

IBM WEBSHERE ADAPTER FOR JDBC V7.0 – LAB EXERCISE

JDBC outbound lab – Hierarchy business objects

What this exercise is about	1
Lab requirements	1
What you should be able to do	1
Introduction	2
Exercise instructions	2
Part 1: Create the JDBCTEST database using Derby.....	4
Part 2: Set up the development environment	6
Part 3: Create the JDBC Outbound application	7
Part 4: Create J2C authentication alias and configure data sources.....	21
Part 5: Test the application using the WebSphere test environment and component test.....	25
What you did in this exercise	33
Task: Adding remote server to the WebSphere Integration Developer test environment	34

What this exercise is about

The objective of this lab is to provide you with an understanding of the WebSphere Adapter for JDBC and outbound request processing.

Lab requirements

- WebSphere Integration Developer V7.0 installed
- WebSphere Process Server V7.0 test environment installed
- WebSphere Adapter for JDBC V7.0 installed
- Sample code in the directory C:\Labfiles70\JDBC (Windows) or /tmp/LabFiles70/JDBC (Linux)

What you should be able to do

At the end of this lab you should be able to:

- Install and deploy the Adapter for JDBC and integrate it into an SCA application for use with outbound request processing. This exercise demonstrate one scenario which is creating a multiple cardinality business objects Hierarchy using External Service Wizard, instead of editing the parent and child Business Objects using Business Object Editor. The 'Keep Relationship', 'Ownership' and 'Required' ASIs can also be specified in Configuration Parameters Screen of External Service Wizard.

Introduction

This lab introduces you to the WebSphere Adapter for JDBC and the processing of outbound requests to a table in a database. It uses a JDBC TEST Derby database that contains a CUSTOMER table. In the lab, you will import the JDBC Adapter into WebSphere Integration Developer and run External Service to input connection information, create a service description, and discover objects existing in the specified database. You will then assemble an SCA application, wiring together a stand-alone reference and the EIS import file. To test your application, you will use the WebSphere Test Environment and Component Test, exercising various outbound requests, such as create, delete, retrieve, and retrieveAll.

Exercise instructions

Some instructions in this lab are Windows operating-system specific. If you plan on running the lab on an operating-system other than Windows, you will need to run the appropriate commands, and use appropriate files (.sh or .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references, as follows:

Reference variable	Windows location	AIX® or UNIX® location
<LAB_NAME>	JDBCOutbound	
<WID_HOME>	C:\Program Files\IBM\WebSphere\ID\7.0	
<WPS_HOME>	<WID_HOME>\runtimes\bi_v70	
<JDBCADAPTER_HOME>	<WID_HOME>\Resource Adapters\JDBC	
<LAB_FILES>	C:\Labfiles70	/tmp/Labfiles70
<TEMP>	C:\temp	/tmp

Windows users note: When directory locations are passed as parameters to a Java™ program such as EJBdeploy or wsadmin, it is necessary to replace the backslashes with forward slashes to follow the Java convention. For example, replace C:\LabFiles70\ with C:/LabFiles70/

Note that the previous table is relative to where you are running WebSphere Integration Developer.

Instructions if using a remote server for testing

Note that the previous table is relative to where you are running WebSphere Integration Developer. The following table is related to where you are running the remote test environment:

Reference variable	Example: Remote Windows test server location	Example: Remote z/OS [®] test server location	Input your values for the remote location of the test server
<SERVER_NAME>	server1	sssr011	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/sscell/AppServer	
<HOSTNAME>	localhost	mvsxxx.rtp.raleigh.ibm.com	
<SOAP_PORT>	8880	8880	
<TELNET_PORT>	N/A	1023	
<PROFILE_NAME>	AppSrv01	default	
<USERID>	N/A	ssadmin	
<PASSWORD>	N/A	fr1day	

Instructions for using a remote testing environment, such as z/OS, AIX[®] or Solaris, can be found at the end of this document. Go to the section "[Task: Adding remote server to the WebSphere Integration Developer test environment](#)".

Part 1: Create the JDBCTEST database using Derby

In this part you will create the JDBCTEST database along with the CUSTOMER and ADDRESS table for user data.

If you choose to run the JDBCInbound lab first, then you only need to create additional table, ADDRESS for this exercise. If you choose to run JDBCOutbound User Defined SQL lab first, then you can skip this part and proceed to Part 2.

If you are using a remote testing environment on z/OS, it is recommended to create the Derby databases on your Windows machine and then FTP the data files to the z/OS environment. Be sure to upload the files in the /log and /seg0 directories in binary format, and the "service.properties" file in ASCII format. You used a directory on the host called /tmp/LabFiles70/DerbyDB. Also be sure to use the CHMOD command to change all the files to 777.

1. Start Derby ij command

Note: The Derby embedded driver that is being used in the lab, supports a connection from only one JVM at a time. You can have either the server running and connected to the JDBCTEST database, or the vendor GUI tool connected to the JDBCTEST database; but not both at the same time.

- a. Open a command prompt window, navigate to this subdirectory, and run Derby's interactive JDBC scripting tool.

```
<WPS_HOME>\derby\bin\embedded>ij
```

2. Using the command line, create the JDBCTEST database if it does not already exist. If you have already completed the JDBCOutbound lab, the database, tables, and triggers are the same.

- a. Type in this script to create JDBC database as user 'Wbidemo' and password 'Wbi15Demo1' in the Console view.

```
connect 'jdbc:derby:databases\JDBCTEST;create=true;user=Wbidemo;password=Wbi15Demo1';
```



```
G:\IBM\70GM\WID7_WTE\runtimes\bi_v7\derby\bin\embedded>ij
ij version 10.3
ij> connect 'jdbc:derby:databases\JDBCTEST;create=true;user=WBI demo;password=WBI
15Demo1';
ij>
```

3. Enter SQL to create two tables: CUSTOMER and ADDRESS tables for user data.

NOTE: For your convenience, these SQL code snippets can be found in <LAB_FILES>\JDBC\snippets\CUSTOMERSQL.txt

- a. Paste this script into the command window:

```
CREATE TABLE CUSTOMER
(
    pkey VARCHAR(10) NOT NULL PRIMARY KEY,
    LName VARCHAR(20),
    FName VARCHAR(20),
    ccode VARCHAR(10)
);
```

```
C:\IBM\70GM\WID7_WTE\runtimes\bi_v7\derby\bin\embedded>ij
ij version 10.3
ij> connect 'jdbc:derby:databases\JDBCTEST;create=true;user=WBI demo;password=WBI
15Demo1';
ij> CREATE TABLE CUSTOMER
(
    pkey VARCHAR(10) NOT NULL PRIMARY KEY,
    LName VARCHAR(20),
    FName VARCHAR(20),
    ccode VARCHAR(10)
);
0 rows inserted/updated/deleted
ij>
```

NOTE: For your convenience, these SQL code snippets can be found in
<LAB_FILES>\JDBC\snippets\ADDRESSSQL.txt

__ b. Paste this script into the command window:

```
ij> CREATE TABLE ADDRESS
(
    addrid VARCHAR(10) NOT NULL PRIMARY KEY,
    custid VARCHAR(10),
    city VARCHAR(20),
    zipcode VARCHAR(10)
);
0 rows inserted/updated/deleted
ij>
```

___ 4. Disconnect and exit the ij command window

```
ij> disconnect;
ij> exit;
C:\IBM\70GM\WID7_WTE\runtimes\bi_v7\derby\bin\embedded>
```

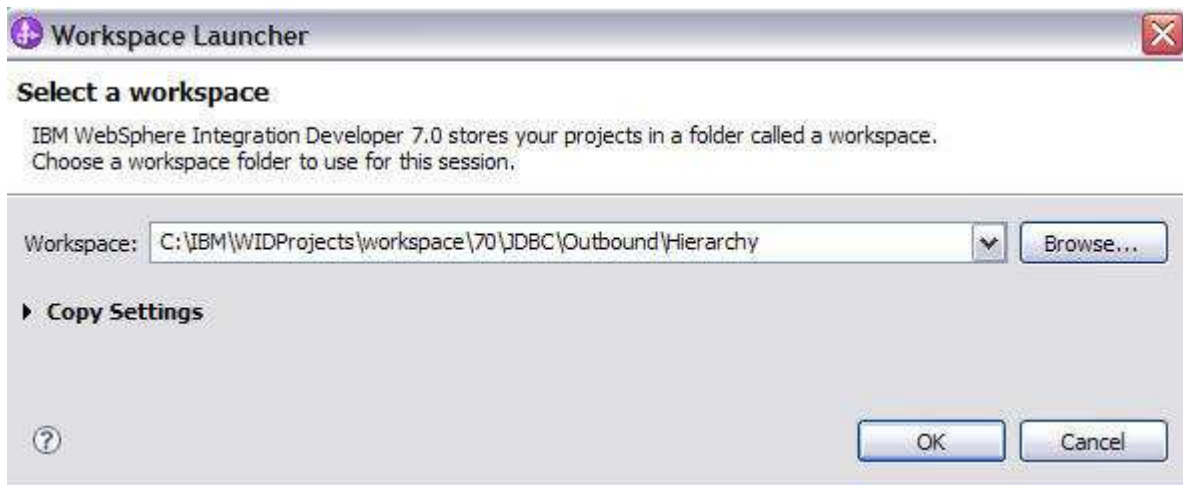
Note: You must exit the ij command window, before starting the server as the Derby “embedded” database driver used in this lab allows for only one JVM connection at a time.

Remember that if you are using a remote test environment for this lab, you need to FTP the Derby database files to the remote system, as explained at the beginning of this section.

Part 2: Set up the development environment

In this part, you will start WebSphere Integration Developer, using a new workspace, import RAR file and set up the WebSphere Process Server to be used as the WebSphere Test Environment (WTE).

- ___ 1. Start WebSphere Integration Developer V7.0 with a new workspace.
 - ___ a. From the start menu select **Start > Programs → IBM WebSphere Integration Developer → IBM WebSphere Integration Developer 7.0 → IBM WebSphere Integration Developer V7.0**
 - ___ b. When prompted enter **<LAB_FILES>\JDBC\outbound-hierarchy** for your workspace and click **OK**



- ___ c. When WebSphere Integration Developer V7.0 opens, close the **Getting Started** page



Part 3: Create the JDBC Outbound application

In this part you will run External Service to discover objects and create the necessary SCA artifacts, and assemble the adapter into an SCA application.

NOTE: If you are using a remote test environment on z/OS, you will first need to enable your Derby database for network access using these steps:

On the host, login to the system using telnet.

In order to allow connections to the network server, you need to edit the /<WPS_HOME>/cloudscape/db2j.properties file. Delete the '#' (pound sign) to uncomment this line:

```
db2j.drda.host=0.0.0.0
```

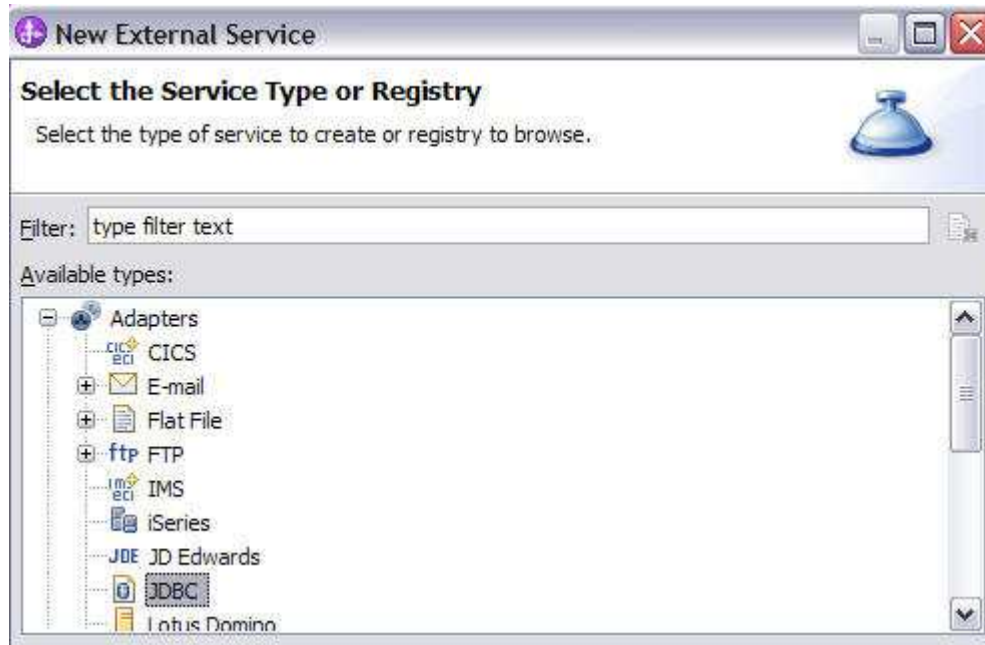
Navigate to /<WPS_HOME>/derby/bin/networkServer

Issue the command **./startNetworkServer.sh**

Keep this telnet window open during the next few steps; this window will show when a connection has been successfully made from WebSphere Integration Developer to the database server.

****NOTE** that there are security concerns with enabling the network server for Derby in this manner. Read the information in the db2j.properties file regarding these security concerns to determine if your installation will allow for the use of the network server for Derby. If not, you can continue doing the lab using the integrated test client in WebSphere Integration Developer instead of your remote test environment.

-
- ___ 1. In the Business Integration Perspective, run the External Service wizard. A Business Integration project is created for you during this process.
 - ___ a. From the top Menu bar, select **Window > Open Perspective > Other ... > Business Integration (default)** click **OK**
 - ___ 2. From within the Business Integration Perspective, select **File > New > External Service**. This opens an External Service wizard that helps you obtain a service which establishes connectivity with other systems. The wizard provides four connectivity options – Adapters, Java, Registers, and Messaging
 - ___ a. Expand **Adapters**, select **JDBC**, and click **Next**

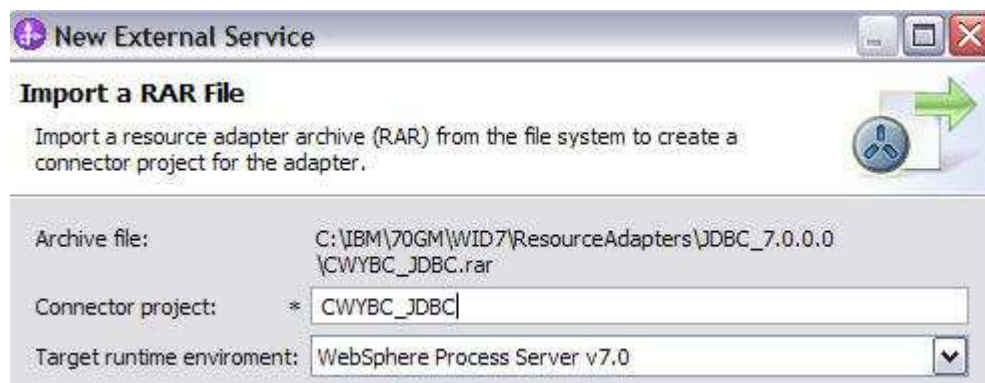


___ 3. Highlight **IBM WebSphere Adapter for JDBC (IBM: 7.0.0.0)** and click **Next**

___ 4. Adapter Import screen:

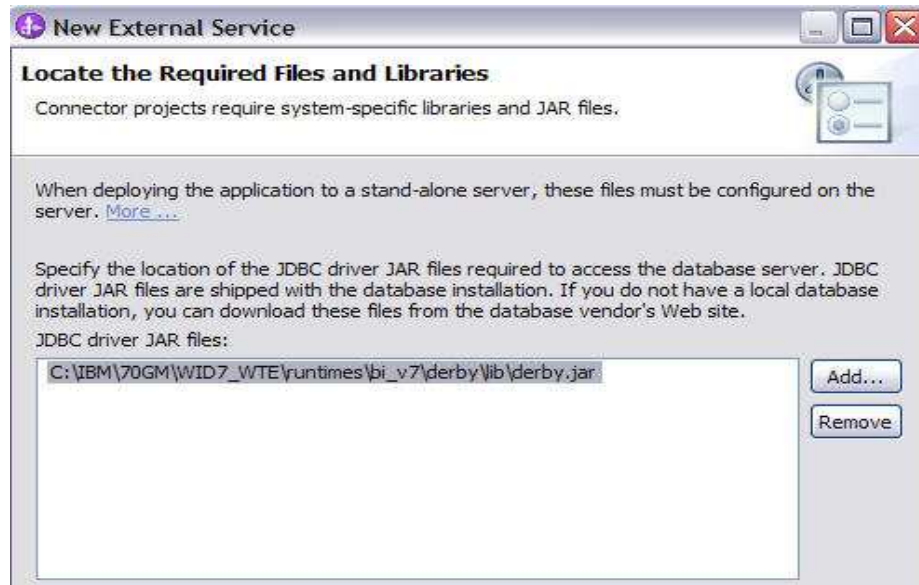
In this step, you will import a connector resource adapter archive from the file system into your WebSphere Integration Developer workspace. The adapter RAR file already exists under **<JDBCADAPTER_HOME>**.

- ___ a. The default Connector file is selected which is shipped along with WebSphere Integration Developer
- ___ b. Accept the default name for Connector project, **CWYBC_JDBC**. You can change it to any other name, but for this lab, you can leave the default name.
- ___ c. For Target server, ensure that **WebSphere Process Server V7.0** is selected



___ 5. Add any external dependencies your adapter has to the imported project. These are dependencies that the adapter can have on the JDBC applications (adapter-specific).

- ___ a. Click **Add** and browse to the location of c:\<WPS_HOME>\derby\lib and select the **derby.jar**. If you are using a remote test environment on z/OS, you will also need to select **db2jcc.jar** and **db2jcc.license.jar**. The db2jcc.jar file is sometimes located in directory \derby\lib\otherJars.



- ___ b. Click **Next**

___ 6. Select the type of processing the adapter will perform at runtime

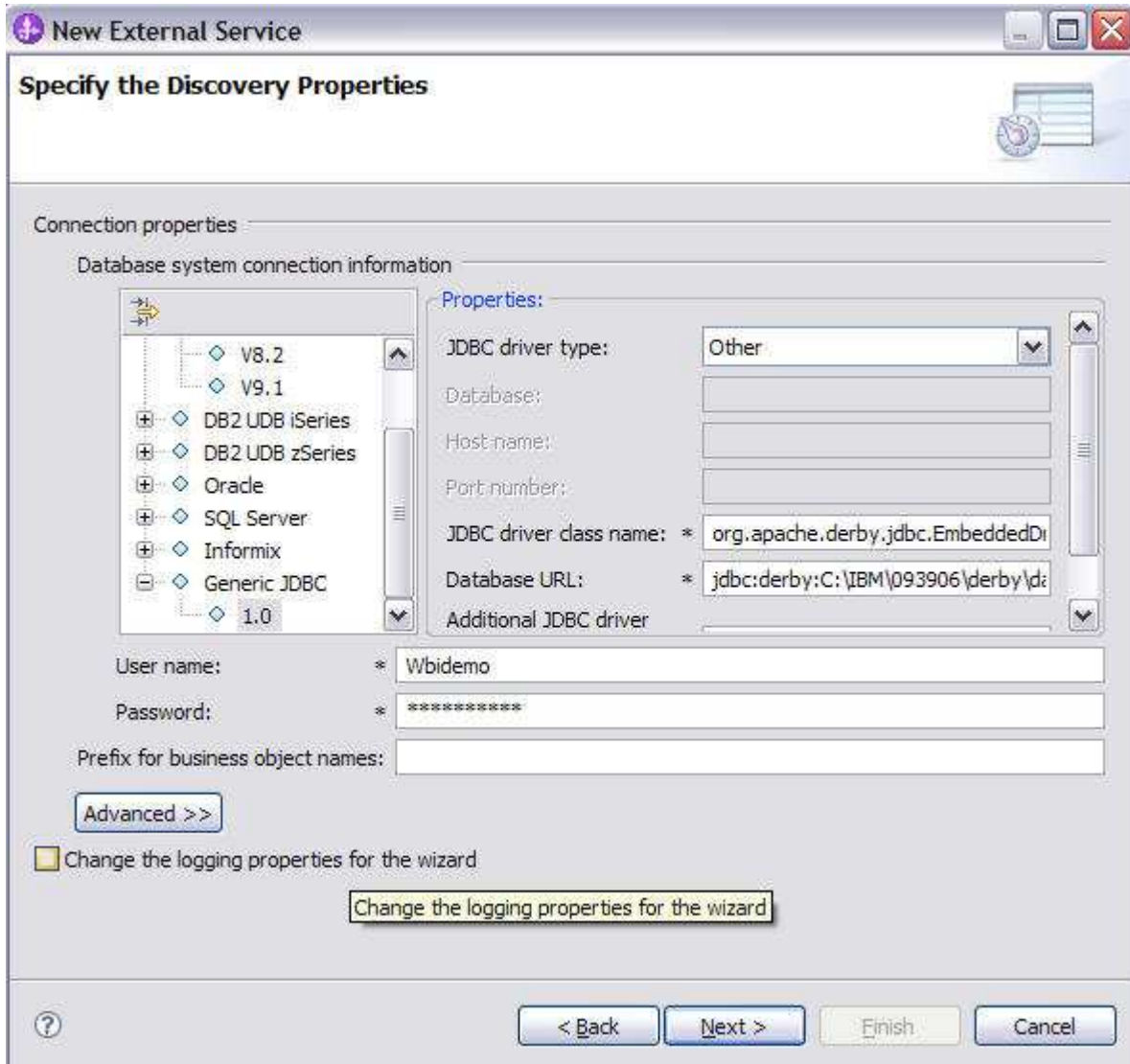
- ___ a. Select **Outbound** and click **Next**



- ___ 7. Complete the Connection Configuration for Discovery Agent Configuration panel to connect to the JDBCTEST database and discover the available services. To connect to the database, these information are necessary: username, password, database URL and JDBC driver class. Check the driver manual for the appropriate values for driver URL and driver class.
- ___ a. From the left panel, expand **Generic JDBC** and select **1.0**
 - ___ b. From the right panel, enter/select these following values
 - 1) JDBC Driver type: **Other**
 - 2) JDBC Driver classname: **org.apache.derby.jdbc.EmbeddedDriver**
 - 3) Database URL: **jdbc:derby:<WPS_HOME>derby\databases\JDBCTEST**
 - ___ c. Enter valid user ID and password values, for example
 - Username: **Wbidemo**
 - Password: **Wbi15Demo1**

NOTE If you are using a remote test environment, use a valid user ID and password for the remote system and enter these values for DatabaseURL and JdbcDriverClass
DatabaseURL: jdbc:db2j:<WPS_HOME>\runtimes\bi_v70\derby\databases\JDBCTEST
JdbcDriverClass: com.ibm.db2j.jdbc.DB2jDriver

If using a remote test environment on z/OS use these values
DatabaseURL: jdbc:db2j:net://<HOST_NAME>:1527//<remote_derbydb_path>/JDBCTEST
JdbcDriverClass: COM.ibm.db2os390.sqlj.jdbc.DB2SQLJDriver



__c. Click **Next**

___ 8. Complete the **Find and Discover Enterprise Services** panel


__ a. Click the **Edit Query** button

__ b. Check option **Prompt for additional configuration settings when adding business object**

__ c. Click **OK**.




__ d. Click **Run Query** button. A connection is made to the Derby JDBCTEST database and a selection of Meta data objects is presented in a tree-like structure.

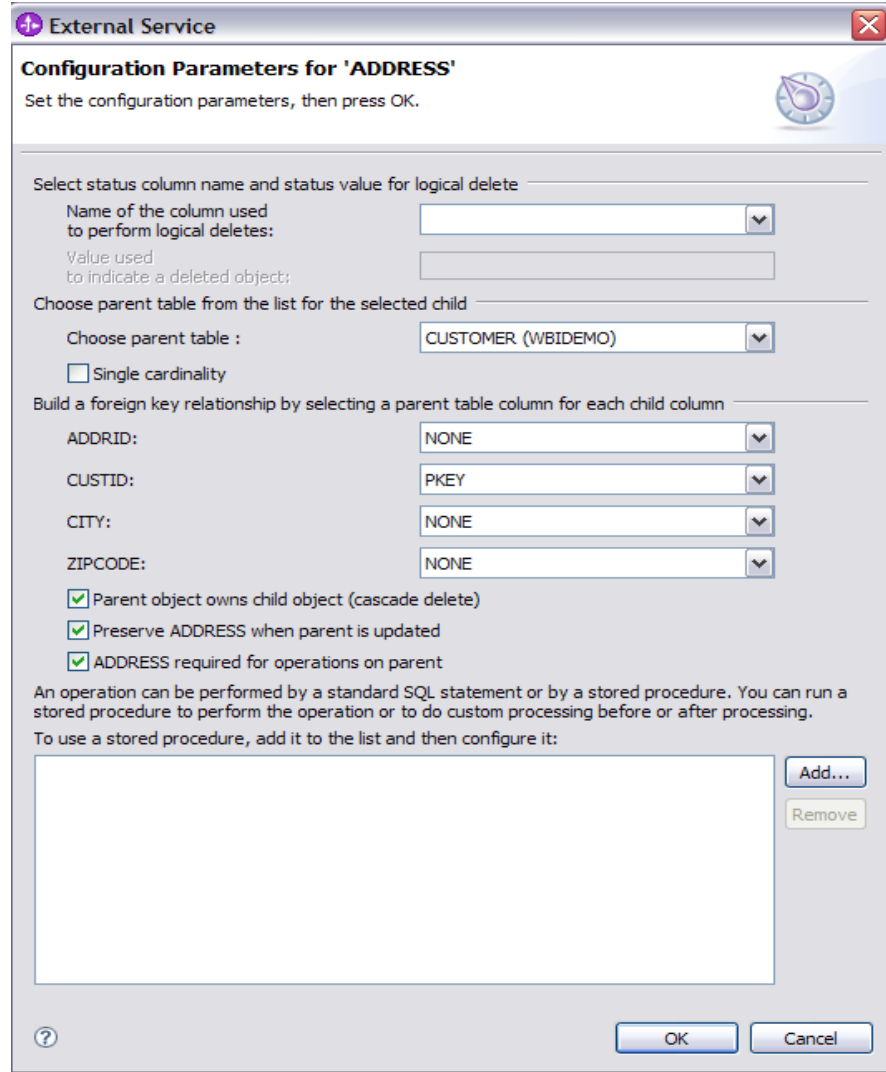
__ e. Expand the schema named **WBIDEMO**, expand **Tables**, highlight **CUSTOMER**, click the  button.

- ___ f. It opens up a window for setting the Configuration Parameters for the **CUSTOMER** table. All the controls, like Choose Parent, for Hierarchy BO Generation are not available as no object is imported yet. Click **OK** button to add **CUSTOMER** table.



- ___ g. Select **Address** table and click the  button.
- ___ h. It opens up a window for setting the Configuration Parameters for the **ADDRESS** table where you can select Parent table and map parent-child attributes.
- ___ i. In the Configuration Parameters screen for **Address**, select these following values
 - 1) Choose **CUSTOMER** (WBIDEMO) as the Parent table for **ADDRESS** from the list box.
 - 2) This particular scenario creates business object with multiple cardinality between **CUSTOMER** and **ADDRESS**, therefore do not check **Single Cardinality** check box.
 - 3) To Map Parent Child Attributes, choose **PKEY** of **CUSTOMER** for **CUSTID** from the drop-down list.

- 4) Check **Parent object owns child object, Preserve ADDRESS when parent is updated, ADDRESS required for operations on parent** check boxes to specify the respective ASIs.
- 5) Click **Ok**



___ j. Click **Next**

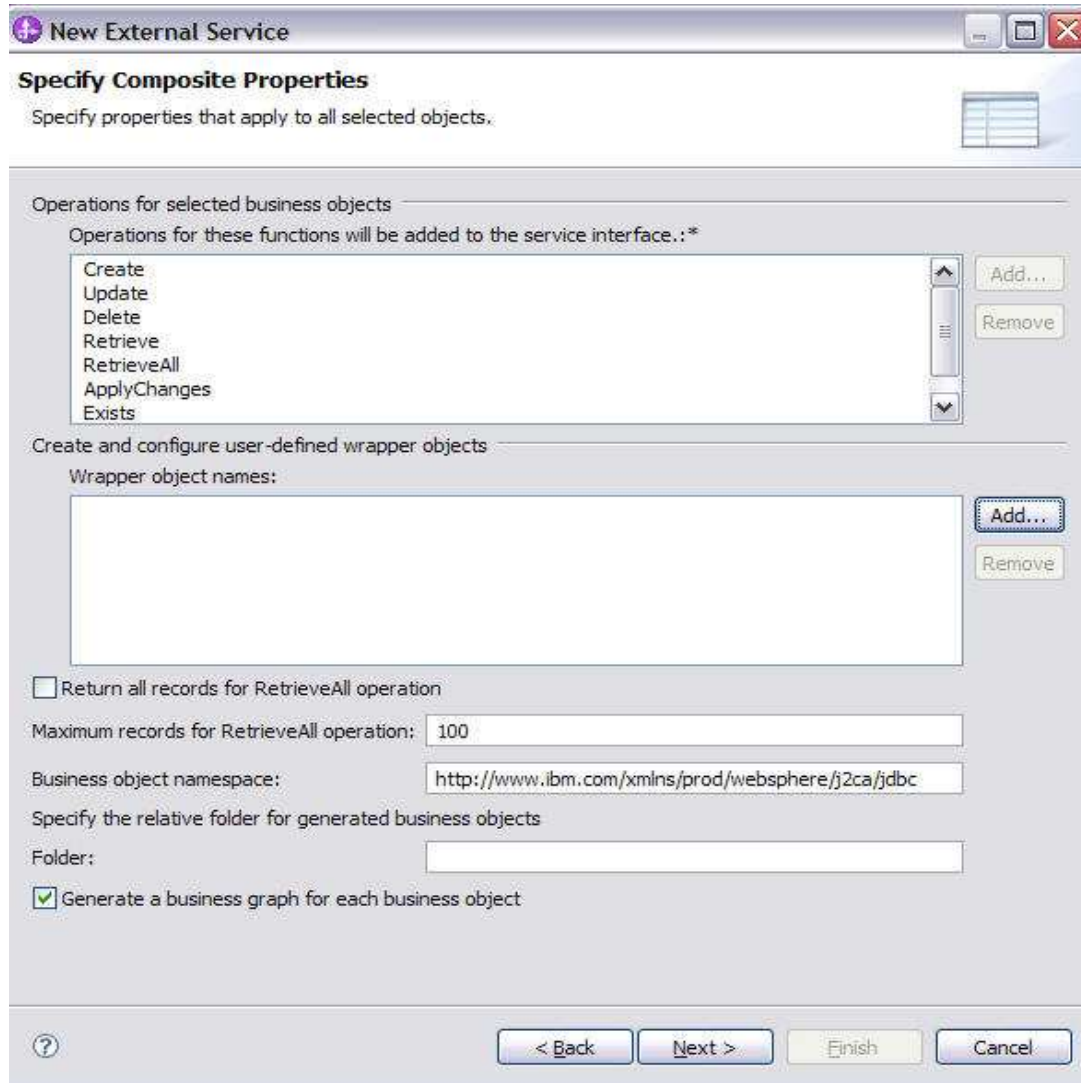
___ 9. Complete the **Configure Objects** panel

___ a. A new operation Exists has been added in addition to the current list of supported operation.

___ b. No wrapper object is required at this time.

___ c. Leave the default value for Namespace. Note the Operations available.

___ d. Note that Business Graph is optional - It will determine if you want to generate business graph or not. If the property is checked then business graphs is generated. By default, the property is checked.



__ e. Click **Next**

___ 10. Complete Publishing Object Configuration Properties

__ a. Specify the security credentials

1) Uncheck **JAAS alias** option and select **Other** to define the data source JNDI name

__ b. Leave the default “**With module for use by single application**” option in this exercise.

__ c. Under **Connection properties**

1) Leave the default “**Specify predefined XA Datasource**”

2) Database Vendor is pre-populated with value **Other** (If you were using DB2, Oracle, or MSSQLServer, you see those values instead as specific adapter processing is available with those specific databases.)

3) Enter **jdbc/Derby XA for JDBC** for XA Data source JNDI name

New External Service

Specify the Service Generation and Deployment Properties

Specify properties for generating the service and running it on the server.

Service Operations

To modify the names, or add a description to the operations to be generated in the interface file, click Edit Operations. Edit Operations...

Deployment Properties

How do you want to specify the security credentials?

Using an existing JAAS alias (recommended)

Using an existing JAAS alias (recommended)

A Java Authentication and Authorization Services (JAAS) alias is the preferred method.

J2C authentication data entry:

Using security properties from the managed connection factory

Using security properties from the managed connection factory

The properties will be stored as plain text; no encryption is used.

User name:

Password:

Other

Other

Use if no security is required or will be handled by the EIS system, or the RAR will be deployed on the server and security will be specified by the properties in the JNDI lookup name.

The quality of service to join the transaction provides a higher degree of data integrity, especially in the event of a failure. To participate in a transaction, a predefined XA DataSource or XA database connection information must be specified in the Connection Properties. [More ...](#)

Join the transaction (recommended)

Deploy connector project:

Specify the settings used to connect to JDBC at run time:

Connection settings:

Connection Properties

Database connection information:

Database system connection information

Database vendor:

XA DataSource JNDI name: *

Note: If you are using a remote test environment, the values for **databaseURL** and **JdbcDriverClass** should match the values provided earlier during the External Service wizard, not what is pictured below.

__ d. Click **Advanced** button for additional properties options

1) **Optional** for logging and tracing

- a) Enter any numeric value for **Adapter ID**. This property identifies the adapter instance in log and trace files and for PMI events. The adapter ID is used with an adapter-specific identifier, MyAdapterRA, to form the component name used by Log and Trace Analyzer.
- b) If you set the property to disguise user data as "XXX", the adapter will replace user data with a string of x's when writing to log and trace files. For inbound processing, this property is retrieved from the resource adapter properties. For outbound processing, it is retrieved from the managed connection factory properties.

The screenshot shows a configuration panel with several sections:

- Additional connection configuration:**
 - Set Auto Commit on database connection
 - Additional JDBC driver connection properties: [name:value;name:value]:
 - SQL query to verify the connection:
 - Query timeout (seconds):
 - Return business object even when the stored procedure result set is empty: false
- Logging and tracing:**
 - Adapter ID: 001
 - Disguise user data as "XXX" in log and trace files. (This checkbox is highlighted with a red box in the original image)
- Bidi properties:** (collapsed)
- Connection retry settings:**
 - Maximum retries on connection failure: 0
 - Connection retry interval (in milliseconds): 60000

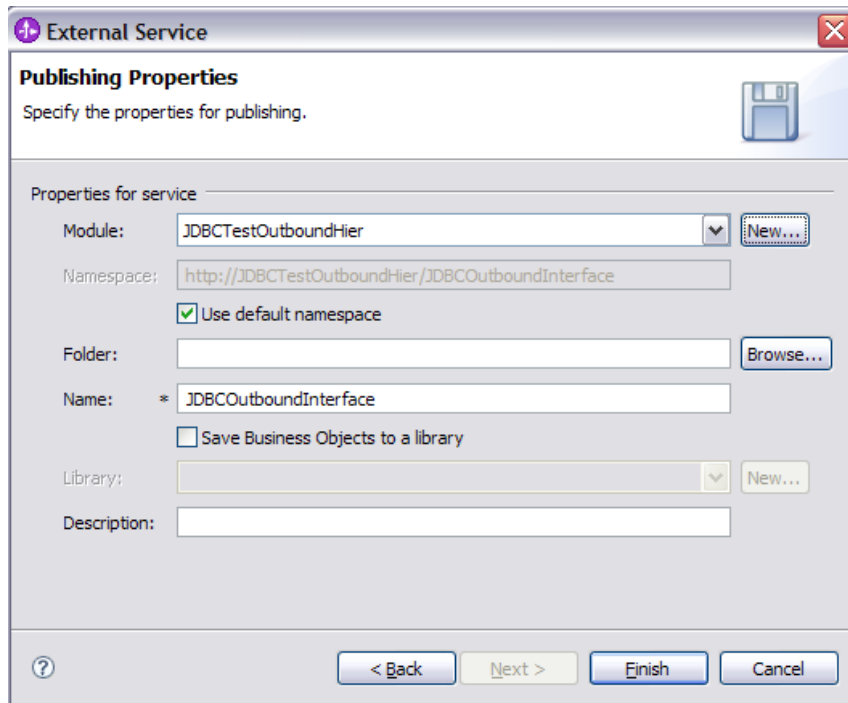
__ e. Click **Next**

___ 11. Complete Publishing Properties panel

__ a. A Business Integration Module has not yet been created. You can define the module here also.

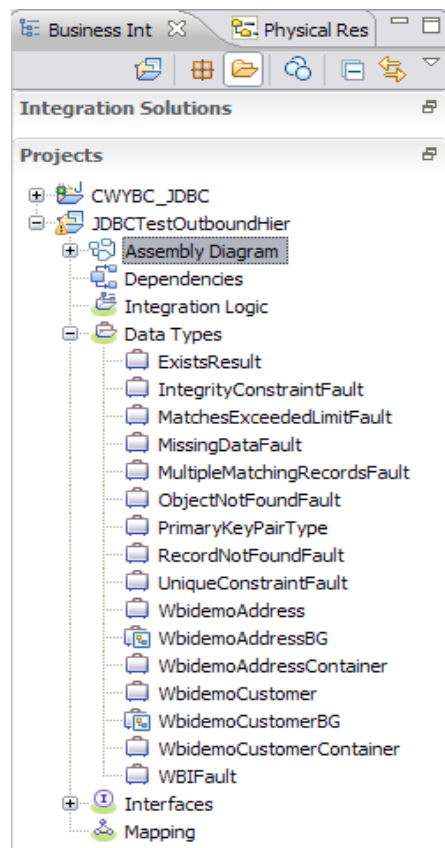
- 1) Select the **New** button Select the radio button "**Create a module project**" from new **Integration Project** pop up window.
- 2) Enter in the name **JDBCTestOutboundHier** for the Module Name
- 3) Accept the defaults and click **Finish**

__ b. Leave the default **JDBCOutboundInterface** for the Name



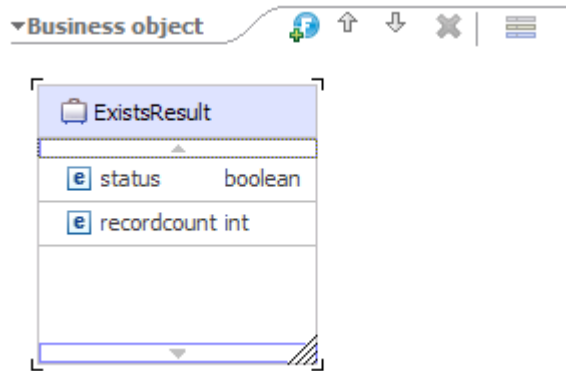
__ c. Click **Finish**

___ 12. Verify the result as given below

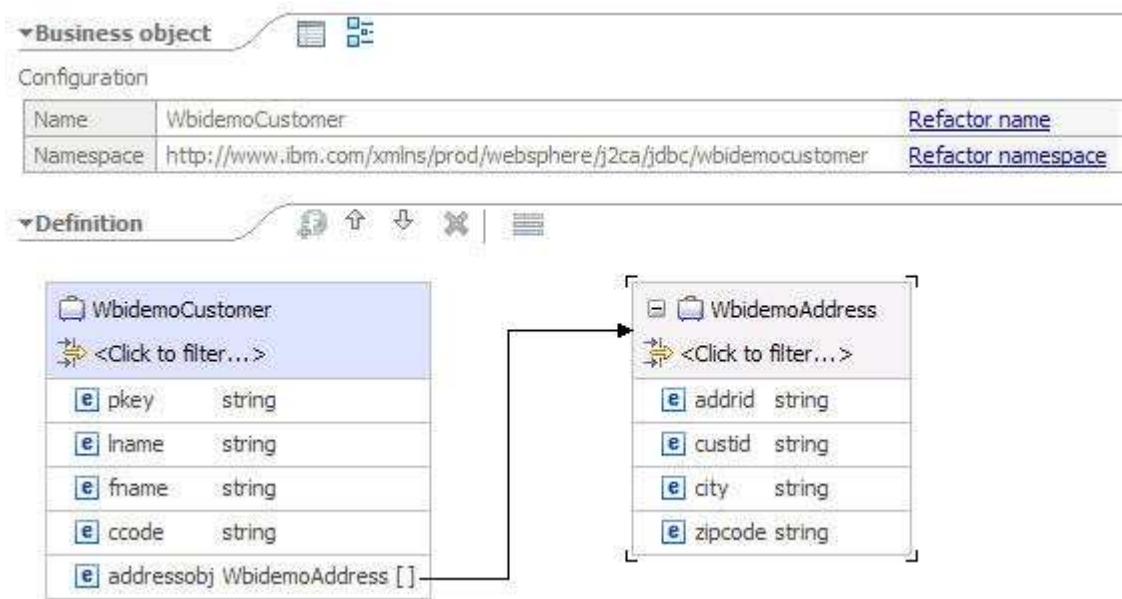


___ a. Double click new **ExistsResult** BO listed under Data Types

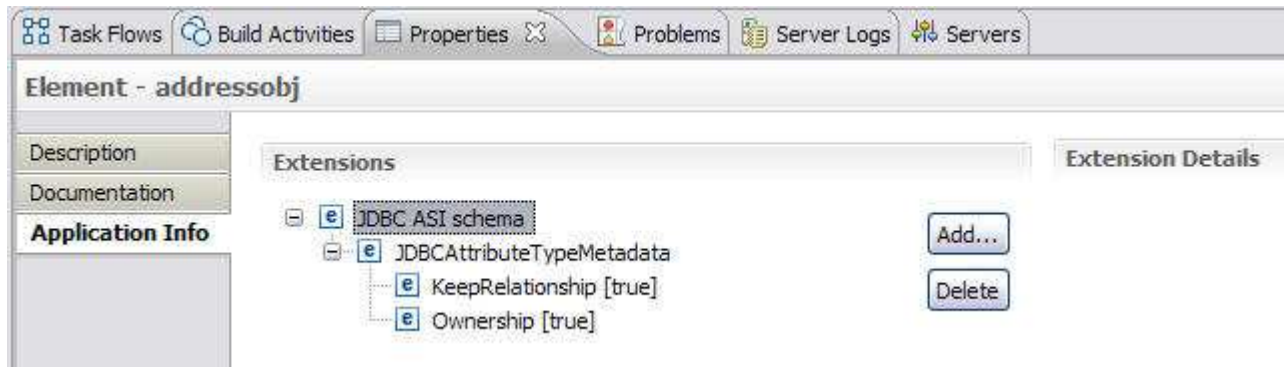
- 1) Status returns a Boolean value if the record is found in the database
- 2) Recordcount returns records found in the database for a specified query.



___ 13. Double click **WbidemoCustomer** business object listed under **Data Types**

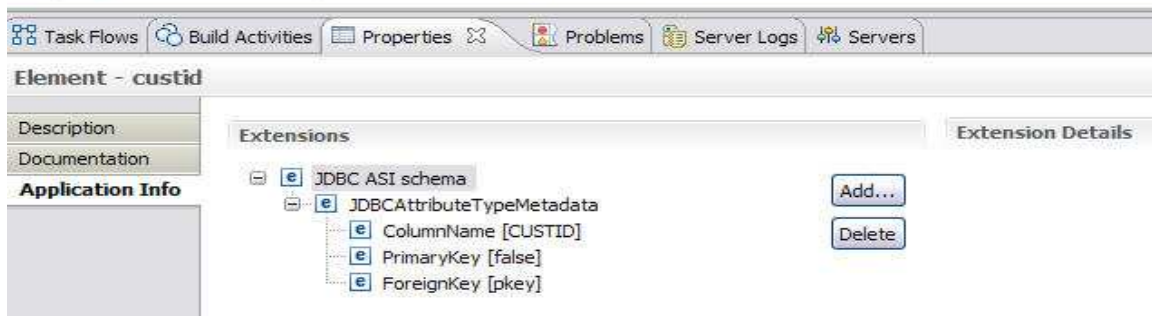
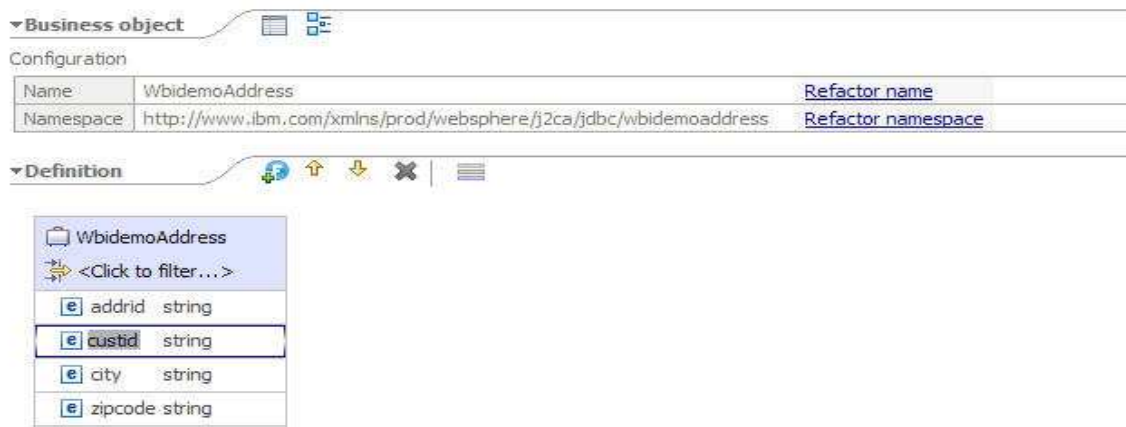


___ a. Click **Application Info** in the **Properties** tab and verify that values against **jdbcas:Ownership** and **jdbcas:KeepRelationship** are true as you selected Ownership and Keep Relationship check boxes in External Service.



___ 14. Double click **WbidemoAddress** business object listed under **Data Types**

___ a. Select attribute **custid** attribute and in the **Properties** tab, click **Application Info**. Verify that **pkey** is specified against **jdbcasi:ForeignKey**.



___ b. Save your work by selecting **File -> Save** from the top menu, or using the shortcut key sequence **Ctrl + S**. Close the file.

___ 15. Release the connection to the Derby database by using **Switch Workspace**

Note: Switch Workspace is a way to release the existing connection to the Derby database from the External Service process. In the next part, you will start the WebSphere Process Server and it will need a connection to the database to create and retrieve records. This step is necessary only because you are using the Derby embedded driver in this exercise which supports a connection from a single JVM.

___ a. From the top menu bar, select **File > Switch Workspace** and select the same workspace from which you have been working.

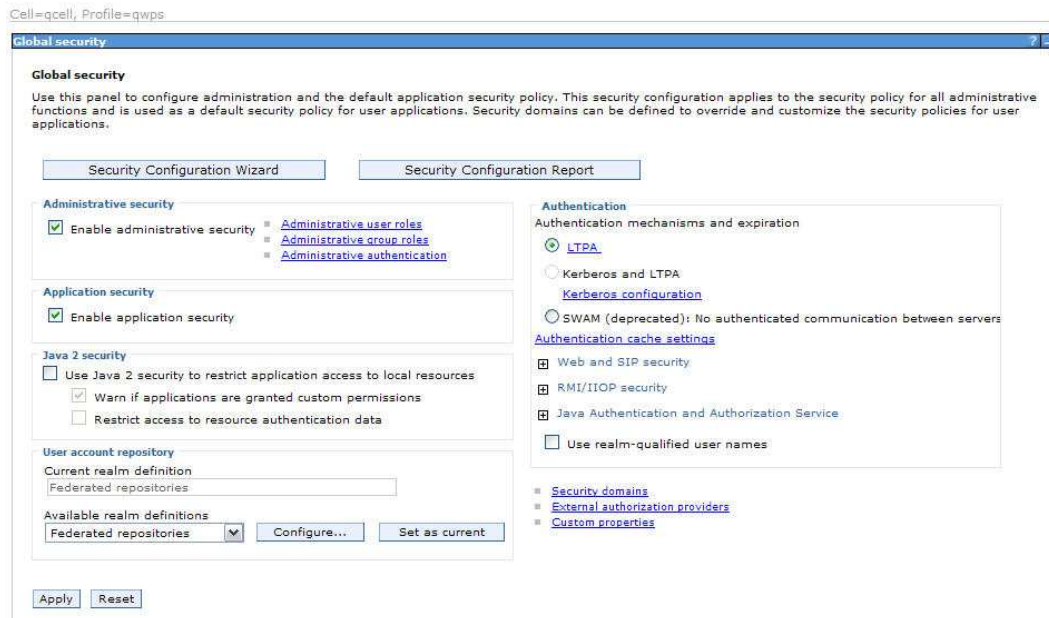
Part 4: Create J2C authentication alias and configure data sources

In this part you will create a J2C Authentication Alias which is required for connection to the database. You will also create the JNDI data source name that is used by the Adapter to configure itself to the endpoint. You will then use the WebSphere Test Environment and Component Test to test the SCA application by retrieving several records from the CUSTOMER table in the JDBCTEST database.

- ___ 1. Switch to Servers view by selecting Windows → Show View → Servers.
- ___ 2. Set the authentication alias from the administrative console
 - ___ a. In the **Servers** tab in the lower-right corner pane, right click the **Server** and then select **Start**
 - ___ b. When the server status is **Started**, right click the **Servers**, and then select **Administration** → **Run Administrative console**



- ___ c. Log in to the administrative console by clicking the “Log in” button
- ___ d. Click **Security** → **Global Security**
- ___ e. On the right, expand **Java Authentication and Authorization Service** under the Authentication heading.



- ___ f. Click **J2C authentication alias**. It gives the list of existing aliases.

1) Click **New** and enter these following values

a) User ID: **Wbidemo**

b) Password: **Wbi15Demo1**

[Global security](#) > [JAAS - J2C authentication data](#) > **New**

Specifies a list of user identities and passwords for Java(TM) 2 connector security to use.

General Properties

* Alias

* User ID

* Password

Description

__ g. Click **Ok** and save the changes

___ 3. Create a JDBC Provider to which is used to create data source

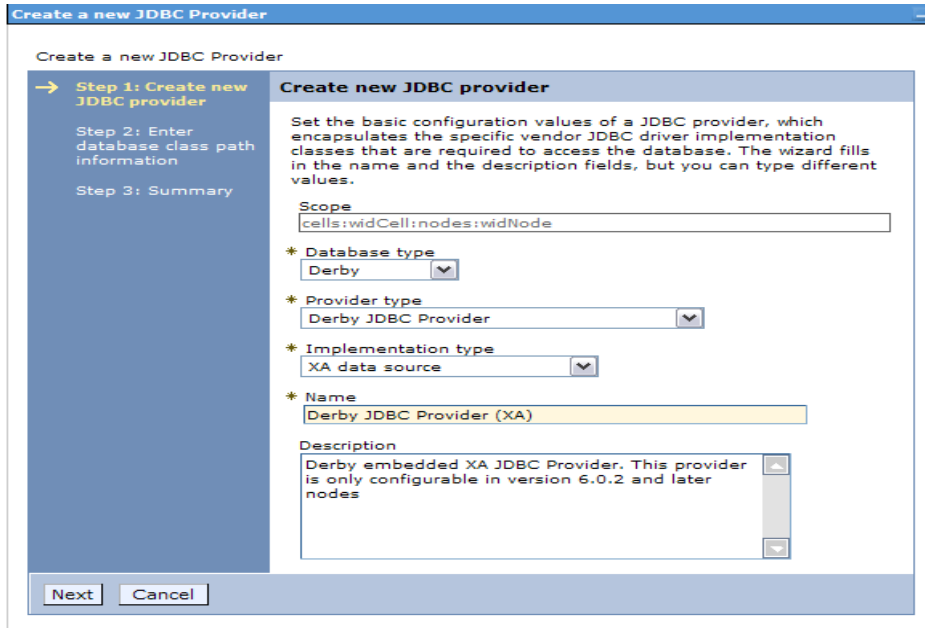
__ a. From administrative console, click **Resources** → **JDBC** → **JDBC Providers**

__ b. On the right, click **New** and choose these following values

1) Database type: **Derby**

2) Provider type: **Derby JDBC Provider**

3) Implementation type: **XA data source**



__ c. Click **Next** and review the summary of changes.

__ d. Click **Finish** and save the changes.

___ 4. Create JDBC data source

__ a. On the right, click **Data Sources** under **Additional Properties** heading

__ b. Click **New** to set a data source and enter these following values

1) Data source name: **Derby JDBC Driver XA DataSource**

2) JNDI name: **jdbc/Derby XA for JDBC**

__ c. Click **Next** to Step 2

__ d. In **Database specific properties for the data source**, enter **JDBCTEST** as database name

__ e. Click **Next** to Step 3

__ f. Set the Component-managed authentication alias to the one created in the earlier section.

__ g. Click **Next** and view the Summary

Create a data source

Step 1: Enter basic data source information

Step 2: Enter database specific properties for the data source

Step 3: Setup security aliases

→ Step 4: Summary

Summary

Summary of actions:

Options	Values
Scope	cells:qcell:nodes:qnode
Data source name	Derby JDBC Driver DataSource
JNDI name	jdbc/Derby XA for JDBC
Select an existing JDBC provider	Derby JDBC Provider
Implementation class name	org.apache.derby.jdbc.EmbeddedConnectionPoolDataSource
Database name	JDBCTEST
Use this data source in container managed persistence (CMP)	true
Component-managed authentication alias	qnode/DerbyDS
Mapping-configuration alias	(none)
Container-managed authentication alias	(none)

Previous Finish Cancel

___ h. Click **Finish** and save the changes.

___ 5. Test the JDBC Data Source connection

___ a. Check the box next to **Derby JDBC Driver XA DataSource** and click **Test connection** from the top of the screen.

___ b. You should see this success message on the top of the screen

The screenshot shows the 'JDBC providers' console window. At the top, a message box states: 'The test connection operation for data source Derby JDBC Driver XA DataSource on server server1 at node widNode was successful.' Below this, the breadcrumb path is 'JDBC providers > Derby JDBC Provider (XA) > Data sources > Data sources'. A paragraph explains the purpose of the page. Under the 'Preferences' section, there are buttons for 'New', 'Delete', 'Test connection' (highlighted with a red box), and 'Manage state...'. Below the buttons is a table with columns: Select, Name, JNDI name, Scope, Provider, Description, and Category. The table contains one entry: 'Derby JDBC Driver XA DataSource' with JNDI name 'jdbc/Derby XA for JDBC', Scope 'Node=widNode', and Provider 'Derby JDBC Provider (XA)'. The 'Select' checkbox for this entry is checked. At the bottom of the table, it says 'Total 1'.

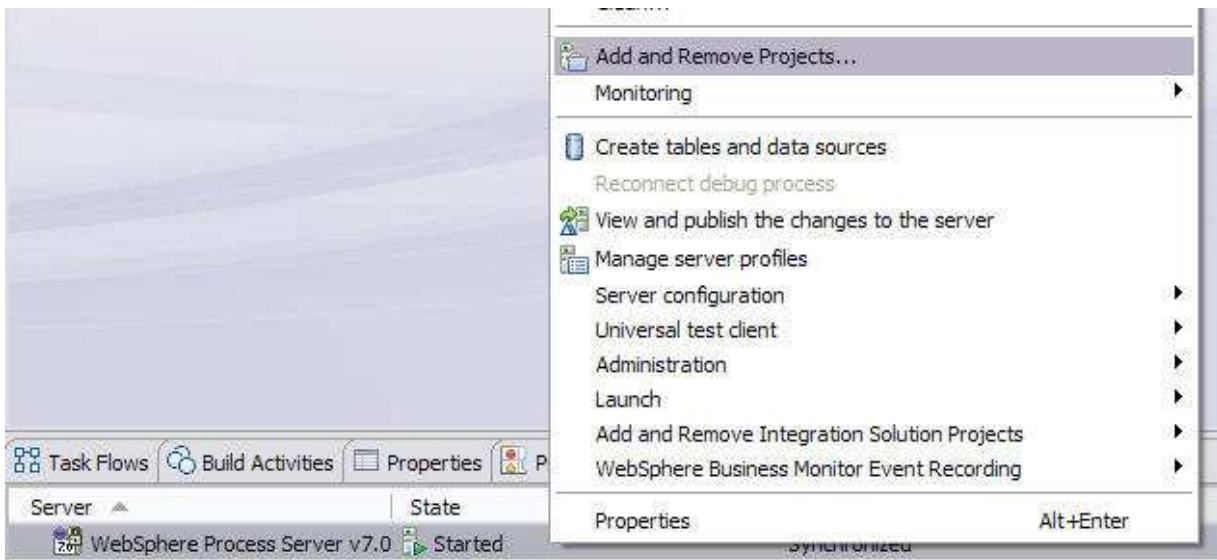
___ 6. Release the connection to the Derby database by using **Switch Workspace**

___ a. From the top menu bar, select **File > Switch Workspace** and select the same workspace from which you have been working.

Part 5: Test the application using the WebSphere test environment and component test

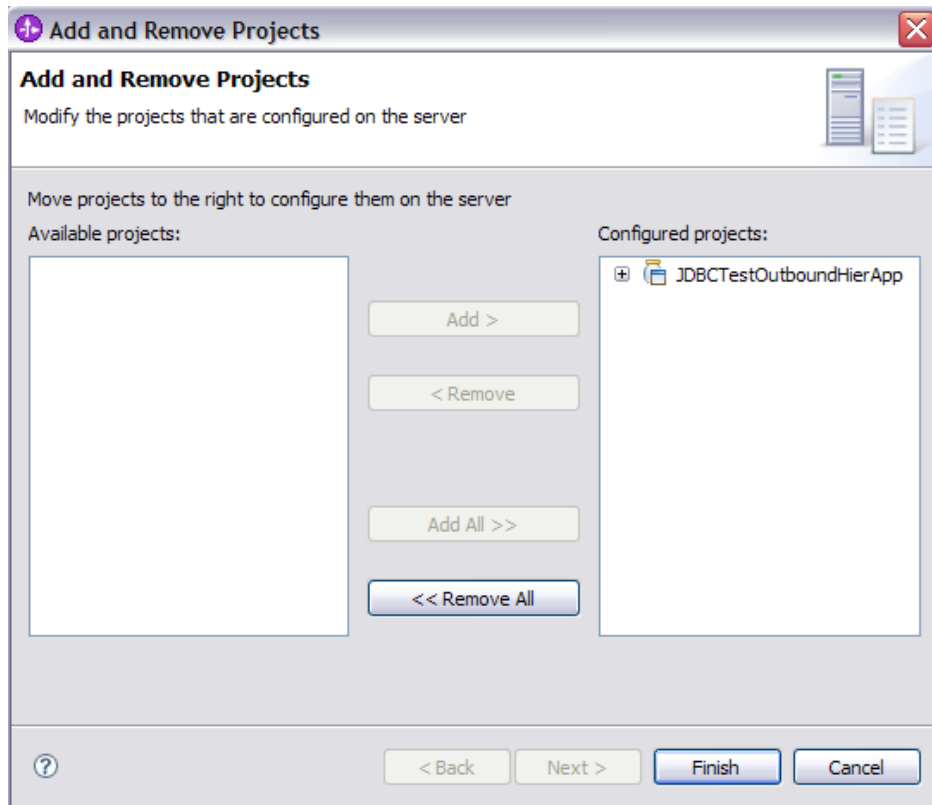
This lab is using the Derby embedded database driver which allows a connection from only a single JVM at a time. ij.bat, the External Service process, and a running Server configured with an application that connects to the database, each are an active connection. This means you can have only one of these active connections to the database at a time. For example, in this part, you will start the server and run several tests creating and retrieving records from the database. You are not able to use ij.bat to view the database, while the server is active. You need to first stop the server.

- ___ 1. In the **Servers** tab in the lower-right corner pane, right click the **Server** and then select **Start**
- ___ 2. Add the project to the server for the WebSphere Test Environment.
 - ___ a. Start the server by right click Server and select **Start**.
 - ___ b. Right click the server in the server view and select **Add and remove projects ...**



- ___ c. In the Add and Remove Projects dialog, select the **JDBCTestOutboundHierApp** project from the Available projects panel.

___ d. Click **Add >** to add it to the Configured projects panel. Click **Finish**



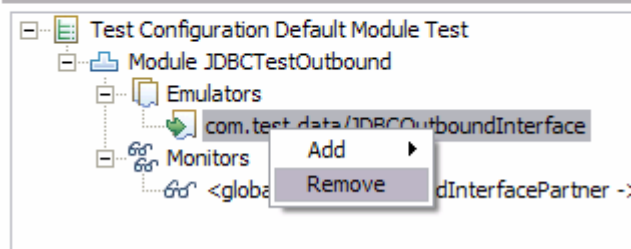
___ 3. Use the Test Component to test the application. The create scenario is to create a record in the database.

___ a. Right click the **JDBCTestOutboundHier module** and select Test→**Test Component**

___ b. In JDBCTestOutboundHier_Test, switch to the **Configurations** tab, if you see **JDBCOutboundInterface** underneath **Emulators**, right click, and remove it. You might not see anything under Emulators. You want to connect to and drive the real JDBCTEST database and not have Test Component emulate for you.

Testing Module: JDBCTestOutbound

Configurations

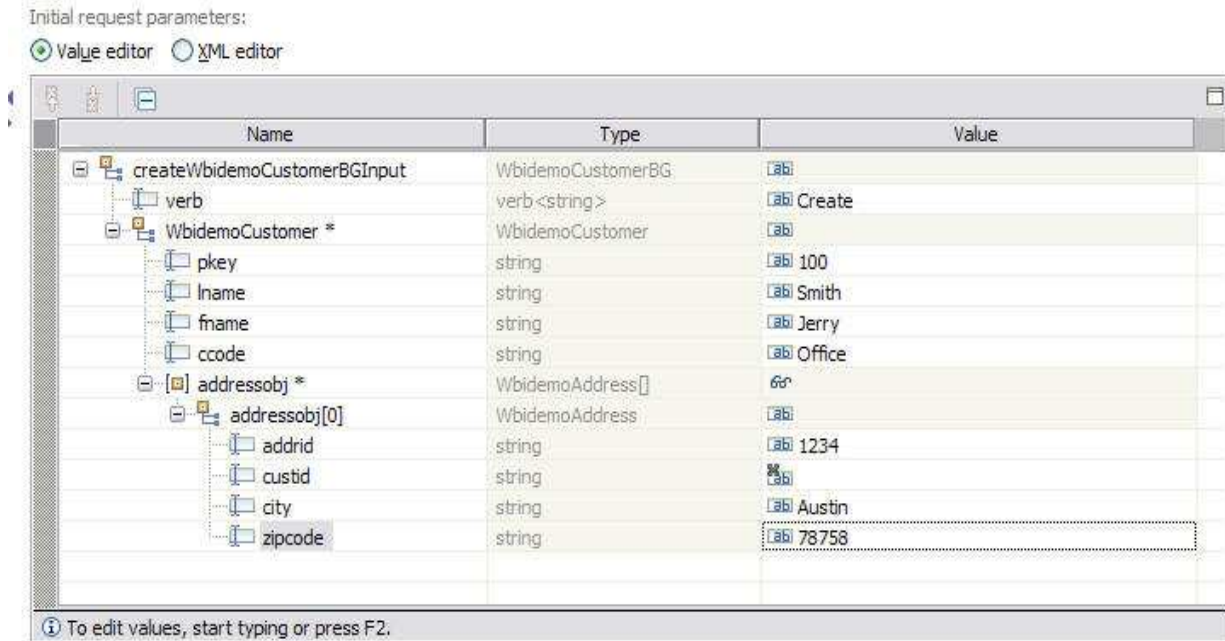


___ c. Switch back to the **Events** tab

1) Under **Detailed Properties**, choose createWbidemoCustomerBG as the Operation.

2) Under **Initial request parameters**, enter these following values

a) Since **custid** is foreign key of **pkey** in Customer table, set **custid** to unset.

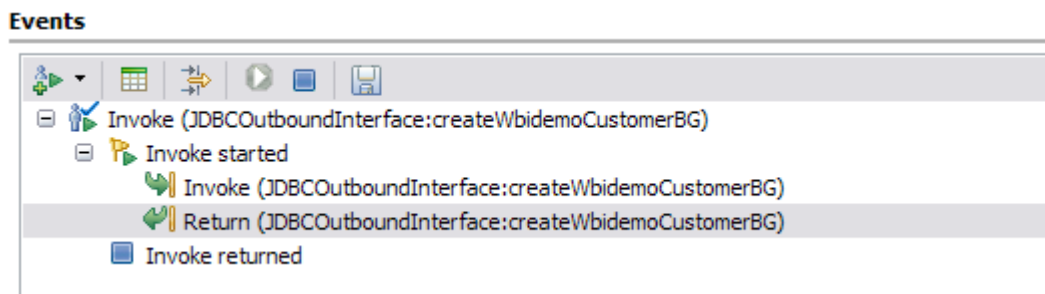


___ d. Click  to continue



___ e. In the **Choose a deployment location dialog**, select the **WebSphere Process Server V7.0** server. Select **Finish**

___ 4. In the Events window you will see that Invoke has returned. Check the data in the EIS to ensure it matches expected values.



- 5. To retrieve the record that you have just created, select the **Invoke** button in the top right corner again, under **Detailed Properties**, select the Operation **retrieveWbidemoCustomerBG**. Since Address object is a child of Customer object, you only need to enter value for Customer and that retrieves the data for child Address object as well.
 - a. Under **Initial request parameters**, enter a value for pkey of one of the previously created customers. Click **Continue**

Configuration:	Default Module Test
Module:	JDBCTestOutboundHier
Component:	JDBCOutboundInterface1
Interface:	JDBCOutboundInterface1
Operation:	retrieveWbidemoCustomerBG

Initial request parameters:

Value editor XML editor

Name	Type	Value
retrieveWbidemoCustomerBGInput	WbidemoCustomerBG	[ab]
verb	verb<string>	[ab] Create
WbidemoCustomer *	WbidemoCustomer	[ab]
pkey	string	[ab] 100
lname	string	[ab]
fname	string	[ab]
ccode	string	[ab]
addressobj *	WbidemoAddress[]	[ab]
addressobj[0]	WbidemoAddress	[ab]
addrid	string	[ab]
custid	string	[ab]
city	string	[ab]
zipcode	string	[ab]

To edit values, start typing or press F2.

- b. Upon return, the values matching the ID specified should be displayed in the Return parameters.

Name	Type	Value
retrieveWbidemoCustomerBGOutput	WbidemoCustomerBG	✓
verb	string	✗✓
WbidemoCustomer	WbidemoCustomer	✓
pkey	string	✓ 100
lname	string	✓ Smith
fname	string	✓ Jerry
ccode	string	✓ Office
addressobj	WbidemoAddress[]	✓
addressobj[0]	WbidemoAddress	✓
addrid	string	✓ 1234
custid	string	✓ 100
city	string	✓ Austin
zipcode	string	✓ 78758

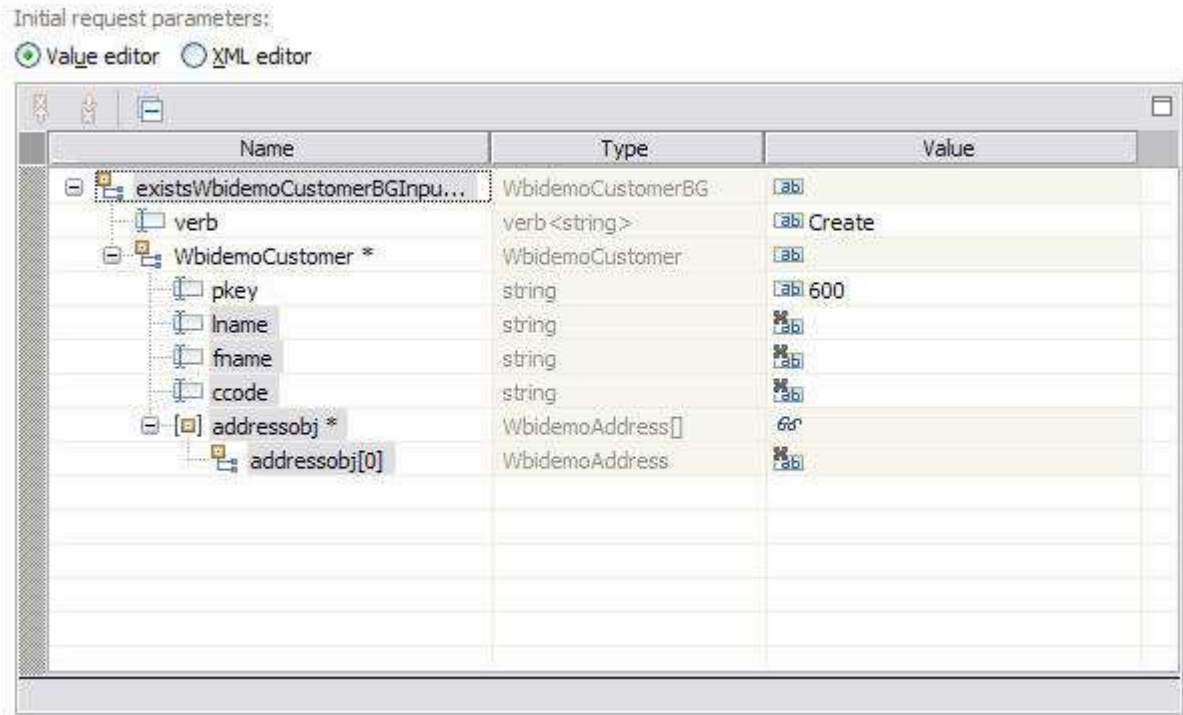
- ___ 6. To test a retrieveAll, select the **Invoke** button, then under **Detailed Properties**, select the Operation **retrieveAllWbidemoCustomerBG**
- ___ a. Set everything to <unset> by highlighting all the fields and selecting the <unset> from the drop down menu.
- ___ b. Click **Continue**.

Initial request parameters:

Value editor XML editor

Name	Type	Value
retrieveallWbidemoCustomerBGInput	WbidemoCustomerBG	[ab]
verb	verb <string>	[ab] Create
WbidemoCustomer *	WbidemoCustomer	[ab]
pkey	string	✗ [ab]
lname	string	[ab]
fname	string	✗ [ab]
ccode	string	✗ [ab]
addressobj *	WbidemoAddress[]	✗ [ab]
addressobj[0]	WbidemoAddress	✗ [ab]

- ___ c. Upon return, the values for Customer table existing in the database should be displayed including values from the Address table.
- ___ d. View the return parameters box to check for the returned records scrolling as needed.
- ___ 7. To test whether records exist or not, select the **Invoke** button, then under **Detailed Properties**, select the Operation **existsWbidemoCustomerBG**. This returns the status of the query and numbers of records based on the input criteria.
 - ___ a. Under **Initial request parameters**, enter a value for pkey of one of the previously created customers.
 - ___ b. Set lname, fname, and ccode to <unset> by clicking on the field and selecting the <unset> option from the drop down menu. Click **Continue**.



- ___ c. Upon return, the result should be displayed with Boolean value whether a particular record exists in the database or not. RecordCount returns the number of records found in the database for the input criteria.

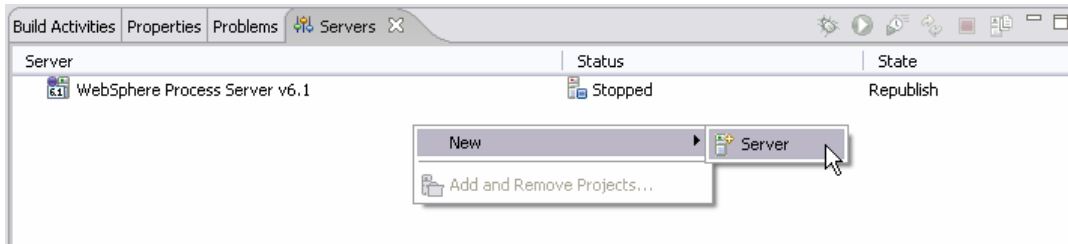
What you did in this exercise

- In this exercise, you learned how to create a multiple cardinality business objects Hierarchy using External Service Wizard, instead of editing the parent and child Business Objects using Business Object Editor.

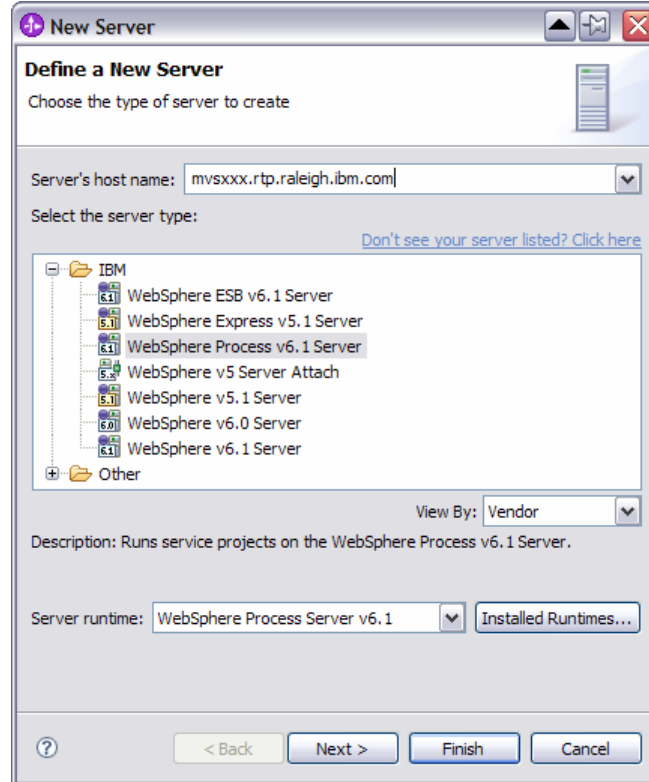
Task: Adding remote server to the WebSphere Integration Developer test environment

This task describes how to add a remote server to the WebSphere Integration Developer Test environment. This example uses a z/OS machine.

- ___ 1. Define a new remote server to WebSphere Integration Developer.
 - ___ a. Right click the background of the **Servers** view to access the pop-up menu.
 - ___ b. Select **New → Server**.

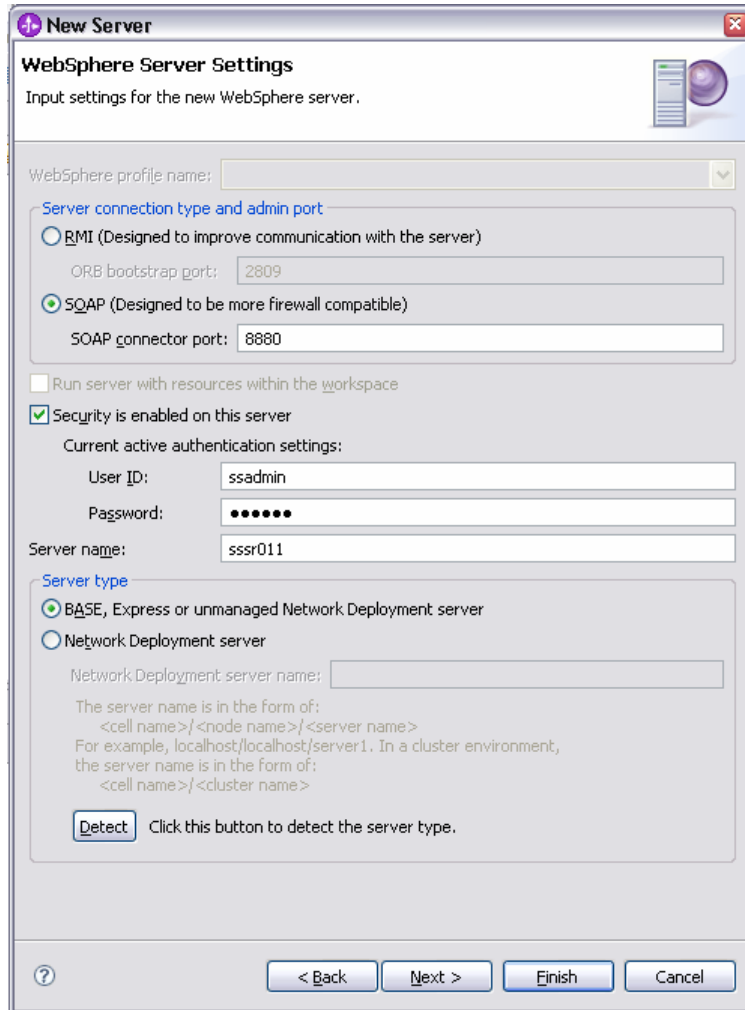


- ___ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.
- ___ d. Ensure that the appropriate server type, **'WebSphere Process V7.0 Server'** or **'WebSphere ESB V7.0 Server'**, is highlighted in the server type list

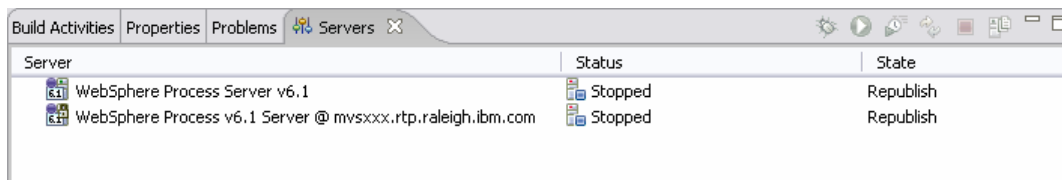


- ___ e. Click **Next**.

- ___ f. On the WebSphere Server Settings page, leave the radio button for **SOAP** selected, changing the **SOAP connector port** to the correct setting (<SOAP_PORT>). If security is on in your server, check the box for 'Security is enabled on this server' and input <USERID> for userID and <PASSWORD> for the password.



- ___ g. Click **Finish**.
- ___ h. The new server should be seen in the Server view.



- ___ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server View.
- ___ a. From a command prompt, telnet to the remote system if needed:

'telnet <HOSTNAME> <TELNET_PORT>'

User ID : **<USERID>**

Password : **<PASSWORD>**

__ b. Navigate to the bin directory for the profile being used:

cd <WAS_HOME>/profiles/<PROFILE_NAME>/bin

__ c. Run the command file to start the server: **./startServer.sh <SERVER_NAME>**

__ d. Wait for status message indicating server has started:

```
ADMU3200I: Server launched. Waiting for initialization status
```

```
ADMU3000I: Server sssr01 open for e-business; process id is 0000012000000002
```