SeeBeyond ICAN Suite

HP NonStop Server SQL eWay Intelligent Adapter User's Guide

Release 5.0.1

Version 20040524160427.

Contents

Appendix A

Chapter 1

Introducing the HP NonStop SQL eWay

This document describes how to install and configure the HP NonStop SQL eWay Intelligent Adapter.

This chapter includes

1.1 Overview

The HP NonStop SQL eWay enables the eGate system to exchange data with external HP NonStop SQL databases. This document describes how to install and configure the HP NonStop SQL eWay.

1.2 Supported Operating Systems

The HP NonStop SQL eWay is available on the following operating systems:

- HP NonStop Server, G06.22.

1.3 System Requirements

To enable the Enterprise Designer's editors to communicate with the external system, the following items must be installed on any host machines running the Graphical User Interface (GUI) editors:

- Windows Server 2003, Windows XP SP1a, and Windows 2000 SP3. This is a requirement for the Enterprise Designer.
- Microsoft Internet Explorer 6.0 SP1 or later.
- TCP/IP network connection.

- ODBC DataSourceName configured and named **NSODBCMX**.
- FTP or Telnet capabilities.

The HP NonStop SQL eWay installation program installs the DataDirect Sequelink 1.0 for HP NonStop SQL driver required to connect to the external HP NonStop SQL database.

1.4 External System Requirements

The HP NonStop SQL eWay supports the following external systems:

- HP NonStop SQL 1.8.5 AAP
- JDBC/MX T1225 V2.1 AAD required for runtime.
- HP NonStop SQL/MP G07

Note: *For a full list of requirements, see the SeeBeyond ICAN Suite Installation Guide.*

1.5 HP NonStop Server Requirements

Refer to the *SeeBeyond ICAN Suite Installation Guide* for a list of HP NonStop Server requirements.

Chapter 2

Installing the HP NonStop SQL eWay

This chapter describes how to install the HP NonStop SQL eWay.

This Chapter Includes

2.1 Installing the HP NonStop SQL eWay from a Windows System

The installation process includes installing the following components:

- Installing the Repository.
- Upload products to the Repository.
- Download components (including Enterprise Designer and Logical Host).
- View product information home pages.

Please review the *SeeBeyond ICAN Suite Installation Guide* for complete installation instructions.

To install the HP NonStop SQL eWay from a Windows system

1. You must have Administrator privileges to install this eWay.
2. Exit all Windows programs before running the setup program, including any anti-virus applications.
3. Insert the eGate installation CD-ROM into the CD-ROM drive.
4. Following the installation instructions provided in the *SeeBeyond ICAN Suite Installation Guide*, install the following components:

- Install the Repository.
- From the Enterprise Manager, select the **NSSQLeWay.sar** file. To install and upload this file, see the chapter on uploading files to the Repository in the *SeeBeyond ICAN Suite Installation Guide*.
- Install the eGate Enterprise Designer as instructed

Chapter 3

Properties of the HP NonStop SQL eWay

This chapter describes the properties of the HP NonStop SQL eWay. The properties of the HP NonStop SQL eWay must be configured in both the Connectivity Map and in the Environment.

This chapter includes

3.1 Setting the eWay Properties in the Connectivity Map

The HP NonStop SQL eWay has three possible modes in the Connectivity Map. Each type has a unique set of properties to control the behavior of the eWay. The three types of HP NonStop SQL eWays are:

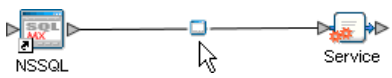
- **Inbound NSSQL eWay**: This eWay is used to poll external databases and retrieve data into ICAN Projects. For more information, see **"Setting the Properties of the Inbound NSSQL eWay" on page 10**.
- **Outbound NSSQL eWay**: This eWay is used to publish data from ICAN Projects into the external database. For more information, see **"Setting the Properties of the Outbound NSSQL eWay" on page 11**.
- **NSSQL eWay**: This outbound eWay is provided for backward compatibility with ICAN Projects created with version 5.0 of the HP NonStop Server SQL eWay. For more information, see **"Setting the Properties of the NSSQL eWay" on page 13**.

Use the eWay properties to configure each eWay.

To set an eWay's properties in the Configuration Map

1. In the Project Explorer, create your Connectivity Map with all the components your Project requires.
2. Double click the properties icon of the eWay you want to configure.

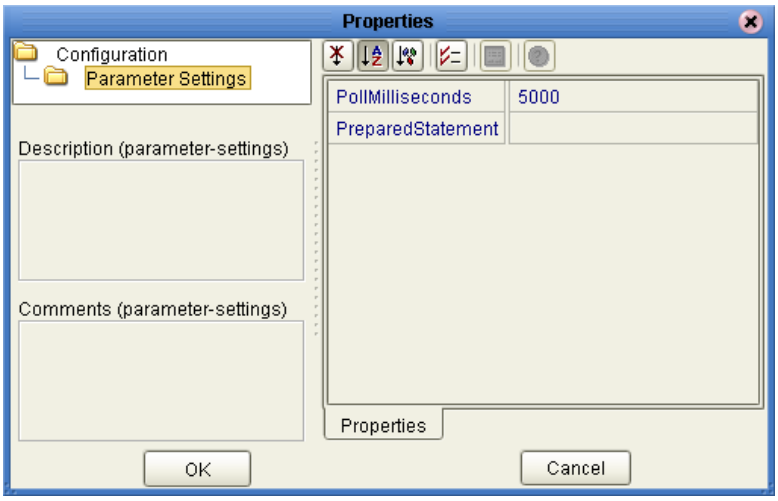
Figure 1 eWay Properties Icon



3 Enter the property settings as described in the following sections.

3.1.1. Setting the Properties of the Inbound NSSQL eWay

Figure 2 Inbound NSSQL eWay Properties



PollMilliseconds

Description

Polling interval in milliseconds.

Required Values

A valid numeric value. The default is 5000.

PreparedStatement

Description

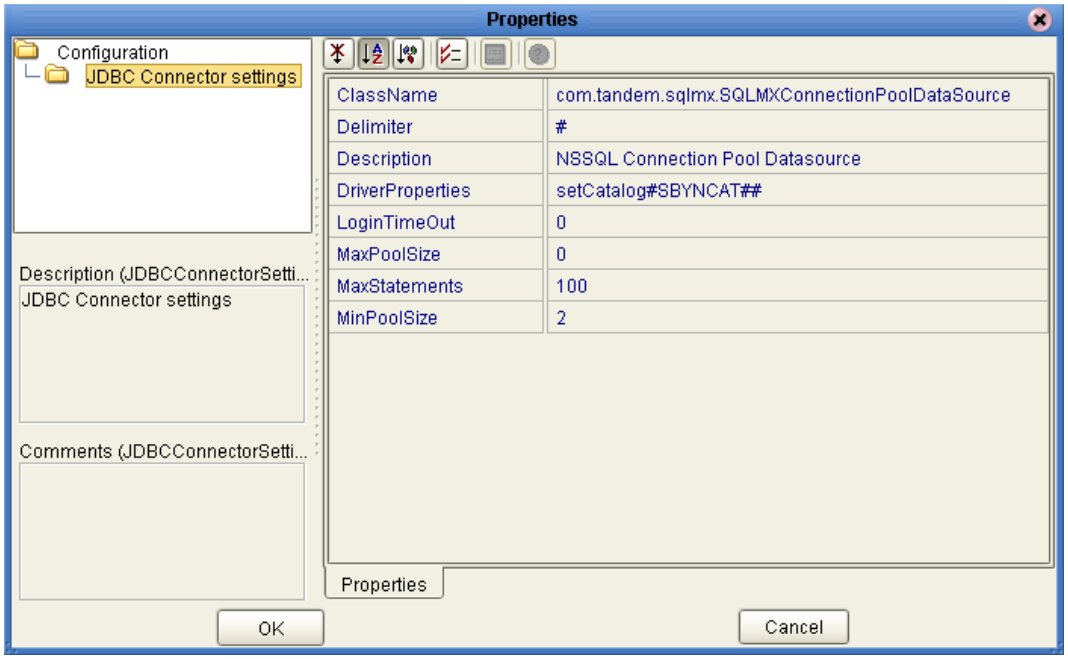
A prepared statement used for polling the database.

Required Values

The prepared statement must be the same prepared statement you created using the Database OTD Wizard. Only "SELECT" statements are allowed. Additionally, no place holders should be specified. There should not be any "?" values in the prepared statement.

3.1.2. Setting the Properties of the Outbound NSSQL eWay

Figure 3 Outbound NSSQL eWay Properties



ClassName

Description

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

If you use any driver other than the JdbcMx driver specified as the default, you must also specify the length of time to wait for the connection to be established (see **"LoginTimeOut" on page 12**).

Required Values

A valid class name. The default value is **com.tandem.sqlmx.SQLMXConnectionPoolDataSource**.

Delimiter

Description

Defines the delimiter character to be used in the **DriverPorperties** property. (See **"DriverProperties" on page 12**).

Required Values

A valid ASCII character. The default is **#**.

Description

Description

Specifies the description for the database.

Required Values

A valid string. The default is **NSSQL Connection Pool Datasource**.

DriverProperties

Description

Specifies the properties for the JDBC driver being used with this eWay. Use this in the case where a DataSource implementation needs to execute additional methods to establish and maintain a connection.

Required Values

Any valid method name(s) and parameter(s). You must use the Delimiter you specified in **"Delimiter" on page 11**.

The format for the method names and parameters is:

```
methodName1#param1##param2##...paramN#methodName2##param1##param2
```

The default value is **setCatalog#SBYNCAT##**.

LoginTimeOut

Description

Specifies the maximum length of time in seconds to wait while establishing a connection with the external database. This value has no effect if you use the default JdbcMx driver (see **"ClassName" on page 11**).

Required Values

A valid number. The default is **0**.

MaxPoolSize

Description

Defines the maximum connection pool size. To specify an unlimited connection pool, use a value of **0**. This option is not required.

Required Values

A valid number. The default is **0** (unlimited).

MaxStatements

Description

Specifies the maximum number of prepared statements to be cached. This option is not required.

Required Values

A valid number. The default is **100**.

MinPoolSize

Description

Specifies the minimum number of connections in the connection pool. This option is not required.

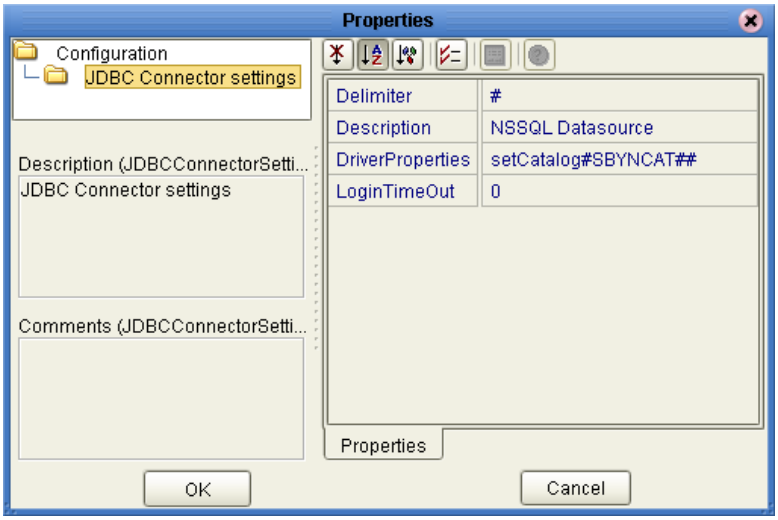
Required Values

A valid number. The default is **2**.

3.1.3. Setting the Properties of the NSSQL eWay

This outbound eWay is provided for backward compatibility with ICAN Projects created with version 5.0 of the HP NonStop Server SQL eWay.

Figure 4 NSSQL eWay Properties



Delimiter

Description

Defines the delimiter character to be used in the **DriverProperties** property. (See **"DriverProperties" on page 14**).

Required Values

A valid ASCII character. The default is **#**.

Description

Description

Specifies the description for the database.

Required Values

A valid string. The default is **NSSQL Datasource**.

DriverProperties

Description

Specifies the properties for the JDBC driver being used with this eWay. Use this in the case where a DataSource implementation needs to execute additional methods to establish and maintain a connection.

Required Values

Any valid method name(s) and parameter(s). You must use the Delimiter you specified in **"Delimiter" on page 13**.

The format for the method names and parameters is:

```
methodName1#param1##param2##...paramN#methodName2##param1##param2
```

The default value is **n##**.

LoginTimeOut

Specifies the maximum length of time in seconds to wait while establishing a connection with the external database.

Required Values

A valid number. The default is **0**.

3.2 Setting the External Properties in the Environment

The HP NonStop SQL eWay has three possible modes in the Environment Explorer. Each type has a unique set of properties to control the behavior of the eWay. The three types of HP NonStop SQL eWay Externals are:

- **Inbound NSSQL eWay**: This eWay is used to poll external databases and retrieve data into ICAN Projects. For more information, see **"Setting the Properties of the Inbound NSSQL eWay External" on page 16**.
- **Outbound NSSQL eWay**: This eWay is used to publish data from ICAN Projects into the external database. For more information, see **"Setting the Properties of the Outbound NSSQL eWay External" on page 17**.
- **NSSQL eWay**: This outbound eWay is provided for backward compatibility with ICAN Projects created with version 5.0 of the HP NonStop Server SQL eWay. For

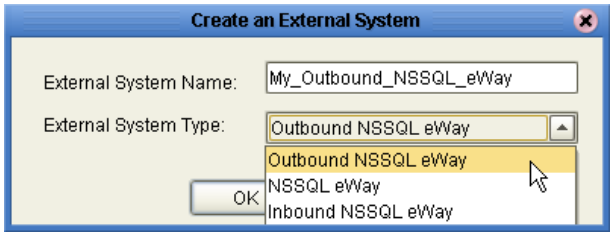
more information, see **"Setting the Properties of the NSSQL eWay External" on page 19**.

Use the Environment Explorer to configure the properties of the HP NonStop SQL eWay External.

To set the properties of the eWay External

1. In the Environment Explorer, create a new Environment (if necessary).
2. Right click your environment and click **New NSSQL External System**.
3. Type the name of the External System and select one of the three External System Types:
   - **Outbound NSSQL eWay**
   - **NSSQL eWay**
   - **Inbound NSSQL eWay**

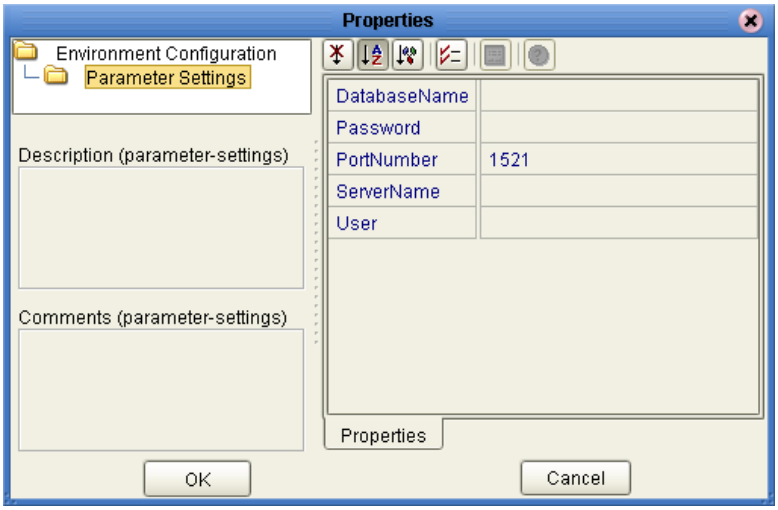
Figure 5 Create an External System



4. Click **OK** to create the External System.
5. Right click the newly created External System and click **Properties**.
6. Enter the property settings as described in the following sections.

3.2.1. Setting the Properties of the Inbound NSSQL eWay External

Figure 6 Inbound NSSQL eWay External Properties



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 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.

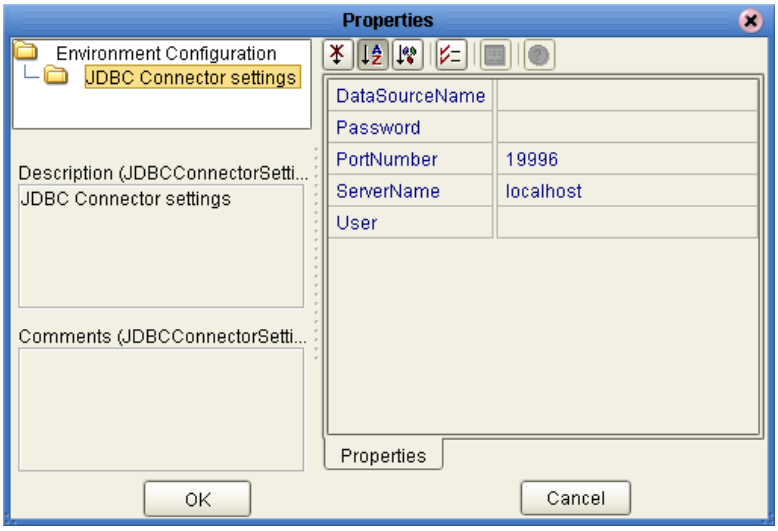
Required Values

Any valid string.

3.2.2. Setting the Properties of the Outbound NSSQL eWay External

This outbound eWay is provided for backward compatibility with ICAN Projects created with version 5.0 of the HP NonStop Server SQL eWay.

Figure 7 Outbound NSSQL eWay External Properties



DataSourceName

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 19996

ServerName

Description

Specifies the host name of the external database server.

Required Values

Any valid string.

User

Description

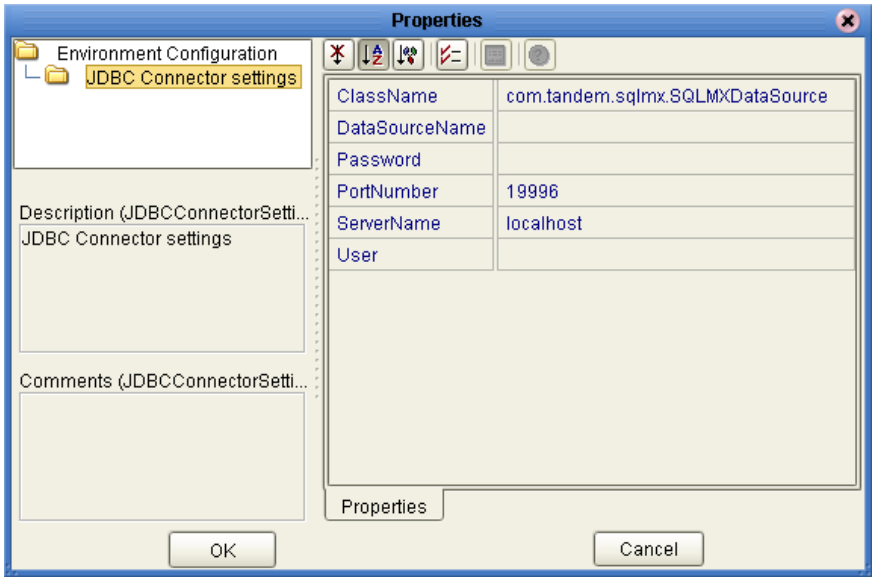
Specifies the user name the eWay uses to connect to the external database.

Required Values

Any valid string.

3.2.3. Setting the Properties of the NSSQL eWay External

Figure 8 NSSQL eWay External Properties



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 value is **com.tandem.sqlmx.SQLMXDataSource**.

DataSourceName

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 external database.

Required Values

Any valid string.

Chapter 4

Using the HP NonStop SQL eWay Database Wizard

This chapter describes how to use the HP NonStop SQL eWay Database Wizard to build OTDs.

This Chapter Includes

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 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.*

Note: *Prior to selecting tables using this wizard you must have created an MP Alias for each table. For more information regarding an MP Alias for tables, see your NonStop SQL MX documentation.*

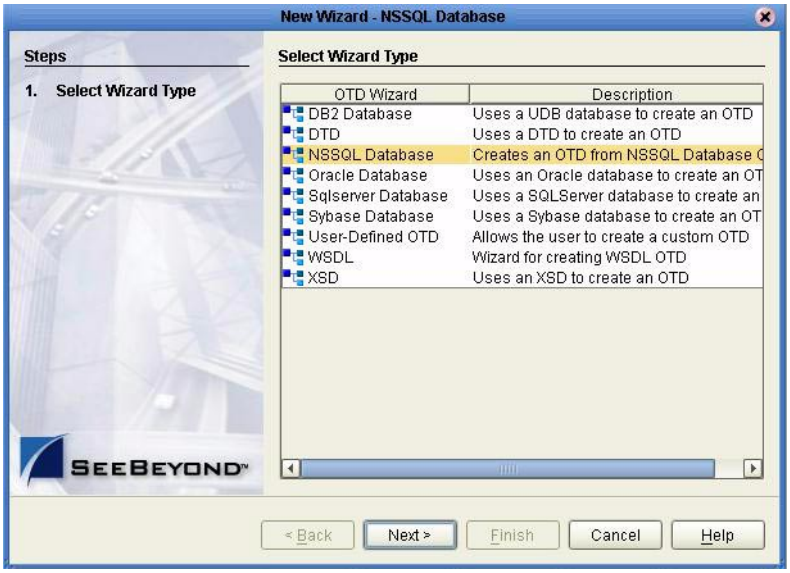
To create a new OTD using the Database Wizard

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 **HPSQL Database** and click **Next**. See Figure 9.

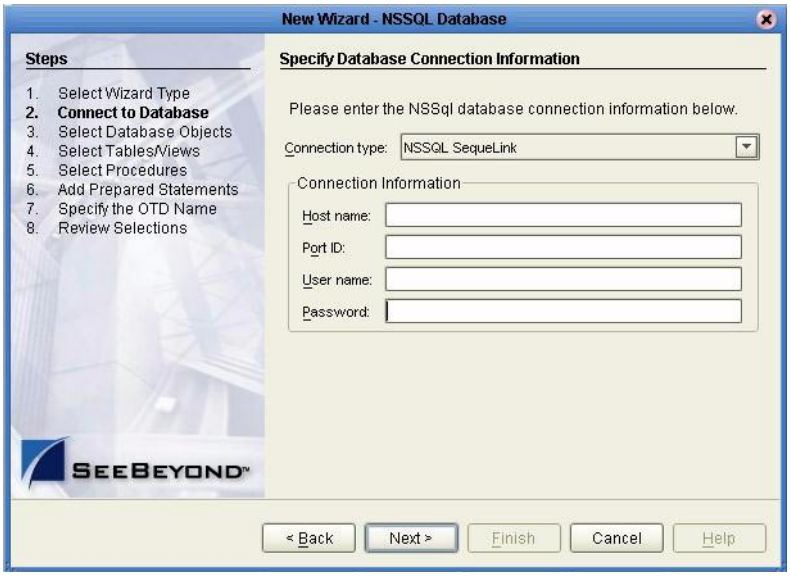
Figure 9 OTD Wizard Selection



Connect to Database

3 Specify the connection information for your database including your **UserName** and **Password** and click **Next**. See Figure 10.

Figure 10 Database Connection Information

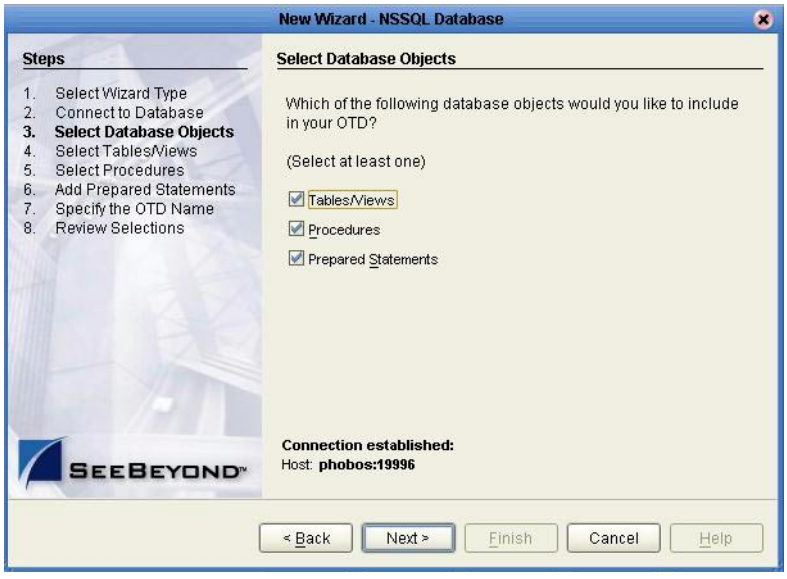


Select Database Objects

1 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. See **Figure 11**.

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

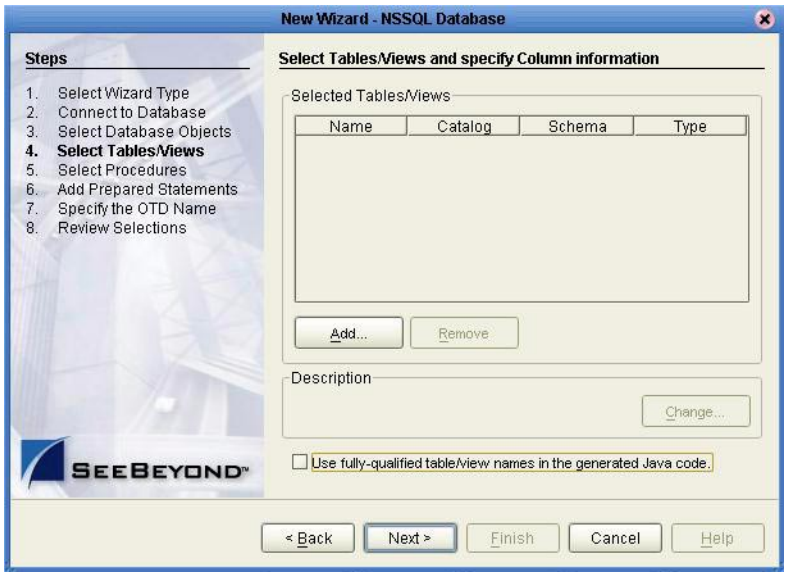
Figure 11 Select Database Objects



Select Table/Views

1 In the **Select Tables/Views** window, click **Add**. See **Figure 12**.

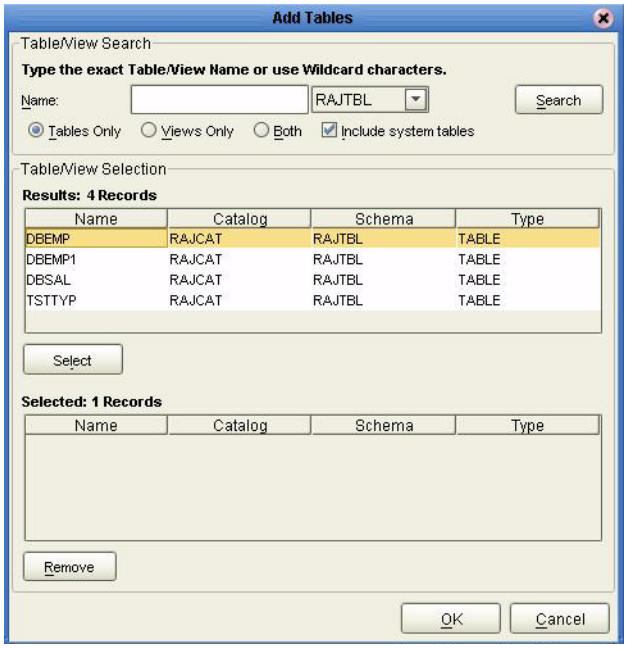
Figure 12 Select Tables/Views



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 13**. You can search for **Table/View Names** by entering a table name.

Figure 13 Database Wizard - All Schemes

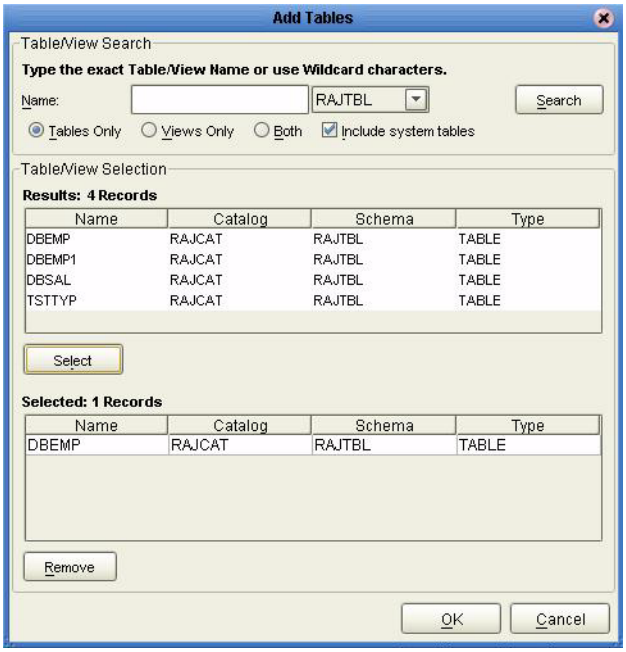


4 Select the table of choice and click **OK**.

Note: *Prior to selecting tables using this wizard you must have created an MP Alias for each table. For more information regarding an MP Alias for tables, see your NonStop SQL MX documentation.*

The table selected is added to the **Selected Tables/Views** window. See **Figure 14**.

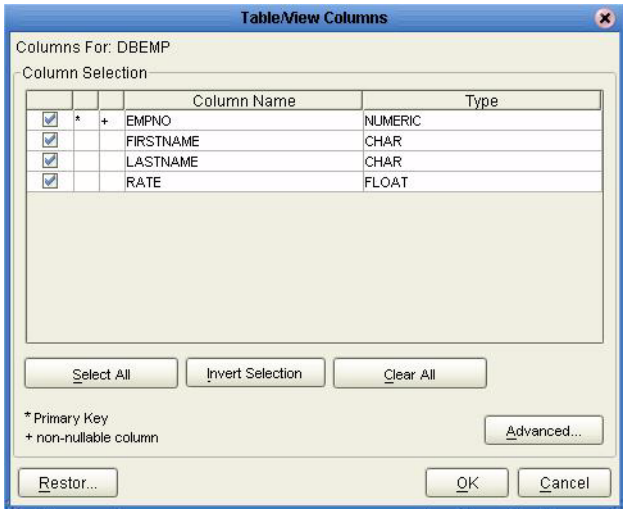
Figure 14 Selected Tables/Views window with a table selected



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.

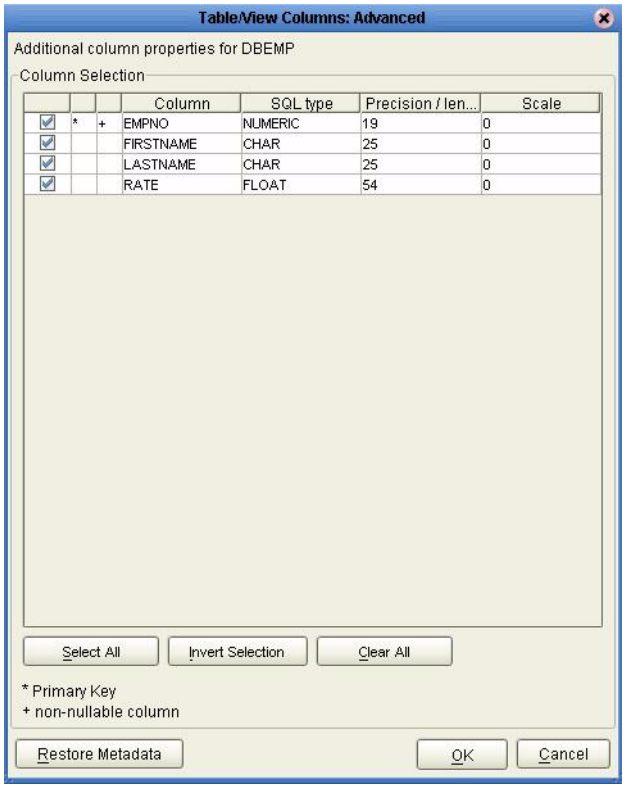
6 In the **Table/View Columns** window, you can select or deselect your table columns. You can also change the data type for each table by highlighting the data type and selecting a different one from the drop down list. If you would like to change any of the tables columns, click **Change**. See **Figure 15**.

Figure 15 Table/View Columns



7 Click **Advanced** to change the data type, precision/length, or scale. Once you have finished your table choices, click **OK**. In general, you will not need to change these settings. See **Figure 16**.

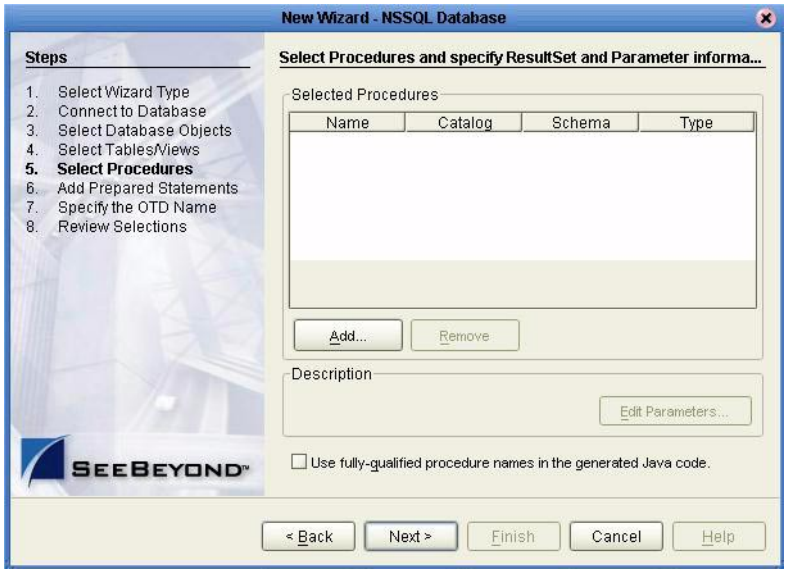
Figure 16 Table/View Columns — Advanced



Select Procedures

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

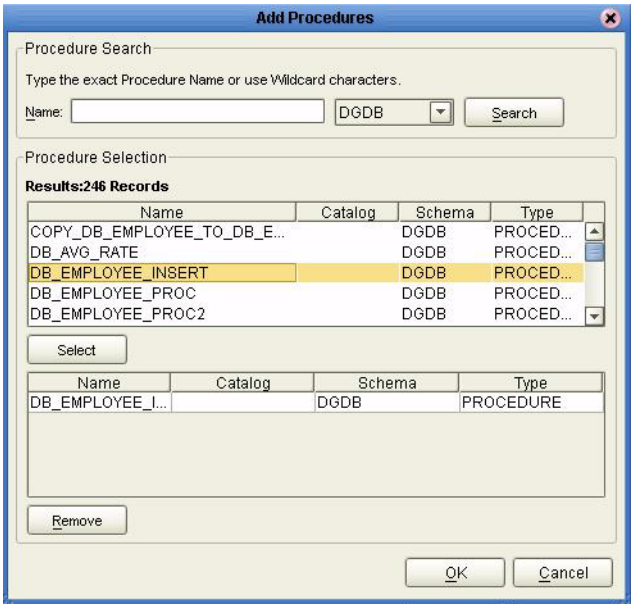
Figure 17 Select Procedures and specify ResultSet and Parameter Information



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.

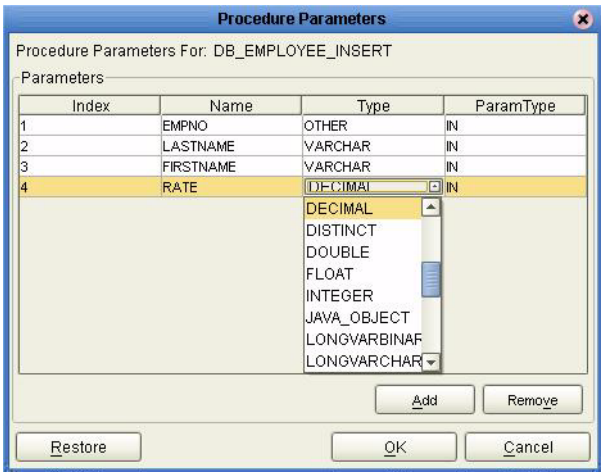
3 In the resulting **Procedure Selection** list box, select a Procedure. Click **OK**.

Figure 18 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 19**.

Figure 19 Procedure Parameters



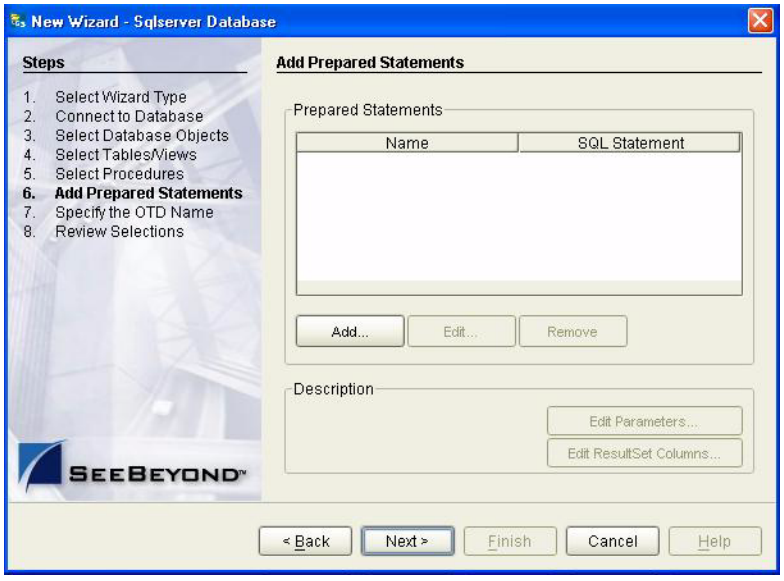
5 To restore the data type, click **Restore**. When finished, click **OK**.

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 20 Prepared Statement

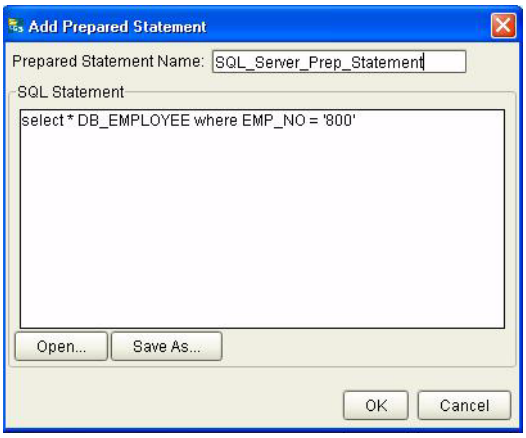


2 Enter the name of a Prepared Statement or create a SQL statement by clicking in the SQL Statement window. When finished creating the statement, click **Save As** giving

the statement a name. This name will appear as a node in the OTD. Click **OK**. See **Figure 21**.

Note: *When creating a SQL statement, you must use fully qualified names; for example <Catalog.Schema.Table Name> would be RAJCAT.RAJTBL.DBEMP. These names can be found under the corresponding headings on the Select Tables/Views window. See* **Figure 14 on page 26**

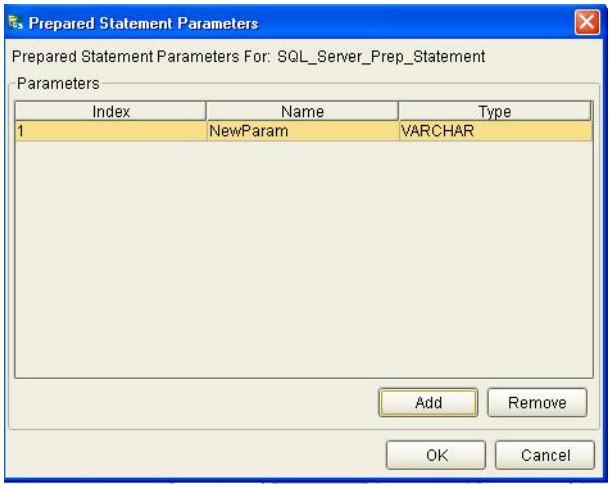
Figure 21 Prepared SQL Statement



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 22**.

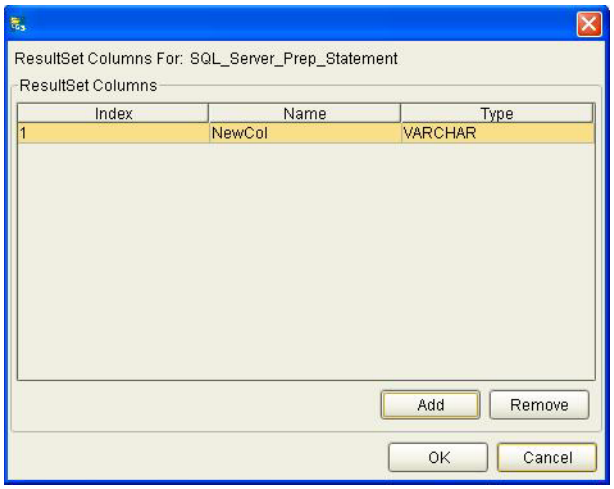
Figure 22 Edit the Prepared Statement Parameters



5 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 23**.

Figure 23 ResultSet Columns



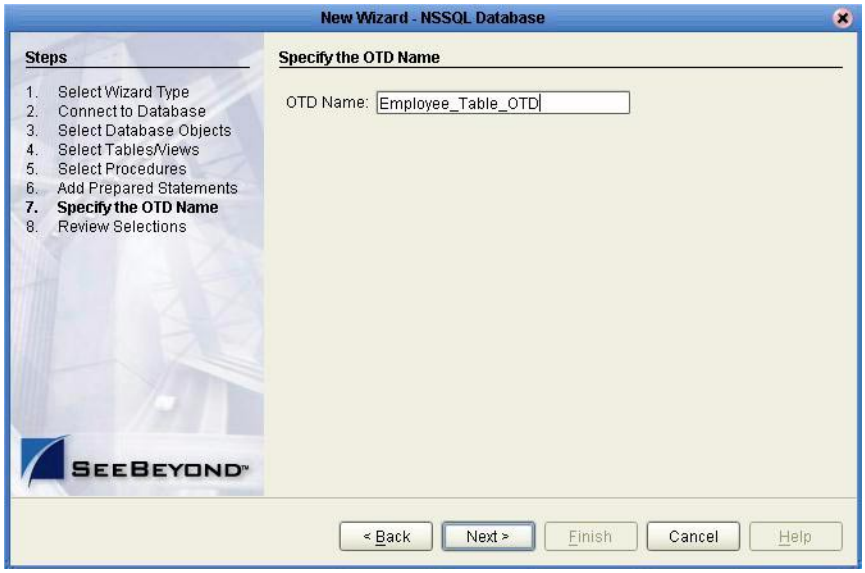
6 Click **OK** to return to the **Database Wizard - Prepared Statements** window.

7 Add additional prepared statements or click **Next** to continue.

Specify the OTD Name

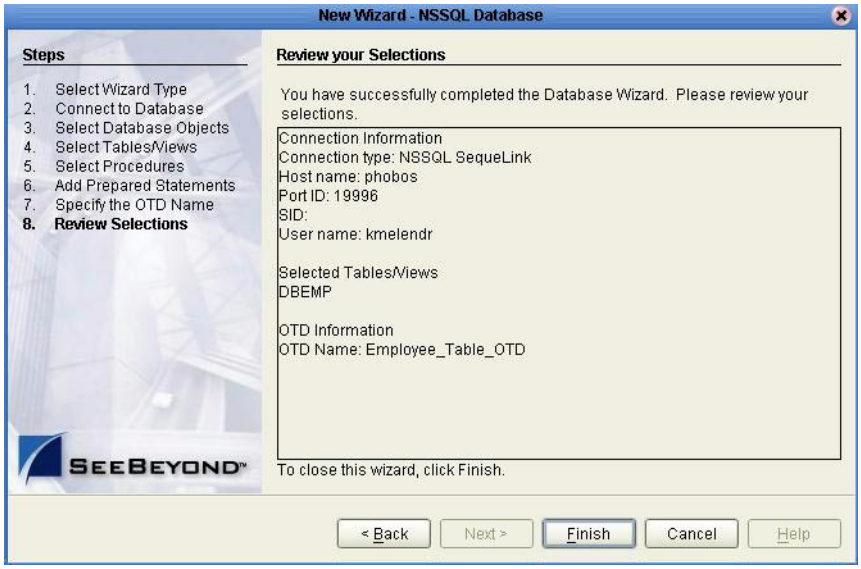
1 Enter a name for the OTD. The OTD contains the selected tables and the package name of the generated classes. See **Figure 24**.

Figure 24 Specific OTD Name



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 25**.

Figure 25 Database Wizard - Summary



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

Chapter 5

Building an eWay Project

This chapter discusses how to build an HP NonStop SQL eWay project in a production environment.

This chapter includes

5.1 Building a Project

To meet your specific business challenge, you first need to identify your business needs and then create a Project that satisfies those needs. To create a Project, you need to identify which of the following components to include to control the flow of data:

- Services—A component that uses Collaboration Definitions to define how data should be routed between Project components. Collaborations are used when data translation is required.
- Topics—A component that is a shareable object conforming to the publish-and subscribe (pub/sub) messaging domain. This means that one publisher broadcasts messages to multiple subscribers, ensuring that all subscribers receive a message.
- Queues—A component that is a sharable object conforming to the point-to-point messaging domain. This means that one publisher sends a message to exactly one subscriber.
- External Applications—A component that connects to an external file. Different external applications allow eGate to connect with various types of databases and servers.

5.2 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***.

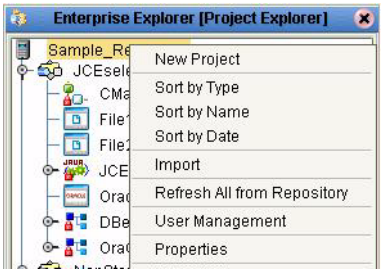
5.3 Creating the Project

Before beginning a project, you need to create a Project Folder. This folder will be comprised of all the components you identified as being part of your project or business process. For information on how to create a Project folder, please see the *eGate Tutorial*.

Importing the Sample Project

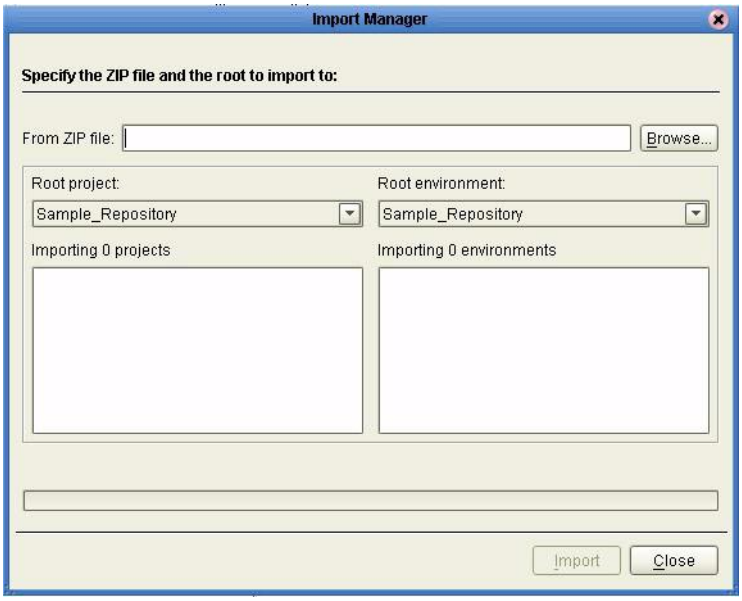
1 On the Enterprise Explorer highlight the repository and right click. Select **Import**. See **Figure 26**.

Figure 26 Importing the sample project



2 In the **Import Manager** window, **From ZIP file** browse to the location of the sample folder and select the following .zip file **HPNonStop_SQL_eWay_Sample.zip** and click **Import**. See **Figure 27**.

Figure 27 Select the project file



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

5.3.1. Working with the Sample Project in eGate

This sample project selects the EMPNO, LASTNAME, FIRSTNAME, and the RATE columns from the table DBEmployee and publishes the record to an output file.

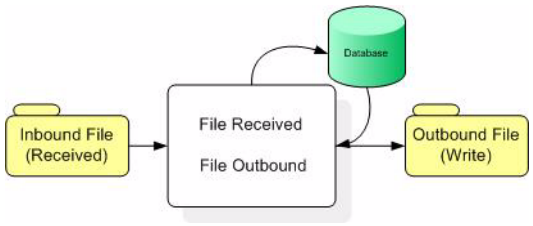
The data used for this projects is within a table called DBEmployee. The table contains the following columns:

Table 1 Sample project data
Table 2

Column Name	Mapping	Data Type	Data Length
EMPNO	Empno	largeint	
LASTNAME	Lastname	char2	25
FIRSTNAME	Firstname	char2	25
RATE	Rate	float	

The sample project consists of topic message containing data that is passed into a collaboration and out to the database.

Figure 28 Database project flow



To work with the sample project, follow the instructions given in the *eGate Tutorial*.

5.3.2. Configuring the eWays

The sample uses a topic as well as an HP NonStop SQL eWay. To configure the sample project, use the following information. For additional information on the properties of the NonStop SQL eWay, see **"Setting the eWay Properties in the Connectivity Map" on page 9**.

To configure the HPNonStop eWay:

1. On the Connectivity Map canvas, double click the eWay icon located between the **Service1** and **NSSQL1**.
2. On the resulting **Templates** window, select **Outbound NSSQL eWay** and click **OK**.
3. On the **Properties** window, enter the appropriate configurations for the Outbound NSSQL eWay. See 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**.

5.3.3. Creating the External Environment

To review the components of the Sample project, there is an Outbound HP NonStop SQL eWay, and a topic.

To create the external environment for the Sample project:

1. On the Environment Explorer, highlight and right-click the HP NonStop SQL eWay profile. Select **Properties**.
2. Enter the configuration information required for your Outbound HP NonStop SQL eWay. See **"Setting the External Properties in the Environment" on page 14**.

5.3.4 Deploying a Project

To deploy a project, please see the "*eGate Integrators User's Guide*."

5.3.5. Running the Sample

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

Once the process has completed, the database in the target directory configured in the Outbound HP NonStop eWay will update the database.

5.4 Common DataType Conversions

Figure 29 The HP NonStop SQL eWay Datatype Conversions

SQL Data Type	Methods to Use
LargeInt	**java.math.BigDecimal**
Char(n)	**java.lang.String**
Date	**java.sql.Date**
Decimal(m,n)	**java.math.BigDecimal**
Float	**double**
Int	**int**
Numeric(n,m)	**java.math.BigDecimal**
* see NonStop Server for Java Programmer's Reference(426947-003)	**java.lang.String**
SmallInt	**short**
Time	**java.sql.Time**
TimeStamp	**java.sql.Timestamp**
Varchar(n)	**java.lang.String**

5.5 Using OTDs with Tables, Views, and Prepared Statements

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

5.5.1 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 ScrollTypeForwardOnly. 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. 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:

```
package SelectSales;
public class Select
{

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,db_employee.Db_employeeOTD
db_employee_1,employeedb.Db_employee employeedb_db_employee_1 )
throws Throwable
  {
//@map:Db_employee.select(Text)
   db_employee_1.getDb_employee().select( input.getText() );
//while
  while (db_employee_1.getDb_employee().next()) {
//@map:Copy EMP_NO to Employee_no
   employeedb_db_employee_1.setEmployee_no(
java.lang.Integer.toString(
db_employee_1.getDb_employee().getEMP_NO() ) );
//@map:Copy LAST_NAME to Employee_lname
   employeedb_db_employee_1.setEmployee_lname(
db_employee_1.getDb_employee().getLAST_NAME() );
//@map:Copy FIRST_NAME to Employee_fname
   employeedb_db_employee_1.setEmployee_fname(
db_employee_1.getDb_employee().getFIRST_NAME() );
//@map:Copy RATE to Rate
   employeedb_db_employee_1.setRate( java.lang.Double.toString(
db_employee_1.getDb_employee().getRATE() ) );
//@map:Copy LAST_UPDATE to Update_date
   employeedb_db_employee_1.setUpdate_date(
db_employee_1.getDb_employee().getLAST_UPDATE().toString() );
}
//@map:Copy employeedb_db_employee_1.marshalToString to Text
   FileClient_1.setText(
   employeedb_db_employee_1.marshalToString() );
//@map:FileClient_1.write
```

```
        FileClient_1.write();
      }

    }
```

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 selectStream Method

The selectStream method of the NSSQL eWay is a limited implementation of the SQL/MX Pub/Sub functionality. The select Stream method performs continuous selects on the any new data inserted into the table. Any new data selected by the selectStream is then subsequently deleted automatically from the table.

The selectStream method treats a SQL table as a work queue whereby all existing and any new rows inserted into the table are returned in the subsequent next() method call and any rows returned by the selectStream are removed automatically from the table as one integral operation. A time-out feature is also available allowing you to programmatically determine if your program will continue receiving rows from the table after a period of inactivity. The selectStream method relies on the NS SQL/MX advanced queuing feature.

```
try {
//EMPLOYEE.selectStream("")
  empTables_1.getEMPLOYEE().selectStream( "" );
...
//while EMPLOYEE.next
  while (empTables_1.getEMPLOYEE().next()) {
// Do work for each row
}
  } catch ( java.sql.SQLException sqlExp ) {
if (sqlExp.getErrorCode() == -8006) {
// Timeout
} else {
logger.info("....Timeout" + sqlExp.getErrorCode())
                }
...
}
```

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()**.

```
//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();

//Copy "Update completed" to Text
  FileClient_1.setText( "Update completed" );

//FileClient_1.write
  FileClient_1.write();
}
```

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(
  employeedb_with_top_db_employee_1.getEMP_NO() ).concat( "'" ) );
}
```

5.5.2 Prepared Statement

A Prepared Statement OTD represents a SQL statement that has been compiled. Fields in the OTD correspond to the input values that users need to provide.

Prepared statements can be used to perform insert, update, delete and query operations. A prepared statement uses a question mark (?) as a place holder for input. For example:

```
insert into EMP_TAB(Age, Name, Dept No) value(?, ?, ?)
```

To execute a prepared statement, set the input parameters and call **executeUpdate()** and specify the input values if any.

```
getPrepStatement().getPreparedStatementTest().setAge(23);
getPrepStatement().getPreparedStatementTest().setName("Peter Pan");
getPrepStatement().getPreparedStatementTest().setDeptNo(6);
getPrepStatement().getPreparedStatementTest().executeUpdate();
```

5.6 Deploying a Project to an HP NonStop Server

To deploy a project on an HP NonStop Server, please see the *"eGate Integrator User's Guide"*.

Appendix A

Known Limitations

The JDBC Driver for SQL/MX conforms to the Sun Microsystems JDBC 2.1 API specification. This appendix describes the JDBC methods that are not supported. The JDBC features that conform to the specification are not described in this appendix.

This chapter includes

1.1 Overview

The JDBC Driver for SQL/MX conforms to the Sun Microsystems JDBC 2.1 API specification. This appendix describes the JDBC methods that are not supported. The JDBC features that conform to the specification are not described in this appendix.

For additional information on the following non-supported packages and JDBC/MX compliance in general, see the following documents available from HP.

- **SQL/MX Reference Manual**
- **JDBC Driver for SQL/MX Programmer's Reference (ver 2.1)**

The selectStream Method

The selectStream functionality of the NonStop SQL eWay is a limited implementation of the SQL/MX Pub/Sub functionality. The **selectStream** functionality performs continuous selects on any new data inserted into the table and any data selected by the **selectStream** is subsequently deleted from the table.

The **selectStream** method views a SQL table as a work queue whereby all existing and any new rows inserted into the table are returned in the subsequent **next()** method call automatically removing any rows returned by the **selectStream** from the table as one integral operation. A time-out feature allows you to programmatically decide whether to continue receiving rows from the table after a period of inactivity. The **selectStream** method relies on the NonStop SQL/MX advanced queuing feature.

1.2 Non-Implemented Interfaces

The following interfaces in the **java.sql** package are not implemented in the JDBC Driver.

- **java.sql.Array**
- **java.sql.Blob**
- **java.sql.CallableStatement**
- **java.sql.Clob**
- **java.sql.Ref**
- **java.sql.SQLData**
- **java.sql.SQLInput**
- **java.sql.SQLOutput**
- **java.sql.Struct**

Note: *This eWay supports only those data types handled by the JDBC driver for NonStop SQL/MX.*

Index