Connectivity Map - Components on the Canvas



Creating the Object Type Definition

The next step in creating sample project is to create the **Object Type Definitions** or **OTDs**. The Object Type Definition (OTD) Wizard is used to generate a set of rules that define an object that encodes events as they move through the eGate system.

The Parts of the OTD

An Object Type Definition (OTD) is comprised of:

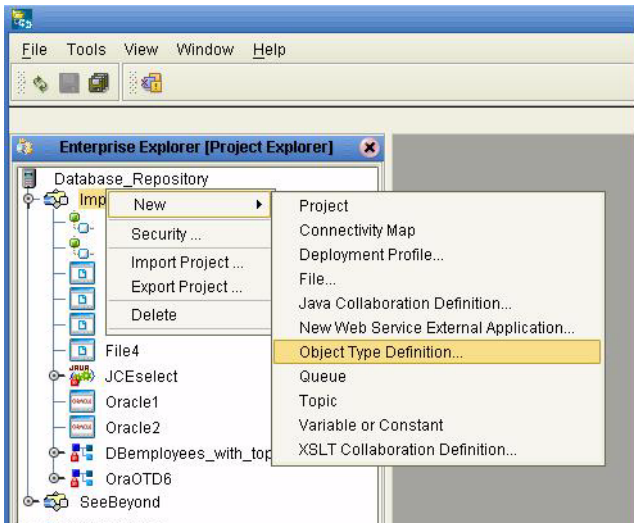
- **Element** – This is the highest level in the OTD tree. The element is the basic container that holds the other parts of the OTD. The element can contain fields and methods.
- **Field** – Fields are used to represent data. A field can contain data in any of the following formats: string, boolean, int, double, or float.
- **Method** – Method nodes represent actual Java methods.
- **Parameter** – Parameter nodes represent the Java methods' parameters.

Creating the Sample Project OTDs

For the sample project, you need to create two OTDs and one database OTD.

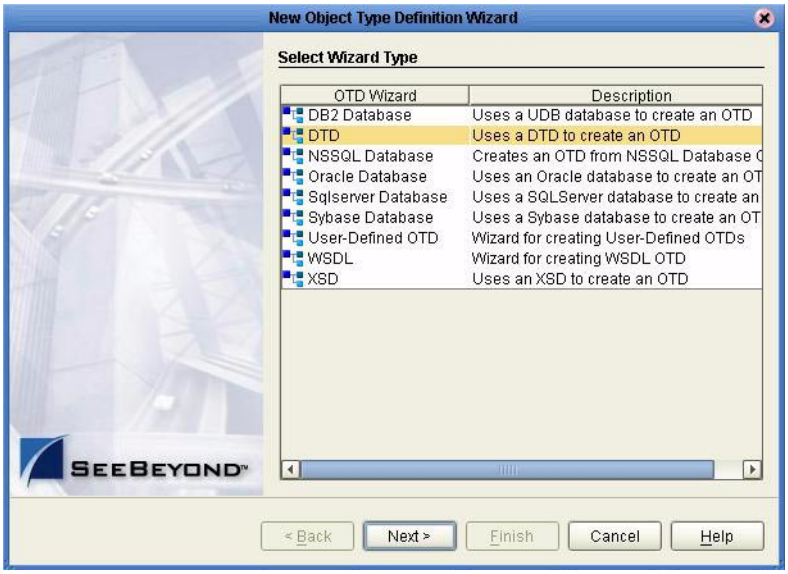
1 On the Enterprise Explorer, highlight the sample project and right click. From the submenu, click New Object Type Definition. See **Figure 51**.

Figure 51 Create an Object Type Definition



2 On the **New Object Type Definition Wizard** window, select the **DTD - Use a DTD to build an OTD**. Click **Next** See **Figure 52**.

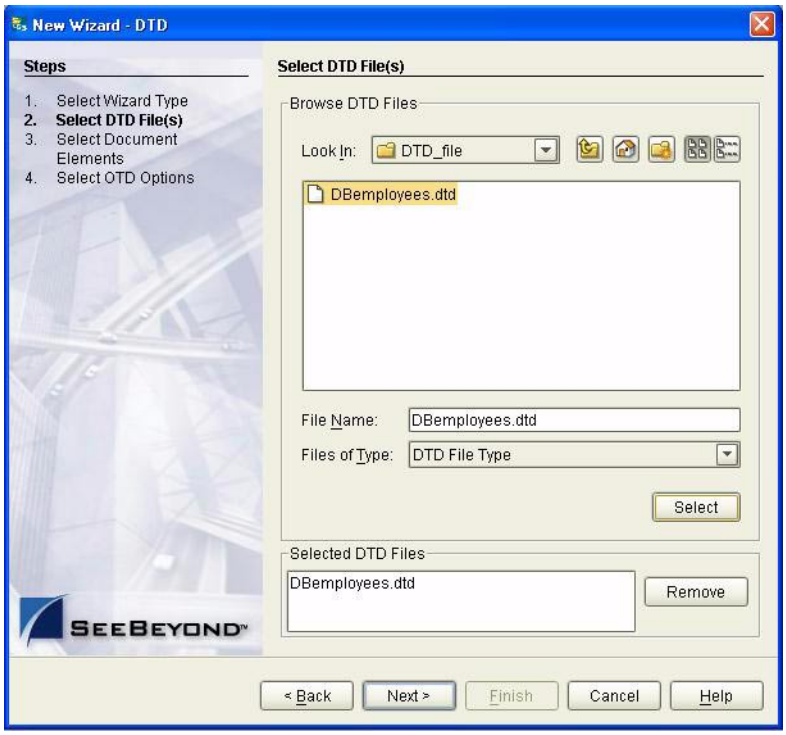
Figure 52 Create an OTD



3 From the **Include DTDs to Selected List** window, browse to the **DBemployees.dtd** located in the sample folder. Highlight **DBemployees.dtd** and click **Select**.

4 The **DBemployees.dtd** file should appear in the **Selected DTD Files** window. If it does not, repeat Step 3. Once the **DBemployees.dtd** appears in the **List of Selected DTDs** window, click **Next**. See **Figure 53**.

Figure 53 Include DTDs to Selected List

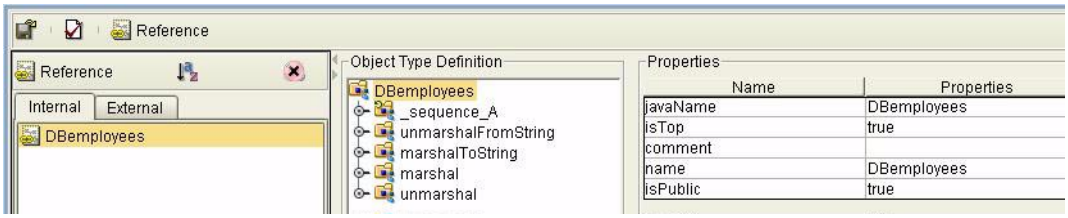


5 On the **Select Documents Elements** window click **Next**.

6 On the **Select OTDs Options** window, click **Finish**. No other options need to be selected for the sample project's **DBEmployees.otd**.

7 The resulting OTD will appear on the Enterprise Designer's canvas. See **Figure 54**.

Figure 54 DBEmployees.otd OTD



5.3.2. The Database OTD Wizard

The Database OTD Wizard can create an editable Object Type Definition (OTD) and a non-editable Object Type Definition (OTD). These types of OTDs can also be combined with each other. The types of OTDs are:

- **The Table/View OTD** – The Table/View OTD contains fields for each of the columns in the selected Table as well as the methods required to exchange data with

the external data source. The View OTD contains selected columns from selected tables. View OTDs are read-only.

- **The Stored Procedure OTD** – The Stored Procedure OTD contains fields which correspond to the input and output fields in the procedure.

Note: *Stored Procedure Result sets are not supported.*

- **The Prepared Statement OTD** – The Prepared Statement OTD contains a result set for the prepared statement.

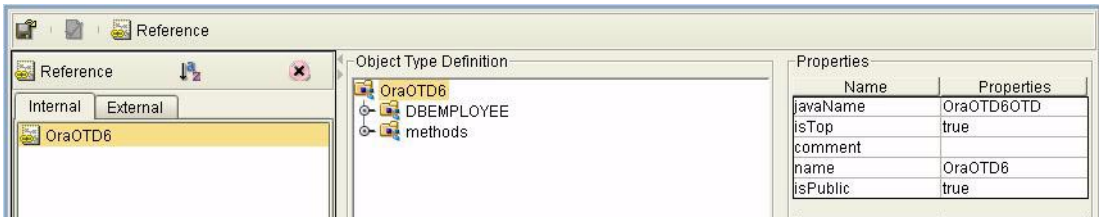
To create a new OTD using the Database Wizard

The sample contains the **OraOTD6** OTD file. You will not need to create this file.

1 To build a database OTD, see **Using the Oracle eWay Database Wizard** on page 25 to create the OTD.

The resulting **OraOTD6** appears as follows. See **Figure 55**.

Figure 55 OraOTD6 OTD



5.3.3. Creating the Collaboration

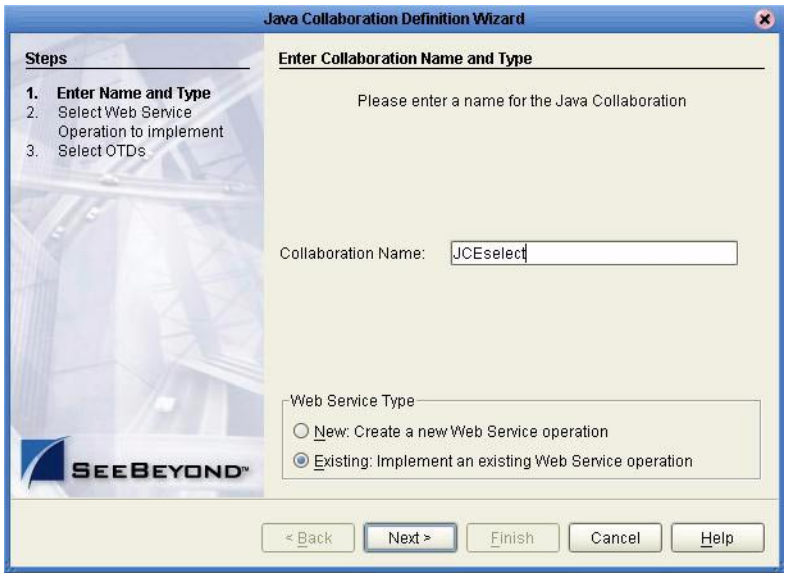
Before eGate can recognize the source and destination data relationships, you need to create a Collaboration.

1 On the Enterprise Designer, right-click on the name of your Project.

2 On the Project's sub-menu, select **New Java Collaboration Definition**.

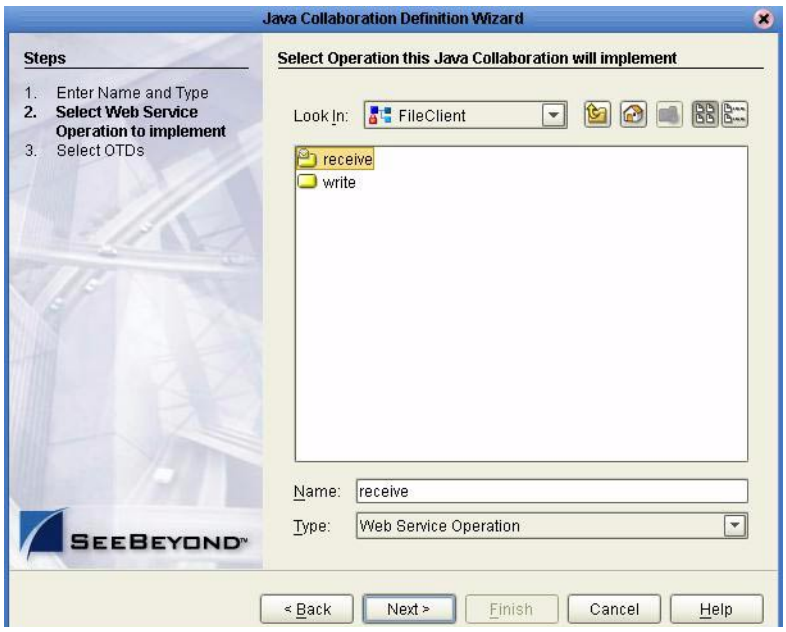
3 From the **Java Collaboration Definition Wizard** Step 1 window, enter a name for your collaboration. For the sample, enter **JCESelect** and click **Next**. See **Figure 56**.

Figure 56 Java Collaboration Definition Wizard - Step 1



4 From the **Java Collaboration Definition Wizard** Step 2 window, select and double-click **SeeBeyond**. Select and double-click **eWays**. Select and double-click **File**. Select and click **receive**. Click **Next**. See **Figure 57**

Figure 57 Java Collaboration Definition Wizard - Step 2



5 On the Collaboration Definition Wizard Step 3 window, select and double-click **Java_Service_Sample**. Select and double-click **DBEmployees_with_top_DBEmployees** adding it to the **OTDs Selected** pane.

6 Select **Ora6OTD** and double click adding it to the **OTDs Selected** pane.

7 Click the **Up one level** icon. Select and double click **Seebeyond**. Select and double click **eWays**. Select and double click **File**. Select and double click **FileClient**. Click **Finish**.

8 On the Enterprise Explorer, select and double-click the connectivity map **CMap1**.

9 On the Enterprise Explorer, select the **JCEselect OTD** and while holding down the left mouse button, drag-and-drop the **JCEselect OTD** onto the **Service1** icon on the connectivity map canvas.

10 The Collaboration Editor window will open and you can begin making your data associations.

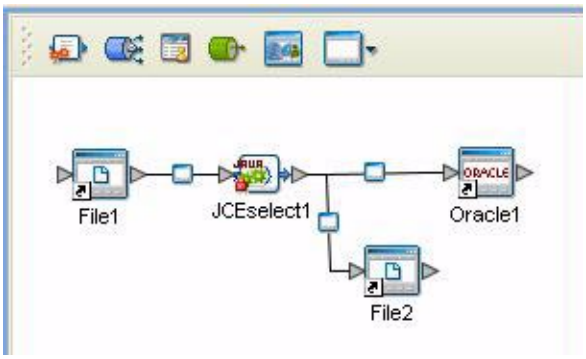
Creating Data Connections on the Connectivity Map

Create connections between the components as follows:

1 To assign inbound and outbound data for the File eWays, double click on the small square icon that represents an eWay. See Figure 50.

- File1 to JCESelect1 as an Inbound File eWay.
- JCESelect1 to the Oracle1.
- JCESelect1 and File2 as an Outbound File eWay.

Figure 58 Connectivity Map - External Components



5.3.4. Configuring the eWays

To configure the Inbound File eWay:

1 On the Connectivity Map canvas, double click the eWay icon located between the **File1** and **JCESelect1** service.

2 On the resulting **Templates** window, select **Inbound File eWay** and click **OK**.

3 On the **Properties** window, enter the appropriate configurations for the Inbound File eWay. See the ***File eWay User's Guide*** for information on how to specifically configure the File eWay. For this sample, the default settings are used.

4 When you have completed your selections, click **OK**.

To configure the Outbound Oracle1 eWay:

1 On the Connectivity Map canvas, double click the eWay icon located between the **JCESelect1** service and **Oracle1** database.

2 On the resulting **Templates** window, select **Outbound Oracle eWay** and click **OK**.

3 On the Properties window, enter the appropriate configurations for the Outbound Oracle eWay and click OK. See **Working with eWay Property Sheets** on page 10. For this sample, the default settings are used.

4 When you have completed your selections, click **OK**.

To configure the Outbound File eWay:

1 On the Connectivity Map canvas, double click the eWay icon located between the **JCESelect1** and **File2** service.

2 On the resulting **Templates** window, select **Outbound File eWay** and click **OK**.

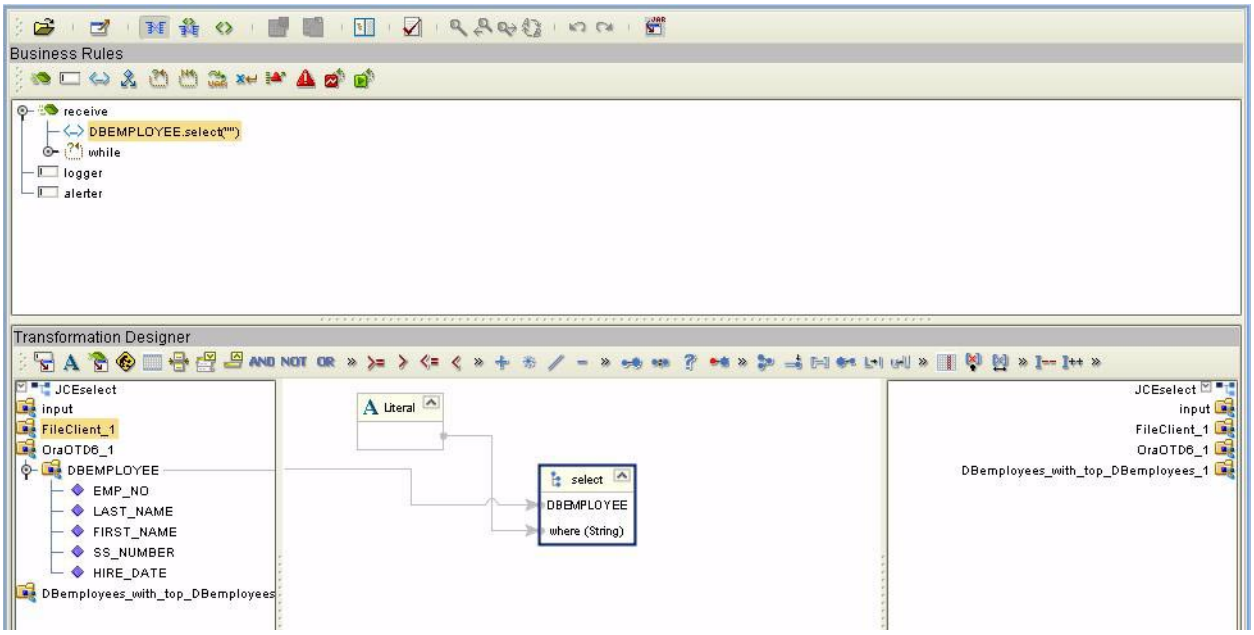
3 On the **Properties** window, enter the appropriate configurations for the Outbound File eWay. See the ***File eWay User's Guide*** for information on how to specifically configure the File eWay. For this sample, change the Directory field to **<valid path to the directory where the output file will be stored>**. The Output File Name to **JCESelect_output%d.dat**. For the remaining parameters, the default settings are used.

4 When you have completed your selections, click **OK**.

Creating a Rules within the Collaboration

1 On the Transformation Designer window's left pane, expand the **OraOTD6** node and the **DBEMPLOYEE** sub-node by selecting them and double-clicking. See **Figure 59**.

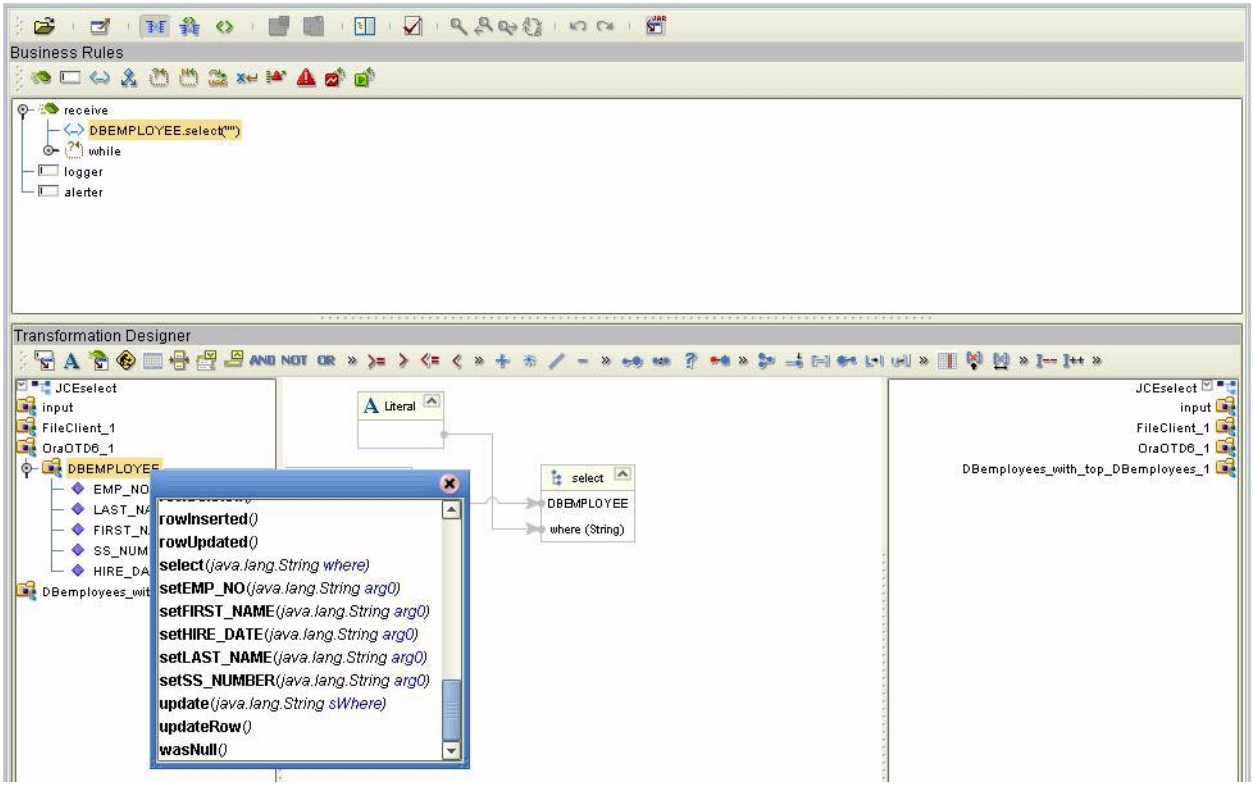
Figure 59 DBEMPLOYEE Select



2 To create a new rule, highlight **DBEMPLOYEE** and right click to open the Methods sub-menu.

3 From the **Select a method to call** submenu, highlight the **select(java.lang.String where)** method and enter. See **Figure 60**.

Figure 60 Selecting a method

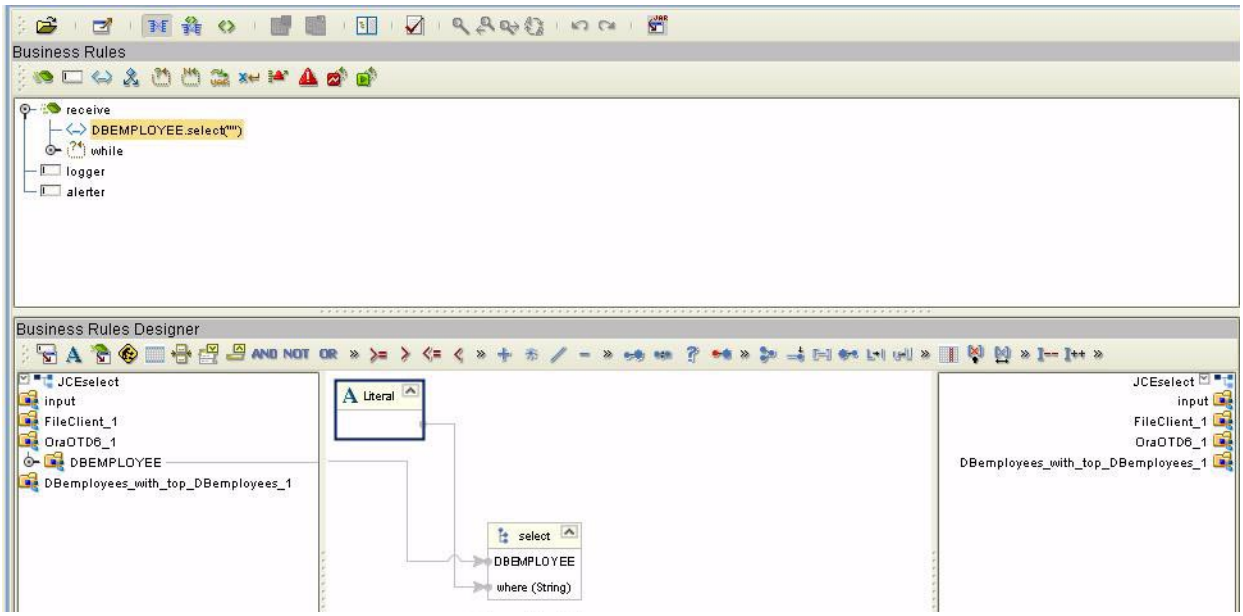


4 Next you need to create a **Literal** by selecting the **Literal (A)** icon on the Transformation Designer toolbar.

5 On the **Create Literal** popup window, select **String** from the drop-down list and click **OK**.

6 On the **Literal** box, place your cursor inside the box and while depressing the left mouse key, drag a new connection line between the **Literal** box to the **where(String)** method node located next to the **select()** box. See **Figure 61**.

Figure 61 Literal mapping



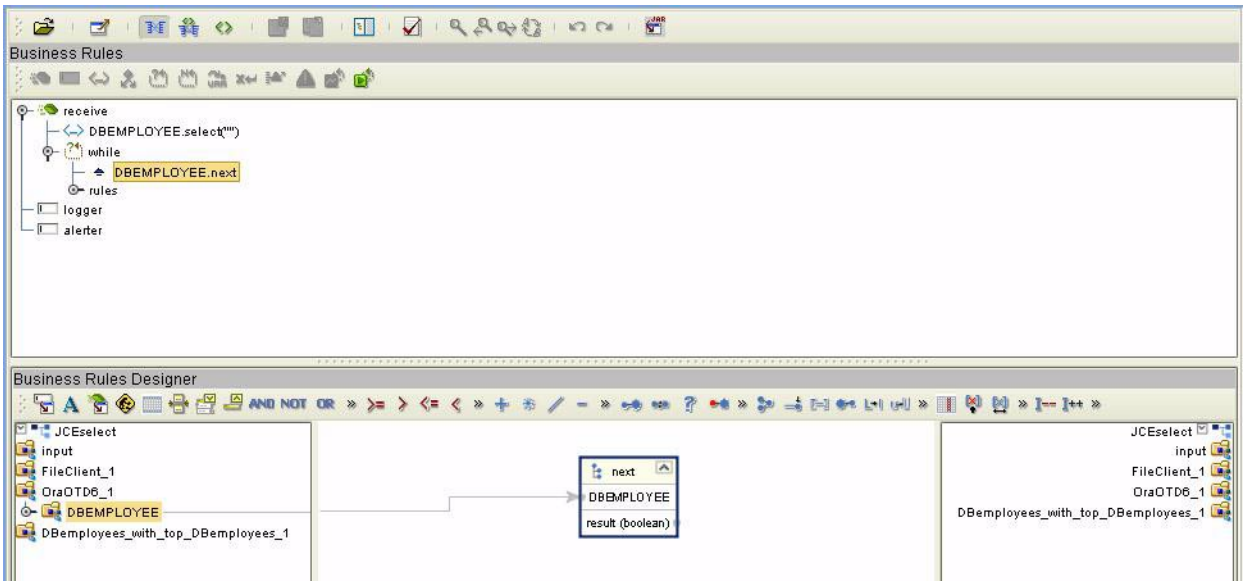
A new rule is created.

```
//DBEMPLOYEE.select("")
        OraOTD6_1.getDBEMPLOYEE().select("");
```

7 On the Business Rules canvas, highlight the `<_>DBEMPLOYEE.select("")` rule. Click the **while()** icon located on the toolbar. A new **while() condition** is waiting to be created within the list.

8 On the **Business Rules** list, highlight the condition. While highlighted, select DBEMPLOYEE located under the OraOTD6_1 node. Right click and select the next() method from the sub-menu. See **Figure 62**.

Figure 62 next()

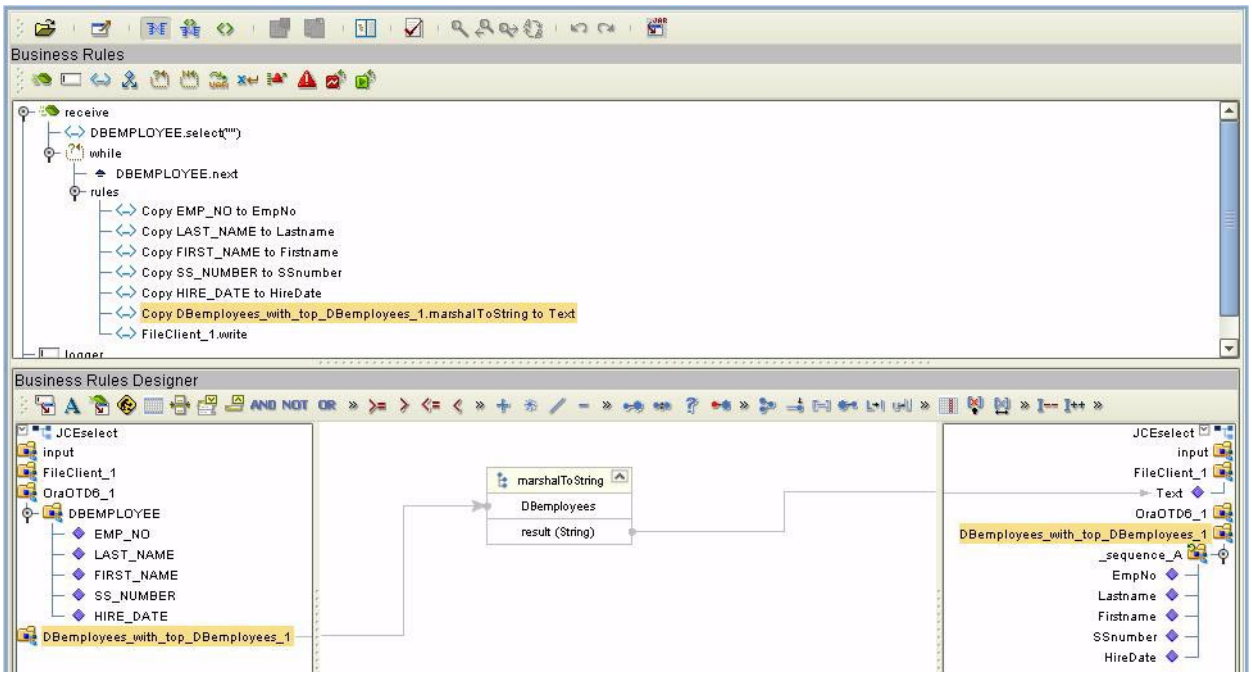


9 On the Business Rules list, highlight the **New Rule** located under the **DBEMPLOYEE.next** condition. On the left pane of the Business Rules canvas, expand the **DBEMOPLOYEE** sub-node located under the **OraOTD6_1** OTD node to expose the database columns. On the right side of the Business Rules canvas, expand the **DBemployees_with_top_DBemployees_2** OTD node. Expand the **_sequence_A** node exposing the table's columns. Drag-and-drop the following fields from the left pane under the **DB_EMPLOYEE** sub-node to the right pane under **_sequence_A**:

EMP_NO	---------->	EmpNo
LAST_NAME	---------->	Lastname
FIRST_NAME	--------- >	Firstname
SS_NUMBER	---------->	SSnumber
HIRE_DATE	---------->	HireDate

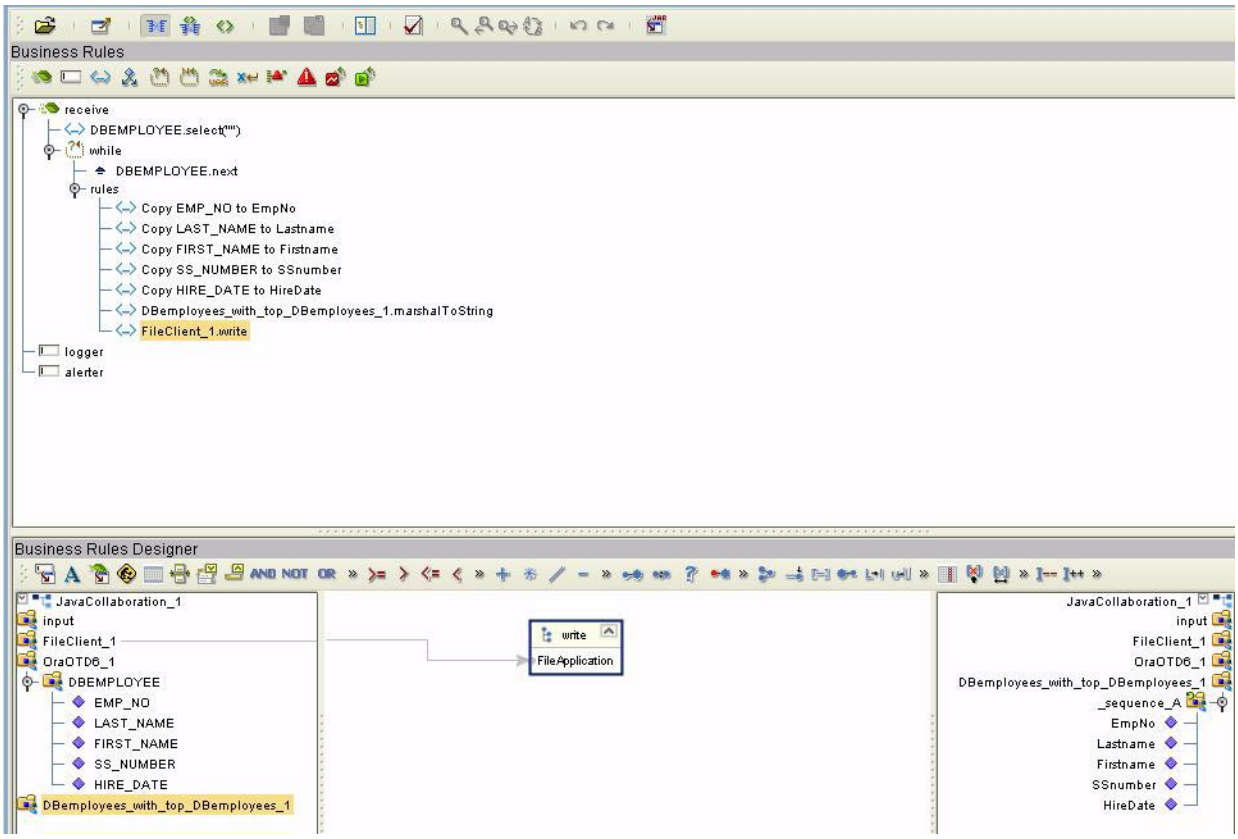
10 On the Business Rules list, highlight **Copy Hire_Date to HireDate**. On the Business Rules Designer's left pane, right-click on the **DBemployees_with_top_DBemployees_1** node. From the sub-menu, select and click **Select a method to call**. From the sub-menu, select and double click **marshalToString()**. See Figure 63.

Figure 63 marshalToString()



11 On the Business Rules list, highlight **DBemployees_with_top_DBemployees_1.marshalToString**. On the Business Rules Designer's left pane, right-click on the **FileClient_1** node. From the sub-menu, select and click **Select a method to call**. From the sub-menu, select and double-click **write()**. See

Figure 64 write()



12 Click **Save** to save the Collaboration.

5.3.5. Creating a Deployment Profile

To review the components of the Sample project, there is an Inbound and an Outbound File eWay, an Oracle eWay, and a Service.

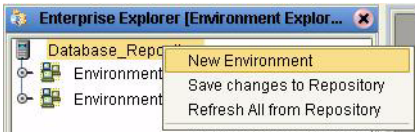
To create the external environment for the Sample project:

1 Click on the Environment tab on the Enterprise Explorer.

2 In the Environment Explorer, right-click on the Repository. The Repository name it the name you gave the Repository during the installation process.

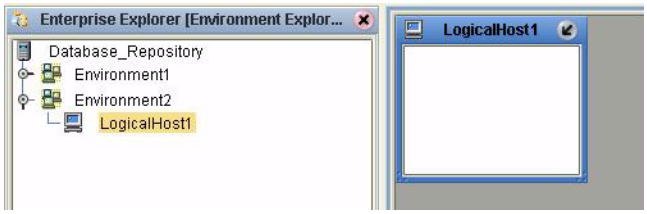
3 Right-click, select New Environment from the sub-menu. See **Figure 65**.

Figure 65 New Environment



4 On the Enterprise Explorer, select the new environment and right-click. Select New Logical Host from the sub-menu.

Figure 66 New Logical Host



5 Follow Step 4 to create a **New File External System** for both the Inbound and the Outbound File eWays, and for a **New Oracle External System**.

6 On the Environment Explorer, highlight and right-click the Oracle profile. Select **Properties**. Enter the configuration information required for your Outbound Oracle eWay. See **Properties of the Oracle eWay** on page 10.

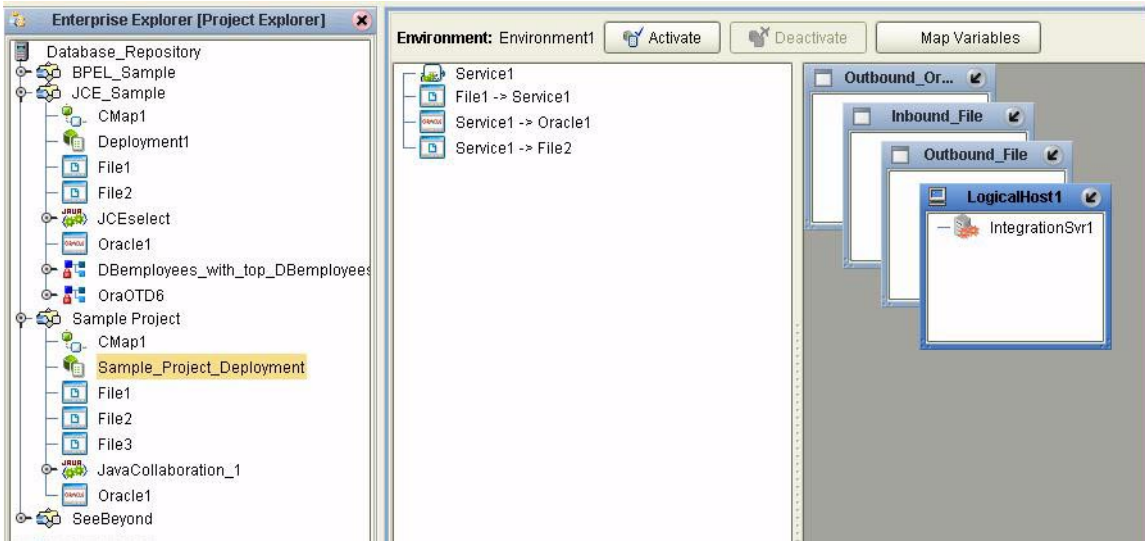
7 Return to the Project Tab.

5.3.6. Deploying the DatabaseSelect Project

1 On the Enterprise Explorer, select the Sample project and right-click selecting **New, Deployment Profile** from the sub-menu.

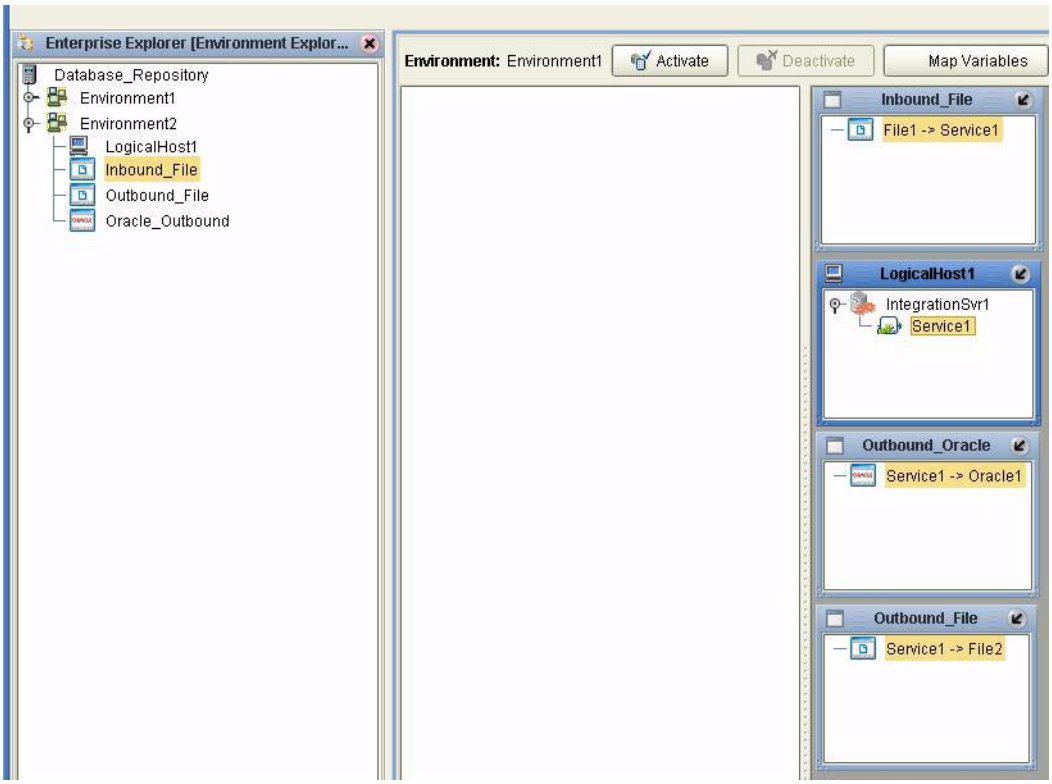
2 On the **Create a Deployment Profile for** window, enter a name for the Profile and click **OK**.

Figure 67 Deployment Profile



3 On the left canvas of the Environment, drag each eWay onto the appropriate external system creating the Deployment Mappings. See the *eGate Tutorial* for additional information. See

Figure 68 Deployment Mappings



4 On the Environments tool bar, click **Activate**.

A message indicating that the deployment activation was successful will display.

5.3.7. Running the Sample

For instruction on how to run the Sample project, see the *eGate Tutorial*.

Once the process has completed, the Output file in the target directory configured in the Outbound File eWay will contain all records retrieved from the database in an .xml format.

5.4 Supported Data Types

The Oracle eWay supports the following data types:

Table 5 Standard Data Types Supported by the Oracle eWay

Data Type	Description	Column Length
Number	Variable-length numeric data. Maximum precision p and/or scale s is 38.	Variable for each row. The maximum space required for a given column is 21 bytes per row.
VarChar2	Variable-length character data, with maximum length size bytes or characters.	Variable for each row, up to 4000 bytes per row. Consider the character set (single-byte or multibyte) before setting size. A maximum size must be specified.
Char	Fixed-length character data of length size bytes or characters.	Fixed for every row in the table (with trailing blanks); maximum size is 2000 bytes per row, default size is 1 byte per row. Consider the character set (single-byte or multibyte) before setting size.
Raw	Variable-length raw binary data.	Variable for each row in the table, up to 2000 bytes per row. A maximum size must be specified. Provided for backward compatibility.
Long	Variable-length character data.	Variable for each row in the table, up to 2^{32} - 1 bytes, or 2 gigabytes, per row. Provided for backward compatibility.
Clob		Up to 2^{32} - 1 bytes, or 64k.
Date	Fixed-length date and time data, ranging from Jan. 1, 4712 B.C.E. to Dec. 31, 4712 C.E.	Fixed at 7 bytes for each row in the table. Default format is a string (such as `DD-MON-RR`) specified by the `NLS_DATE_FORMAT` parameter. Oracle expects a format of: “YYYY-MM-DD; hh:mm:ss.x”.

5.5 Converting Data Types in the Oracle eWay

When working with data in the Oracle eWay OTD's, you may need to do a data conversion. The following tables should help:

Table 6 Insert and Update Operations Datatype Conversions (Text/String input data)

Oracle Data Type	OTD/Java Data Type	Java Method or New Constructor to Use (Default: Java Method)	Sample Data
Int	BigDecimal	**Call a New Constructor BigDecimal**: java.math.BigDecimal(String)	123
Smallint	BigDecimal	**Call a New Constructor BigDecimal**: java.math.BigDecimal(String)	123
Number	BigDecimal	**Call a New Constructor BigDecimal**: java.math.BigDecimal(String)	123
Decimal*	BigDecimal	**Call a New Constructor BigDecimal**: java.math.BigDecimal(String)	147.78
Real	Double	**Double**: java.lang.Double.parseDouble(String)	147.78
Float	Double	**Double**: java.lang.Double.parseDouble(String)	147.78
Double	Double	**Double**: java.lang.Double.parseDouble(String)	147.78
Date	TimeStamp	**TimeStamp**: java.sql.TimeStamp.valueOf(String)	2003-08-11 11:47:39.0
Varchar2	String	Direct Assign	Any character
Char	String	Direct Assign	Any character
Long Char	String	Direct Assign	Any character
Raw	Byte[]	**String**: java.lang.String.getBytes()	Any character
Long Raw	Byte[]	**String**: java.lang.String.getBytes()	Any character
CLOB	Clob	See Appendix	Any character

* When using Prepared Statements with the Inbound Oracle eWay, you will need to change the decimal type to a double data type in the OTD.

Table 7 Select Operation Datatype Conversion (Text/String output data)

Oracle Data Type	OTD/Java Data Type	Methods To Use	Sample Data
Int	BigDecimal	**BigDecimal**: java.math.toString()	123
SmallInt	BigDecimal	**BigDecimal**: java.math.toString()	123
Number	BigDecimal	**BigDecimal**: java.math.toString()	123
Decimal	BigDecimal	**BigDecimal**: java.math.toString()	123.67
Real	Double	**Double**: java.lang.Double.toString(double)	123
Float	Double	**Double**: java.lang.Double.toString(double)	123
Double	Double	**Double**: java.lang.Double.parseDouble(String)	123
Date	TimeStamp	**TimeStamp**: java.sql.TimeStamp.valueof()	2003-08-11 11:47:39.0
Varchar2	String	Direct Assign	Any characters
Char	String	Direct Assign	Any Characters
Raw	Byte[]	N/A	N/A
Long Raw	Byte[]	N/A	N/A
CLOB	Clob	N/A	N/A

5.6 Using OTDs with Tables, Views, and Stored Procedures

Tables, Views, and Stored Procedures are manipulated through OTDs. Common operations include insert, delete, update, and query.

5.6.1. Data Types

Oracle Tables support:

- Real
- Float
- CLOB.

For all others, to use the data types Float, Double, and CLOB build them using a data type of "Other".

Long RAW for Prepared Statements and Stored Procedure support:

The following 2 parameters must be set prior to the Insert/Update/Delete statement.

```
setConcurrencyToReadOnly()
setScrollTypeToForwardOnly()
```

5.6.2. The Table

A table OTD represents a database table. It consists of fields and methods. Fields correspond to the columns of a table while methods are the operations that you can apply to the OTD. This allows you to perform query, update, insert, and delete SQL operations in a table.

By default, the Table OTD has UpdatableConcurrency and ScrollTypeForward only. The type of result returned by the select() method can be specified using:

- SetConcurrencytoUpdatable
- SetConcurrencytoReadOnly
- SetScrollTypetoForwardOnly
- SetScrollTypetoScrollSensitive
- SetScrollTypetoInsensitive

The methods should be called before executing the select() method if used. For example,

```
getDBEmp().setConcurToUpdatable();
getDBEmp().setScroll_TypeToScrollSensitive();
getDBEmp().getDB_EMPLOYEE().select("");
```

The Query Operation

To perform a query operation on a table

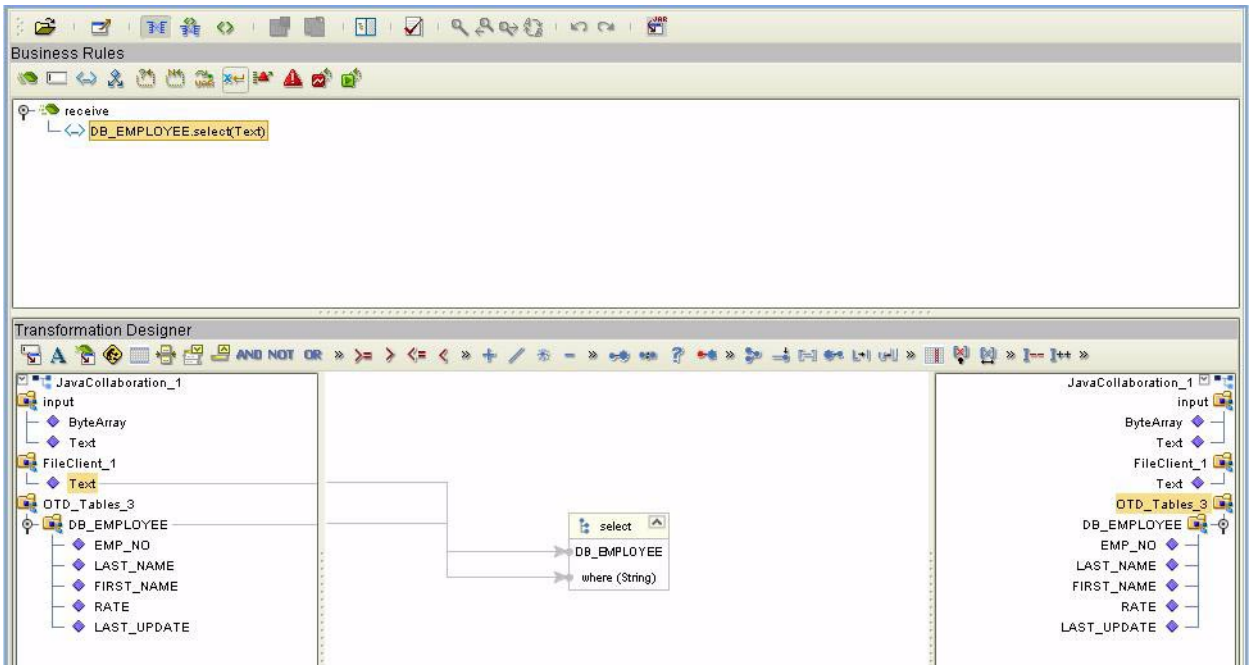
1. Execute the **select()** method with the "**where**" clause specified if necessary.
2. Loop through the ResultSet using the **next()** method.
3. Process the return record within a **while()** loop.

For example:

```
//DB_EMPLOYEE.select(Text)
    Table_OTD_1.getDB_EMPLOYEE().select( read.getText() );

//while DB_EMPLOYEE.next
    while (Table_OTD_1.getDB_EMPLOYEE().next()) {
//Copy LAST_NAME to LAST_NAME
      Table_OTD_1.getDB_EMPLOYEE().setLAST_NAME(
Table_OTD_1.getDB_EMPLOYEE().getLAST_NAME() );
}
```

Figure 69 Select()



The Insert Operation

To perform an insert operation on a table

1 Execute the **insert()** method. Assign a field.

2 Insert the row by calling **insertRow()**

This example inserts an employee record.

```
//DB_EMPLOYEE.insert
    Table_OTD_1.getDB_EMPLOYEE().insert();
//Copy EMP_NO to EMP_NO
   insert_DB_1.getInsert_new_employee().setEmployee_no(
   java.lang.Integer.parseInt(
     employeedb_with_top_db_employee_1.getEmployee_no() ) );

//@map:Copy Employee_lname to Employee_Lname
   insert_DB_1.getInsert_new_employee().setEmployee_Lname(
   employeedb_with_top_db_employee_1.getEmployee_lname() );

//@map:Copy Employee_fname to Employee_Fname
   insert_DB_1.getInsert_new_employee().setEmployee_Fname(
   employeedb_with_top_db_employee_1.getEmployee_fname() );

//@map:Copy java.lang.Float.parseFloat(Rate) to Rate
   insert_DB_1.getInsert_new_employee().setRate(
   java.lang.Float.parseFloat(
   employeedb_with_top_db_employee_1.getRate() ) );

//@map:Copy java.sql.Timestamp.valueOf(Update_date) to Update_date
   insert_DB_1.getInsert_new_employee().setUpdate_date(

java.sql.Timestamp.valueOf(
   employeedb_with_top_db_employee_1.getUpdate_date() ) );
```

```
    Table_OTD_1.getDB_EMPLOYEE().insertRow();

//Table_OTD_1.commit
  Table_OTD_1.commit();
}
```

The Update Operation

To perform an update operation on a table

1. Execute the **update()** method.
2. Using a while loop together with **next()**, move to the row that you want to update.
3. Assign updating value(s) to the fields of the table OTD
4. Update the row by calling **updateRow()**.

   In this example, we move to the third record and update the SALES_ORDERS Table and Cust_Name, and Cust_phone columns.

```
//SalesOrders_with_top_SalesOrders_1.unmarshalFromString(Text)
  SalesOrders_with_top_SalesOrders_1.unmarshalFromString(
  input.getText() );

//SALES_ORDERS.update("SO_num =99")
  DB_sales_orders_1.getSALES_ORDERS().update( "SO_num ='01'" );

//while
  while (DB_sales_orders_1.getSALES_ORDERS().next()) {

//Copy SalesOrderNum to SO_num
  DB_sales_orders_1.getSALES_ORDERS().setSO_num(
  SalesOrders_with_top_SalesOrders_1.getSalesOrderNum() );

//Copy CustomerName to Cust_name
  DB_sales_orders_1.getSALES_ORDERS().setCust_name(
  SalesOrders_with_top_SalesOrders_1.getCustomerName() );

//Copy CustomerPhone to Cust_phone
  DB_sales_orders_1.getSALES_ORDERS().setCust_phone(
  SalesOrders_with_top_SalesOrders_1.getCustomerPhone() );

//SALES_ORDERS.updateRow
  DB_sales_orders_1.getSALES_ORDERS().updateRow();
}
//DB_sales_orders_1.commit
  DB_sales_orders_1.commit();

   }
```

The Delete Operation

To perform a delete operation on a table

1. Execute the **delete()** method.
2. Move to the row that you want to delete.
3. Delete the row by calling **deleteRow()**.

   In this example DELETE an employee.

```
//DB_EMPLOYEE.delete("EMP_NO = '".concat(EMP_NO).concat("'"))
    Table_OTD_1.getDB_EMPLOYEE().delete( "EMP_NO = '".concat(
Table_OTD_1.getDB_EMPLOYEE().getEMP_NO() ).concat( "'" ) );
  }
```

5.6.3. The Stored Procedure

A Stored Procedure OTD represents a database stored procedure. Fields correspond to the arguments of a stored procedure while methods are the operations that you can apply to the OTD. It allows you to execute a stored procedure. Remember that while in the Collaboration Editor you can drag and drop nodes from the OTD into the Collaboration Editor.

Note: *When creating a Package Stored Procedure in the Database Wizard, you must select* ***Use fully qualified names****.*

Note: *Stored Procedure ResultSets are not supported.*

Executing Stored Procedures

The OTD represents the Stored Procedure "LookUpGlobal" with two parameters, an inbound parameter (INLOCALID) and an outbound parameter (OUTGLOBALPRODUCTID). These inbound and outbound parameters are generated by the DataBase Wizard and are represented in the resulting OTD as nodes. Within the Transformation Designer, you can drag values from the input parameters, execute the call, collect data, and drag the values to the output parameters.

Below are the steps for executing the Stored Procedure:

1. Specify the input values.
2. Execute the Stored Procedure.
3. Retrieve the output parameters if any.

For example:

```
package Storedprocedure;

public class sp_jce
{

public com.stc.codegen.logger.Logger logger;

public com.stc.codegen.alerter.Alerter alerter;

public void receive( com.stc.connector.appconn.file.FileTextMessage
input,com.stc.connector.appconn.file.FileApplication
FileClient_1,employeedb.Db_employee
employeedb_with_top_db_employee_1,insert_DB.Insert_DBOTD insert_DB_1
)
  throws Throwable
    {

//@map:employeedb_with_top_db_employee_1.unmarshalFromString(Text)
   employeedb_with_top_db_employee_1.unmarshalFromString(
   input.getText() );
//@map:Copy java.lang.Integer.parseInt(Employee_no) to Employee_no
```

```
      insert_DB_1.getInsert_new_employee().setEmployee_no(
      java.lang.Integer.parseInt(
      employeedb_with_top_db_employee_1.getEmployee_no() ) );
   //@map:Copy Employee_lname to Employee_Lname
      insert_DB_1.getInsert_new_employee().setEmployee_Lname(
      employeedb_with_top_db_employee_1.getEmployee_lname() );
   //@map:Copy Employee_fname to Employee_Fname
      insert_DB_1.getInsert_new_employee().setEmployee_Fname(
      employeedb_with_top_db_employee_1.getEmployee_fname() );
   //@map:Copy java.lang.Float.parseFloat(Rate) to Rate
      insert_DB_1.getInsert_new_employee().setRate(
      java.lang.Float.parseFloat(
      employeedb_with_top_db_employee_1.getRate() ) );
   //@map:Copy java.sql.Timestamp.valueOf(Update_date) to Update_date
      insert_DB_1.getInsert_new_employee().setUpdate_date(
      java.sql.Timestamp.valueOf(
      employeedb_with_top_db_employee_1.getUpdate_date() ) );
   //@map:Insert_new_employee.execute
      insert_DB_1.getInsert_new_employee().execute();
   //@map:insert_DB_1.commit
      insert_DB_1.commit();
   //@map:Copy "procedure executed" to Text
      FileClient_1.setText( "procedure executed" );
   //@map:FileClient_1.write
      FileClient_1.write();
       }

   }
```

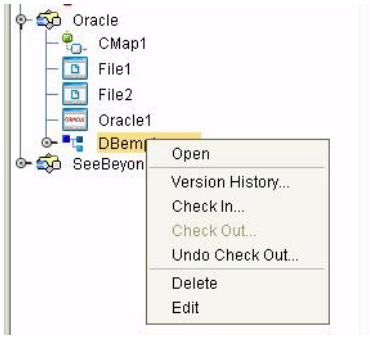
5.7 Creating or Editing an OTD from an Exhisting OTD

Editing and Exhisting OTD

To edit an OTD do the following:

1 On the Enterprise Explorer, right click on the OTD. From the submenu, click Edit. See **Figure 70**.

Figure 70 OTD Edit Menu Item



The Database Wizard will open allowing you to change the OTD. You may select additional Tables or edit the existing Tables.

Caution: *If during the edit process you delete a table that is included in a Collaboration, the Collaboration may fail at run time.*

You can also edit your Prepared Statements or Stored Procedures.

2 Save the OTD with the same name. Do not change the OTD name.

3 Close the Enterprise Explorer and reopen for your changes to take effect.

Note: *You can not edit the OTD of an imported Project.*

Creating an OTD from an Exhisting OTD

You can also create and OTD from an exhisting OTD. Open an OTD that you would like to be the basis for your new OTD. Following the steps in **Editing and Exhisting OTD** on page 73, make your edits and save the OTD with a new name.

5.8 Using XA

When using XA, do not call the commit() method.

5.9 Alerting and Logging

eGate provides an alerting and logging feature. This allows monitoring of messages and captures any adverse messages in order of severity based on configured severity level and higher. To enable Logging, please see the ***eGate Integrator User's Guide***.

Appendix A

Using CLOB in the Oracle eWay

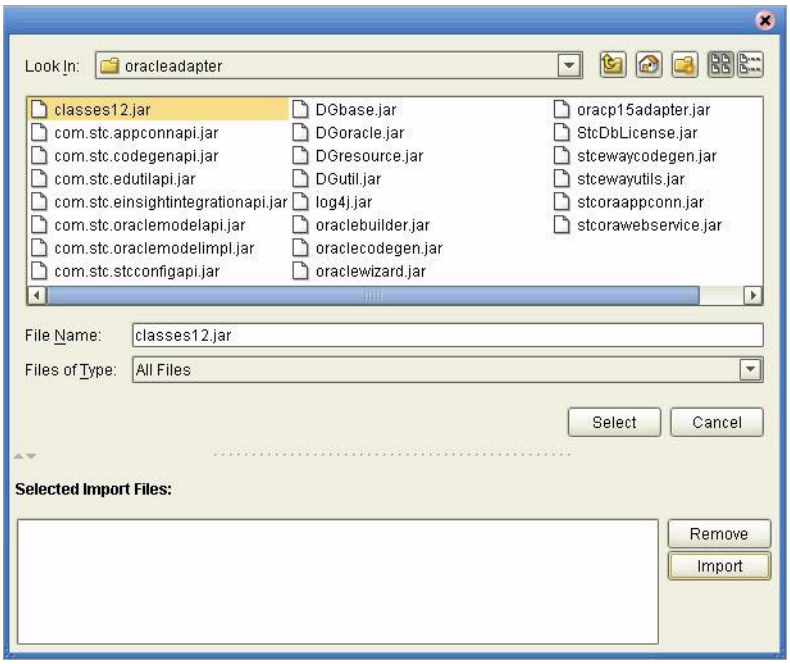
To use a CLOB in the Oracle eWay you will need to:

1 On the Enterprise Designer, select the project and right click. Select New, File. See **Figure 71**.

Figure 71 File Import Feature

2 Navigate to the classes12.jar file, <Client eDesigner>\usrdir\modules\ext\oracleadapter**classes12.jar** using the Enterprise Designer's Project File Import feature. See **Figure 72**.

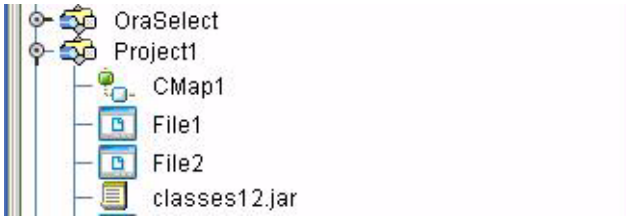
Figure 72 Importing Classes12.jar



3 Click Select.

4 Click Import. See **Figure 73**.

Figure 73 Classes12.jar in a Project

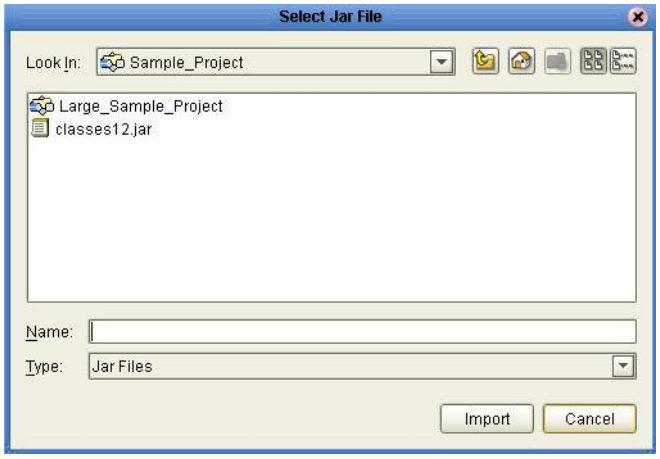


5 To load the **classes12.jar** file into your Java Collaboration, select the Import JAR File icon. On the **Add/Remove Jar Files** click **Add**. See **Figure 74**.

Figure 74 Add/Remove Jar Files

6 On the Select Jar File window, select the classes12.jar file and click Import. See **Figure 75**.

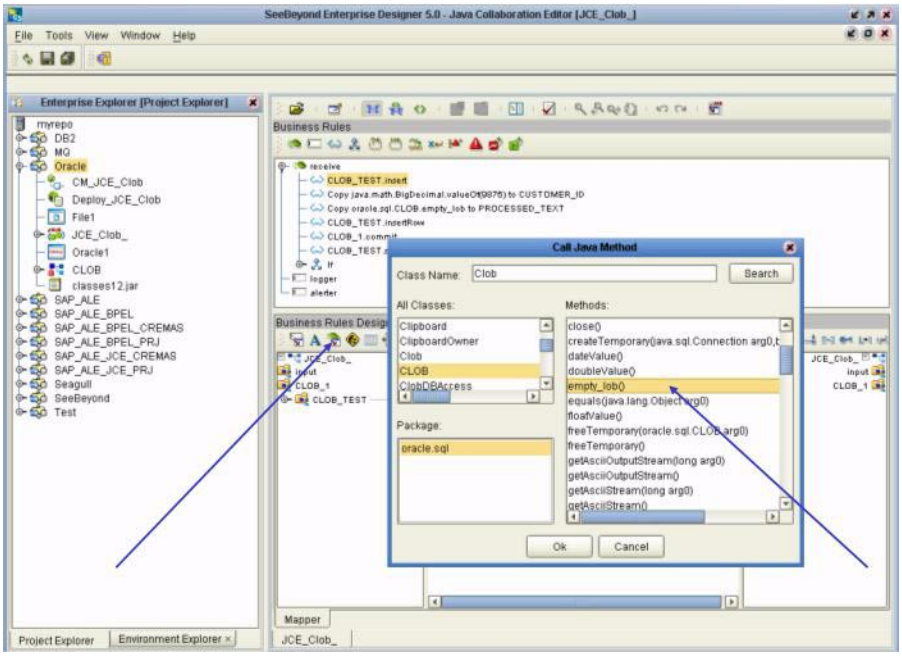
Figure 75 Select Jar File



7 On the **Add/Remove Jar Files** window, click **Close**.

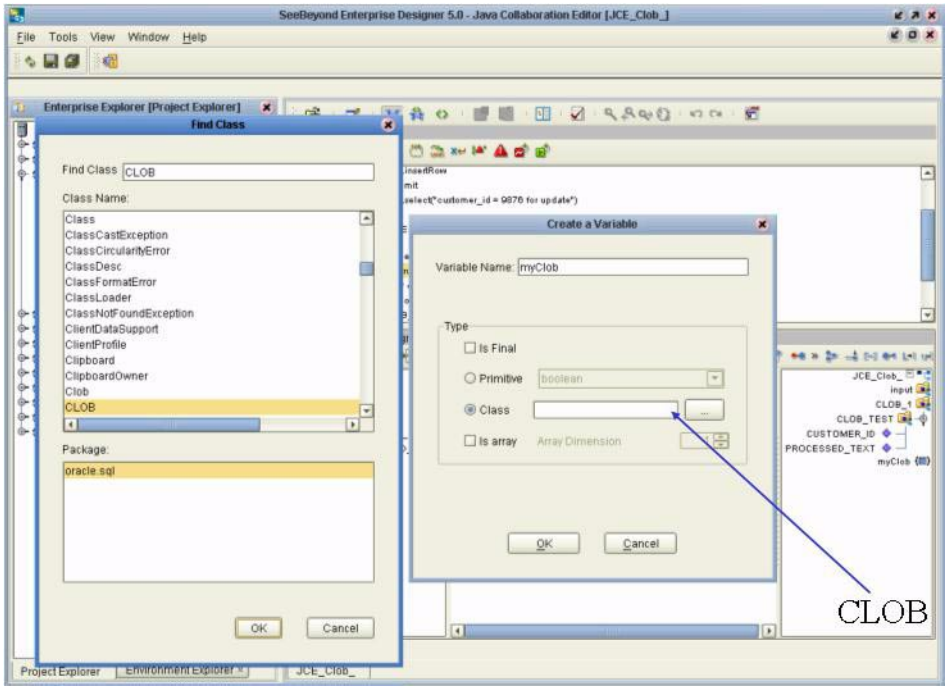
8 On the Business Rules Designer, call the CLOB method. See **Figure 76**.

Figure 76 Call the CLOB Method



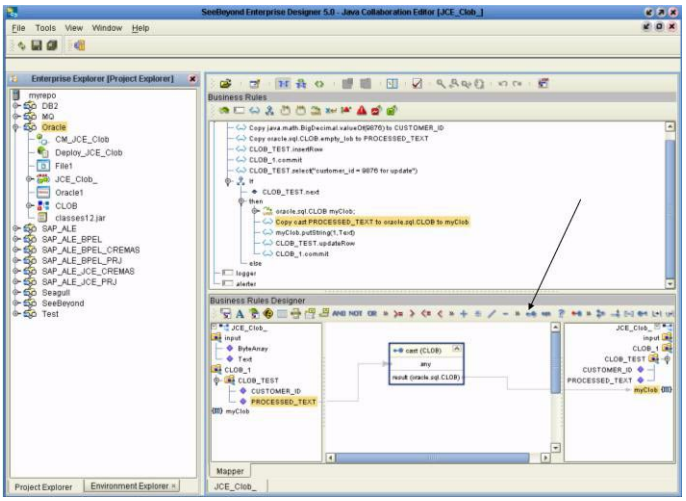
9 Create a local variable. See **Figure 77**.

Figure 77 Create a Local Variable



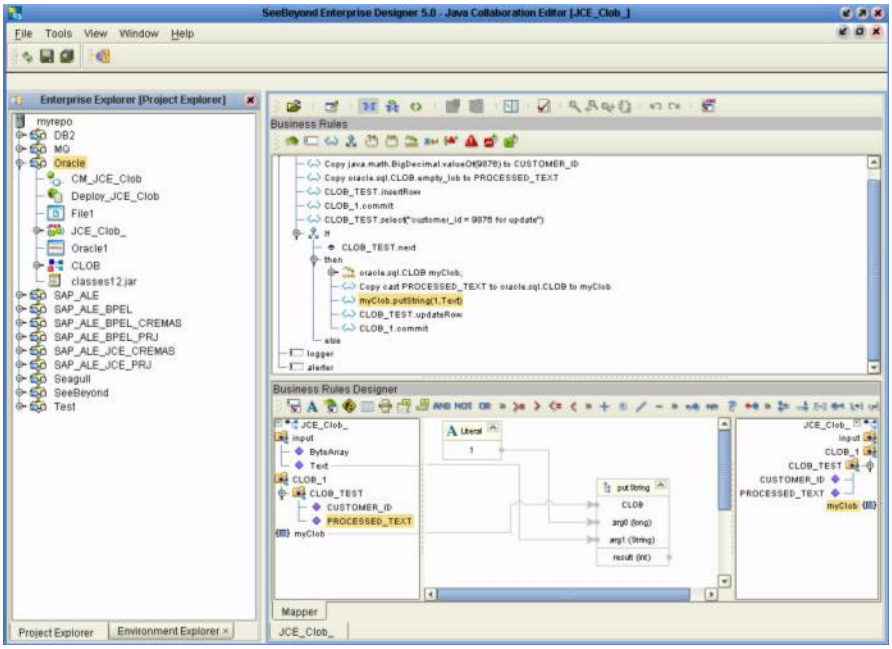
10 In the Business Rules Designer, drag the **CLOB** to the Local Variable using the Cast method. Click **Yes** to the Incompatible DataType warning. See **Figure 78**.

Figure 78 Drag CLOB to the Local Variable



11 Use the CLOB putString method assigning 1 to Arg(). See **Figure 79**.

Figure 79 CLOB putString Method



12 In the Java Collaboration Editor, the java code should look something like:

```
public void receive( com.stc.connector.appconn.file.FileTextMessage
input,cLOB.CLOBOTD CLOB_1 )
throws Throwable
    {
        //@map:Copy java.math.BigDecimal.valueOf(9876) to CUSTOMER_ID
        CLOB_1.getCLOB_TEST().setCUSTOMER_ID(
java.math.BigDecimal.valueOf( 9876 ) );
```

```
            //@map:Copy oracle.sql.CLOB.empty_lob to PROCESSED_TEXT
            CLOB_1.getCLOB_TEST().setPROCESSED_TEXT(
oracle.sql.CLOB.empty_lob() );

            //@map:CLOB_TEST.insertRow
            CLOB_1.getCLOB_TEST().insertRow();

            //@map:CLOB_1.commit
            CLOB_1.commit();

            //@map:CLOB_TEST.select("customer_id = 9876 for update")
           CLOB_1.getCLOB_TEST().select( "customer_id = 9876 for update"
);
//If
            if (CLOB_1.getCLOB_TEST().next()) {
                //@map:oracle.sql.CLOB myClob;
                oracle.sql.CLOB myClob;

                //@map:Copy cast PROCESSED_TEXT to oracle.sql.CLOB to
myClob
                myClob = (oracle.sql.CLOB)
CLOB_1.getCLOB_TEST().getPROCESSED_TEXT();

               //@map:myClob.putString(1,Text)
                myClob.putString( 1,input.getText() );

                //@map:CLOB_TEST.updateRow
                CLOB_1.getCLOB_TEST().updateRow();

                //@map:CLOB_1.commit
                CLOB_1.commit();
            }
        }
```

Index

A

Add Prepared Statements **32**

C

ClassName **11**, **19**
Connect to Database **26**

D

Database Wizard **54**
DatabaseName **15**, **18**
DataSourceName **15**, **22**
Delimiter **16**, **23**
Description **11**, **16**, **19**, **23**
driver class, JDBC **11**, **19**
DriverProperties **16**, **23**

E

Element **51**
Environment Properties
  DatabaseName **15**
  DataSourceName **15**, **22**
  Delimiter **16**, **23**
  Description **16**, **23**
  DriverProperties **16**, **23**
  Password **16**, **23**
  PortNumber **16**, **23**
  ServerName **17**, **24**
  User **17**, **24**
External System Requirements **7**

F

Field **51**

I

Inbound Environment Properties
  Database **18**
  Password **18**
  PortNumber **18**
  ServerName **18**
  User **18**
InitialPoolSize **12**, **20**

J

JDBC
  driver class **11**, **19**

L

LoginTimeOut **12**, **20**

M

MaxIdleTime **12**, **20**
MaxPoolSize **12**, **20**
MaxStatements **12**, **20**
Method **51**
MinPoolSize **13**, **21**

N

NetworkProtocol **13**, **21**

O

Object Type Definition **51**
Oracle eWay Database Wizard **25**
Oracle eWay With eInsight **36**
Outbound Properties
  ClassName **11**, **19**
  Description **11**, **19**
  InitialPoolSize **12**, **20**
  LoginTimeOut **12**, **20**
  MaxIdleTime **12**, **20**
  MaxPoolSize **12**, **20**
  MaxStatements **12**, **20**
  MinPoolSize **13**, **21**
  NetworkProtocol **13**, **21**
  PropertyCycle **13**, **21**
  RoleName **13**, **21**

P

Parameter **51**
Password **16**, **18**, **23**
PortNumber **16**, **18**, **23**
Property settings, Environment
  DatabaseName **15**
  DataSourceName **15**, **22**
  Delimiter **16**, **23**
  Description **16**, **23**
  DriverProperties **16**, **23**

Password **16**, **23**
PortNumber **16**, **23**
ServerName **17**, **24**
User **17**, **24**
Property settings, Inbound Environment
Database **18**
Password **18**
PortNumber **18**
ServerName **18**
User **18**
Property settings, Outbound
ClassName **11**, **19**
Description **11**, **19**
InitialPoolSize **12**, **20**
LoginTimeOut **12**, **20**
MaxIdleTime **12**, **20**
MaxPoolSize **12**, **20**
MaxStatements **12**, **20**
MinPoolSize **13**, **21**
NetworkProtocol **13**, **21**
PropertyCycle **13**, **21**
RoleName **13**, **21**
PropertyCycle **13**, **21**

R

RoleName **13**, **21**

S

Select Database Objects **27**
Select Procedures **31**
Select Table/Views **27**
Select Wizard Type **25**
ServerName **17**, **18**, **24**
Specify the OTD Name **34**
Supported Operating Systems **6**, **67**

U

User **17**, **18**, **24**