

IBM WEBSHERE ADAPTER FOR JDBC V6.2 – LAB EXERCISE

## JDBC outbound lab – User defined SQL

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.....	20
Part 5: Test the application using the WebSphere test environment and component test.....	25
What you did in this exercise .....	30
Task: Adding remote server to the WebSphere Integration Developer test environment .....	31

### 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 V6.2 installed
- WebSphere Process Server V6.2 test environment installed
- WebSphere Adapter for JDBC V6.2 installed
- Sample code in the directory C:\Labfiles62\JDBC (Windows) or /tmp/LabFiles62/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 business objects for user-defined insert/update/delete SQL statements and executing the batch of SQL statements using EXECUTE operation .

---

## 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 JDBCTEST Derby database that contains a CUSTOMER table. In the lab, you 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. Then, you continue to assemble an SCA application, wiring together a stand-alone reference and the EIS import file. To test your application, you 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 can be Windows operating-system specific. If you plan on running the lab on an operating-system other than Windows, you 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®/UNIX® Location
<LAB_NAME>	JDBCOutbound	
<WID_HOME>	C:\Program Files\IBM\WebSphere\ID\6.2	
<WPS_HOME>	<WID_HOME>\runtimes\bi_v62	
<JDBCADAPTER_HOME>	<WID_HOME>\Resource Adapters\JDBC_6.2.0.0\	
<LAB_FILES>	C:\Labfiles62	/tmp/Labfiles62
<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, C:\LabFiles62\ is replaced by C:/LabFiles62/.

---

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 <sup>®</sup> 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<sup>®</sup> or Solaris, can be found at the end of this document, in 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 are going to the JDBCTEST database along with the CUSTOMER and ADDRESS table for user data.

If you choose to run the JDBC inbound lab first, then you only need to create additional table, ADDRESS for this exercise. If you choose to run JDBC Outbound Hierarchy BO lab first, then you can skip this section 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/LabFiles62/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.

---

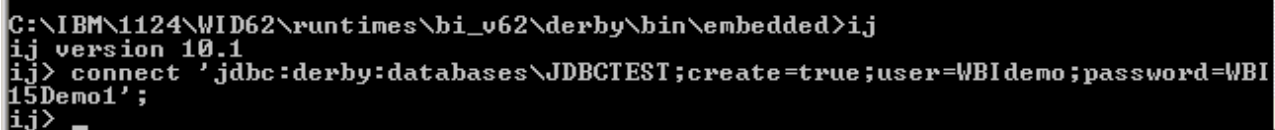
- a. Open a command prompt window, navigate to this subdirectory, and run Derby's interactive JDBC scripting tool.

```
<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. You can skip to step7, where you add records to the CUSTOMER table.

- 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';
```



```
C:\IBM\1124\WID62\runtimes\bi_v62\derby\bin\embedded>ij
ij version 10.1
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, the following SQL code snippets can be found in two locations: <LAB\_FILES>\JDBC\snippets\EVENTSTORESQL.txt or <JDBCADAPTER\_HOME>\Scripts

---

- a. Paste this into the command window:

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

```
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, the following SQL code snippets can be found in two locations:  
<LAB\_FILES>\JDBC\snippets\EVENTSTORESQL.txt or <JDBCADAPTER\_HOME>\Scripts

---

\_\_\_ b. Paste this into the command window:

```
CREATE TABLE ADDRESS  
(  
    addrid VARCHAR(10) NOT NULL PRIMARY KEY,  
    custid VARCHAR(10),  
    city VARCHAR(20),  
    zipcode VARCHAR(10)  
);
```

```
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\1124\WID62\runtimes\bi_v62\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.

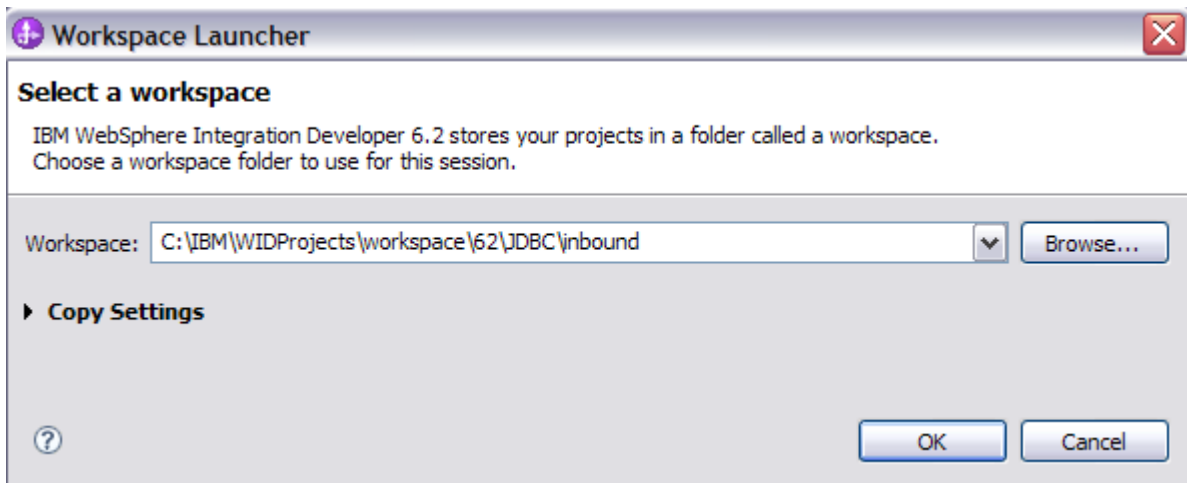
---

Rememer 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 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 V6.2 with a new workspace.
  - \_\_\_ a. From the start menu select **Start > Programs > IBM Software Development Platform > IBM WebSphere Integration Developer 6.2 > IBM WebSphere Integration Developer V6.2**
  - \_\_\_ b. When prompted enter **<LAB\_FILES>\JDBC\outbound-UserDefinedSQL** for your workspace and click **OK**



- \_\_\_ c. When WebSphere Integration Developer V6.2 opens, close the **Welcome page**



- \_\_\_ 2. **Add remote server to the Servers view:** If you have your WebSphere Process Server V6.2 on a remote machine, follow the instructions of the lab "**WPIv62\_AdapterAddRemoteServerTask**" to add the remote server to your Servers view of WebSphere Integration Developer V6.2

---

## Part 3: Create the JDBC Outbound application

In this part you 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 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**

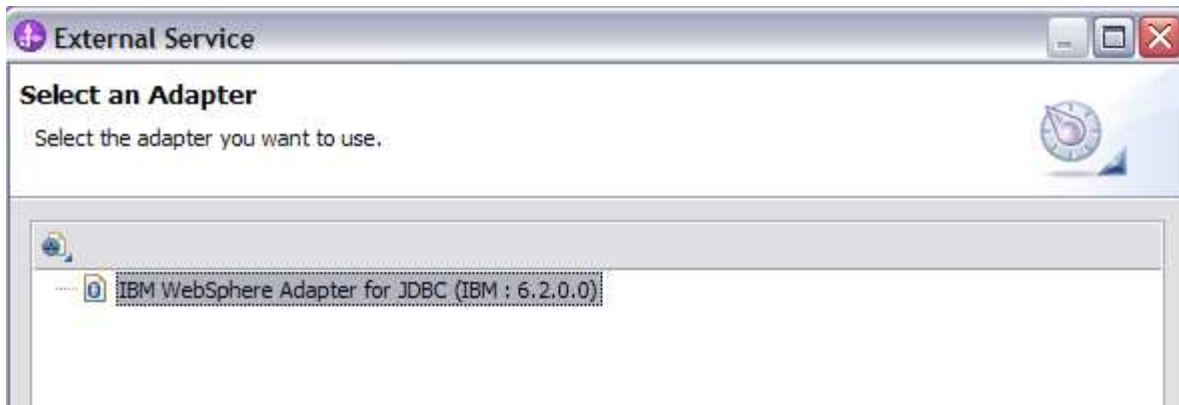
Keep this telnet window open during the next few steps; this window shows when a connection has been successfully made from WebSphere Integration Developer to the database server.

**\*\*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 allows 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: 6.2.0.0)** and click **Next**

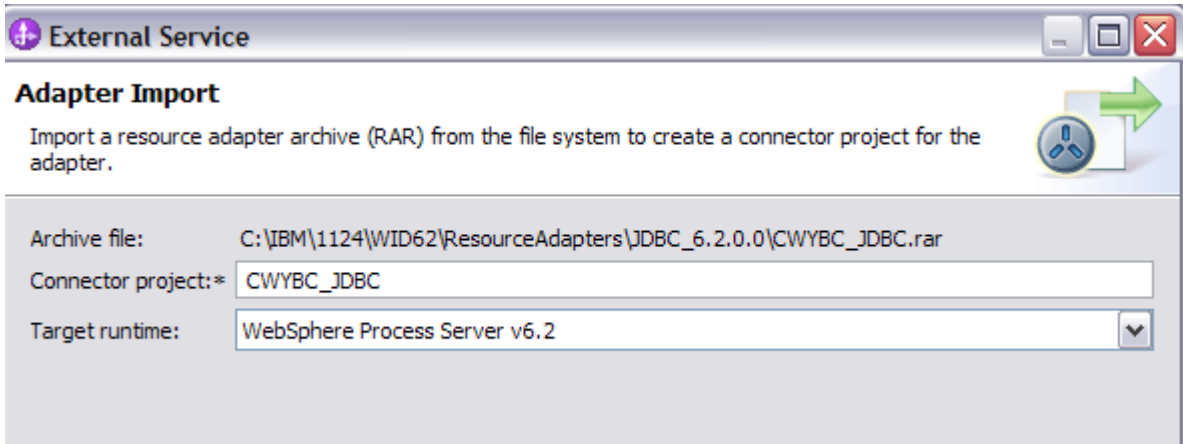


\_\_\_ 4. Adapter Import screen:

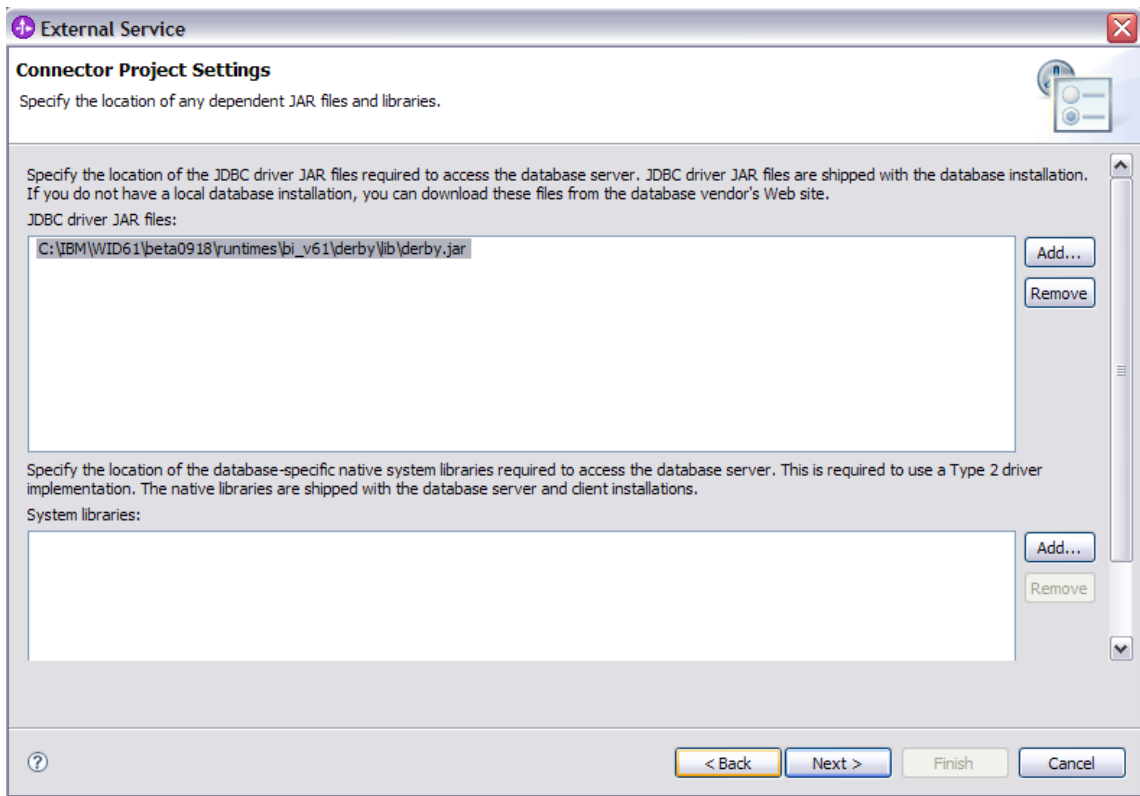
In this step, you will import a connector resource adapter archive from the file system into your WebSphere Integration Developer workspace. The adapter RAR file already exists under **<JDBCADAPTER\_HOME>**.

- \_\_\_ 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 v6.2** is selected

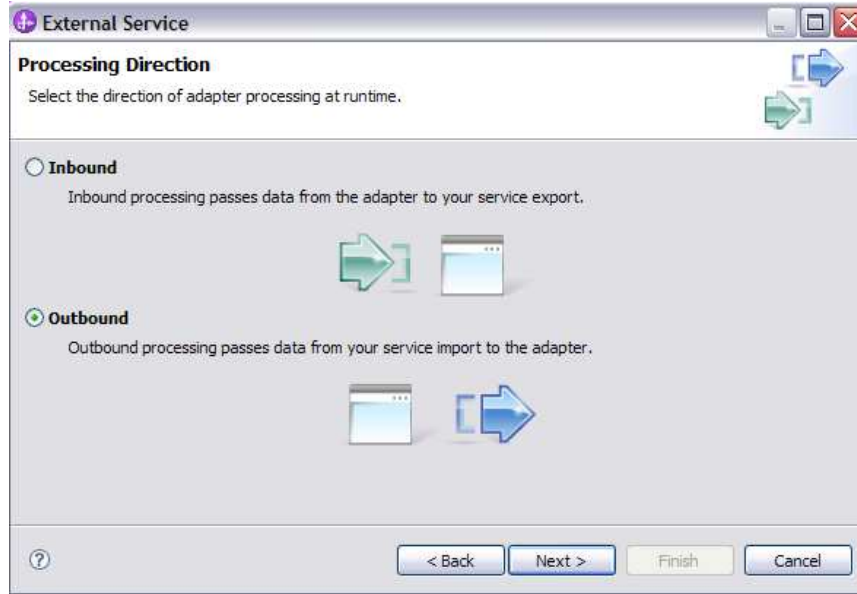




- \_\_\_ 5. Add any external dependencies your adapter has to the imported project. These are dependencies that the adapter can have on the JDBC applications (adapter-specific).
  - \_\_\_ a. Click **Add** and browse to the location of c:\<WPS\_HOME>\derby\lib and select the **derby.jar**. If you are using a remote test environment on z/OS, you 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 performs 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\_v62\derby\databases\JDBCTEST  
 JdbcDriverClass: com.ibm.db2j.jdbc.DB2jDriver

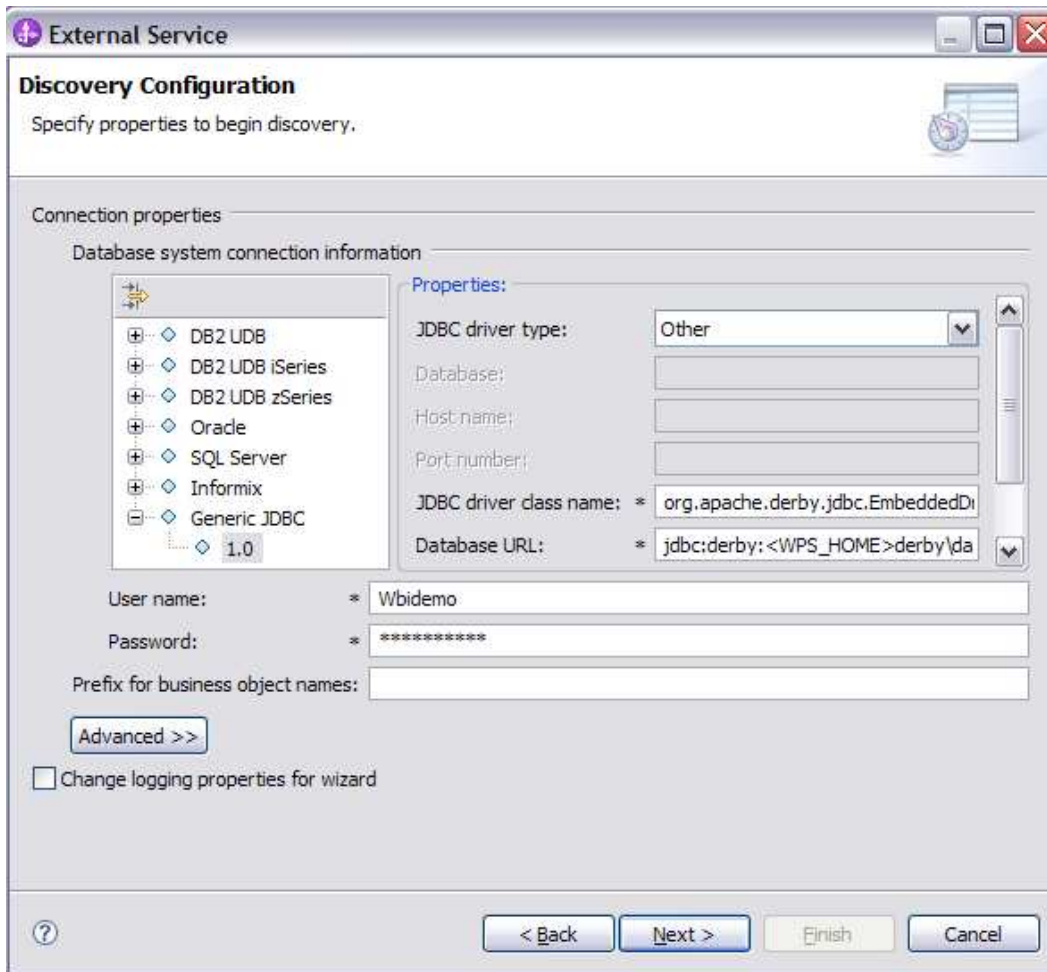
---



---

**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**.

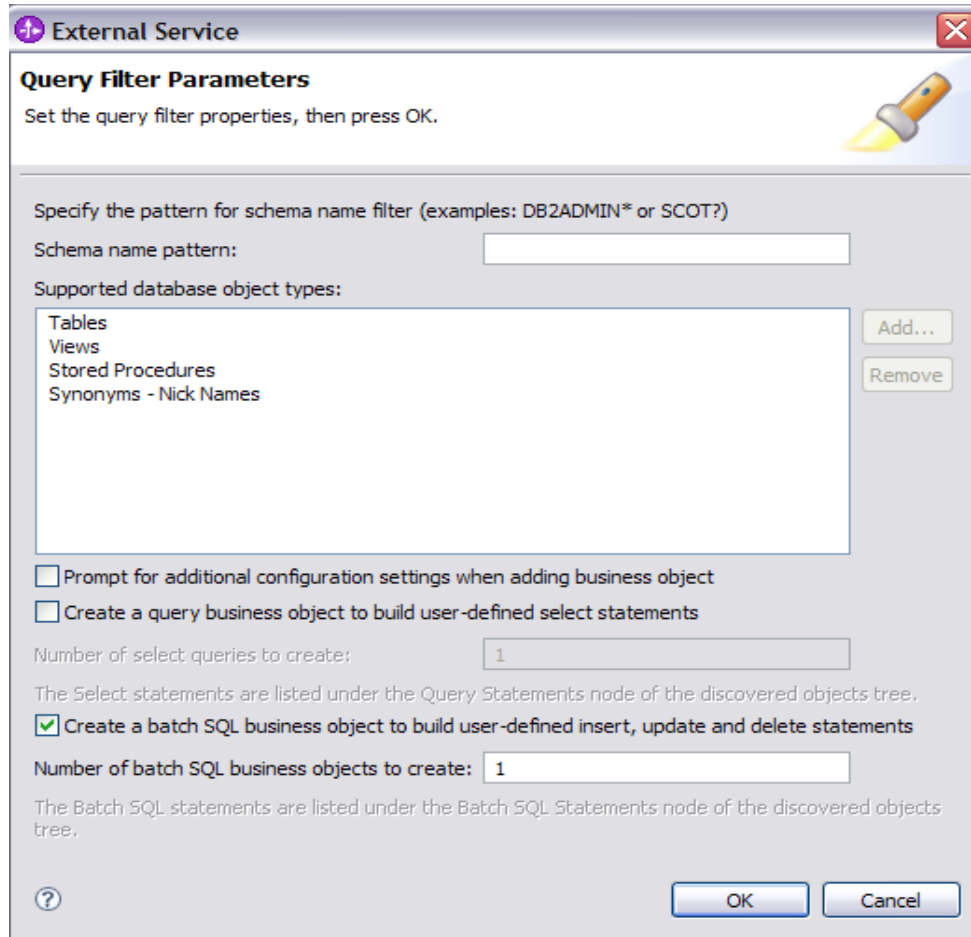
\_\_\_ 8. Complete the **Find and Discover Enterprise Services** panel

\_\_ a. Click the **Edit Query** button


\_\_ b. In the Query Filter Properties window that opens up, check **Create a batch SQL business object to build user-defined insert, update and delete statement** check box,

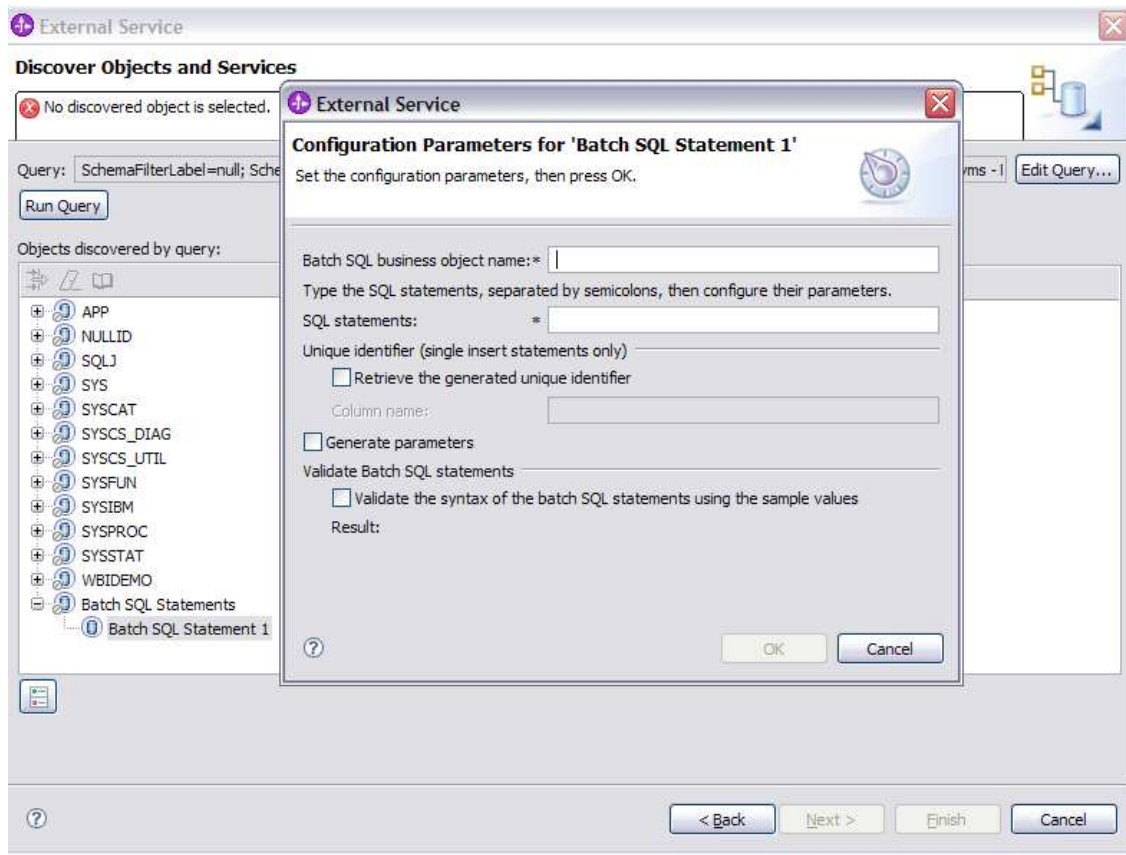
\_\_ c. You can enter the number of user-defined queries that you want to enter. In this exercise, there is only one user-defined query so keep the default value.

\_\_ d. Click **OK**.



- \_\_\_ e. Click **Run Query** button. A connection is made to the Derby JDBCTEST database and lists the tables, stored procedures, etc, sorted by schema name. It also lists 'Batch SQL statements' as a top level node for creation of user-defined insert/update/delete BOs.

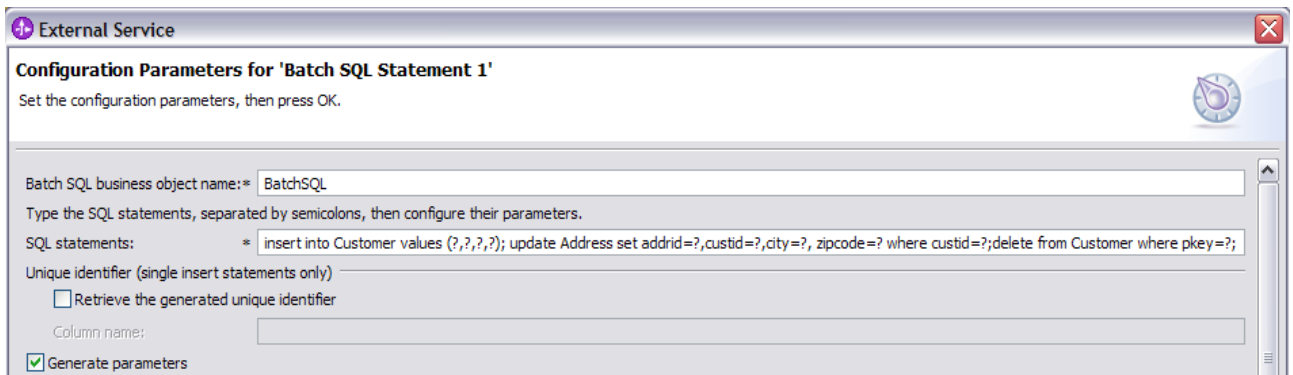
- \_\_\_ f. Expand **Batch SQL Statements** and highlight **Batch SQL Statements 1**, click the  button.
- \_\_\_ g. It opens up a window for setting the Configuration Parameters for **Batch SQL Statement 1**.



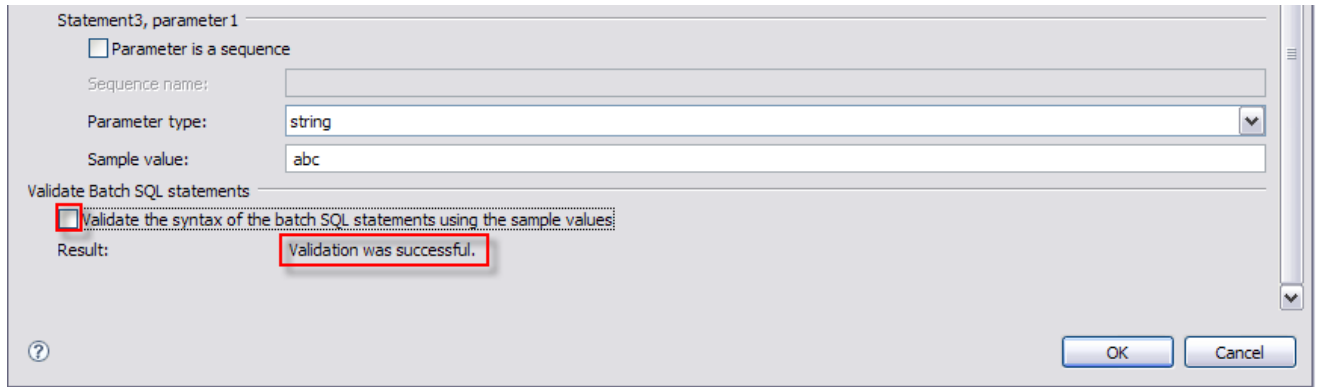
- \_\_\_ h. Enter the name for the **Batch SQL Statements** as **BatchSQL** and this statement in the **SQL Statements** property separated by a semicolon.

*insert into Customer values (?, ?, ?, ?);  
 update Address set addrid=? where custid=?;  
 delete from Customer where pkey=?;*

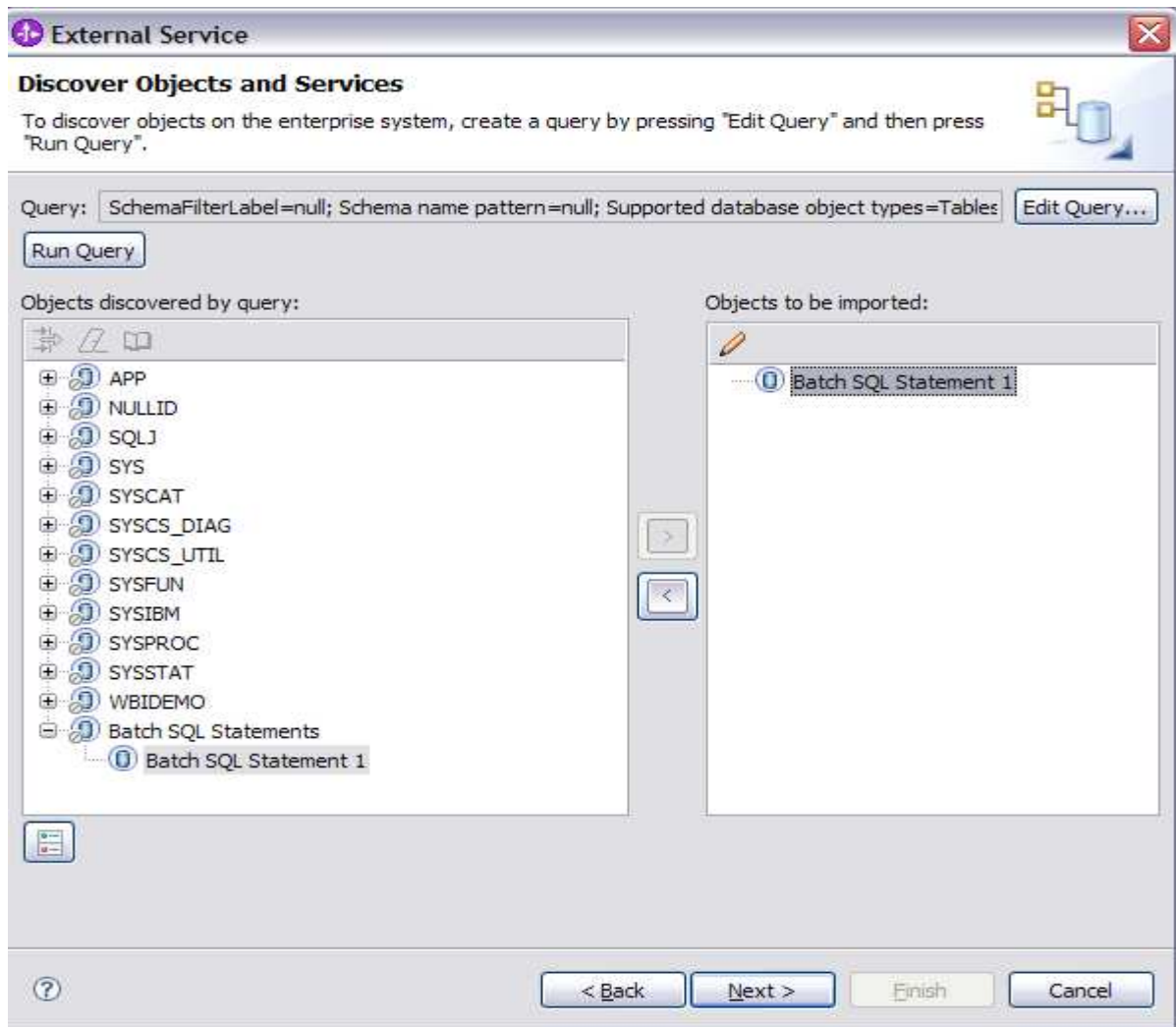
- \_\_\_ i. Check **Generate Parameters**



- \_\_\_ j. Enter dummy values for the **Sample value** for each parameter value and check **Validate the syntax of the batch SQL statements using the sample values** to verify if the user-defined statement is validated.

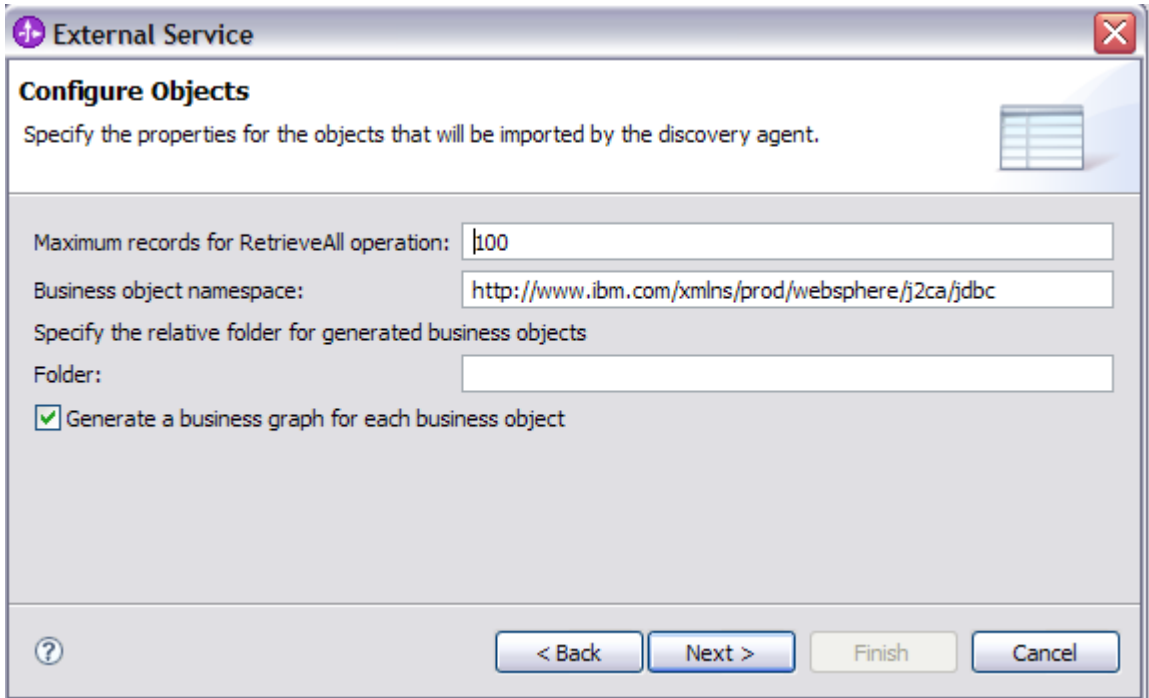


- \_\_\_ k. Click **Ok**



- \_\_\_ l. Click **Next**

- \_\_\_ 9. Complete the **Configure Objects** panel
  - \_\_\_ a. Leave the default value for Namespace, leave blank for **BOLocation**. Note the Operations is not available as there is only one operation **Execute** for user defined SQL statements.
  - \_\_\_ b. Note that Business Graph is optional - It determines 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.



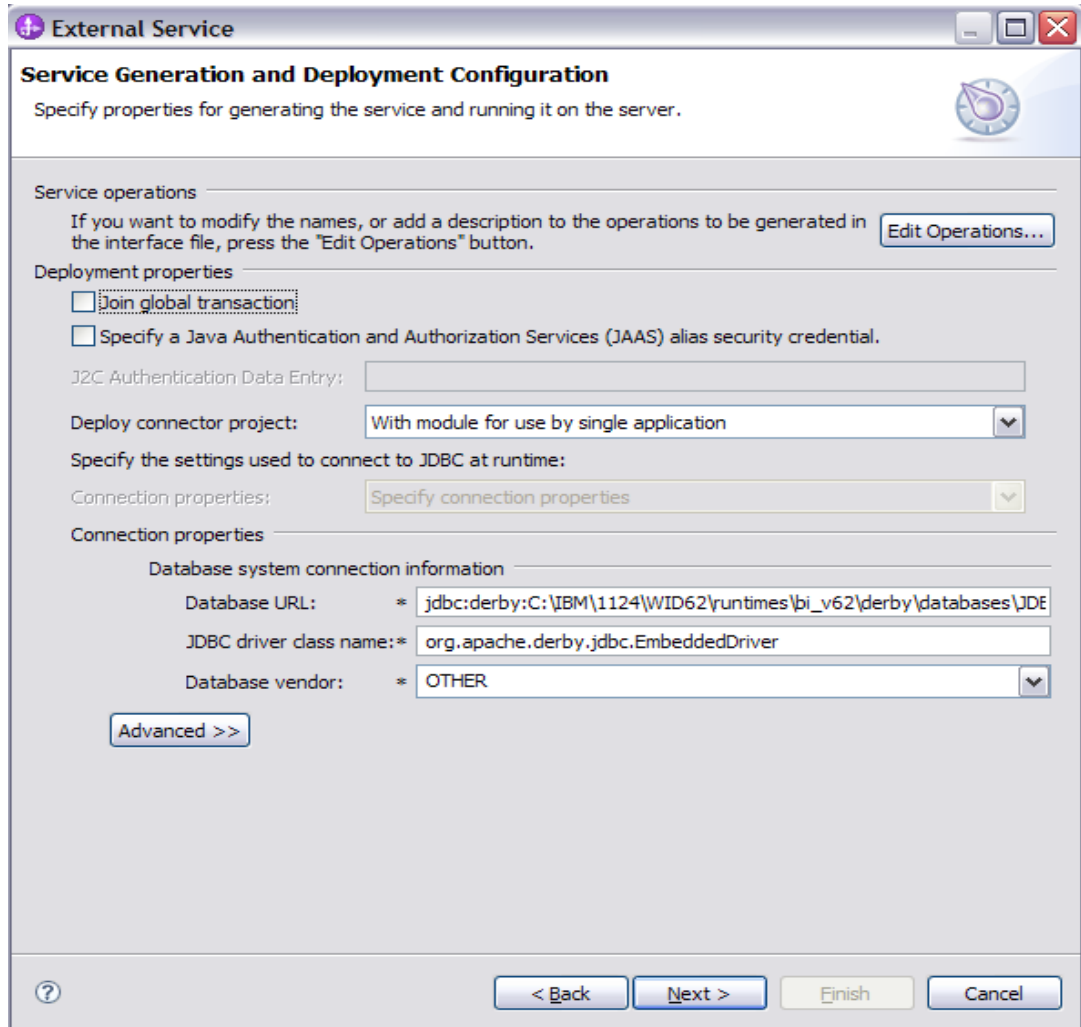
- \_\_\_ c. Click **Next**
- \_\_\_ 10. Complete Publishing Object Configuration Properties
    - \_\_\_ a. Leave the default **“With module for use by single application”** option in this exercise.

---

**Note:** All IBM WebSphere Adapters are now supported deploying RAR separately. To do so, select “With module for use by single application” from the Deploy connector project drop down list. If you want the adapter to be deployed with the module which is packaged as an Enterprise Archive file (EAR file), then the “On Server for use by multiple applications” option should be selected.

---

- \_\_\_ b. Clear the check box for **Join Global Transaction**. By clearing it, you are going to run this component in a new global transaction.
- \_\_\_ c. Specify the **JAAS Alias security credential** as JNDI data source name is optional for Outbound service. For this lab, you are going to define the data source JNDI name, so clear the JAAS option.
- \_\_\_ d. Under **Connection properties of Deployment Properties**
  - 1) 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.)



- \_\_\_ e. Click **Advanced** button for additional properties options
- \_\_\_ f. **Optional** - expand **Advanced connection configuration** and enter this following value
  - 1) Data Source JNDI Name: **jdbc/Derby XA for JDBC**

---

**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.

---



\_\_\_ g. **Optional** for logging and tracing

- 1) 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.
- 2) 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:

- Alternative ways to specify connection information:** Includes fields for Datasource JNDI name (jdbc/Derby XA for JDBC), XA DataSource name, and XA database name.
- Advanced connection configuration:** Includes a checkbox for 'Set auto commit on database connection', a field for 'Additional JDBC driver connection properties [name:value;name:value]', a field for 'SQL query to verify the connection', a field for 'Query timeout (seconds)', and a dropdown for 'Return business object even when the stored procedure result set is empty' (set to false).
- Logging and tracing:** Includes a field for 'Adapter ID' (001) and a checked checkbox 'Disguise user data as "XXX" in log and trace files.' which is circled in red.
- Bidi properties:** A section header at the bottom.

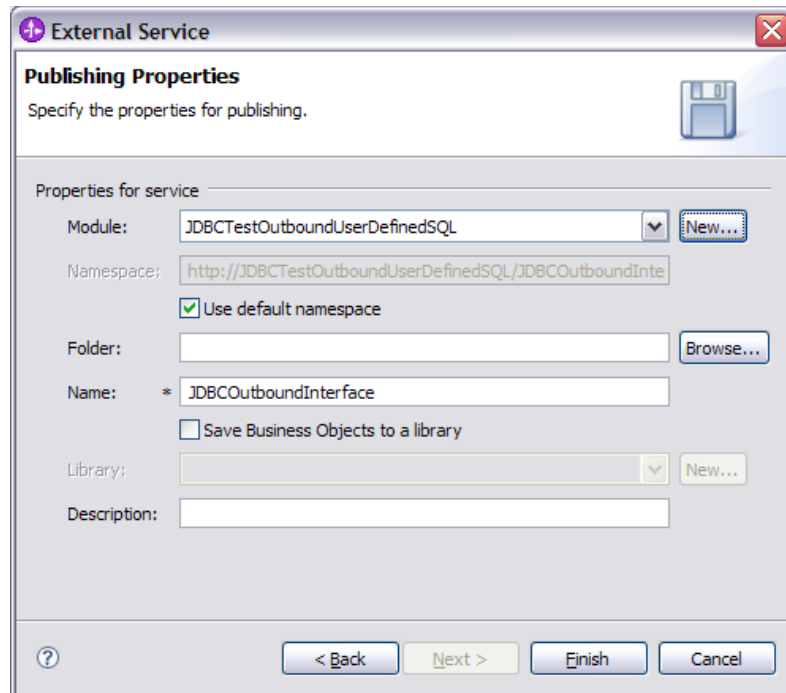
\_\_\_ h. Click **Next**

\_\_\_ 11. Complete Publishing Properties panel

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

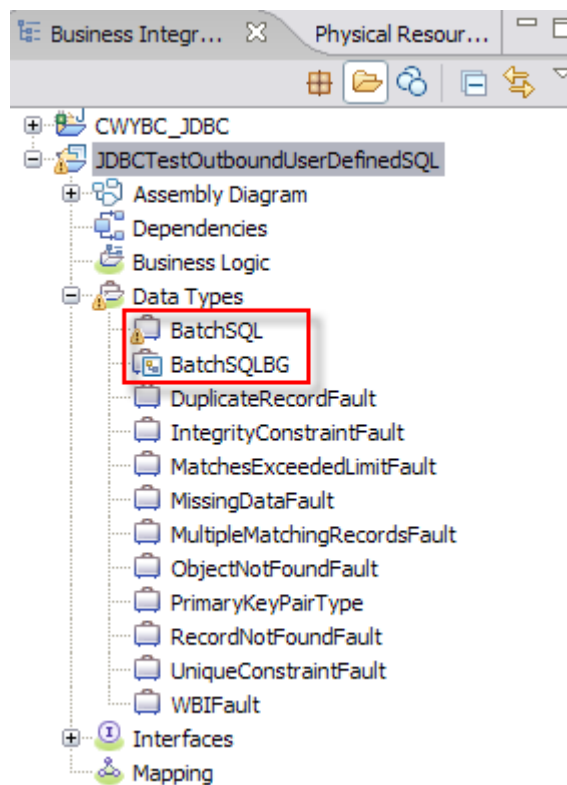
- 1) Select the **New** button Select the radio button "**Create a module project**" from new **Integration Project** pop up window.
- 2) Enter in the name **JDBCTestOutboundUserDefinedSQL** for the Module Name
- 3) Accept the defaults and click **Finish**

\_\_\_ b. Leave the default **JDBCOutboundInterface** for the Name value.



\_\_\_ c. Click **Finish**

\_\_\_ 12. Verify the result as given below



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

\_\_\_\_ 13. 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 needs 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.

---

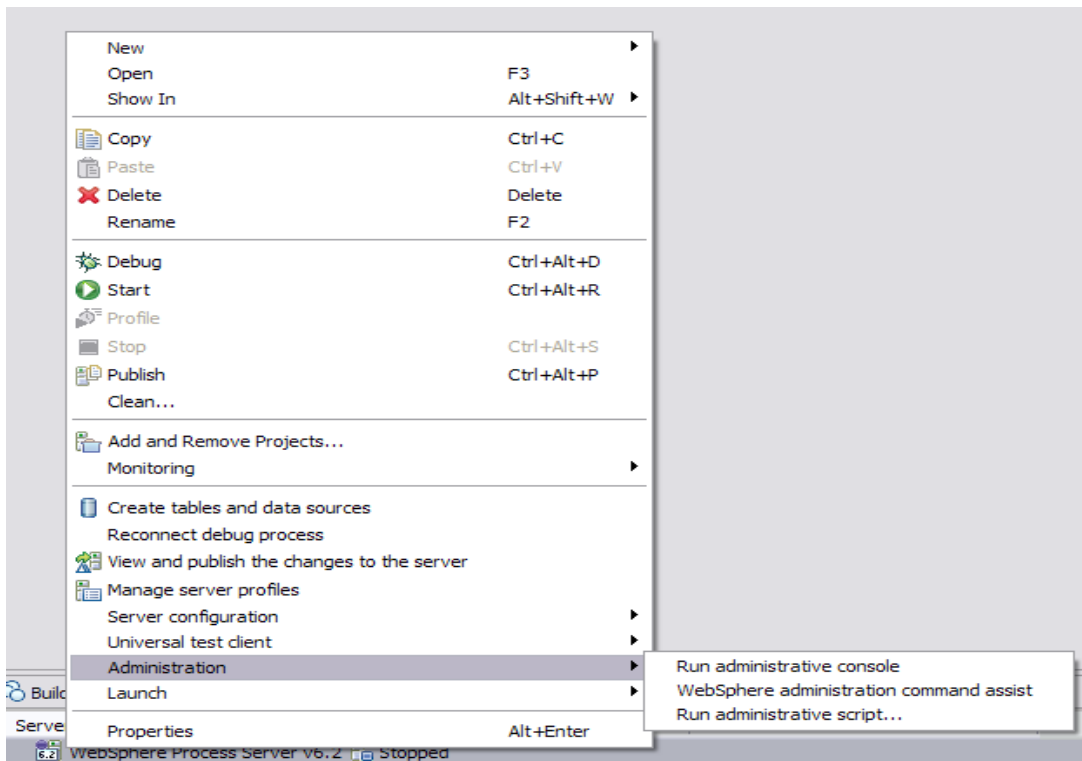
- \_\_\_ a. From the top menu bar, select **File > Switch Workspace** and select the same workspace from which you have been working.

## Part 4: Create J2C authentication alias and configure data sources

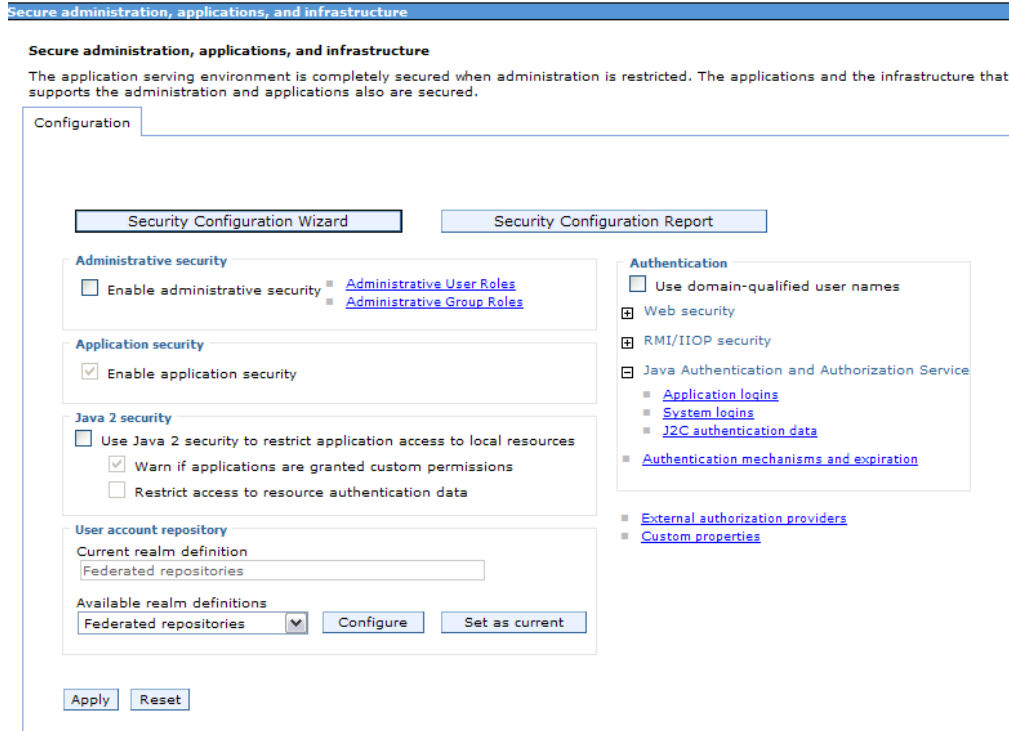
**Note:** This section is optional for outbound lab, you can skip this section and go to Part 5.

In this part, create a J2C Authentication Alias which is required for connection to the database. Also create the JNDI data source name that is used by the Adapter to configure itself to the endpoint. 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 **Run administrative console**.



- \_\_\_ c. Log in to the administrative console by clicking the “Log in” button
- \_\_\_ d. Click **Security** → **Secure administration, application, and infrastructure**
- \_\_\_ e. On the right, expand **Java Authentication and Authorization Service** under the Authentication heading.

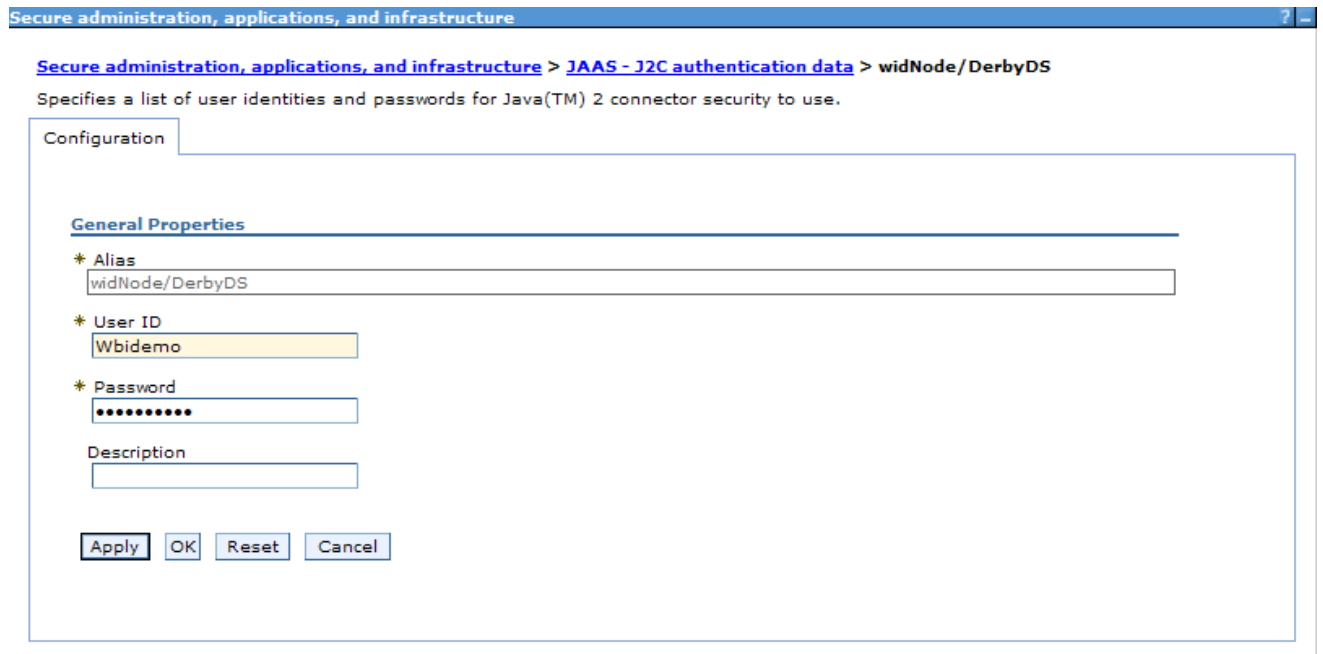


\_\_ f. Click **J2C authentication alias**. It gives the list of existing aliases.

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

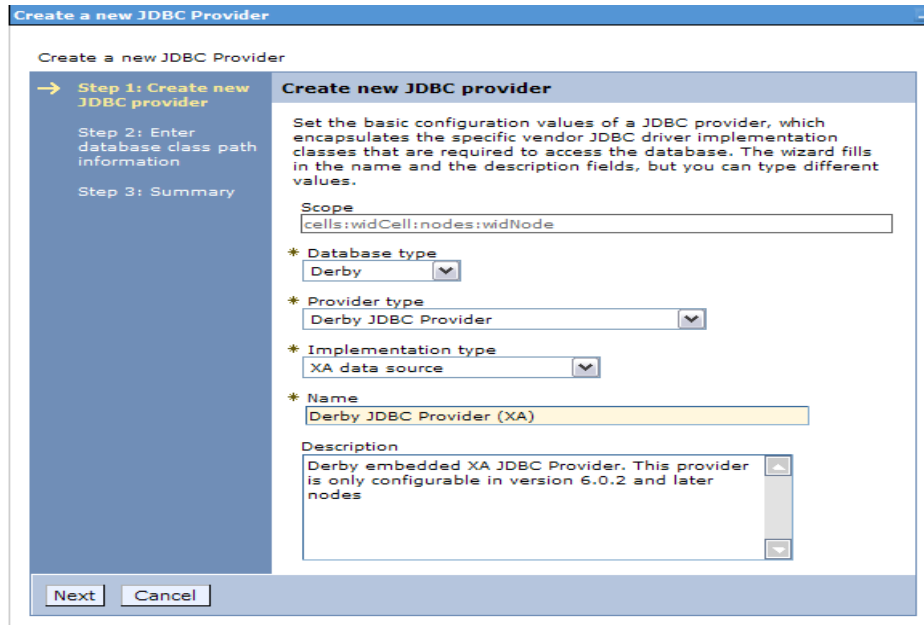
a) User ID: **Wbidemo**

b) Password: **Wbi15Demo1**



\_\_ 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. Set the Component-managed authentication alias to the one created in the earlier section.

**Create a data source**

Step 1: Enter basic data source information

Step 2: Enter database specific properties for the data source

Step 3: Summary

**Enter basic data source information**

Set the basic configuration values of a data source for association with your JDBC provider. A data source supplies the physical connections between the application server and the database.

Requirement: Use the Data sources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans (TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

Scope  
cells:widCell:nodes:widNode

JDBC provider name  
Derby JDBC Provider (XA)

\* Data source name  
Derby JDBC Driver XA DataSource

\* JNDI name  
jdbc/Derby XA for JDBC

Component-managed authentication alias and XA recovery authentication alias  
Select a component-managed authentication alias. The selected authentication alias will also be set as the XA recovery authentication alias if your JDBC Provider supports XA. If you choose to [create a new J2C authentication alias](#), the wizard will be canceled.

widNode/DerbyDS

Next Cancel

\_\_\_ d. Click **Next**.

\_\_\_ e. Enter the full path to the location of database name **JDBCTEST**.

**Create a data source**

Step 1: Enter basic data source information

Step 2: Enter database specific properties for the data source

Step 3: Summary

**Enter database specific properties for the data source**

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through this data source.

\* Database name  
\${WBI\_INSTALL\_ROOT}/derby/databases/JDBC

Use this data source in container managed persistence (CMP)

Previous Next Cancel

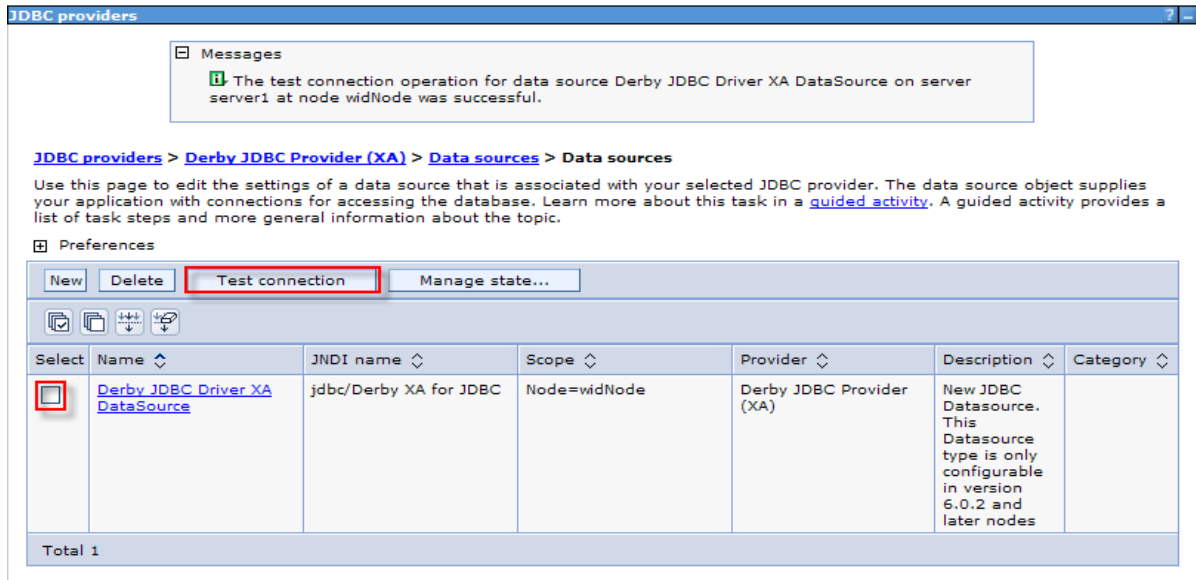
\_\_\_ f. Click **Next** and view the summary

\_\_\_ g. 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



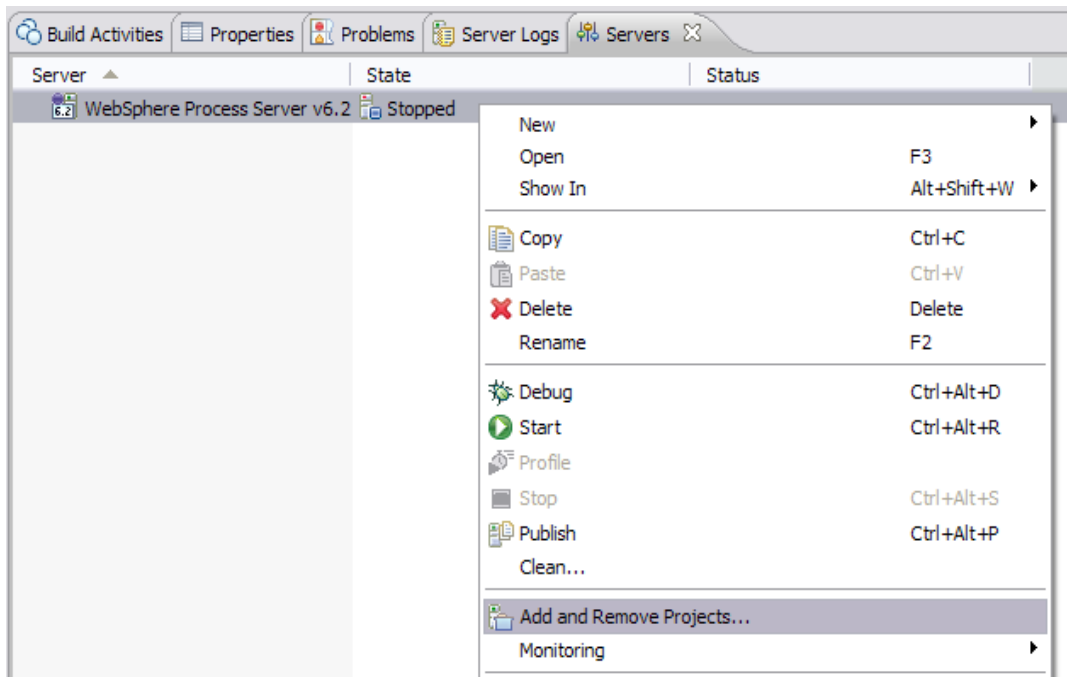
- \_\_\_ 6. Release the connection to the Derby database by using **Switch Workspace**
  - \_\_\_ a. From the top menu bar, select **File > Switch Workspace** and select the same workspace from which you have been working.



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

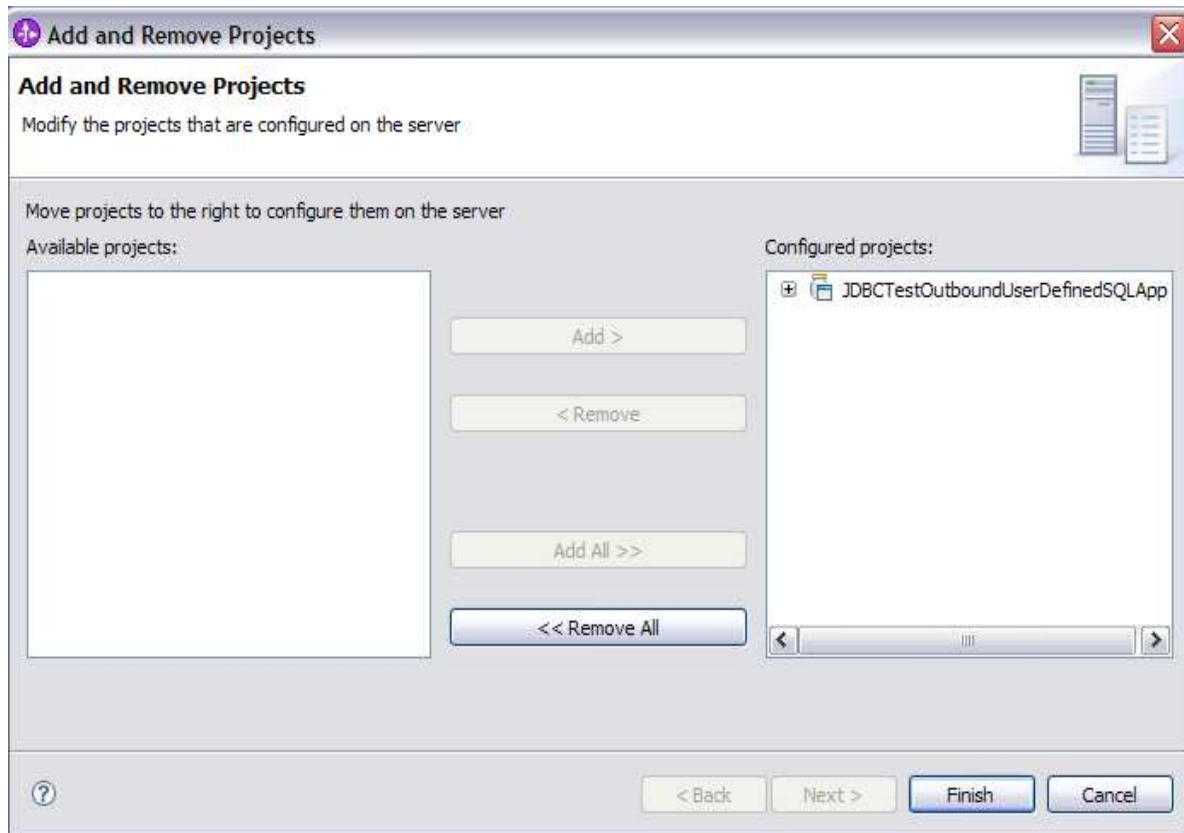
This lab is using the Derby embedded database driver which allows a connection from only a single JVM at a time. ij.bat, the External Service process, and a running Server configured with an application that connects to the database, each are an active connection. This means you can have only one of 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 clicking Server and selecting **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 **JDBCTestOutboundUserDefinedSQLApp** project from the Available projects panel.

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



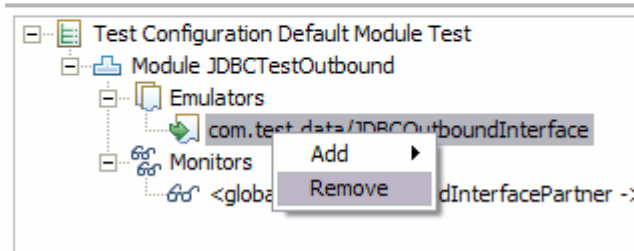
\_\_\_ 3. Use the Test Component to test the application. The create scenario is to create a record in the database.

\_\_\_ a. Right click the **JDBCTestOutboundUserDefinedSQL module** and select **Test → Test Component**

\_\_\_ b. In JDBCTestOutboundUserDefinedSQL\_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

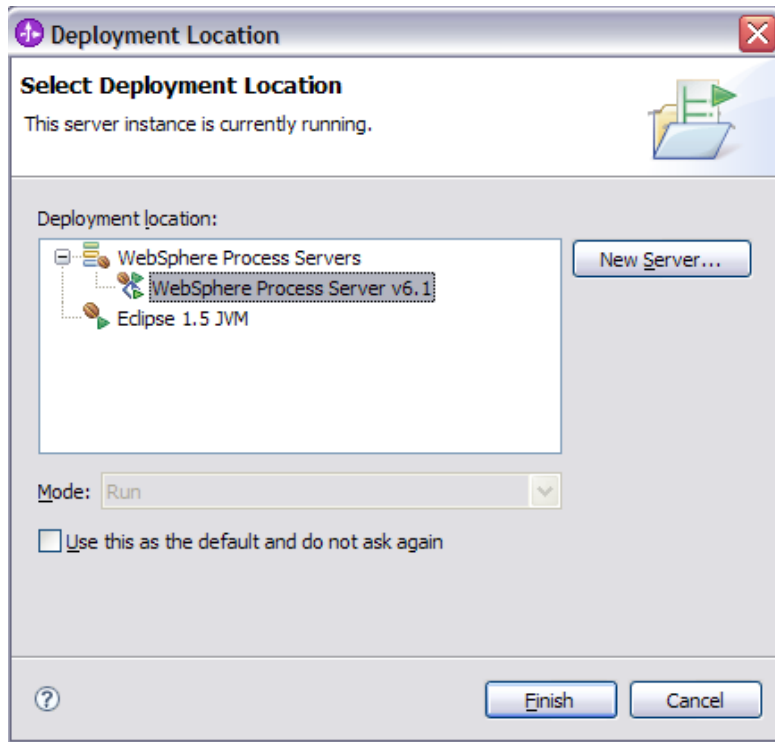
- 1) Under **Detailed Properties**, choose executeBatchSQLBG as this is the only operation.
- 2) Under **Initial request parameters**, enter these following values
  - a) **Verb** is depreciated so it is no longer required to enter.
  - b) Enter the values for each parameter. You do not see SQL statements here because SQL statements can not be change, and those statements are not stored as attributes.

Name	Type	Value
executeBatchSQLBGInput	BatchSQLBG	✓
verb	verb <string>	✗
BatchSQL	BatchSQL	✓
statement1parameter1	string	✓ 110
statement1parameter2	string	✓ Walker
statement1parameter3	string	✓ Jane
statement1parameter4	string	✓ Office
statement2parameter1	string	✓ 110
statement2parameter2	string	✓ 100
statement3parameter1	string	✓ 100
statement1status	int	✓ 0
statement2status	int	✓ 0
statement3status	int	✓ 0

\_\_ d. Click  to continue

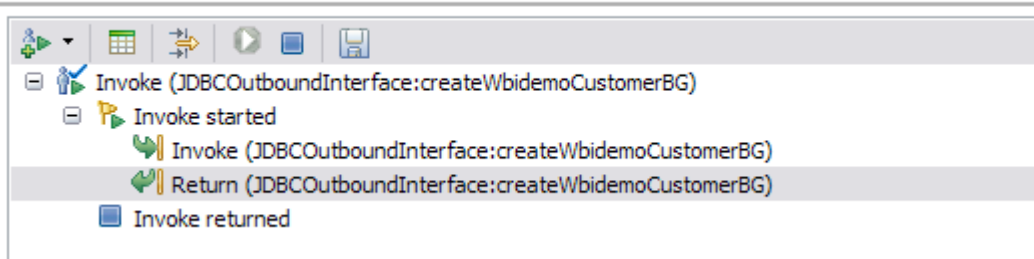


\_\_ e. In the **Choose a deployment location dialog**, select the **WebSphere Process Server V6.2** server. Select **Finish**



- \_\_\_ 4. In the Events window you should see that Invoke has returned.

**Events**



- \_\_\_ 5. Check the data in the EIS to ensure it matches expected values
- \_\_\_ a. Stop the **Server** by right clicking WebSphere Process Server V6.2 and selecting **Stop**.
  - \_\_\_ b. From the ij command window, connect to JDBCTEST database with user name as **Wbidemo** and password as **Wbi15Demo1**.
  - \_\_\_ c. After ran SQL statements in this exercise, this is the result you should see from JDBCTEST database.

1) Table **Customer**

- a) New record of pkey as **110**, lname as **Walker**, fname as **Jane**, and ccode as **Office**.
- b) Existing record which has pkey as 100 has been deleted from the table.

2) Table **Address**

- a) AddrId value in record which has custid as **100** is changed to **110** from **100**.

```

ij> connect 'jdbc:derby:databases\JDBCTEST;user=Wbidemo;password=Wbi15Demo1';
ij> select * from customer;
PKEY      !LNAME      !FNAME      !CCODE
-----
10        !Johnson   !Jerry      !Office
20        !Smith     !John       !Office
30        !Walker    !Jane       !Office
40        !Smith     !Johnson   !Office
50        !Smith     !Johnson   !Office
60        !Smith     !Johnson   !Office
110       !Walker    !Jane       !Office

7 rows selected
ij> select * from Address;
ADDRID    !CUSTID    !CITY      !ZIPCODE
-----
110       !100      !Austin    !78758

1 row selected
ij>

```

- \_\_\_ 6. Disconnect the database and exit.
- \_\_\_ 7. Exit the Test Component panel, remove the JDBCTestOutboundUserDefinedSQL project from the server, and stop the server.

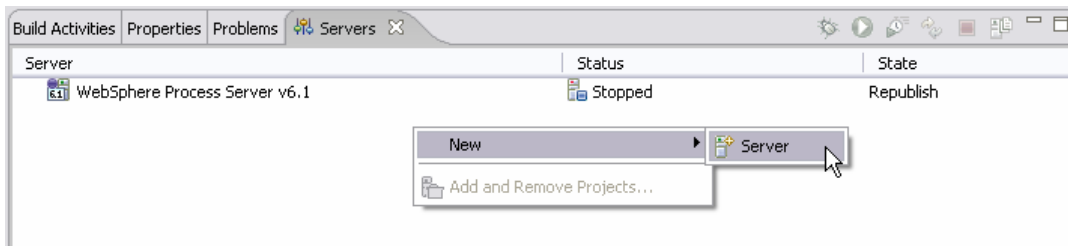
## What you did in this exercise

- In this exercise, you learned how to create a business object for an user defined insert/update/delete statements by using the Execute operation to process the records based on your user-defined SQL statements.

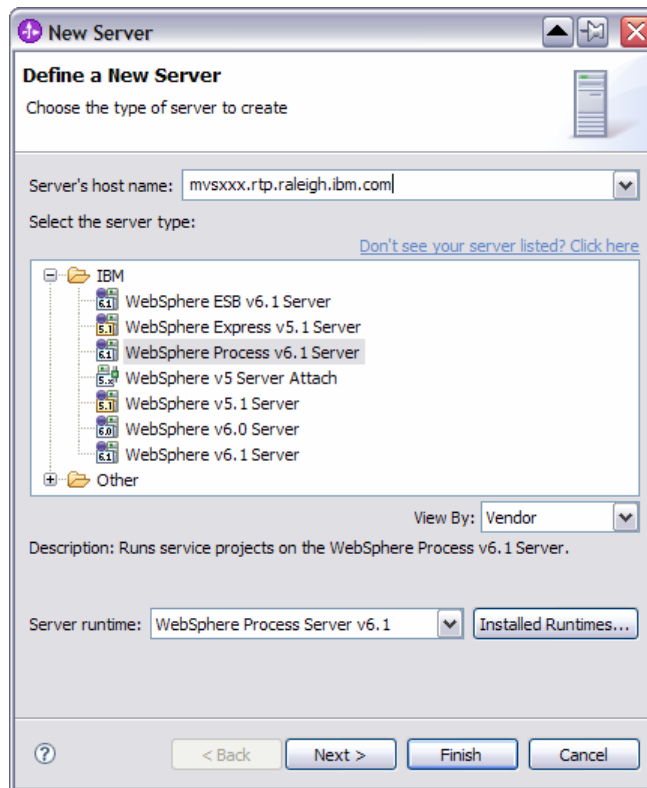
## Task: Adding remote server to the WebSphere Integration Developer test environment

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**.

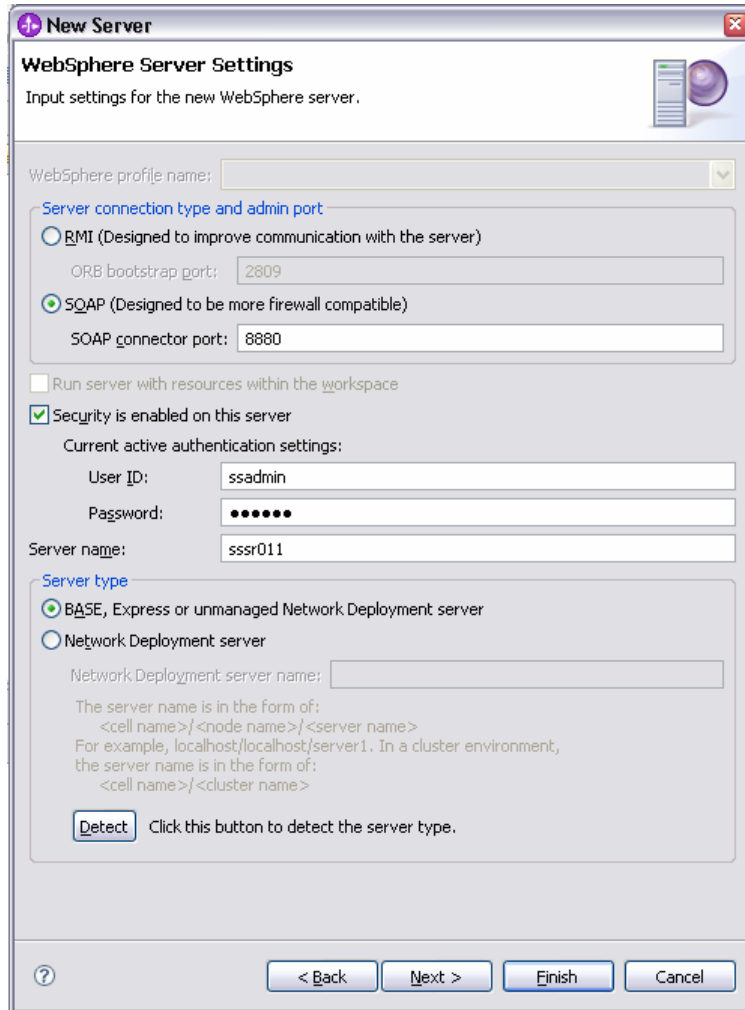


- \_\_\_ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.
- \_\_\_ d. Ensure that the appropriate server type, **'WebSphere Process v6.2 Server'** or **'WebSphere ESB v6.2 Server'**, is highlighted in the server type list

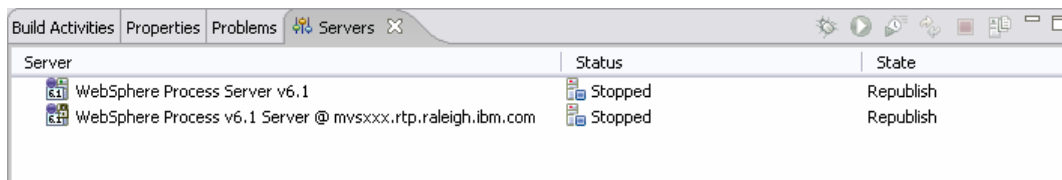


- \_\_\_ e. Click **Next**.

- \_\_\_ 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 the user ID 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
```