

Oracle EBS Adapter Sample – Outbound processing

This edition applies to version 6, release 1, modification 0 of IBM® WebSphere® Adapter for Oracle E-Business Suite on WebSphere Application Server (product 5724-T73) and to all subsequent releases and modifications until otherwise indicated in new editions. To send us your comments about this document, email <mailto://doc-comments@us.ibm.com>. We look forward to hearing from you. When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you. © **Copyright International Business Machines Corporation 2007. All rights reserved.** US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

This sample uses Oracle database interface tables, which is the standard outbound scenario for the Oracle E-Business Suite, to populate data to the Oracle base tables, and then shows how to do a retrieve of the customer data from those base tables. The sample uses the Create operation and then the Retrieve operation through the Adapter for JDBC.

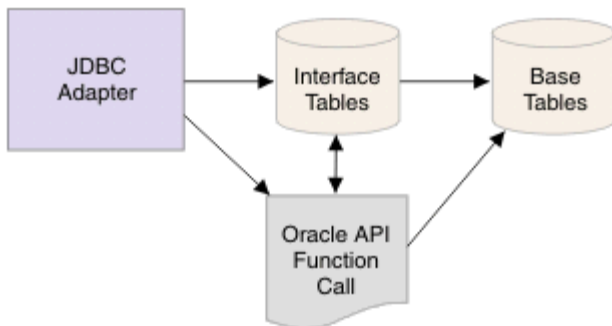
Outbound processing using Interface tables

The first outbound request processing sample has two parts:

- Creating a generated Java™ data binding (business object) by using a Create operation
- Retrieving an object by using a Retrieve operation

The Oracle database permits the retrieval of data from the application's base tables. The Java data bindings used for the Retrieve operation reflect the base tables' representation of the data. The Oracle database does not permit the manual modification of data in the base tables. Thus, this sample uses the interface tables when the Adapter for JDBC sends business data that change the content in the Oracle database. This is the standard outbound request processing scenario. After the interface tables are updated with the business data, a function is called by the adapter, through a stored procedure, that moves the data from the interface to the base tables.

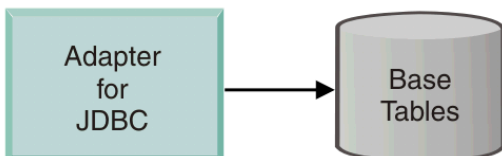
Figure 1. Create scenario



The Adapter for JDBC is used to populate the interface tables with business data. Because updates cannot be made directly to the Oracle base tables, the Update operation cannot be used. The Create operation is used to process both new and changed data through the Oracle interface tables. The business data can be moved into the Oracle base tables by invoking an internal, standard Oracle function. The function is called through a stored procedure to move the data to the base tables using AfterCreateSP application-specific information on the business object.

The Create and Retrieve operations use different sets of objects, Java data bindings, because they represent different tables in the Oracle database. This sample application includes the wrapper stored procedure for the function call to move the data to the base tables, plus the sample content to use for the procedure.

Figure 2. Retrieve scenario



Business object selection and linking

This sample uses one set of primary and child objects to populate customer data into the Oracle E-Business Suite interface tables, and a second set of objects to process the retrieve. The Create operation is used to post new customer data and updates to existing customer data. You must configure five business objects for the Create operation. The Retrieve operation is used to retrieve business data from the Oracle base tables. You must configure nine business objects for the Retrieve operation.

Business objects in the database tables for the Create operation

The Customer Profile object is the primary object, and the rest of the objects are structured as child business objects. You link the business objects by setting the application-specific information foreign key relationship between the parent and child objects. The following table lists the database tables for the Accounts Receivable (AR) schema including the table name, suggested synonym, and a brief description of the information that each object contains. Note: For more information about the table data, refer to your documentation for Oracle Applications, particularly the section of the Receivables user guide that describes the customer interface.

Table 1. Database tables for the AR schema

Table name	Synonym	Description
AR.RA_CUSTOMER_PROFILES_INT_ALL	Ibm_C_Cust_Profile	Customer Profile interface table, which contains high-level information about the customer
AR.RA_CUSTOMERS_INTERFACE_ALL	Ibm_C_Cust_Interface	Primary table for customers, as well as customer addresses
AR.RA_CONTACT_PHONES_INT_ALL	Ibm_C_Contacts	Contact and phone information
AR.RA_CUST_PAY_METHOD_INT_ALL	Ibm_C_Pay_Methods	Payment methods associated with the customer
AR.RA_CUSTOMER_BANKS_INT_ALL	Ibm_C_Cust_Banks	Banks associated with the customer

Business objects in the database for the Retrieve operation

The following table lists the base tables for the Accounts Receivable (AR) schema, including the table name, suggested synonym, and description. The next section shows a diagram of the relationship of the business objects that you configure for the Retrieve operation.

Table 2. Base tables for the AR schema

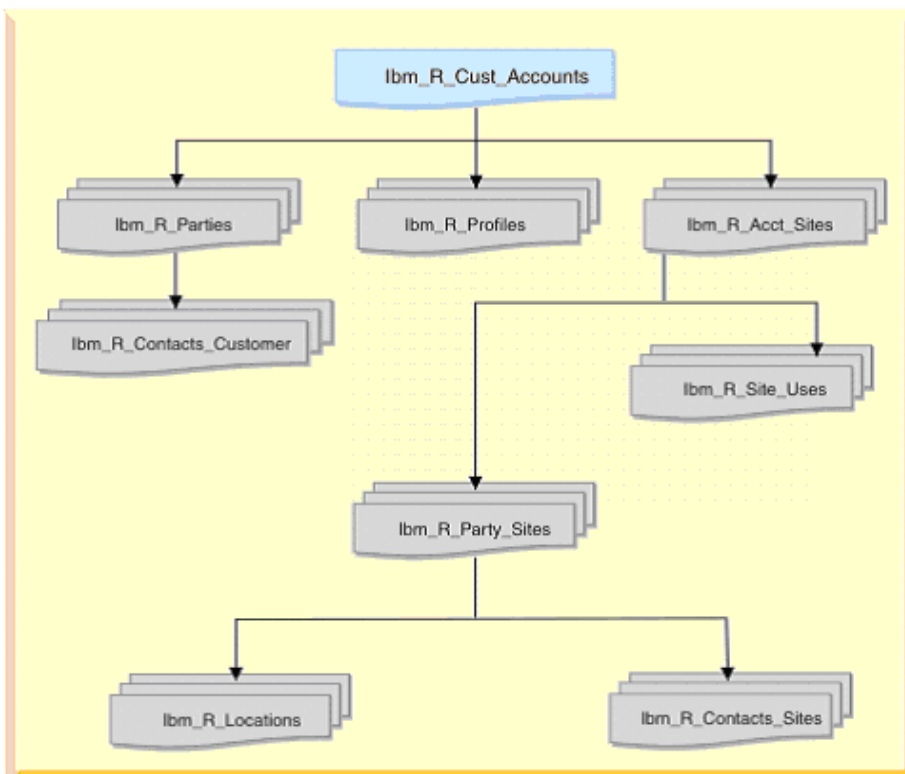
Table name	Synonym	Description
AR.HZ_CUST_ACCOUNTS	Ibm_R_Cust_Accounts	Customer number, type, and other customer information.
AR.HZ_CUSTOMER_PROFILES	Ibm_R_Profiles	High-level profile information for the customer.
AR.HZ_PARTIES	Ibm_R_Parties	Customer name and if it is a person, not a business, the person's details.

AR.HZ_PARTY_SITES	Ibm_R_Party_Sites	Links between customer's address IDs and address location IDs.
AR.HZ_CUST_ACCT_SITES_ALL	Ibm_R_Acct_Sites	Address IDs.
AR.HZ_CUST_SITE_USES_ALL	Ibm_R_Site_Uses	Customer business address purposes (BILL_TO, SHIP_TO, etc.).
AR.HZ_LOCATIONS	Ibm_R_Locations	Address details.
AR.HZ_CONTACT_POINTS	Ibm_R_Contacts_Sites	Contact and phone details. The table is listed twice because both sites and customers (PARTY_SITES and PARTIES) use the same table for contact details.
AR.HZ_CONTACT_POINTS	Ibm_R_Contacts_Customer	Contact and phone details. The table is listed twice because both sites and customers (PARTY_SITES and PARTIES) use the same table for contact details.

Relationship of the business objects

For the Retrieve operation to succeed, the links connecting the customer object to its child objects must match those in the Oracle database. The following diagram shows the relationship of the objects you must configure for the Retrieve operation.

Figure 3. Relationship of objects for the Retrieve operation



To link the objects, you create the child objects and set the application-specific information foreign key relationship between the parent and child objects.

Database and applications user account requirements

To use the samples, you must use a database account that gives you rights to the artifacts needed to run the sample content, and use an Oracle E-Business Suite account that allows you to perform responsibilities of the System Administrator and Receivables Manager. For the purposes of these samples, the user account running all scripts is assumed to be the APPS user for the Oracle database. This user has the following rights:

- To change and create content in the APPS schema
- To add and remove data from tables
- To run the required executables in the APPS schema

Check with your Oracle database administrator to determine the account that will be used to run the sample content. If you want to choose a different user account, work with your database administrator to ensure that the user has rights to all of the database artifacts needed to run the sample content. Oracle E-Business Suite requires you to have an account with rights to the following responsibilities:

- System Administrator
- Receivables Manager

Note: If the Oracle E-Business Suite account that you are using does not have access to these responsibilities, they can be added. To change the responsibilities assigned to the account you will use, log onto an account that has System Administrator responsibility rights and go to the Security->User->Define menu option. For specific information about changing user responsibilities, refer to your documentation for Oracle Applications.

Run the sample scripts

Before you will be able to use the Oracle EBS Adapter solution you need to run two database scripts against the Oracle EBS database.

Table 3. Sample files

Filename	Description
lhm_submit_request.sql	SQL script to insert a special procedure to be called after event entries have been made to the interface tables to move the data to the base tables
lhm_create_synonyms.sql	SQL script to create the synonym names used

Preparing for outbound processing

You must run SQL script files to prepare for outbound processing. You need to create synonyms to save you time later when you generate business objects from the Oracle database. You also need to insert artifacts into Oracle E-Business Suite before you can process objects using the database tables.

1. Create synonyms

Open the SQL*Plus tool (or similar program for processing SQL statements). Log on to SQL*Plus using a database account that has been granted database administrator (DBA) rights. Run the file `ibm_create_synonyms.sql`. For example: `SQL> @C:\samplecontent\ibm_create_synonyms.sql`; The size of the database and number of database artifacts in the Oracle database can lengthen the process of searching through large lists when running the external service wizard to generate objects. Creating the synonyms helps to make the search quicker. In addition, the synonyms help to resolve an issue related to the Oracle database driver. The issue is documented in IBM® Technote 1218775. Refer to "Related information " in the "Reference" section for a link to WebSphere Adapters Technotes.

2. Insert artifacts into Oracle E-Business Suite

Before you process business objects, run the SQL script `ibm_submit_request.sql`. This places a stored procedure called `IBM_WEBSPPHERE_CUSTOMER_IMP` into the Oracle APPS schema. This procedure is used to move customer data from the Oracle interface tables to the base tables. You will use the interface tables when you construct the Create business object.

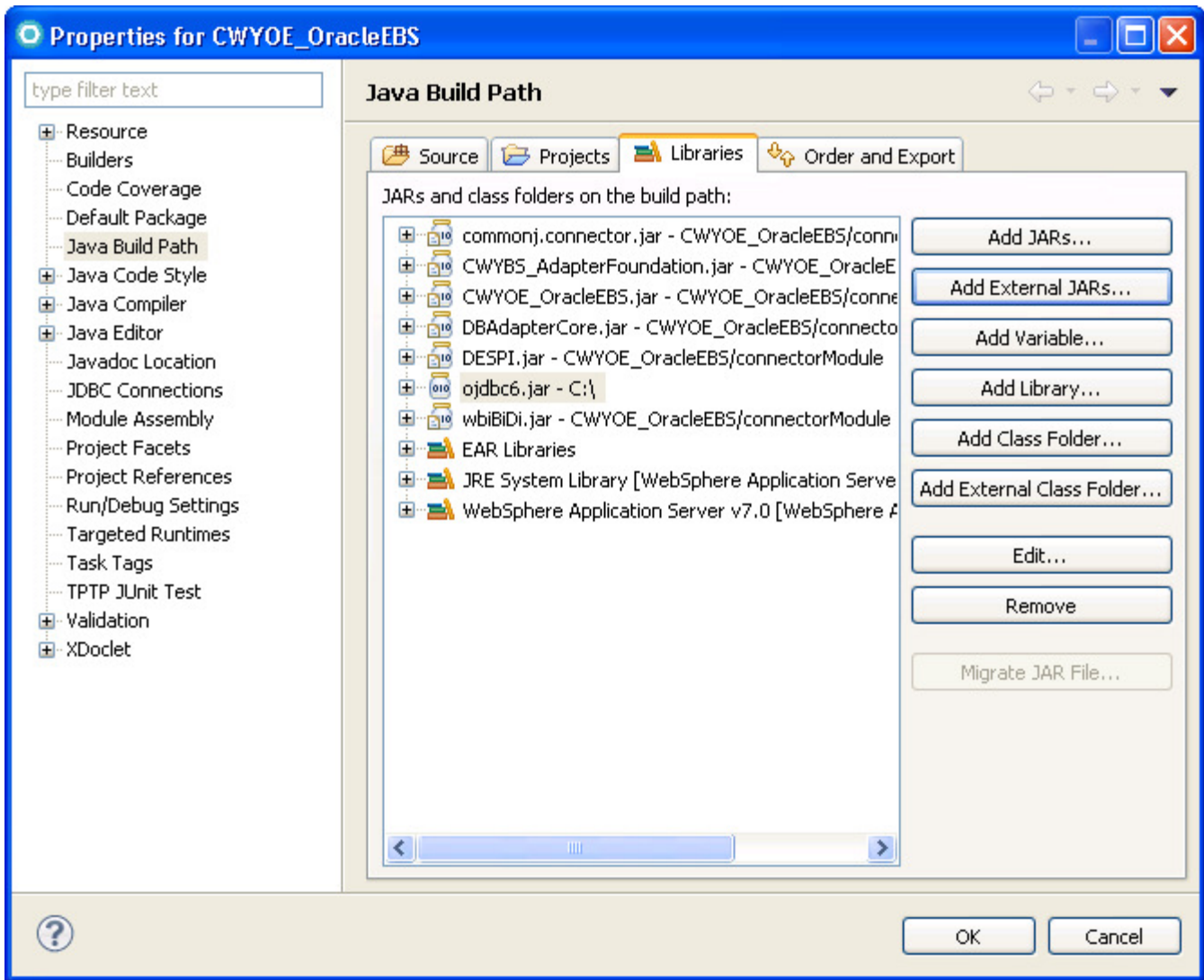
Running the prepared sample

A sample Project Interchange file is provided to allow you to quickly run a customer creation and retrieve using the Oracle E-Business Suite adapter. **Restore the Project Interchange File**

1. In a clean workspace in Rational Application Developer, go to File -> Import.
2. Import the sample by clicking "Import Sample" link.

Add your JDBC Driver to the CWYBC_JDBC project

1. Right-click on the CWYBC_JDBC project. Choose Build Path -> Configure Build Path.
2. In the Properties for CWYBC_JDBC window, click on the Libraries tab.
3. Click Add External Jars and browse to your JDBC driver file. Click OK



Modify the server connection

1. In the Project Explorer pain in RAD, expand the OracleEBSCustomer project.
2. Expand src -> customer -> CustomerInfoImpl
3. Right-Click on CustomerInfoImpl.Java and choose Open. In this file, you need to configure your connection information for your Oracle EBS server database. Modify the values at the beginning of this section to match your connection information. You will at least need to modify the databaseURL, password, and userName values.

```

/** * @j2c.connectionFactory jndi-name="OracleEBSCF"
 * @j2c.managedConnectionFactory class="com.ibm.j2ca.jdbc.JDBCManagedConnectionFactory"
 * @j2c.managedConnectionFactory-property name="databaseURL" value="jdbc:oracle:thin:@my_host:1521:my_db"
 * @j2c.managedConnectionFactory-property name="databaseVendor" value="ORACLE"
 * @j2c.managedConnectionFactory-property name="jdbcDriverClass" value="oracle.jdbc.driver.OracleDriver"
 * @j2c.managedConnectionFactory-property name="password" value="my_password"
 * @j2c.managedConnectionFactory-property name="returnDummyBOForSP" value="false"
 * @j2c.managedConnectionFactory-property name="userName" value="my_username"
 * @generated
 */

```

4. Save the Java class file.

Run the CreateCustomer class

1. In the OracleEBSCustomer project, expand src and expand the create.customer package.
2. Right-click on the CreateCustomer class and choose Open.
3. Now you'll need to modify the string values at the beginning of the class.
 - a. These values represent unique keys that come from the source where the customer data originated.

```
String CustomerReferenceNum = "Cust_01";
String AddressReference1Num = "Addr_01";
String AddressReference2Num = "Addr_02";
String TelephoneReferenceNum = "Tel_01";
String CustomerName = "CustNameTest";
```
 - b. The date value should be set to today's date.

```
String CurrentDate = "2010-05-02 12:00:00";
```
 - c. Save the file.
4. Now that the class has been created, we'll run the class to create the customer in Oracle using the adapter.
 - a. Right-click on the class CreateCustomer.Java, and choose Run As -> Java Application.
 - b. The class should run and in the Console view you should see the text 'Create Finished'.

Getting the customer key value for the Retrieve There are two ways in which you can verify the imported customer. Please use one of the two steps below:

1. Getting the retrieve key from the Oracle EBS application.
 - a. Log into the Oracle E-Business Suite and switch to the 'Receivables Manager' responsibility.
 - b. Go to the Customers -> Standard menu to open the Find Customers screen.
 - c. Query for the imported customer.
 - d. Once you have found the customer, write down the **Customer Number** from the customer screen in Oracle.
 - e. Open a database editor, such as SQL*Plus, and run the following command:

```
SQL> select cust_account_id from ar.hz_cust_accounts where account_number = the
customer number from the previous step
```
 - f. Write down the cust_account_
2. Getting the retrieve key directly from the database.
 - a. Open a database editor, such as SQL*Plus, and connect to the Oracle database you are using for the scenario.
 - b. Find the row of your inserted customer by looking at the creation_date attribute in the AR.HZ_CUST_ACCOUNTS table. (It should be today's date, or the date you inserted for the customer profile object.)
 - c. Once you have located the row, write down the value for **cust_account_id**. You'll use it in the Retrieve operation.

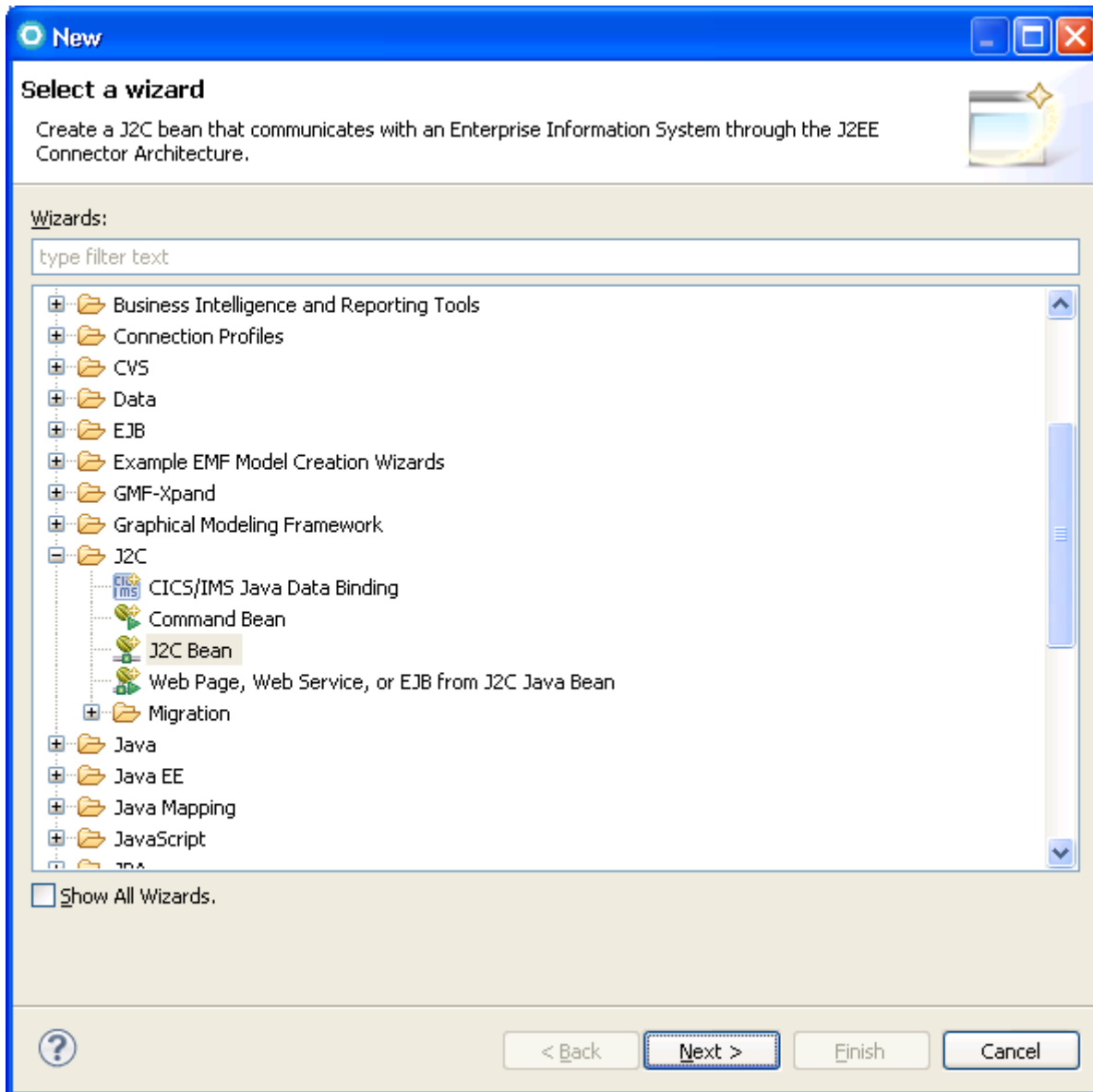
Run the RetrieveCustomer class

1. In the OracleEBSCustomer project, expand src and expand the retrieve.customer package.
2. Right-click on the RetrieveCustomer class and choose Open.
3. Modify the string Cust_Account_ID and set the value to match the value you discovered in the last section.
4. Now that the class has been created, we'll run the class to create the customer in Oracle using the adapter.
 - a. Right-click on the class RetrieveCustomer.Java, and choose Run As -> Java Application.
 - b. The class should run and in the Console view you should see the text 'CustomerName : *the customer name you inserted in the create*'.

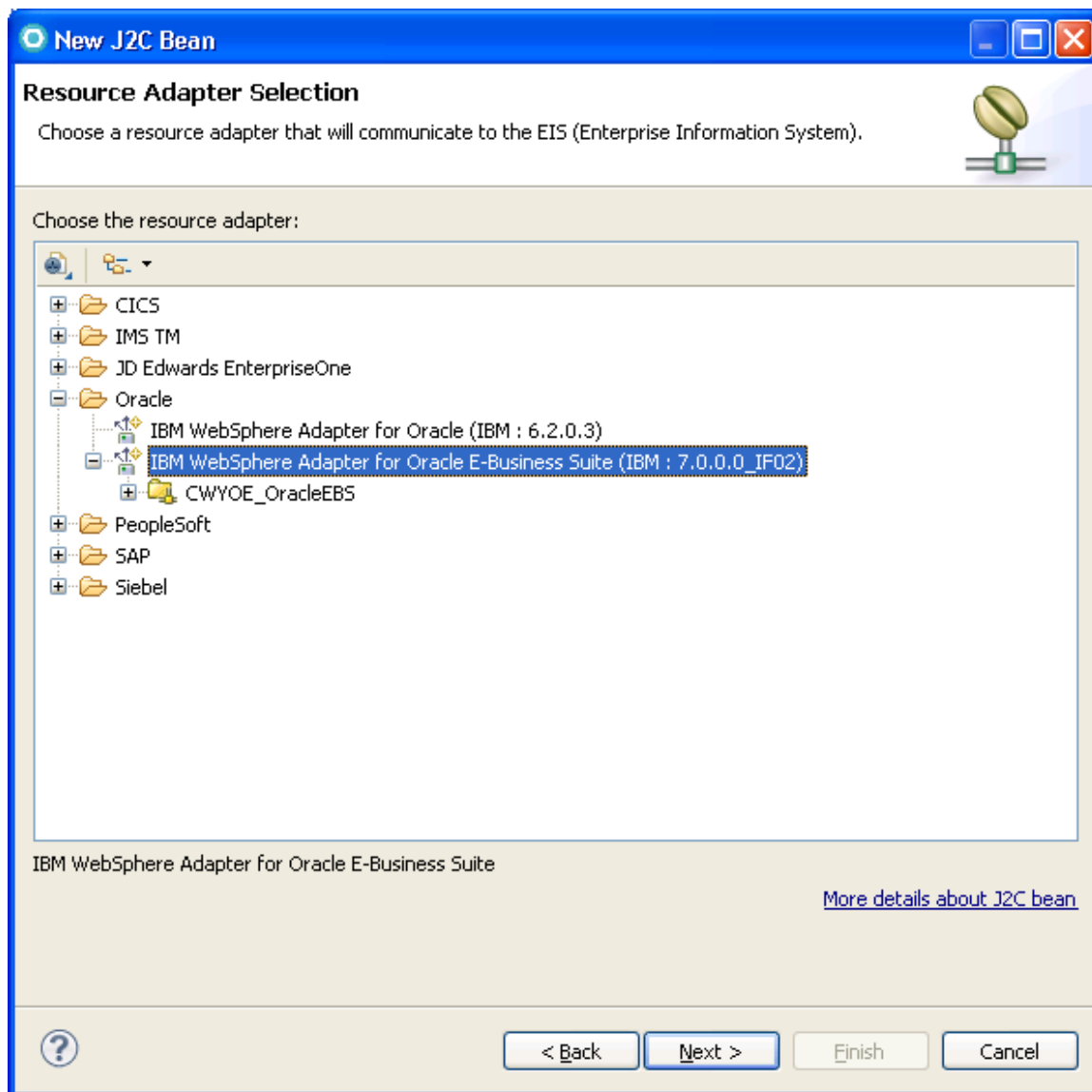
Developing the complete Customer Create and Retrieve sample

The remaining sections of this sample guide walk you through creating the customer content yourself.

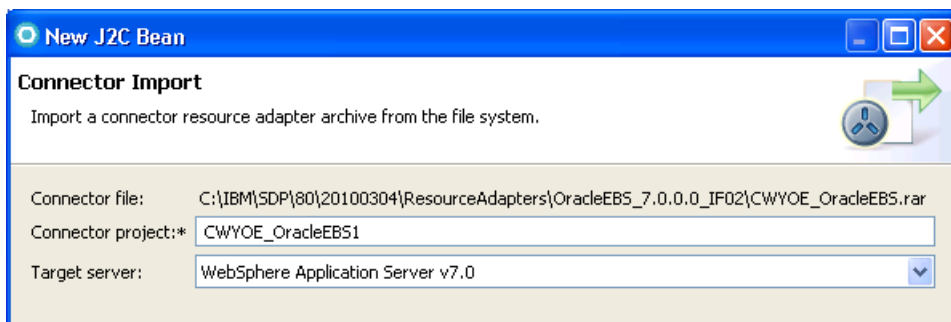
1. In Rational Application Developer, at the J2EE perspective, select File > New > Other.
2. At the New, Select a Wizard window, scroll down to the J2C entry, expand it and select J2C Java Bean. Click Next.



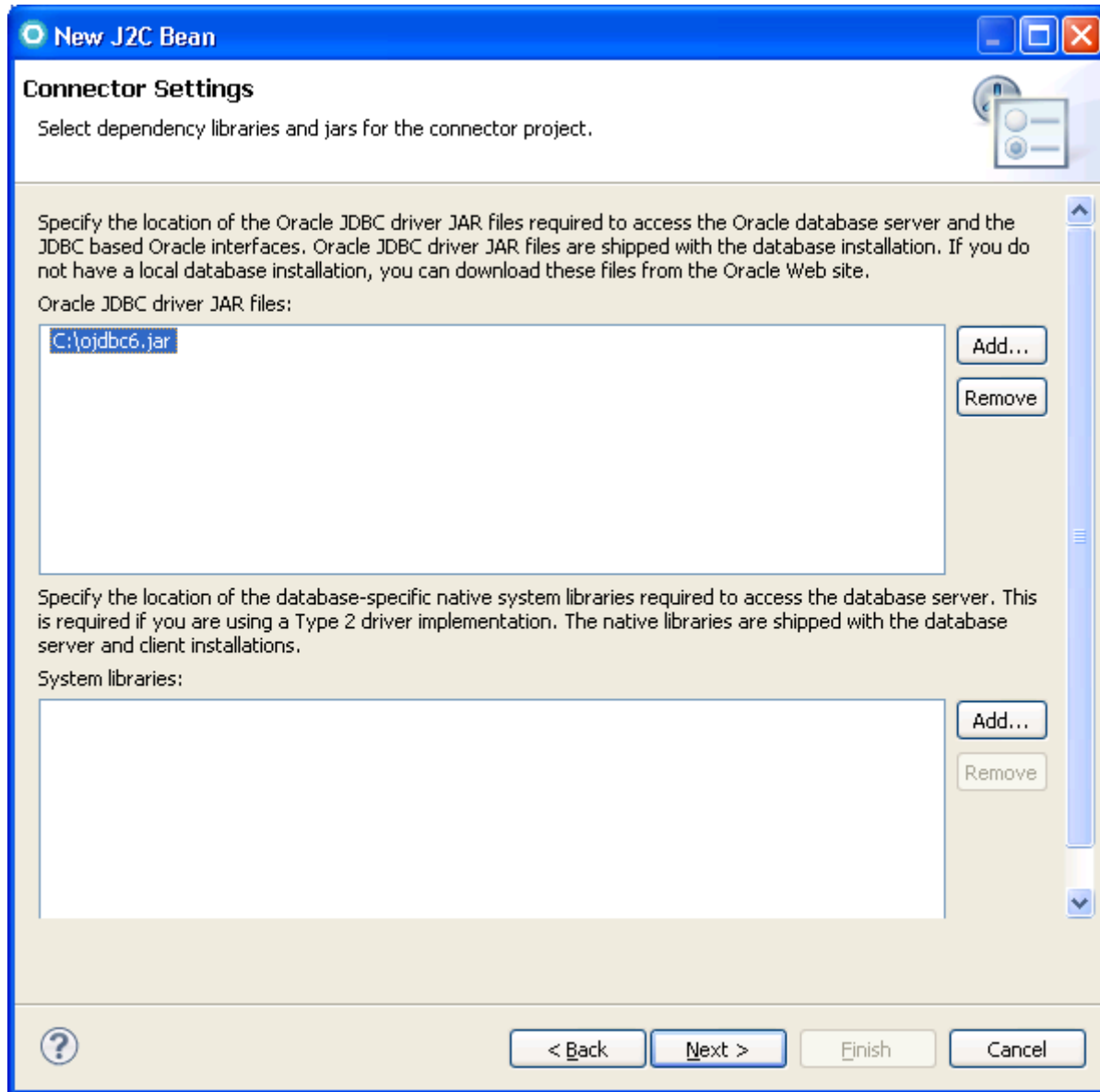
3. Expand the entry for JDBC and select IBM WebSphere Adapter for JDBC (IBM : 7.0.0.0). Click Next.



4. In the New J2C Java Bean, Connector Import window, leave the Connector Project with the default value, and choose your server for the Target Server. Click Next.



5. In the New J2C Java Bean, Connector Settings window, you need to select your JDBC driver files. To do this, click the Add button next to the window for JDBC driver JAR files and select your JDBC driver. Click Next.



6. Here you configure the connection information to connect to the server that hosts your Oracle EBS database. In the left pane, select your database vendor, driver, and version. Enter the information for your database in the Properties fields.

This table lists the properties required for the sample and their descriptions:

Table 4. Connection properties for the external service wizard

Property	Description
Database name	Along with host name and port number, this generates the Database

Property	Description
	URL that is used to connect to the database.
Host name	Along with database name and port number, this generates the Database URL that is used to connect to the database.
Port number	Along with database name and host name, this generates the Database URL that is used to connect to the database.
JDBC driver class name	The class name of the JDBC driver that is used to connect to the database.
UserName	The database account you are using.
Password	The password for the account you are using.

6. Click Next. On the Adapter Style page, click Outbound:

New J2C Bean

Adapter Style

Select the style of this adapter

☐ Inbound

Inbound adapters allow data to be passed into the application from the adapter.

☐ Enable Inbound Event Monitor

☒ Outbound

Outbound adapters allow data to be passed from the application to the adapter.

?

< Back

Next >

Finish

Cancel

New J2C Bean

Discovery Configuration

The wizard can now guide you through discovery of objects to communicate with. Specify settings to begin discovery.

Connection Configuration

Oracle system connection information

Properties:

JDBC driver type: Oracle Thin Driver

System ID: * sample_db_name

Host name: * oracle_host_machine

Port number: * 1521

JDBC driver class name: * oracle.jdbc.driver.OracleDriver

Database URL: jdbc:oracle:thin:@oracle_host_mach

Additional JDBC driver connection properties [name:value;name:value]:

User name: * user_name

Password: * *****

Prefix for business object names:

Advanced >>

☐ Specify the level of the logging desired

< Back Next > Finish Cancel

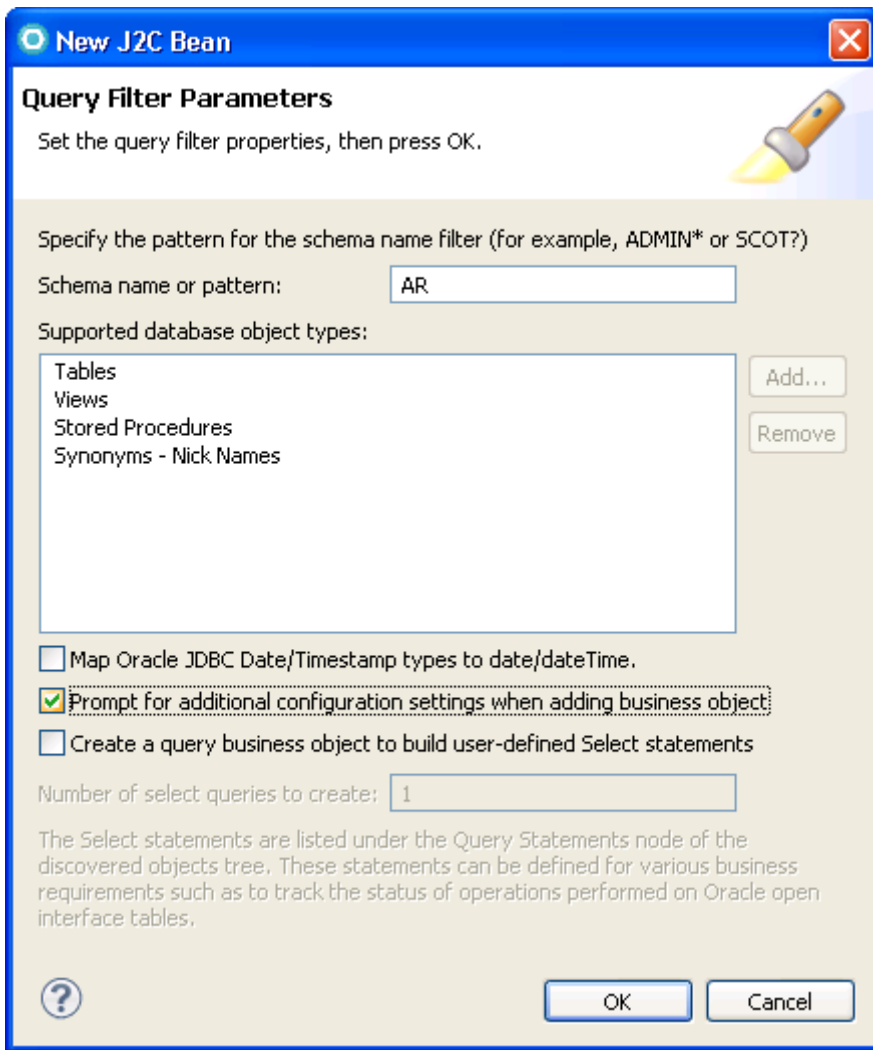
Selecting business data

For this sample, you run a query on the accounts receivable (AR) schema, and select synonyms that represent data in the Oracle E-Business Suite interface tables.

Selecting business objects for Create operation

1. Specify filter properties

- In the Object Discovery and Selection window, click Edit Query.
- In the Query Filter Parameters window, type AR in the Schema name pattern field to display the accounts receivable (AR) schema.
- Select the check box Prompt for additional configuration settings when adding business object, and click OK.



Now whenever you add an object when you run the metadata query in step 2, you are prompted to enter application-specific information for the object.

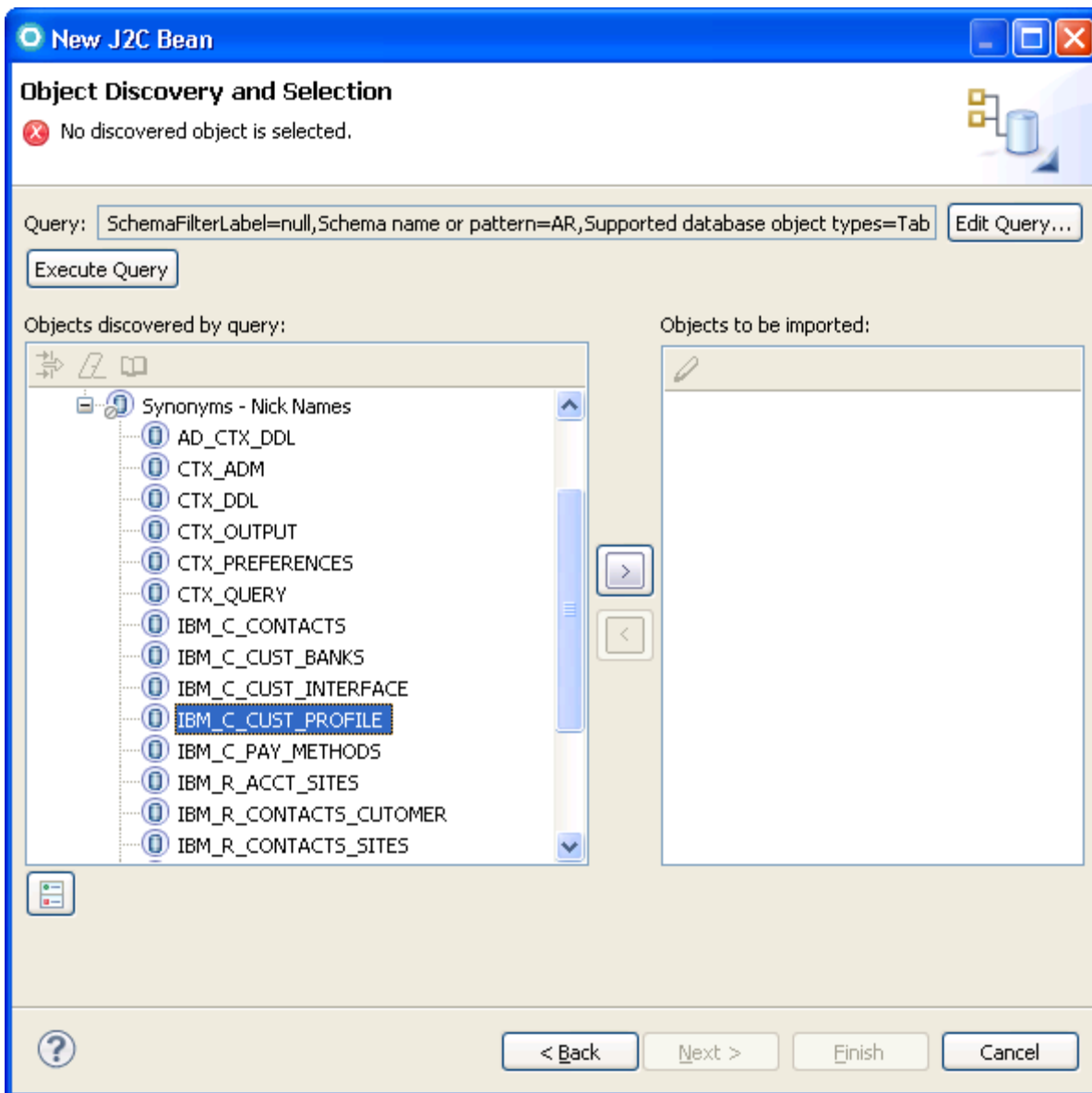
2. Run the metadata query

a. Display objects covered by the query

Click Execute Query. The AR schema and data elements of the schema are displayed.

b. Select the object for import

Expand the AR schema. Expand Synonyms - Nicknames. Highlight the synonym IBM_C_CUST_PROFILE, and click >, the Add icon, to select this object to be imported.



c. Add business object application-specific information

i. In the Configuration Parameters for IBM_C_CUST_PROFILE window, click Add to select the primary key for the table related to IBM_C_CUST_PROFILE. Select REQUEST_ID and click OK.

ii. Click the Add button next to the window titled 'To use a stored procedure, add it to the list then configure it'. Select AfterCreateSP and click OK.

In the AfterCreateSP area of the Configuration Parameters for IBM_C_CUST_PROFILE window, set the following values:

- Type APPS in the Schema name pattern field.
- Select APPS from the Schema list.
- Note: There may be long delay after selecting the APPS schema before the stored procedure name filter field is active.
- Type IBM_WEBSPPHERE_CUSTOMER_IMP in the stored procedure name filter field.
- Select IBM_WEBSPPHERE_CUSTOMER_IMP from the stored procedure name list.

iii. Click OK.

New J2C Bean

Configuration Parameters for 'IBM_C_CUST_PROFILE'

Set the configuration parameters, then press OK.

Select status column name and status value for logical delete

Name of the column used to perform logical deletes:

Value used to indicate a deleted object:

Select primary key for table IBM_C_CUST_PROFILE

Select one or more columns:*

An operation can be performed by a stored procedure. You can run a stored procedure to perform the operation or to do custom processing before or after processing.

To use a stored procedure, add it to the list and then configure it.:

AfterCreateSP

Select the schema name to display the matching schema

Schema name:

You must specify a name or pattern to display the matching stored procedures. The pattern cannot consist of only asterisks (*) or percent signs (%).

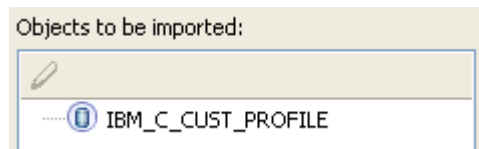
Stored procedure name or pattern:

Stored procedure name:

For each stored procedure parameter, select a table column

None:

After a delay, the selected object is displayed in the Objects to be imported pane.



In this sample, you process data using Oracle's built-in interface and base tables. Only the Create operation is necessary to post both new customer data and updates to existing customer data. As part of calling the Create operation of the object, the stored procedure IBM_WEBSPPHERE_CUSTOMER_IMP is run using the AfterCreateASI value for the object to process. This stored procedure moves the data from the interface to the base tables.

d. Select another object for import and link child object to parent

i. In the Object Discovery and Selection window, highlight the synonym IBM_C_CUST_INTERFACE, and click > to select this object to be imported.

ii. In the Configuration Parameters for IBM_C_CUST_INTERFACE window, click Add to select the primary key. Select CUSTOMER_KEY.

iii. Set the value for Choose Parent Table to IBM_C_CUST_PROFILE (AR).

iv. Under the 'Build a foreign key relationship...' section, scroll to the REQUEST_ID field and select request_id as the field in the parent object.

v. At the bottom of the list, click the check box for 'Parent object owns child object (cascade delete)'

vi. Click OK.

New J2C Bean

Configuration Parameters for 'IBM_C_CUST_INTERFACE'

Set the configuration parameters, then press OK.

Select status column name and status value for logical delete

Name of the column used to perform logical deletes:

Value used to indicate a deleted object:

Select primary key for table IBM_C_CUST_INTERFACE

Select one or more columns:*

Choose parent table from the list for the selected child

Choose parent table:

☐ Single cardinality

Build a foreign key relationship by selecting a parent table column for each child column

ORIG_SYSTEM_CUSTOMER_REF:
 SITE_USE_CODE:
 ORIG_SYSTEM_ADDRESS_REF:
 INTERFACE_STATUS:
 REQUEST_ID:

☒ Parent object owns child object (cascade delete)
☐ Preserves IBM_C_CUST_INTERFACE when the parent is updated.
☐ IBM_C_CUST_INTERFACE required for operations on parent

An operation can be performed by a stored procedure. You can run a stored procedure to perform the operation or to do custom processing before or after processing.
 To use a stored procedure, add it to the list and then configure it.:

e. Select remaining synonyms. Repeat step 2d for each of the remaining synonyms in the following table. The primary key is specified for each synonym. The foreign key is the same for all objects.

Table 5. Remaining Synonymsto select for import and to link to the parent object

Synonym	Primary Key
IBM_C_CONTACTS	CONTACT_KEY
IBM_C_CUST_BANKS	BANK_ACCOUNT_NUM
IBM_C_PAY_METHODS	ORG_ID

Selecting and linking business objects for Retrieve

Now that you have selected the tables for the Create operation, you need to select the tables that will be used for the Retrieve.

1. Select the top-level object for import

a. In the Object Discovery and Selection window, after you have expanded the AR schema and expanded Synonyms - Nicknames, highlight the synonym IBM_R_CUST_ACCOUNTS and click > (the Add icon) to select the object to be imported.

b. Add business object application-specific information

In the Configuration Parameters for IBM_R_CUST_ACCOUNTS window, click Add to select the primary key for the table related to IBM_R_CUST_ACCOUNTS. Select CUST_ACCOUNT_ID and click OK.

2. Select another object for import and link child object to parent

a. Highlight the synonym IBM_R_CUST_PROFILES and click > (the Add icon).

b. In the Configuration Parameters for IBM_R_CUST_PROFILES window, click Add to select the primary key for the table related to IBM_R_CUST_PROFILES.

Select APPLICATION_ID and click OK.

c. In the Choose parent table select IBM_R_CUST_ACCOUNTS.

d. Under the Build a foreign key relationship... section, scroll to the CUST_ACCOUNT_ID field and select CUST_ACCOUNT_ID as the field in the parent object for the foreign key attribute.

e. Check 'Parent object owns child object (cascade delete)' and click Next.

3. Select remaining synonyms

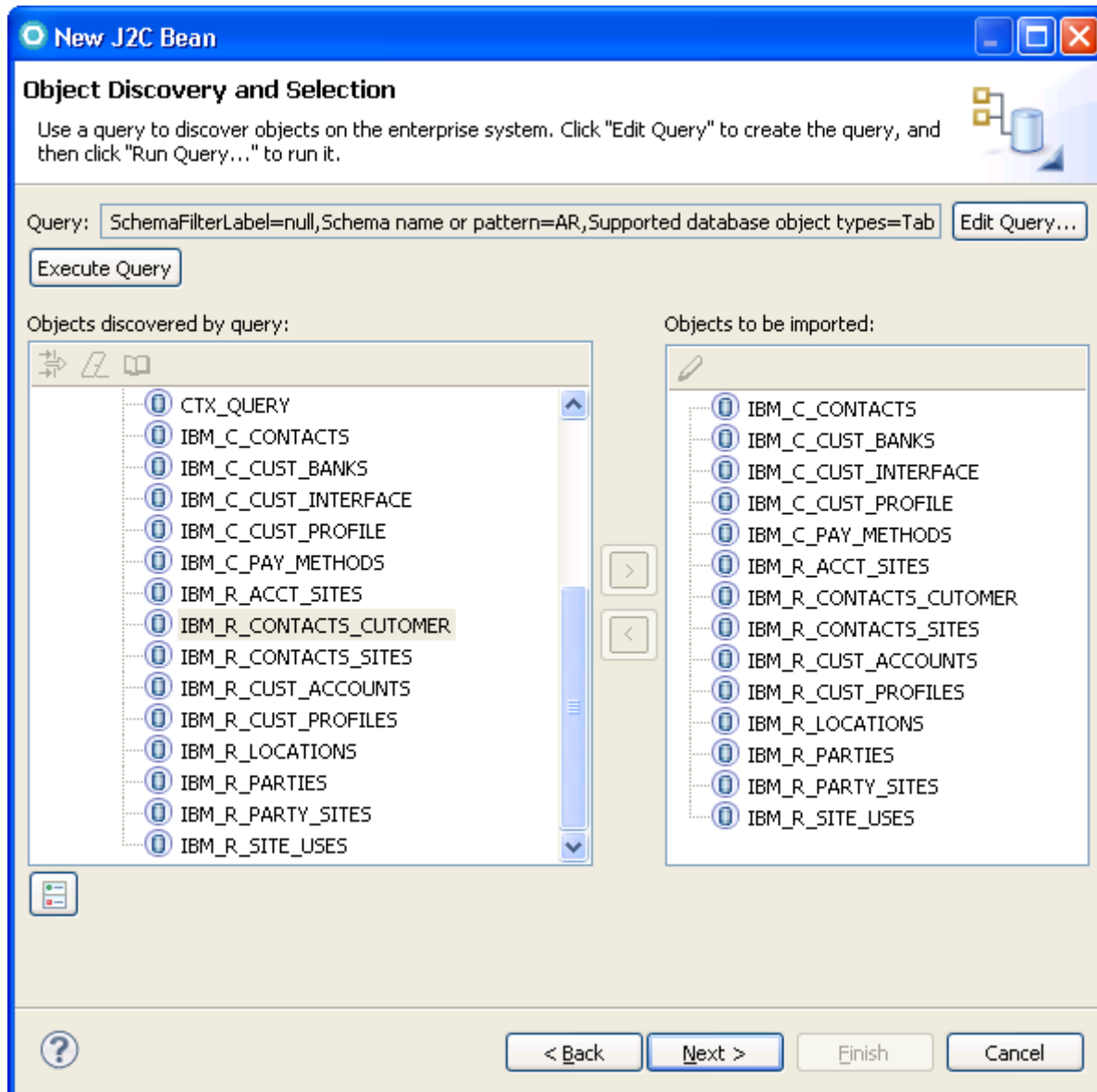
Repeat step 2 for each of the remaining synonyms in the order the synonyms are presented in the following table. Use the primary key, parent, foreign key attribute, and foreign key attribute value shown for each synonym in the table.

Table 6. Objects requiring Foreign Key values and child objects

Synonym	Primary key	Parent	Foreign key attribute	Foreign key attribute value
Ibm_R_Parties	party_id	Ibm_R_Cust_Accounts	party_id	party_id
Ibm_R_Acct_Sites	party_site_id and cust_acct_site_id	Ibm_R_Cust_Accounts	cust_account_id	cust_account_id
Ibm_R_Party_Sites	location_id and party_site_id	Ibm_R_Acct_Sites	party_site_id	party_site_id
Ibm_R_Site_Uses	application_id	Ibm_R_Acct_Sites	cust_acct_site_id	cust_acct_site_id

Ibm_R_Locations	address_key	Ibm_R_Party_Sites	location_id	location_id
Ibm_R_Contacts_Sites	application_id	Ibm_R_Party_Sites	owner_table_id	party_site_id
Ibm_R_Contacts_Customer	application_id	Ibm_R_Parties	owner_table_id	party_id

When you're done, your Object Discovery and Selection window will look like this:



After you've completed these steps, click Next.

Configuring the selected objects

After you have selected and linked database objects, specify values for the selection properties for the import file.

1. Select operations: In the Configure Composite Properties window of the New J2C Java Bean wizard, the Operations for selected business objects pane lists the operations that the adapter supports for the outbound service type. Remove the following operations: Update, Delete, and RetrieveAll by highlighting them and clicking Remove. The following operations remain:

- Create
- Retrieve

2. Modify the Business object namespace value to be: `http://company.com/Oracle/sample` to simplify the package names for generated records.

3. Leave the default values for Maximum records for RetrieveAll operation and Specify the relative folder for generated business object.

4. Click Next.

The screenshot shows the 'New J2C Bean' wizard window, specifically the 'Configure Composite Properties' step. The window title is 'New J2C Bean'. The main heading is 'Configure Composite Properties' with a subtitle 'Specify properties that apply to all selected objects.' Below this, there is a section titled 'Operations for selected business objects' with a subtext 'Operations for these functions will be added to the service interface.:*'. A list box contains 'Create' and 'Retrieve'. To the right of the list box are 'Add...' and 'Remove' buttons. Below the list box, there are three input fields: 'Maximum records for RetrieveAll operation:' with a value of '100', 'Business object namespace:' with a value of '* http://company.com/Oracle/sample', and 'Specify the relative folder for generated business objects' with a value of 'Folder:'. At the bottom of the window, there are four buttons: '?', '< Back', 'Next >', 'Finish', and 'Cancel'.

New J2C Bean

Configure Composite Properties
Specify properties that apply to all selected objects.

Operations for selected business objects
Operations for these functions will be added to the service interface.:*

Create
Retrieve

Add...
Remove

Maximum records for RetrieveAll operation: 100

Business object namespace: * http://company.com/Oracle/sample

Specify the relative folder for generated business objects
Folder:

? < Back Next > Finish Cancel

Configuring J2C Java Bean Creation and Deployment Configuration properties

Set the values for the Java bean for the Oracle EBS project and custom package. Set the Save properties values

1. Create the project.
 - a. Next to the Project Name field, click the New button on the far right.
 - b. Select Java Project and click Next.
 - c. For the Project Name, enter OracleEBSCustomer.
 - d. Click Finish.
2. Create the package.
 - a. Next to the Package Name field, click the New button on the far right.
 - b. For the package Name enter customer.
 - c. Click Finish.
3. In the Interface Name field, enter CustomerInfo. Note that the Implementation Name will automatically be filled in. Your screen should now look like this:

New J2C Bean

J2C Bean Creation and Deployment Configuration
Specify the properties for creating and running the J2C bean.

Save properties

Project name:	*	OracleEBSCustomer	Browse...	New...
Package name:	*	customer	Browse...	New...
Interface name:	*	CustomerInfo		
Implementation name:	*	CustomerInfoImpl		

Generate Command Bean

☐ Click to launch J2C deployment wizard

4. Scroll to the bottom of the window until Connection Properties becomes visible.
5. Specify OracleEBSCF as the JNDI.
6. Check the box for Non-managed Connection. Verify the properties match those you entered when you started the wizard.

New J2C Bean

J2C Bean Creation and Deployment Configuration

Specify the properties for creating and running the J2C bean.

☐ Managed Connection (recommended)

☒ Non-managed Connection

Database system connection information

To join a global transaction, specify a predefined XA datasource or XA database connection information. When not joining a global transaction, either the XA connection information or the local connection information can be specified.

Database connection information: Specify local database connection information

Database system connection information

Database vendor: ORACLE

Database URL: * jdbc:oracle:thin:@ora220.cn.ibm.com:1521:VIS

JDBC driver class name: * oracle.jdbc.driver.OracleDriver

User name: apps

Password: ****

Advanced >>

☐ Click to launch J2C deployment wizard

< Back Next > Finish Cancel

7. Click Finish. After you click Finish, the wizard will generate the appropriate set of artifacts to access the PeopleSoft Enterprise system.

Testing Create and Retrieve Operations for the customer

Now that your adapter project has been created, you can test the project by creating two new class files that will be used to execute the Create and Retrieve operations. **Running the Create operation**

1. Expand the OracleEBSCustomer project.
2. Right-click on src and choose New -> Package. Name the package 'create.customer'.
3. Right-click on the new package and choose New -> class.
 - a. Name the class CreateCustomer.
 - b. Check the box for 'public static void main(String*+ args)'.
 - c. Click Finish.
4. Right-click on the new class, and choose Open.
5. Insert the following code into the class, and save it.

```
package create.customer;

import javax.resource.ResourceException;

import com.Oracle.sample.aribm_c_contacts.Aribm_C_Contacts;

import com.Oracle.sample.aribm_c_cust_interface.Aribm_C_Cust_Interface;
```

```

package create.customer;

import java.math.BigDecimal;
import javax.resource.ResourceException;
import com.Oracle.sample.aribm_c_contacts.Aribm_C_Contacts;
import com.Oracle.sample.aribm_c_cust_interface.Aribm_C_Cust_Interface;
import com.Oracle.sample.aribm_c_cust_profile.Aribm_C_Cust_Profile;
import customer.CustomerInfo;
import customer.CustomerInfoImpl;

public class CreateCustomer {

    /**
     * @param args
     */
    public static void main(String[] args) {

        CreateCustomer client = new CreateCustomer();
        client.createCustomer();

    }

    public void createCustomer(){

        // This class will create a Customer to be sent into Oracle E-Business Suite.
        // It consists of 1 customer profile with 2 addresses, and one telephone
        // on the first address.

        // These set of strings should be set prior to running the class
        String CustomerReferenceNum = "Cust_01";    // MUST BE UNIQUE: Reference number from the
'original' system
        String AddressReference1Num = "Addr_01";    // MUST BE UNIQUE: Reference number from the
'original' system
        String AddressReference2Num = "Addr_02";    // MUST BE UNIQUE: Reference number from the
'original' system
        String TelephoneReferenceNum = "Tel_01";    // MUST BE UNIQUE: Reference number from the
'original' system
        String CustomerName = "CustNameTest";    // MUST BE UNIQUE: Reference number from the
'original' system
        String CurrentDate = "2007-11-02 12:00:00";    // Today's time and date - does not have to be accurate

        try {

```

```

// Create Contact
Arlbm_C_Contacts contact = new Arlbm_C_Contacts();
contact.setorig_system_contact_ref("");
contact.setorig_system_telephone_ref(TelephoneReferenceNum);
contact.setorig_system_customer_ref(CustomerReferenceNum);
contact.setorig_system_address_ref(AddressReference1Num);
contact.setinsert_update_flag("I");
contact.setcontact_first_name("");
contact.setcontact_last_name("");
contact.setcontact_title("");
contact.setcontact_job_title("");
contact.settelephone("5551212");
contact.settelephone_extension("1234");
contact.settelephone_type("FAX");
contact.settelephone_area_code("650");
contact.setlast_update_date(CurrentDate);
contact.setlast_updated_by(new BigDecimal(-1));
contact.setcreation_date(CurrentDate);
contact.setcreated_by(new BigDecimal(-1));
contact.setemail_address("");
contact.setorg_id(new BigDecimal(204));
contact.setcontact_point_type("PHONE");
contact.setcontact_key("");

// Create Address 1
Arlbm_C_Cust_Interface address1 = new Arlbm_C_Cust_Interface();
address1.setorig_system_customer_ref(CustomerReferenceNum);
address1.setsite_use_code("BILL_TO");
address1.setorig_system_address_ref(AddressReference1Num);
address1.setinsert_update_flag("I");
address1.setcustomer_name(CustomerName);
address1.setcustomer_number("");
address1.setcustomer_status("A");
address1.setprimary_site_use_flag("Y");
address1.setlocation("");
address1.setaddress1("Test Address 1");
address1.setaddress2("");
address1.setaddress3("");
address1.setaddress4("");
address1.setcity("San Mateo");
address1.setstate("CA");
address1.setprovince("");
address1.setcounty("San Mateo");

```

```

address1.setpostal_code("94010");
address1.setcountry("US");
address1.setcustomer_category_code("CUSTOMER");
address1.setlast_updated_by(new BigDecimal(-1));
address1.setlast_update_date(CurrentDate);
address1.setcreated_by(new BigDecimal(-1));
address1.setcreation_date(CurrentDate);
address1.setorg_id(new BigDecimal(204));
address1.setcustomer_name_phonetic(CustomerName);
address1.setcustomer_key("");

// Create Address 2
Arlbm_C_Cust_Interface address2 = new Arlbm_C_Cust_Interface();
address2.setorig_system_customer_ref(CustomerReferenceNum);
address2.setsite_use_code("SHIP_TO");
address2.setorig_system_address_ref(AddressReference2Num);
address2.setinsert_update_flag("I");
address2.setcustomer_name(CustomerName);
address2.setcustomer_number("");
address2.setcustomer_status("A");
address2.setprimary_site_use_flag("Y");
address2.setlocation("");
address2.setaddress1("Test Address 2");
address2.setaddress2("");
address2.setaddress3("");
address2.setaddress4("");
address2.setcity("San Mateo");
address2.setstate("CA");
address2.setprovince("");
address2.setcounty("San Mateo");
address2.setpostal_code("94010");
address2.setcountry("US");
address2.setcustomer_category_code("CUSTOMER");
address2.setlast_updated_by(new BigDecimal(-1));
address2.setlast_update_date(CurrentDate);
address2.setcreated_by(new BigDecimal(-1));
address2.setcreation_date(CurrentDate);
address2.setorg_id(new BigDecimal(204));
address2.setcustomer_name_phonetic(CustomerName);
address2.setcustomer_key("");

// Create array to hold addresses
Arlbm_C_Cust_Interface[] address_array = {address1, address2};

```

```

// Create array to hold contact
Arlbm_C_Contacts[] contact_array = {contact};

// Create Customer
Arlbm_C_Cust_Profile customer = new Arlbm_C_Cust_Profile();
customer.setinsert_update_flag("I");
customer.setorig_system_customer_ref(CustomerReferenceNum);
customer.setcustomer_profile_class_name("DEFAULT");
customer.setcredit_hold("N");
customer.setlast_updated_by(new BigDecimal(-1));
customer.setlast_update_date(CurrentDate);
customer.setcreation_date(CurrentDate);
customer.setcreated_by(new BigDecimal(-1));
customer.setorg_id(new BigDecimal(204));
customer.setrequest_id(new BigDecimal(0));
customer.setibm_c_cust_interfaceobj(address_array);
customer.setibm_c_contactsobj(contact_array);

// Send Create
CustomerInfo info = new CustomerInfoImpl();

info.createArlbm_C_Cust_Profile(customer);

System.out.println("Create Finished");
}
catch (ResourceException e) {

    System.out.println("Exception is execution: " + e.getMessage());
}

}

}

```

6 Now you'll need to modify the string values at the beginning of the class.

- a. These values represent unique keys that come from the source where the customer data originated.

```
String CustomerReferenceNum = "Cust_01"; String AddressReference1Num = "Addr_01"; String  
AddressReference2Num = "Addr_02"; String TelephoneReferenceNum = "Tel_01"; String CustomerName =  
"CustNameTest";
```

- b. The date value should be set to today's date. String CurrentDate = "2010-05-02 12:00:00";
- c. Save the file.

7. Now that the class has been created, we'll run the class to create the customer in Oracle using the adapter.

- a. Right-click on the class CreateCustomer.Java, and choose Run As -> Java Application.
- b. The class should run and in the Console view you should see the text 'Create Finished'.

The customer has now been created. Now you'll need to get the key value you'll use to process the retrieve of the customer info from the Oracle base tables.

Getting the customer key value for the Retrieve

There are two ways in which you can verify the imported customer. Please use one of the two steps below:

1. Getting the retrieve key from the Oracle EBS application.

- a. Log into the Oracle E-Business Suite and switch to the 'Receivables Manager' responsibility.
- b. Go to the Customers -> Standard menu to open the Find Customers screen.
- c. Query for the imported customer.
- d. Once you have found the customer, write down the **Customer Number** from the customer screen in

Oracle.

- e. Open a database editor, such as SQL*Plus, and run the following command:

```
SQL> select cust_account_id from ar.hz_cust_accounts where account_number = the customer number from the  
previous step
```

- f. Write down the cust_account_id. You'll use it in the Retrieve operation.

2. Getting the retrieve key directly from the database.

- a. Open a database editor, such as SQL*Plus, and connect to the Oracle database you are using for the scenario.
- b. Find the row of your inserted customer by looking at the creation_date attribute in the AR.HZ_CUST_ACCOUNTS table. (It should be today's date, or the date you inserted for the customer profile object.)
- c. Once you have located the row, write down the value for **cust_account_id**. You'll use it in the Retrieve operation.

Running the Retrieve operation

Now that we have customer key value, we can use it and the adapter to retrieve the customer back into the server.

1. Expand the OracleEBSCustomer project.
2. Right-click on src and choose New -> Package. Name the package 'retrieve.customer'.
3. Right-click on the new package and choose New -> class.
 - a. Name the class RetrieveCustomer.
 - b. Check the box for 'public static void main(String*+ args)'.
 - c. Click Finish.
4. Right-click on the new class, and choose Open.
5. Insert the following code into the class, and save it.

```

package retrieve.customer;

import java.math.BigDecimal;
import javax.resource.ResourceException;
import com.Oracle.sample.arlbm_r_cust_accounts.Arlbm_R_Cust_Accounts;
import customer.CustomerInfo;
import customer.CustomerInfoImpl;

public class RetrieveCustomer {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // This class performs a retrieve on a customer

        // Set this value to be the value used to retrieve customer data
        String Cust_Account_ID = "1234";

        try {

            //Create customer record to retrieve and populate the customer key
            Arlbm_R_Cust_Accounts customer = new Arlbm_R_Cust_Accounts();
            Arlbm_R_Cust_Accounts returned_customer = null;

            customer.setcust_account_id(new BigDecimal(Cust_Account_ID));

            // Send Retrieve
            CustomerInfo info = new CustomerInfoImpl();
            returned_customer = info.retrieveArlbm_R_Cust_Accounts(customer);

            System.out.println("Customer Name : " +
returned_customer.getibm_r_partiesobj()[0].getorganization_name_phonetic());
        }

        catch (ResourceException e) {

            System.out.println("Exception is execution: " + e.getMessage());

        }

    }
}

```

6. Modify the string Cust_Account_ID and set the value to match the value you discovered in the last section.
7. Now that the class has been created, we'll run the class to create the customer in Oracle using the adapter.
 - a. Right-click on the class RetrieveCustomer.Java, and choose Run As -> Java Application.
 - b. The class should run and in the Console view you should see the text 'CustomerName : *the customer name you inserted in the create*'.

Clearing the sample content

After you have tested the Create and Retrieve operations, clear the sample content to return the data to its original state. To do this, you must deactivate the customer in Oracle E-Business Suite, and then run SQL commands to delete the stored procedure.

1. Use the Oracle E-Business Suite Client to log into the Receivables Manager responsibility. Navigate to the Customers > Standard menu and find the customer. Change the Status value for the customer to "Inactive." Then save the customer.

Note: For specific information about using the Oracle interface, refer to your documentation for Oracle Applications.

2. Use the SQL*Plus tool, or a similar program for processing SQL commands, to log into the database and run commands to clear artifacts from the database. Consult your database administrator if you need help performing this task.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to: IBM Director of Licensing IBM Corporation North Castle Drive Armonk, NY 10504-1785 U.S.A. For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to: IBM World Trade Asia Corporation Licensing 2-31 Roppongi 3-chome, Minato-ku Tokyo 106-0032, Japan **The following paragraph does not apply to the**

United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you. This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you. Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact: IBM Corporation Software Interoperability Coordinator, Department 49XA 3605 Highway 52 N Rochester, MN 55901 U.S.A. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee. The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental. COPYRIGHT LICENSE: This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. 44

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program. General-use programming interfaces allow you to write application software. General-use programming interfaces allow you to write application software that obtain the services of this program's tools. However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software. **Warning:** Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks

IBM and related trademarks: <http://www.ibm.com/legal/copytrade.shtml> Other company, product, or service names may be trademarks or service marks of others. This product includes software developed by the Eclipse Project (<http://www.eclipse.org>).

