

HP OpenView Select Identity

Connector for Oracle[®] E-Business Suite 11i

Installation and Configuration Guide

**Connector Version: 1.0
Select Identity Version: 3.3.1**



August 2005

© 2005 Hewlett-Packard Development Company, L.P.

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this document, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

© 2005 Hewlett-Packard Development Company, L.P.

No part of this document may be copied, reproduced, or translated into another language without the prior written consent of Hewlett-Packard Company. The information contained in this material is subject to change without notice.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>). Portions Copyright (c) 1999-2003 The Apache Software Foundation. All rights reserved.

Select Identity uses software from the Apache Jakarta Project including:

- Commons-beanutils.
- Commons-collections.
- Commons-logging.
- Commons-digester.
- Commons-httpclient.

- Element Construction Set (ecs).
- Jakarta-poi.
- Jakarta-regexp.
- Logging Services (log4j).

Additional third party software used by Select Identity includes:

- JasperReports developed by SourceForge.
- iText (for JasperReports) developed by SourceForge.
- BeanShell.
- Xalan from the Apache XML Project.
- Xerces from the Apache XML Project.
- Java API for XML Processing from the Apache XML Project.
- SOAP developed by the Apache Software Foundation.
- JavaMail from SUN Reference Implementation.
- Java Secure Socket Extension (JSSE) from SUN Reference Implementation.
- Java Cryptography Extension (JCE) from SUN Reference Implementation.
- JavaBeans Activation Framework (JAF) from SUN Reference Implementation.
- OpenSPML Toolkit from OpenSPML.org.
- JGraph developed by JGraph.
- Hibernate from Hibernate.org.
- BouncyCastle engine for keystore management, bouncycastle.org.

This product includes software developed by Teodor Danciu <http://jasperreports.sourceforge.net>). Portions Copyright (C) 2001-2004 Teodor Danciu (teodord@users.sourceforge.net). All rights reserved.

Portions Copyright 1994-2004 Sun Microsystems, Inc. All Rights Reserved.

This product includes software developed by the Waveset Technologies, Inc. (www.waveset.com). Portions Copyright © 2003 Waveset Technologies, Inc. 6034 West Courtyard Drive, Suite 210, Austin, Texas 78730. All rights reserved.

Portions Copyright (c) 2001-2004, Gaudenz Alder. All rights reserved.

Trademark Notices

HP OpenView Select Identity is a trademark of Hewlett-Packard Development Company, L.P. Microsoft, Windows, the Windows logo, and SQL Server are trademarks or registered trademarks of Microsoft Corporation.

Sun™ workstation, Solaris Operating Environment™ software, SPARCstation™ 20 system, Java technology, and Sun RPC are registered trademarks or trademarks of Sun Microsystems, Inc. JavaScript is a trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

This product includes the Sun Java Runtime. This product includes code licensed from RSA Security, Inc. Some portions licensed from IBM are available at <http://oss.software.ibm.com/icu4j/>.

IBM, DB2 Universal Database, DB2, WebSphere, and the IBM logo are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group.

This product includes software provided by the World Wide Web Consortium. This software includes xml-apis. Copyright © 1994-2000 World Wide Web Consortium, (Massachusetts Institute of Technology, Institute National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. <http://www.w3.org/Consortium/Legal/>

Intel and Pentium are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

AMD and the AMD logo are trademarks of Advanced Micro Devices, Inc.

BEA and WebLogic are registered trademarks of BEA Systems, Inc.

VeriSign is a registered trademark of VeriSign, Inc. Copyright © 2001 VeriSign, Inc. All rights reserved.

All other product names are the property of their respective trademark or service mark holders and are hereby acknowledged.

Support

Please visit the HP OpenView web site at:

<http://www.managementsoftware.hp.com/>

This web site provides contact information and details about the products, services, and support that HP OpenView offers.

You can also go directly to the support web site at:

<http://support.openview.hp.com/>

HP OpenView online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valuable support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and log in. Many also require a support contract.

To find more information about access levels, go to:

http://support.openview.hp.com/access_level.jsp

To register for an HP Passport ID, go to:

<https://passport2.hp.com/hpp/newuser.do>

contents

Chapter 1	Installing the Connector	8
	Operations Supported by the Connector	9
	System Requirements	10
	Deploying on the Web Application Server	10
	Installing the Agent on the Oracle Server	12
	Installation of SQL Scripts on the Oracle System	13
	Oracle Application Agent Provision_service Package	16
	Event Alert Definitions	18
	Creating Alerts for Application User Reverse Synchronization	20
	Creating Alerts for Employee Reverse Synchronization	39
	The SPMLProcess.sql script and SPML Messages	60
	Defining a Periodic Alert	60
	Defining a Concurrent Program	64
	Defining a Job Scheduler	69
	Custom Attributes for Employee Modifications	69
	Secure Communication for Reverse Provisioning	73
	Configuring a Secure JDBC Connection Pool	76
Chapter 2	Configuring the Connector	77
Chapter 3	Understanding the Mapping Files	85
	ORAERP-11-5-x.xml Mapping Information	86
	ORAERPMP-11-5-x.xml Mapping Information	89
	Elements in the XML Mapping Files	92
	Elements in the XSL Reverse Mapping File	96

Chapter 4	Uninstalling the Connector	97
	Uninstalling the Connector from WebLogic	97
	Uninstalling the Agent	98
Appendix A	Troubleshooting	99

Installing the Connector

The Oracle E-Business Suite 11i connector — hereafter referred to as the Oracle 11i connector — enables HP OpenView Select Identity to provision users on Oracle E-Business Suite 11i systems. The Oracle 11i connector is a two-way connector and provides an agent that can send changes made to data in Oracle Application and Human Resources systems to Select Identity.



The connector does not provision employees in Oracle Human Resources systems. Instead, information is sent to Select Identity by the agent when an employee is created, modified, terminated, or reverse terminated in Oracle Human Resources.

The Oracle 11i connector is packaged in the following files, which are located in the `Oracle11i` folder on the Select Identity Connector CD:

- `oraerp.rar` — contains the binaries for the connector.
- `schema.jar` — contains the following mapping files, which control how Select Identity and Oracle fields are mapped to each other:
 - `ORAERP-11-5-9.xml` and `ORAERP-11-5-10.xml` — map the Select Identity fields to the Oracle fields, which enables Select Identity to provision data in Oracle E-Business Suite systems. Use one of these files that matches the installed version of Oracle Application server.

- `ORAERPEMP-11-5-9.xml` and `ORAERPEMP-11-5-10.xml` — map the Select Identity attributes to the Oracle Human Resources employee attributes. Use one of these mapping files if you are provisioning to Oracle Human Resources systems (this file also enables you to map attributes on Oracle Application servers). Use the file that matches the installed version of Oracle Application server.
- `oracle11ierp.xsl` — maps attributes on the Oracle 11i Human Resources server and Oracle Applications server to attributes on the Select Identity server. This file is used by the agent during reverse synchronization.
- `oraerpsecattr.zip` — contains the SQL Script for security attributes.
- `oraerpagent.zip` — contains the SQL files that comprise the Oracle 11i agent.

Operations Supported by the Connector

The Oracle 11i connector enables Select Identity to perform the following provisioning tasks on Oracle E-Business Suite 11i systems:

- Add, update, and remove users
- Retrieve user attributes
- Enable and disable users
- Verify a user's existence
- Change user passwords
- Reset user passwords
- Expire passwords
- Retrieve all entitlements
- Retrieve a list of supported user attributes
- Assign and unassign entitlements to and from users

An agent is also provided that can send changes made to user and employee data in Oracle Application server to Select Identity. This is called **reverse synchronization**. Additional configuration steps are required to enable reverse synchronization.

System Requirements

The Oracle 11i connector is supported in the following environment:

Select Identity Version	Application Server	Database
3.3.1	WebLogic 8.1.4 on Windows 2003	SQL Server 2000
	WebSphere 5.1.1 on HP-UX 11i	Oracle 9i
	WebSphere 5.1.1 on Windows 2003	Oracle 9i

This connector supports Oracle E-Business Suite 11.5.9 on HP-UX 11i and 11.5.10 on Windows 2000 and HP-UX 11i. For secure communication, this connector supports secure JDBC. See [Configuring a Secure JDBC Connection Pool on page 76](#) for configuration information.

Deploying on the Web Application Server

To install the Oracle 11i connector on the Select Identity server, complete these steps:

- 1 Create a subdirectory in the Select Identity home directory where the connector's RAR file will reside. For example, you could create the `C:\Select_Identity\connectors` folder on Windows. (A connector subdirectory may already exist.)
- 2 Copy the `oraerp.rar` file from the Select Identity Connector CD to the connector subdirectory.
- 3 If deploying the connector on WebLogic, complete the following steps. If deploying on WebSphere, skip to [Step 4 on page 11](#).
 - a Create a schema subdirectory in the Select Identity home directory where the connector's mapping file(s) will reside. For example, you could create the `C:\Select_Identity\schema` folder. (This subdirectory may already exist.)
 - b Extract the contents of the `schema.jar` file on the Select Identity Connector CD to the schema subdirectory. (If you copied the JAR file to the local system, be sure to remove it after extracting its contents.)

- c Copy the `oracle11ierp.xml` file from the schema subdirectory to the connector subdirectory.
 - d Ensure that the CLASSPATH environment variable in the WebLogic server startup script references the schema subdirectory.
 - e Start the application server if it is not currently running.
 - f Log on to the WebLogic Server Console.
 - g Navigate to *My_domain* → **Deployments** → **Connector Modules**.
 - h Click **Deploy a New Connector Module**.
 - i Locate and select the `oraerp.rar` file from the list. It is stored in the connector subdirectory.
 - j Click **Target Module**.
 - k Select the **My Server** (your server instance) check box.
 - l Click **Continue**. Review your settings.
 - m Keep all default settings and click **Deploy**. The Status of Last Action column should display Success.
- 4 If deploying the connector on WebSphere, complete the following steps:
- a Stop the application server.
 - b Extract the contents of the `schema.jar` file (on the Select Identity Connector CD) to the `WebSphere\AppServer\lib\ext` directory.
 - c Copy the `oracle11ierp.xml` file from the schema subdirectory to the connector subdirectory.
 - d Start the application server.
 - e Log on to the WebSphere Application Server Console.
 - f Navigate to **Resources** → **Resource Adapters**.
 - g Click **Install RAR**.
 - h In the Server path field, enter the path to the `oraerp.rar` file. It is stored in the subdirectory created in [Step 1](#).
 - i Click **Next**.
 - j In the Name field, enter a name for the connector.
 - k Click **OK**.

- l** Click the **Save** link (at the top of the page).
 - m** On the Save to Master Configuraton dialog, click the **Save** button.
 - n** Click **Resources** → **Resource Adapters**.
 - o** Click the new connector.
 - p** Click **J2C Connection Factories** in the Additional Properties table.
 - q** Click **New**.
 - r** In the Name field, enter the name of the factory for the connector. For the SQL connector, enter `eis/oraerp`.
 - s** Click **OK**.
 - t** Click the **Save** link.
 - u** On the Save to Master Configuraton dialog, click the **Save** button.
 - v** Restart WebSphere.
- 5** Modify the `ORAERP-11-5-x.xml` and/or `ORAERPMP-11-5-x.xml` mapping file(s), if necessary. These files are described in detail in [Understanding the Mapping Files on page 85](#).

After installing the connector, refer to [Configuring the Connector on page 77](#) for information about registering and configuring this connector in Select Identity.

Installing the Agent on the Oracle Server

After you install the Oracle 11i connector on the Select Identity server, you must deploy SQL scripts, which comprise the agent, on the target Oracle system. The following sections describe the steps you must take to deploy the SQL scripts.



Do not use the ampersand (&) character when entering text values. This character is used for substitute variables in Oracle PL/SQL. If you specify this character, it will cause XML parsing errors during reverse provisioning. Also, if the Responsibility key uses & as part of its name, change the key to exclude &.

Installation of SQL Scripts on the Oracle System

Install the SQL scripts by completing the following steps on the target Oracle system:



Windows SQL client tools such as SQL Worksheet or SQLPLUS client may not work properly for the following installation steps.

- 1 If required, set the environment variable for the database. An environment setting script is located in the Oracle Application home directory. You can get this location for \$APPL_TOP or %APPL_TOP% from your Oracle Application administrator. (The environment may not be set correctly if you not logged on as an Oracle Application user.)
- 2 Create a directory on the Oracle Application server where the Oracle SQL scripts will reside. For example, on Windows, create the C:\app\selectid directory. On UNIX, you could create the /app/selectid directory.
- 3 Extract the contents of the oraerpsecattr.zip and oraerpagent.zip files to the directory created in Step 2.
- 4 Run the sioraerp.sql script in SQLPLUS as the system user, as follows:

```
Sqlplus system/manager
@ sioraerp.sql;
```

This creates the custom schema called sioraerp that includes the following tables used in reverse provisioning:

SIORAERP.USER_REQUESTS table — contains individual events recorded by Oracle alert actions for a user. Oracle alert stores user IDs for application users and person IDs for employees, an employee or user indicator, and the requested action type. The action type is A for add action, M for modify action, or T for terminate. Here is the table:

```
REQUEST_NUM  NUMBER NOT NULL,
ID NUMBER    NOT NULL, /* User ID for app user and
                       * Person ID for employee */
USER_TYPE    CHAR NOT NULL, /* E for Employee and U for User */
ACTION_TYPE  CHAR NOT NULL /* A for Add, M for Modify,
                       * T for terminate */
```

SIORAERP.PROVISION_REQUEST table — contains the Select Identity provisioning request for a user. At regular intervals, the requests in the **USER_REQUESTS** table are combined to create reverse provisioning requests for a user in the **PROVISION_REQUESTS** table. Here is the table:

```

REQUEST_NUM      NUMBER NOT NULL,
SI_REQUEST_ID    NUMBER, /* Not currently used */
CREATION_DATE    DATE,
LAST_SENT_DATE   DATE, /* Date and time SPML message
                        * sent last time*/
USER_NAME        VARCHAR2(64),
RESOURCE_NAME    VARCHAR2(64),
USER_ID          NUMBER NOT NULL, /* User ID for app user and
                        * PersonID for employee */
USER_TYPE        CHAR NOT NULL, /* E for Employee and
                        * U for User */
ADD_ACTION       NUMBER, /* 1 or 0 */
MODIFY_ACTION    NUMBER, /* 1 or 0 */
TERMINATE_ACTION NUMBER, /* 1 or 0 */
REQ_STATUS       NUMBER, /* 0= ready for process, 1=in process,
                        * 2=completed, -1 = error */
SPML_MESSAGE     VARCHAR2(4000), /* SPML message going out */
RESPONSE         VARCHAR2(4000), /* Response from SI */
ERROR_MSG        VARCHAR2(4000), /* Error Message */
RETRY            NUMBER /* Retry count for this message */

```

SIORAERP.OWMINFO table — contains Oracle Wallet information.

Oracle Wallet contains the security certificates used for secure communication. The Wallet information contains the path to the Wallet directory and password. If the Wallet information is valid and the URL is specified as HTTPS, a secure (SSL) channel is used to communicate with the Select Identity Web Service.

```

OWMPATH  VARCHAR(256) NOT NULL, /* Wallet Directory */
OWMPASS  VARCHAR(64) /* Wallet password */

```

- 5 Run the `oraerp_secAttr.sql` script in SQLPLUS as the apps user. This creates PL/SQL packages, which will handle securing attribute updates from the Select Identity Oracle 11i connector. Here is an example of the commands:

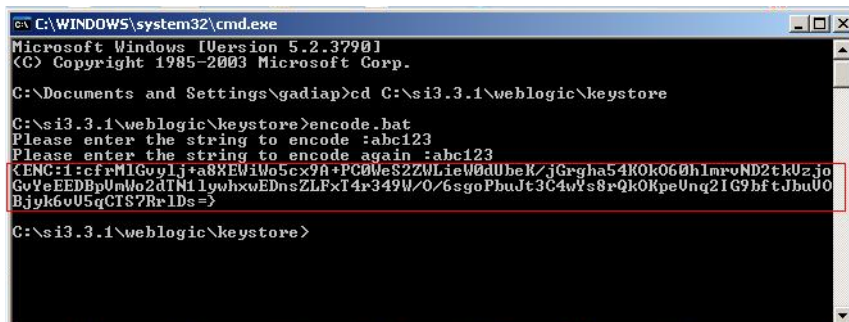
```

Sqlplus apps/apps
@ oraerp_secAttr.sql;

```

- 6 Run the `oraerp_spml.sql` script in SQLPLUS as the apps user. This creates PL/SQL Provision_Service packages, which handle SPML message handling for the agent.
- 7 Edit the `SIConfig.sql` script to enter values for the Select Identity server, administrator's user name, password, and the resource ID for the application user reverse synchronization and employee reverse synchronization. Specifically, you must modify the `SIHOST`, `SIADMIN`, `SIPWD`, `SIFNDRESOURCE`, and `SIEMPRESOURCE` parameters. The application user reverse synchronization resource is created in the `ORAERP-11-5-x.xml` mapping file. The employee reverse synchronization resource is created in the `ORAERPEMP-11-5-x.xml` mapping file.

To encrypt the password, run `encode.bat` (on Windows) or `encode.sh` (on UNIX), which is provided in the `weblogic/keystore` subdirectory in the Select Identity home directory. This utility prompts you for the password to encrypt and will generate the encrypted password. Be sure to copy the entire encrypted password, as shown here:



```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\gadiap>cd C:\si3.3.1\weblogic\keystore

C:\si3.3.1\weblogic\keystore>encode.bat
Please enter the string to encode :abc123
Please enter the string to encode again :abc123
<ENC:1:cfrMIGvylj+a8XEWIWo5cx9A+PC0WeS2ZWLieW0dUbeK/jGrgha54K0k060hlmrvND2tkUzjo
GuVeEEDBpUmWo2dTNl1ywhxwEDnsZLFxT4r349W/O/6sgoPbuJt3C4vYs8rQkOKpeUnq2IG9bftJbuU0
Bjyk6vU5qCTS?Rr1Ds=>

C:\si3.3.1\weblogic\keystore>

```

- 8 *Wallet configuration for SSL communication only:*
After Oracle Wallet configuration is complete (see [Secure Communication for Reverse Provisioning on page 73](#)), run the `updateowminfo.sql` script as the `sioraerp` or `apps` user. Be sure to run the script from the Select Identity installation directory because the script loads the `SIConfig.sql` file to get the URL of the Select Identity server.

`updateowminfo.sql` prompt for the Oracle Wallet Manager (OWM) directory and the password. It then stores the information in the `SIORAERP.OWMINFO` table. When the wallet path is entered, the path

information is stored in the table as `file:$PATH`, such as `file:c:\owm` or `file:/app/owm`. The wallet path is the directory located in the Oracle database server, not the application forms server.

- 9 Copy the SQL scripts from the foundation and employee directory to the Select Identity directory created in [Step 2](#).
- 10 On the Select Identity server, modify the `ORAERP-11-5-x.xml`, `ORAERPEMP-11-5-x.xml`, and `oracle11ierp.xsl` files to verify that the parameters specified in the `SIconfig.sql` script are the same. If new attributes were added to the `SIconfig.sql` script, these attributes must also be added to the XSL and XML files. See [Understanding the Mapping Files on page 85](#) for more information about the XML and XSL files.

Oracle Application Agent Provision_service Package

To enable reverse synchronization on Oracle Application or Oracle Human Resources systems, you must understand the `Provision_service` PL/SQL package. This package handles the SPML message processing that occurs when the agent sends requests to the Select Identity server.

The `Provision_service` package provides the following data types:

- `provision_service.atttab_type` — attribute name/value table
- `provision_service.val_list_type` — entitlement list

The package provides the following procedures:

- `invoke_add_request` — adds a user
- `invoke_modify_request` — modifies a user
- `invoke_delete_request` — deletes a user
- `invoke_enable_svc_request` — assigns a user to a Select Identity Service
- `invoke_disable_svc_request` — removes a user from a Select Identity Service
- `invoke_reset_password_request` — resets a password for a user
- `invoke_enable_user_request` — enables a user
- `invoke_disable_user_request` — disables a user
- `invoke_terminate_user_request` — terminates a user
- `invoke_update_password_request` — updates a user's password

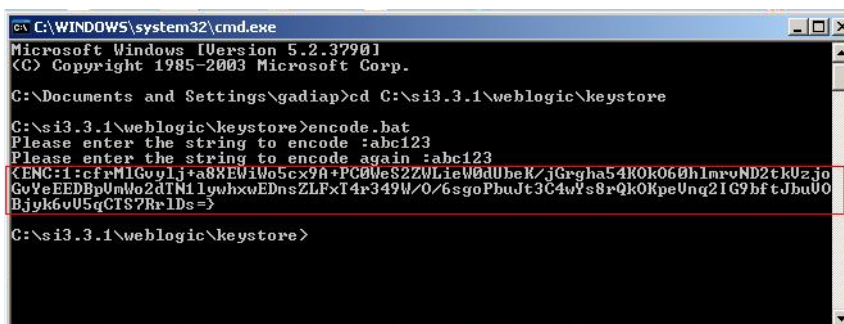
For each procedure, the following arguments must be configured:

`url` — URL of the Select Identity server.

`request_id` — Request ID.

`admin_id` — ID of a Select Identity user under which the service request will be performed.

`admin_pwd` — The password of the Select Identity user. To encrypt the password, run `encode.bat` (on Windows) or `encode.sh` (on UNIX), which is provided in the `weblogic/keystore` subdirectory in the Select Identity home directory. This utility prompts you for the password to encrypt and will generate the encrypted password. Be sure to copy the entire encrypted password in the field, as shown here:



```

C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\gadiap>cd C:\si3.3.1\weblogic\keystore

C:\si3.3.1\weblogic\keystore>encode.bat
Please enter the string to encode :abc123
Please enter the string to encode again :abc123
<ENC:1:cfrM1Gvylj+a8XEWiWo5cx9A+PC0WeS2Z0LieW0dUbeK/jGrgha54K0k060h1mr0ND2tkUzjo
GuVeEEDBpUmWo2dTN11ywhxwEDnsZLFxT4r349W/O/6sgoPbuJt3C4vYs8rQk0KpeUng21G9bf tJbuU0
Bjyk6vU5qCTS?Rr1Ds=>

C:\si3.3.1\weblogic\keystore>

```

`service_name` — The name of the Select Identity Sservice for which this request is performed.

`resource_id` — The name of the Select Identity resource for which this request is performed.

`attrs` — List of attributes.

`entitlements` — List of entitlements to be added during user creation.

`addentitlements` — List of entitlements to be added during user modification.

`deleteentitlements` — List of entitlements to be removed during user modification.

Here is an example for adding a new user:

```

DECLARE
    attab provision_service.atttab_type;
    add_entlist provision_service.val_list_type;

```

```

BEGIN
    attab('UserName') := 'CHPARK69';
    attab('Password') := 'abc123';
    attab('Company') := 'MJM';
    attab('Email') := 'chpark@mjm.com';
    attab('Owner') := 'CUST';
    attab('Person') := '';
    attab('Desc') := 'Description';
    attab('Customer') := '';
    attab('Supplier') := '';
    attab('Fax') := '8225267883';
    attab('Days') := '20';
    attab('Accesses') := '20';
    attab('StartDate') := '2005-2-1';
    attab('EndDate') := '2005-3-1';

    add_entlist := provision_service.val_list_type();
    provision_service.add_entitlement(add_entlist, 'CM_NORWAY');
    provision_service.invoke_add_request('http://helix:7003/lmz/
webservice', '12345', 'sis', 'abc123', '', 'vis', attab,
add_entlist);

END;
/

```

Event Alert Definitions

The agent uses the Oracle Applications Alert system to implement event monitoring on the Oracle system. When information about an Oracle user or employee changes, Oracle Alert provides immediate action based on the criteria defined by the user.

With Oracle Alert, the Oracle system can be monitored without using external programs, and it can be customized easily to suit the needs of the organization for user and employee event monitoring. For the reverse synchronization agent, event alerts are used. An event alert immediately notifies you of activity in the database as it occurs. When you create an event alert, you specify the following:

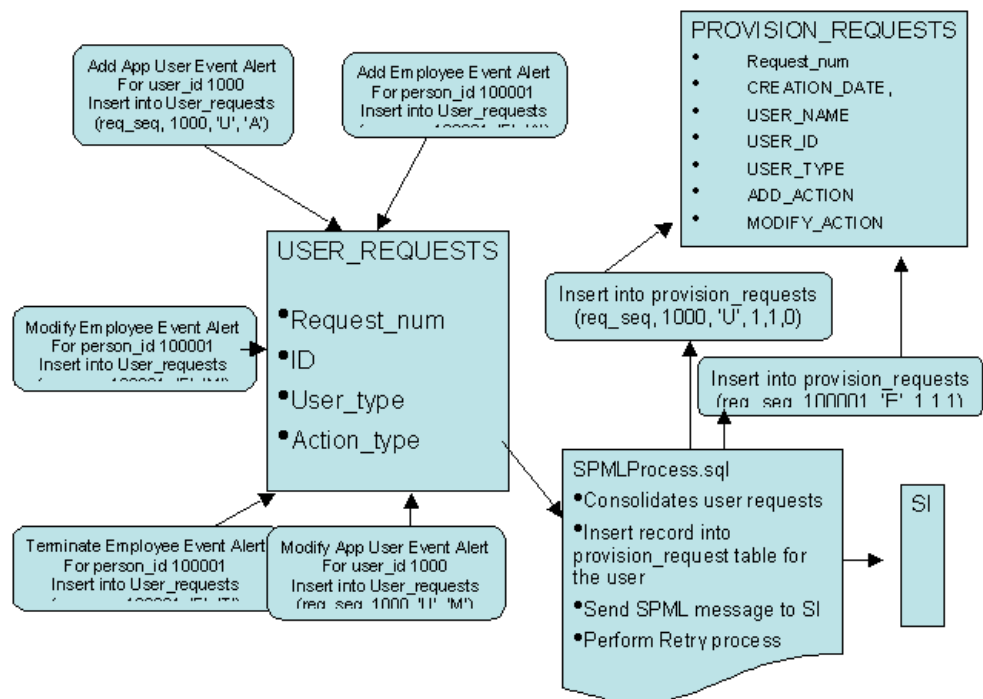
- A database event that you want to monitor (an insert or update to a specific database table).
- A SQL Select statement that retrieves specific database information as a result of the database event.

- Actions that you want Oracle Alert to perform as a result of the database event. An action can entail sending someone an email message, running a concurrent program, running an operating script, or running a SQL statement script. You specify the actions that the Oracle Alert should perform in an action set.

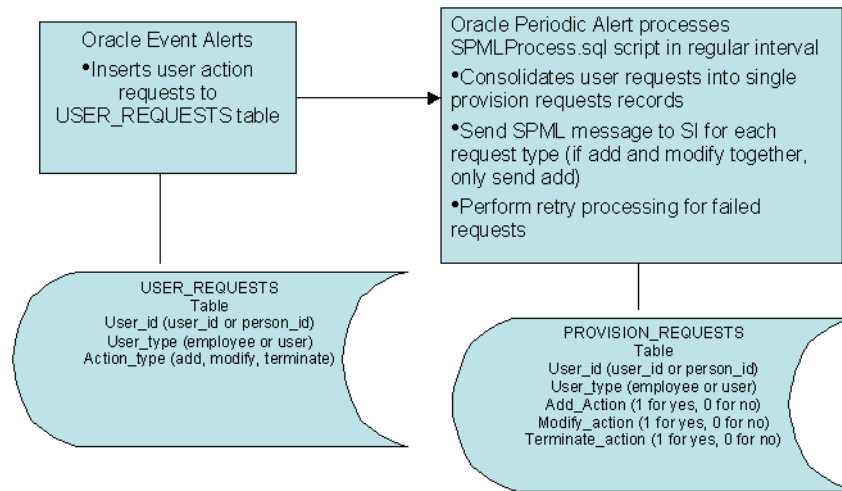
To create an event alert, the following tasks need to be performed:

- 1 Define the database events that will trigger the alert.
- 2 Specify the details of the alert.
- 3 Define actions for the alert.
- 4 Create action sets containing the actions you want the alert to perform.

The Oracle 11i connector agent uses Oracle Event alerts to capture individual event actions on a user (add, modify, terminate) and stores that individual event in a temporary processing table (sioraerp.user_requests table). At regular intervals, a periodic alert processes a consolidation processing script (SPMLProcess.sql) that combines all actions for a user and creates corresponding the SPML message to be sent to Select Identity.



The following diagram illustrates the data flow to and from the agent:



The following sections describe how to create event alerts based on whether an application user or employee is provisioned.

Creating Alerts for Application User Reverse Synchronization

An Oracle Application user uses the Oracle Application system to perform required business processing. An Oracle Application user differs from an Oracle Human Resources employee in that the Application user actively uses Oracle Application to perform business tasks. An Oracle Human Resources employee is simply stored in the Oracle Human Resources system to represent an employee in the organization. However, an Oracle Human Resources employee can be an Oracle Application user, and vice versa.

The agent monitors the following events. If any of these events occur, the agent sends the new data to Select Identity (reverse synchronization). The user name is the key field that identifies each Application user. The user name must be unique in the Oracle Application system, and the user name is not case sensitive (the agent sends the user name in all uppercase letters).

- New Application user
- Change in user attributes
- Change in user responsibilities
- Change in securing attributes for a user

- Termination of a user

The following user attributes are synchronized with Select Identity:

- Description — A description of the user.
- Employee ID — ID of the user if the user is also an employee in the Oracle Human Resources system.
- Customer ID — ID of the customer contact defined in Oracle Application
- Supplier ID — ID of the supplier contact defined in Oracle Application
- E-Mail — Email address for the user
- Fax — Fax number for the user
- Password Access Days — Maximum number of days between password changes. A pop-up window prompts an Application user to change her or his password after the maximum number of days you specify has elapsed.
- Password Accesses — Maximum allowed number of sign-ons to Oracle Applications between password changes. A pop-up window prompts an application user to change her or his password after the maximum number of accesses you specify has elapsed. Password Access Days and Password Accesses are mutually exclusive. Only one of these attributes may be used for the Oracle Application user.
- Start Date — Start date for the user. The user cannot sign onto Oracle Application before the start date and after the end date. The default for the start date is the current date.
- End Date — End date for the user. The user cannot sign onto Oracle Application after the end date. If the end date is not specified, the user name is valid indefinitely. An Application user cannot be deleted from Oracle Application because this information provides an audit trail. An Oracle Application user can be deactivated at any time by setting the End Date to the current date. To reactivate a user, change the End Date to a date after the current date, or clear the End Date field.

The user events are monitored by Oracle event alerts. Alerts are defined to monitor user creation, user modification, responsibility modification, and securing attributes changes. Additional events can be monitored by custom alert definitions. The alert action scripts call the main processing SQL scripts with parameters retrieved from the alert action. The processing script performs further data processing on the attributes and calls the `Provision_service` PL/SQL package to send SPML message to Select Identity.

Scripts are provided by the agent for the following events:

- **Add Application User**
Application Name: Application Object Library
Table Name: FND_USER
Event to Monitor: After Insert
Select SQL Script: AddUserSelect.sql
Action SQL Script: AddUserAction.sql
- **Modify Application User**
Application Name: Application Object Library
Table Name: FND_USER
Event to Monitor: After Update
Select SQL Script: ModifyUserSelect.sql
Action SQL Script: ModifyUserAction.sql
- **Update Responsibility**
Application Name: Application Object Library
Table Name: FND_USER_RESP_GROUPS
Event to Monitor: After Insert After Update
Select SQL Script: ModifyRespSelect.sql
Action SQL Script: ModifyRespAction.sql
- **Modify Securing Attributes**
Application Name: Oracle Common Modules -AK
Table Name: AK_WEB_USER_SEC_ATTR_VALUES
Event to Monitor: After Insert After Update
Select SQL Script: ModifySecAttrSelect.sql
Action SQL Script: ModifySecAttrAction.sql

The following sections describe how to create an alert in Oracle Application for each of these events.



Instead of importing the alert scripts from file, it is possible to copy and paste to update the alert actions and select statements. On some platforms (such as Windows), importing causes in fewer issues due to embedded special characters.

Defining a New Application User Alert

Complete the following steps to define an alert for a new Application user:

- 1 Display the alert definition window from the Oracle Application Alert Manager.

The screenshot shows the Oracle Applications Alert Manager window titled "Oracle Applications - ADS Vision LM0001". The window is divided into several sections:

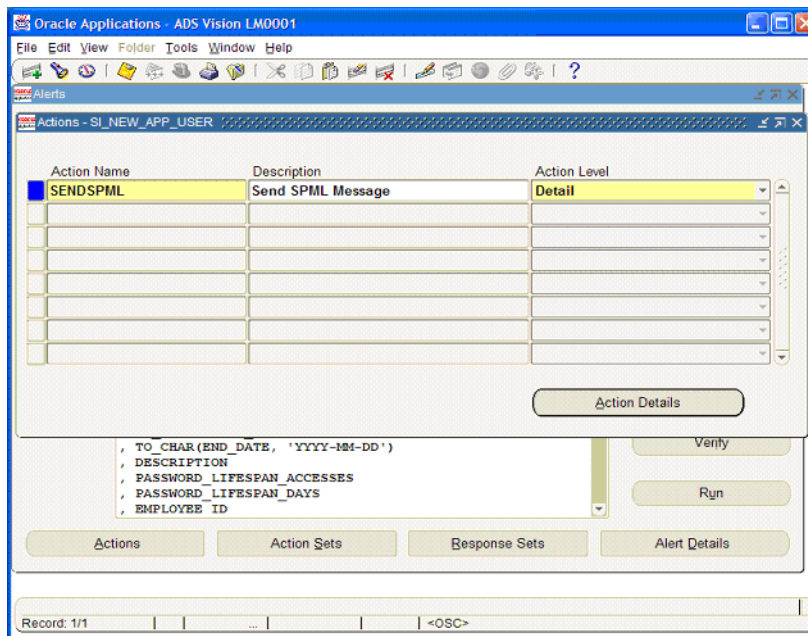
- Alert Information:**
 - Application: **Application Object Library**
 - Description: **HP SI New Application User SPML Ger**
 - Name: **SI_NEW_APP_USER**
 - Enabled
- Event Details:**
 - Application: **Application Object Library**
 - Table: **FND_USER**
 - After Insert (I)
 - After Update
- Keep:** **0** Days
- End Date:** [Empty field]
- Last Checked:** [Empty field]
- Select Statement:**

```
SELECT COLUMN NAME INTO &OUTPUT1 FROM TABLE NAME
```
- Buttons:** Import..., Export..., Verify, Run, Actions, Action Sets, Response Sets, Alert Details.

At the bottom of the window, a status bar displays: "FRM-40400: Transaction complete: 1 records applied and saved. Record: 1/1 <OSC>".

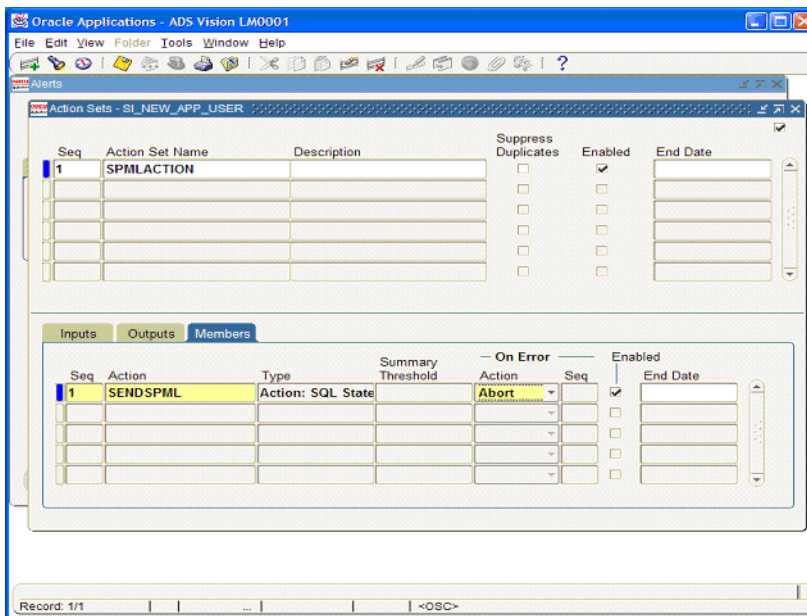
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Application Object Library** in the Application field.
- 4 Enter **FND_USER** in the Table field.
- 5 Select the **After Insert** option. Make sure that the **After Update** option is deselected.
- 6 Enter the alert name and description.
- 7 Save the alert.

- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `AddUserSelect.sql` script.
- 9 Click **OK** on the file upload window.
- 10 Verify the statement by clicking the **Verify** button, on the Select Statement window, which is populated with the SQL script from the imported file.

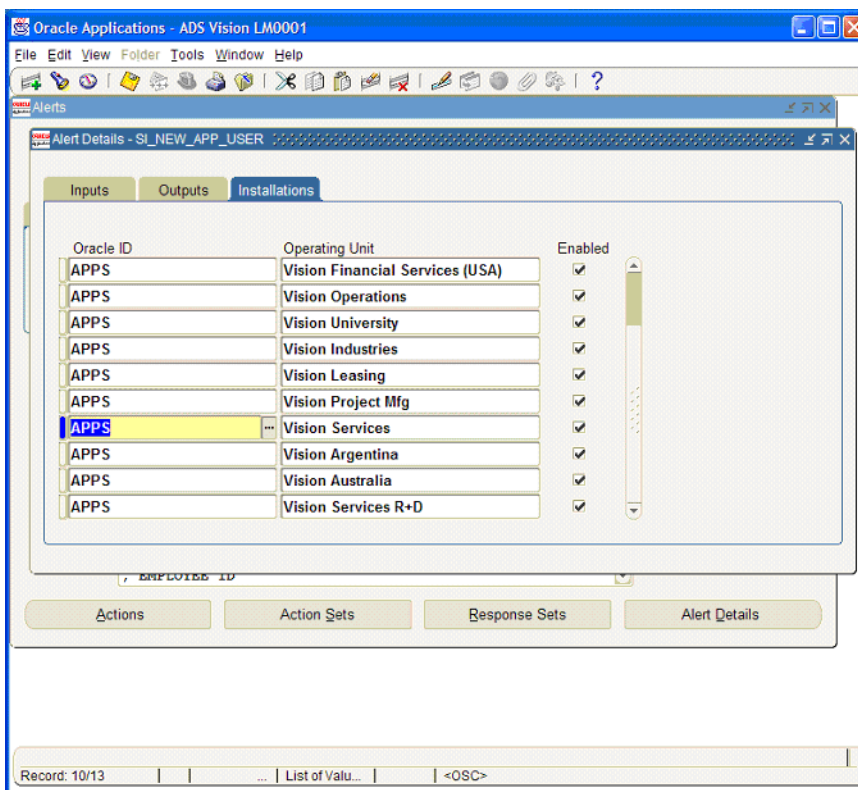


- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Enter the action name, such as `SENDSPML`. Select **Detail** as the action level, then save the alert.
- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.

- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the AddUserAction.sql script. This populates the window with the content of the script. AddUserAction.sql calls the AddUserProcess.sql script with user creation arguments.
- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to /app/selectid. Modify the directory name to specify the correct directory name. Also, modify the path to the SIconfig.sql script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as SPMLACTION, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.



- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).



- 27 Save the alert. The alert is now enabled and active.

Defining an Alert for Application User Modifications

Complete the following steps to define an alert that is triggered when an Application user is modified:

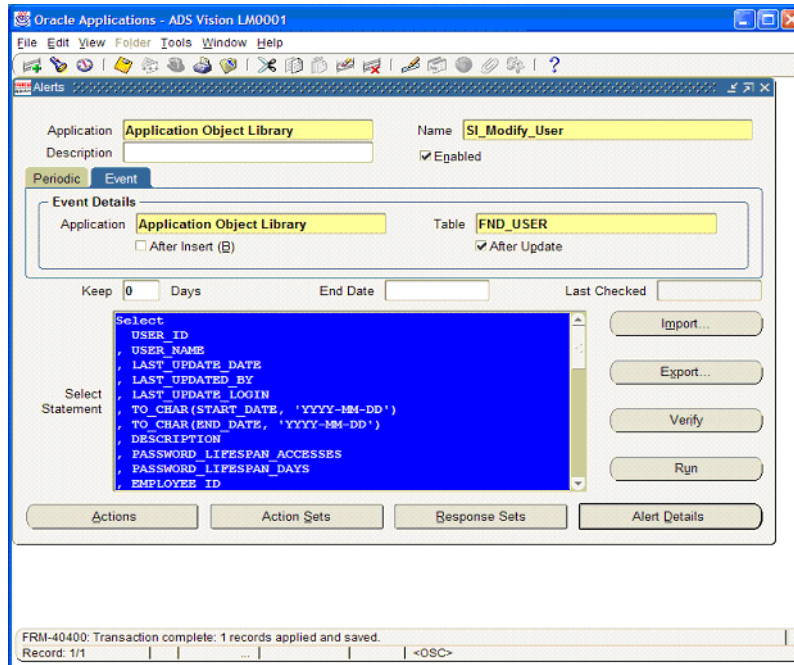
- 1 Display the alert definition window from the Oracle Application Alert Manager.

The screenshot shows the Oracle Applications Alert Manager window. The window title is "Oracle Applications - ADS Vision LM0001". The menu bar includes File, Edit, View, Folder, Tools, Window, and Help. The toolbar contains various icons for file operations and help. The main window is titled "Alerts" and contains the following fields and controls:

- Application:** Application Object Library
- Description:** (empty field)
- Name:** SI_Modify_User
- Enabled:** Enabled
- Event Details:**
 - Application:** Application Object Library
 - Table:** FND_USER
 - After Insert (I):** After Insert (I)
 - After Update:** After Update
- Keep:** 0 Days
- End Date:** (empty field)
- Last Checked:** (empty field)
- Select Statement:** SELECT COLUMN NAME INTO <OUTPUT1 FROM TABLE NAME
- Buttons:** Import..., Export..., Verify, Run, Actions, Action Sets, Response Sets, Alert Details
- Status Bar:** Record: 1/1

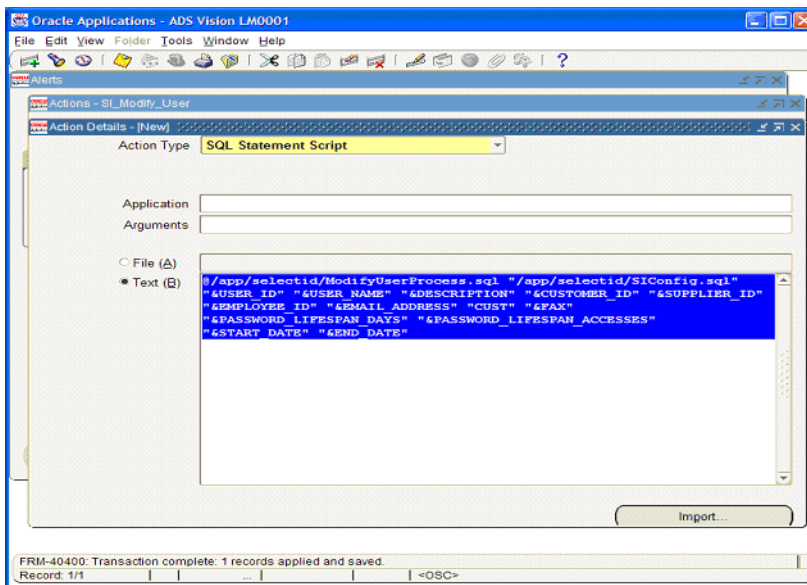
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Application Object Library** in the Application field.
- 4 Enter **FND_USER** in the Table field.
- 5 Select the **After Update** option. Make sure that the **After Insert** option is deselected.
- 6 Enter the alert name and description.
- 7 Save the alert.

- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyUserSelect.sql` script.
- 9 Click **OK** on the file upload window.



- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Enter the action name, such as `SENDSPLM`. Select **Detail** as the action level, then save the alert.
- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.

- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the ModifyUserAction.sql script. This populates the window with the content of the script. ModifyUserAction.sql calls the ModifyUserProcess.sql script with user creation arguments.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to /app/selectid. Modify the directory name to specify the correct directory name. Also, modify the path to the SIConfig.sql script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as SPMLACTION, and save.
- 22 Click the **Members** tab.

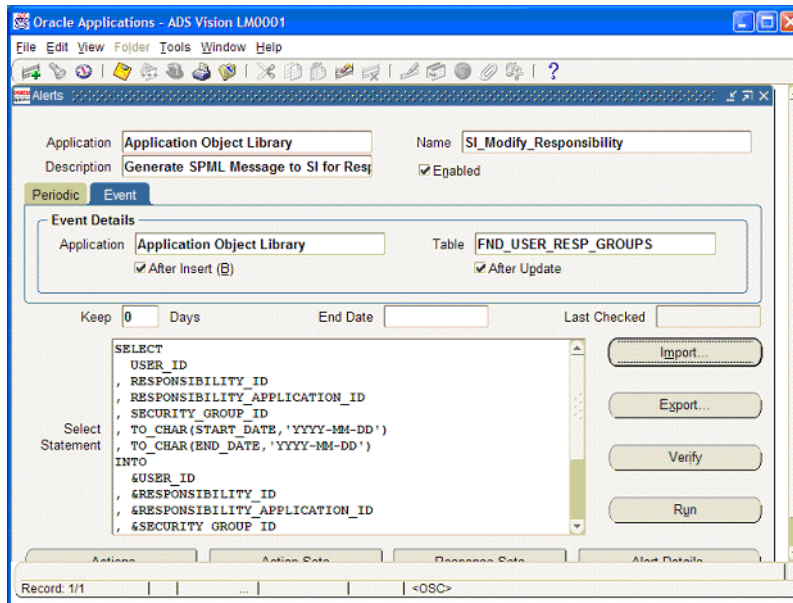
- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).
- 27 Save the alert. The alert is now enabled and active.

Defining an Alert for Application User Responsibility Modifications (v11.5.9)

Complete the following steps on Oracle 11.5.9 to define an alert that is triggered when an Application user's responsibilities are modified:

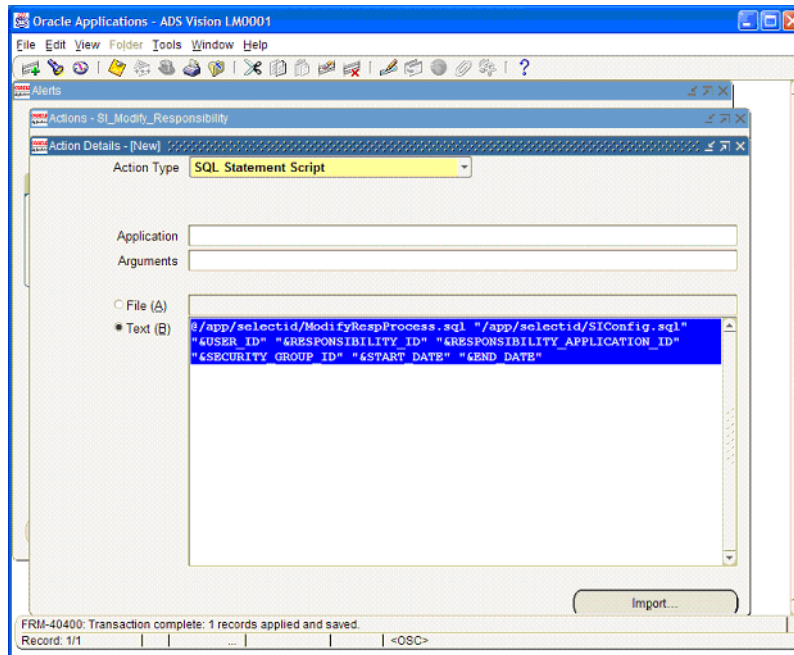
- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Application Object Library** in the Application field.
- 4 Enter **FND_USER_RESP_GROUPS** in the Table field.
- 5 Select the **After Insert** and **After Update** options.
- 6 Enter the alert name and description.
- 7 Save the alert.
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the **Select Identity** directory (created in [Step 2 on page 13](#)) and select the `ModifyRespSelect.sql` script.

- 9 Click **OK** on the file upload window.



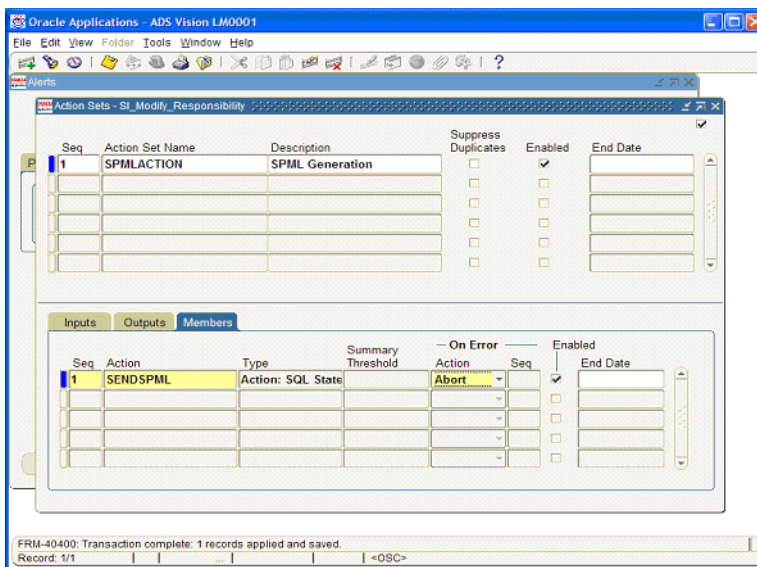
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Enter the action name, such as **SENDSPML**. Select **Detail** as the action level, then save the alert.
- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyRespAction.sql` script. This populates the window with the

content of the script. `ModifyRespAction.sql` calls the `ModifyRespProcess.sql` script.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to `/app/selectid`. Modify the directory name to specify the correct directory name. Also, modify the path to the `SIConfig.sql` script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as `SPMLACTION`, and save.
- 22 Click the **Members** tab.

23 Choose the action defined above and save the alert.



24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.

25 In the alert details window, click the **Installations** tab.

26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).

27 Save the alert. The alert is now enabled and active.

Defining an Alert for Application User Responsibility Modifications (v11.5.10)

Complete the following steps on Oracle 11.5.10 to define an alert that is triggered when an Application user's responsibilities are modified:

- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Application Object Library** in the Application field.

- 4 Enter `wf_local_user_roles` in the Table field.
- 5 Select the **After Insert** and **After Update** options.
- 6 Enter the alert name and description.
- 7 Save the alert.
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyRespSelect.sql` script.
- 9 Click **OK** on the file upload window.
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Select **SQL Statement Script** for the Action Type.
- 13 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 14 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyRespAction.sql` script. This populates the window with the content of the script. `ModifyRespAction.sql` calls the `ModifyRespProcess.sql` script.
- 15 Edit the alert and delete the control characters at the end of each line. The last character of each line may have to be retyped.
- 16 Save the alert.
- 17 Close the Alert Details definition window and Action window and go to main alert definition window.
- 18 Click the **Action Sets** button.
- 19 Enter the action set name, such as `SPMLACTION`, and save.
- 20 Click the **Members** tab.
- 21 Choose the action defined above and save the alert.
- 22 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.

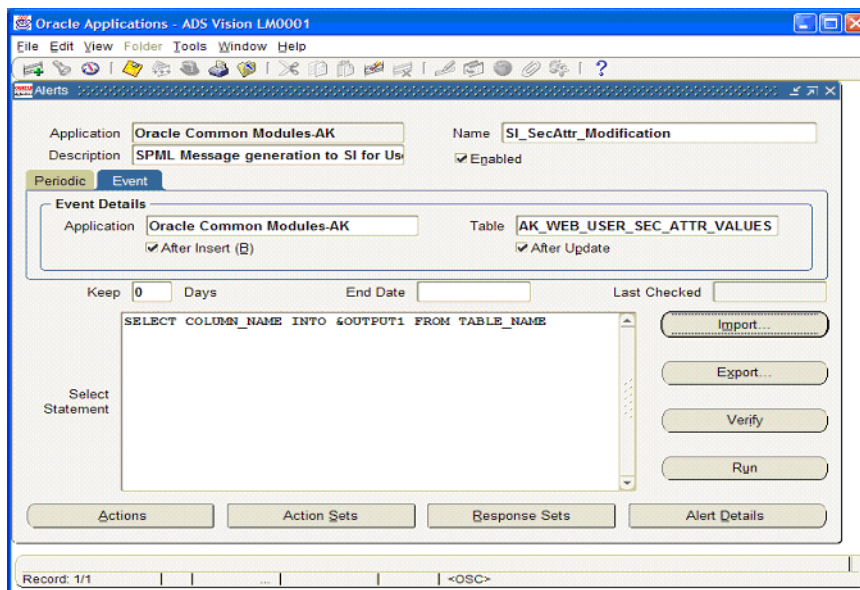
- 23 In the alert details window, click the **Installations** tab.
- 24 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).
- 25 Save the alert. The alert is now enabled and active.

Defining an Alert for Application User Securing Attribute Modifications

Complete the following steps to define an alert that is triggered when an Application user's responsibilities are modified:

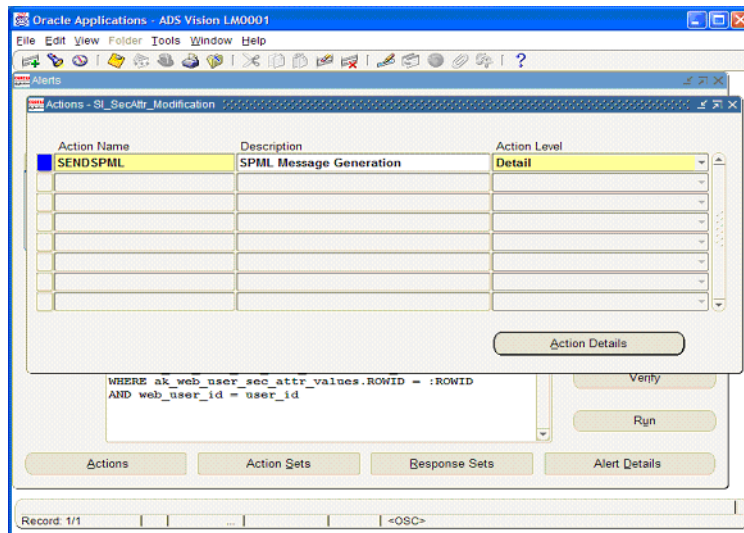
- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Common Modules-AK** in the Application field.
- 4 Enter **AK_WEB_USER_SEC_ATTR_VALUES** in the Table field.
- 5 Select the **After Insert** and **After Update** options.
- 6 Enter the alert name and description.

7 Save the alert.



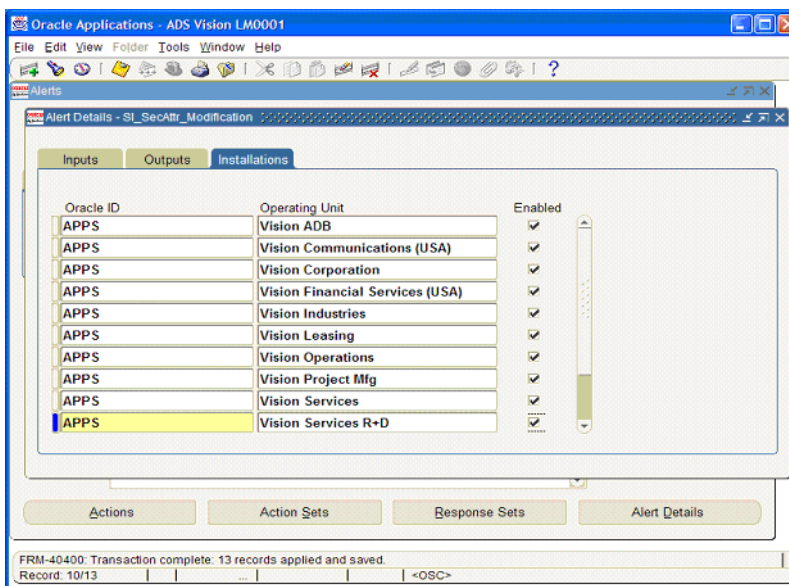
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifySecAttrSelect.sql` script.
- 9 Click **OK** on the file upload window.
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.

- 12 Enter the action name, such as **SENDSPML**. Select **Detail** as the action level, then save the alert.



- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the **Select Identity** directory (created in [Step 2 on page 13](#)) and select the **ModifySecAttrAction.sql** script. This populates the window with the content of the script. **ModifySecAttrAction.sql** calls the **ModifySecAttrProcess.sql** script.
- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to `/app/selectid`. Modify the directory name to specify the correct directory name. Also, modify the path to the **SIConfig.sql** script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.

- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as **SPMLACTION**, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).



- 27 Save the alert. The alert is now enabled and active.

Creating Alerts for Employee Reverse Synchronization

An Oracle Human Resources employee is stored in the Oracle Human Resources system to represent an employee in the organization. An employee is, therefore, a managed entity, not a real user on an Oracle Application system. However, an Oracle Human Resources employee can be an Oracle Application user, and vice versa.

The agent monitors the following events. If any of these events occur, the agent sends the new data to Select Identity (reverse synchronization). Oracle Human Resources employees are not assigned user names; thus, the agent creates a default user name by combining the first three characters of the last name with the person ID for each employee. To change the process for generating the user name, you must modify the alert select and process scripts.

- Creation of a new employee
- Changes in employee attributes
- Termination of an employee

The following employee attributes are synchronized with Select Identity:

- CompanyName
- Email
- First Name
- Last Name
- Work Phone
- Job
- Position
- Grade
- Location
- Address1
- Address2
- City
- State
- Zipcode
- Country
- Home Phone
- Date of Birth
- Manager Flag (indicates whether the employee is a manager)

Scripts are provided by the agent for the following events:

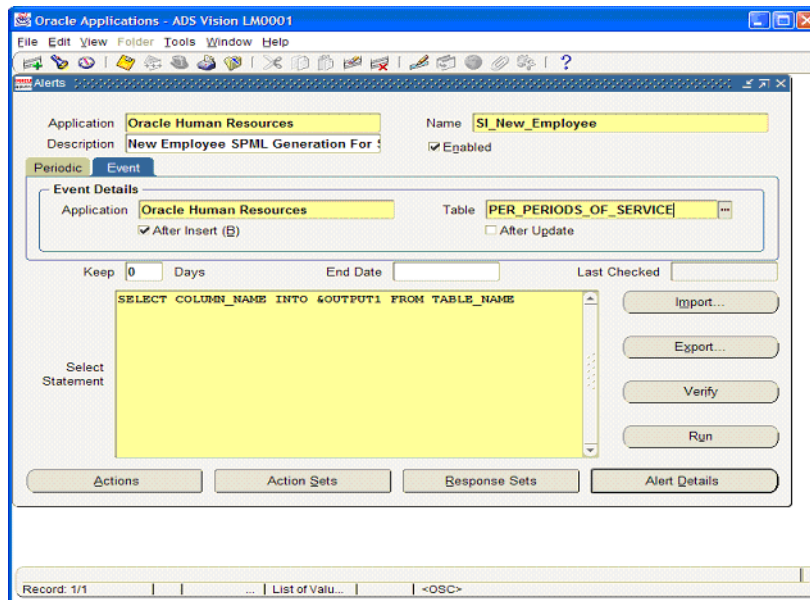
- **Add New Employee**
Application Name: Oracle Human Resources
Table Name: PER_PERIODS_OF_SERVICE
Event to Monitor: After Insert
Select SQL Script: AddEmployeeSelect.sql
Action SQL Script: AddEmployeeAction.sql
- **Modify Employee Info**
Application Name: Oracle Human Resources
Table Name: PER_ALL_PEOPLE_F
Event to Monitor: After Update
Select SQL Script: Modify_EMP_PAPF_Select.sql
Action SQL Script: ModifyEmployeeAction.sql
- **Modify Employee Address**
Application Name: Oracle Human Resources
Table Name: PER_ADDRESSES
Event to Monitor: After Insert After Update
Select SQL Script: Modify_EMP_ADDR_Select.sql
Action SQL Script: ModifyEmployeeAction.sql
- **Modify Employee Job**
Application Name: Oracle Human Resources
Table Name: PER_ALL_ASSIGNMENTS_F
Event to Monitor: After Insert After Update
Select SQL Script: Modify_EMP_ASGN_Select.sql
Action SQL Script: ModifyEmployeeAction.sql
- **Terminate Employee**
Application Name: Oracle Human Resources
Table Name: PER_ALL_PEOPLE_F
Event to Monitor: After Insert
Select SQL Script: TerminateEmployeeSelect.sql
Action SQL Script: TerminateEmployeeAction.sql

The following sections describe how to create an alert in the Oracle Application system for each of these events.

Defining a New Employee Alert

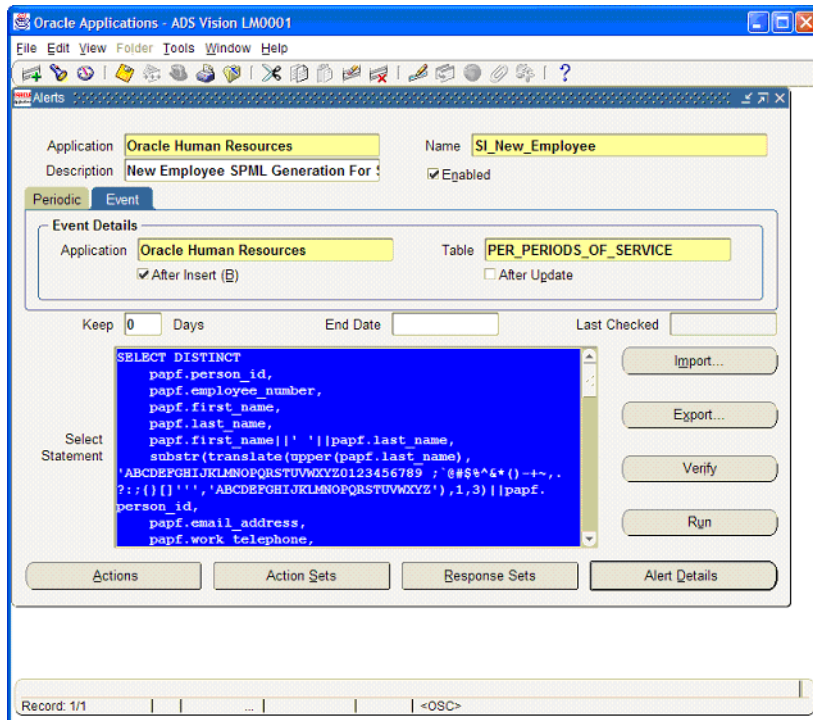
Complete the following steps to define an alert that is triggered when a new employee is created in the Oracle Human Resources system:

- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Human Resources** in the Application field.
- 4 Enter **PER_PERIODS_OF_SERVICE** in the Table field.
- 5 Select the **After Insert** option. Make sure the **After Update** option is deselected.
- 6 Enter the alert name and description.
- 7 Save the alert.



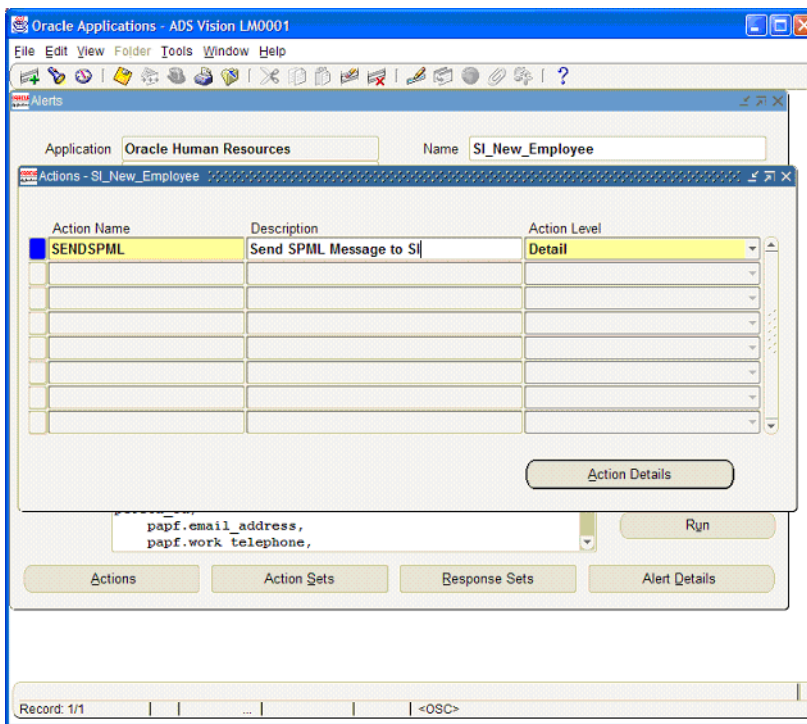
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `AddEmployeeSelect.sql` script.

9 Click **OK** on the file upload window.



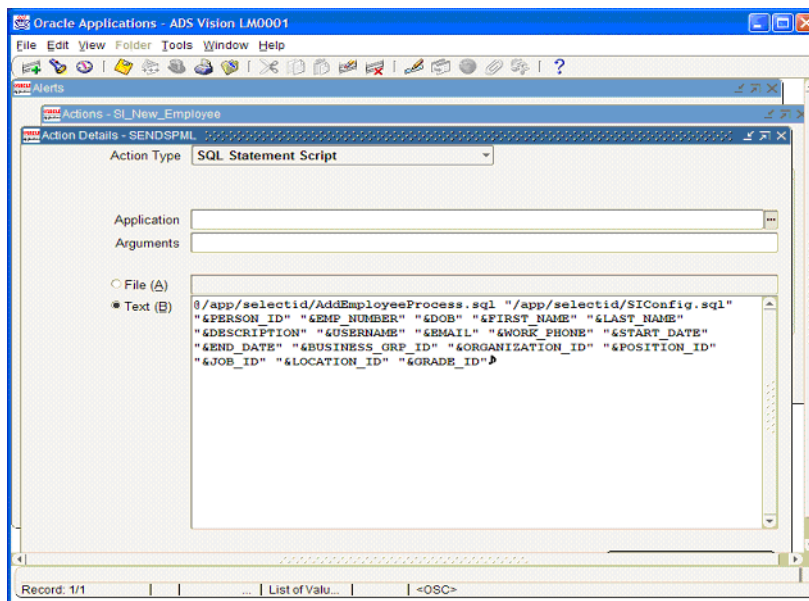
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.

- 12 Enter the action name, such as **SENDSPML**. Select **Detail** as the action level, then save the alert.



- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the

AddEmployeeAction.sql script. This populates the window with the content of the script.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to /app/selectid. Modify the directory name to specify the correct directory name. Also, modify the path to the SIConfig.sql script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as SPMLACTION, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.

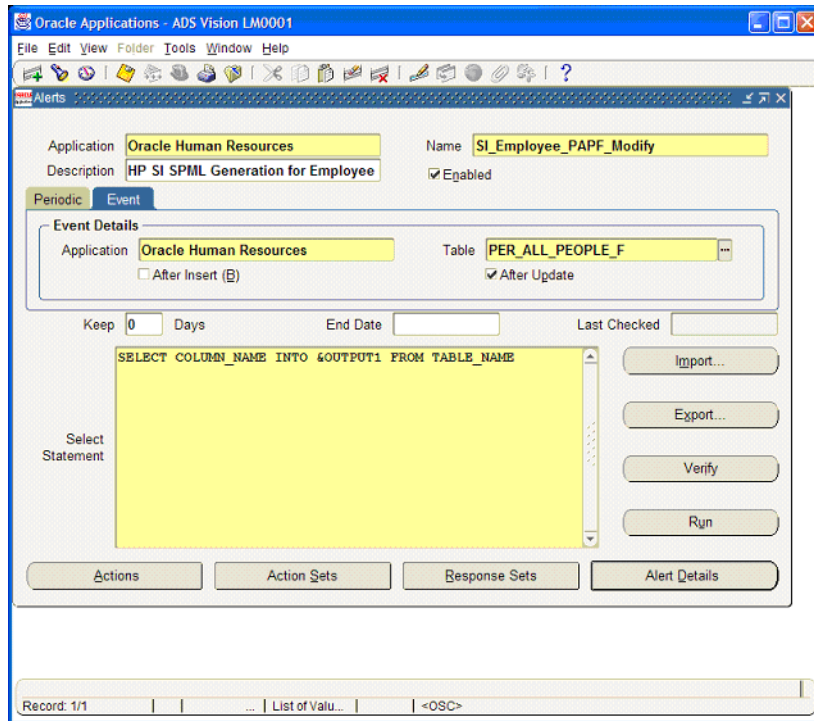
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).
- 27 Save the alert. The alert is now enabled and active.

Defining an Alert for Employee Modifications

Complete the following steps to define an alert that is triggered when an employee is modified in the Oracle Human Resources system:

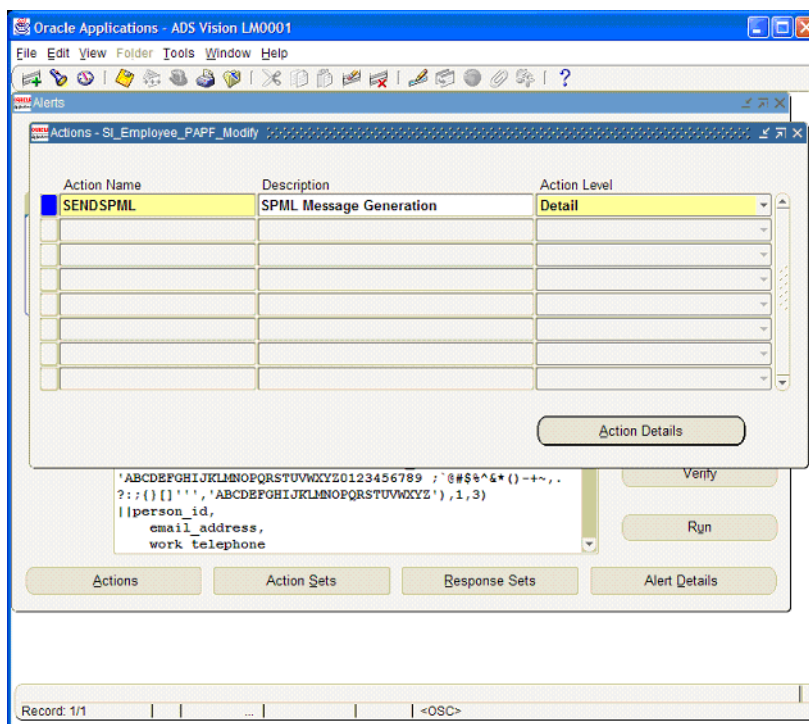
- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Human Resources** in the Application field.
- 4 Enter **PER_ALL_PEOPLE_F** in the Table field.
- 5 Select the **After Update** option. Make sure the **After Insert** option is deselected.
- 6 Enter the alert name and description.

7 Save the alert.



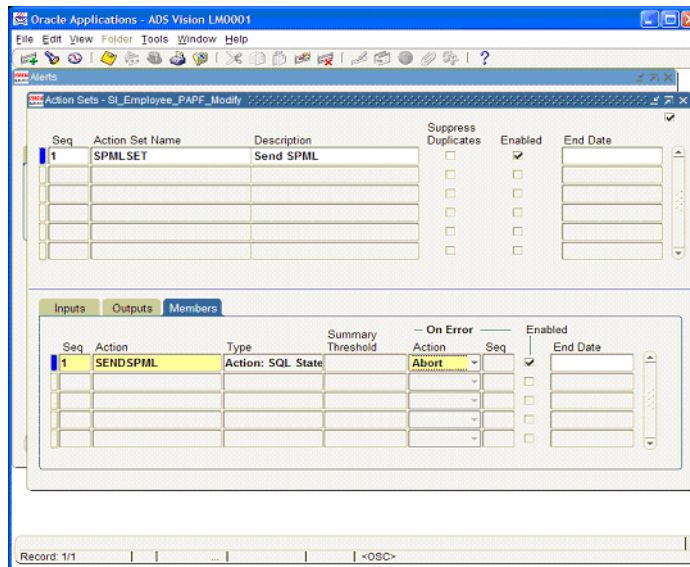
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyEMP_PAPF_Select.sql` script.
- 9 Click **OK** on the file upload window.
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.

- 12 Enter the action name, such as **SENDSPML**. Select **Detail** as the action level, then save the alert.



- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyEmployeeAction.sql` script. This populates the window with the content of the script.
- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to `/app/selectid`. Modify the directory name to specify the correct directory name. Also, modify the path to the `SIconfig.sql` script to specify the correct installation directory.

- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as **SPMLACTION**, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.



- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).
- 27 Save the alert. The alert is now enabled and active.

Defining an Alert for Employee Address Modifications

Complete the following steps to define an alert that is triggered when an employee's address is modified in the Oracle Human Resources system:

- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Human Resources** in the Application field.
- 4 Enter **PER_ADDRESSES** in the Table field.
- 5 Select the **After Insert** and **After Update** options.
- 6 Enter the alert name and description.
- 7 Save the alert.

The screenshot shows the Oracle Alerts configuration window for defining an event alert. The window title is "Oracle Applications - ADS Vision LM0001". The "Alerts" window has a menu bar (File, Edit, View, Folder, Tools, Window, Help) and a toolbar. The main configuration area is divided into "Periodic" and "Event" tabs, with the "Event" tab selected.

Alert Configuration:

- Application:** Oracle Human Resources
- Name:** SI_Employee_Addr_Modify
- Description:** HP SI SPML Generation for Employee
- Enabled:**

Event Details:

- Application:** Oracle Human Resources
- Table:** PER_ADDRESSES
- After Insert:** (B)
- After Update:**

Additional Fields:

- Keep:** 0 Days
- End Date:** [Empty field]
- Last Checked:** [Empty field]

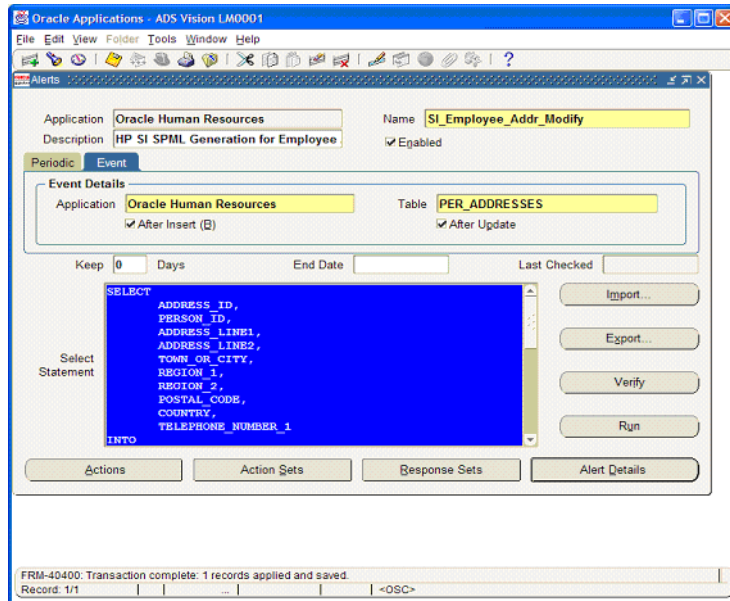
Select Statement:

```
SELECT COLUMN_NAME INTO &OUTPUT1 FROM TABLE_NAME
```

Buttons: Import..., Export..., Verify, Run, Actions, Action Sets, Response Sets, Alert Details.

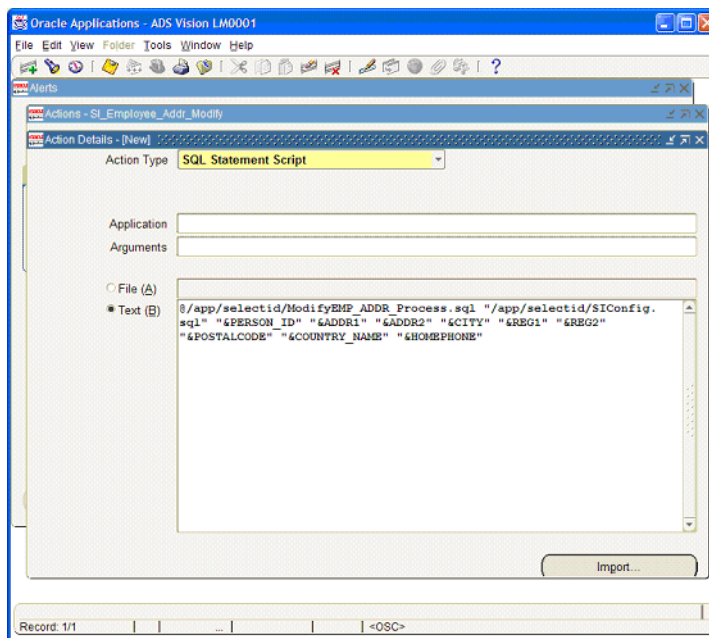
Status Bar: Record: 1/1 | ... | <-OSC>

- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyEMP_ADDR_Select.sql` script.
- 9 Click **OK** on the file upload window.



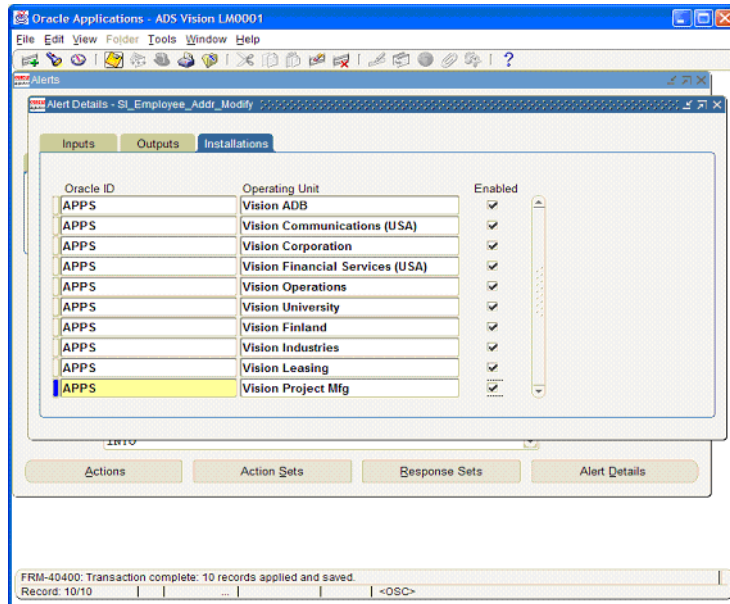
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Enter the action name, such as `SENDSPML`. Select **Detail** as the action level, then save the alert.
- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the

ModifyEmployeeAction.sql script. This populates the window with the content of the script.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to /app/selectid. Modify the directory name to specify the correct directory name. Also, modify the path to the SIConfig.sql script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as SPMLACTION, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.

- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).



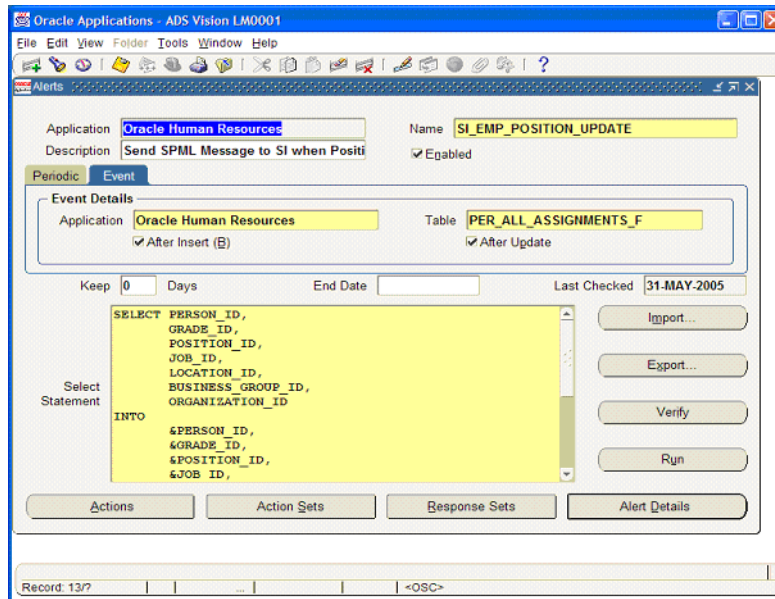
- 27 Save the alert. The alert is now enabled and active.

Defining an Alert for Employee Position Modifications

Complete the following steps to define an alert that is triggered when an employee's assignments are modified in the Oracle Human Resources system:

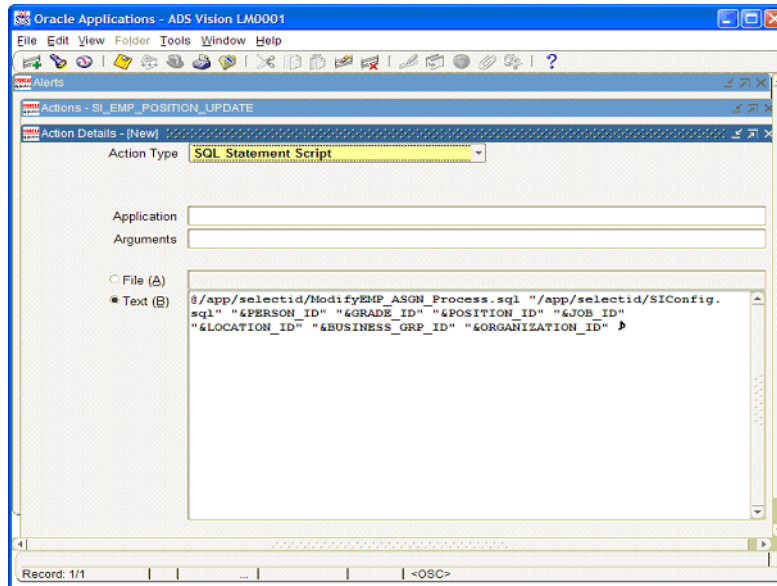
- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Human Resources** in the Application field.
- 4 Enter **PER_ALL_ASSIGNMENTS** in the Table field.
- 5 Select the **After Insert** and **After Update** options.
- 6 Enter the alert name and description.

- 7 Save the alert.
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyEMP_ASGN_Select.sql` script.
- 9 Click **OK** on the file upload window.



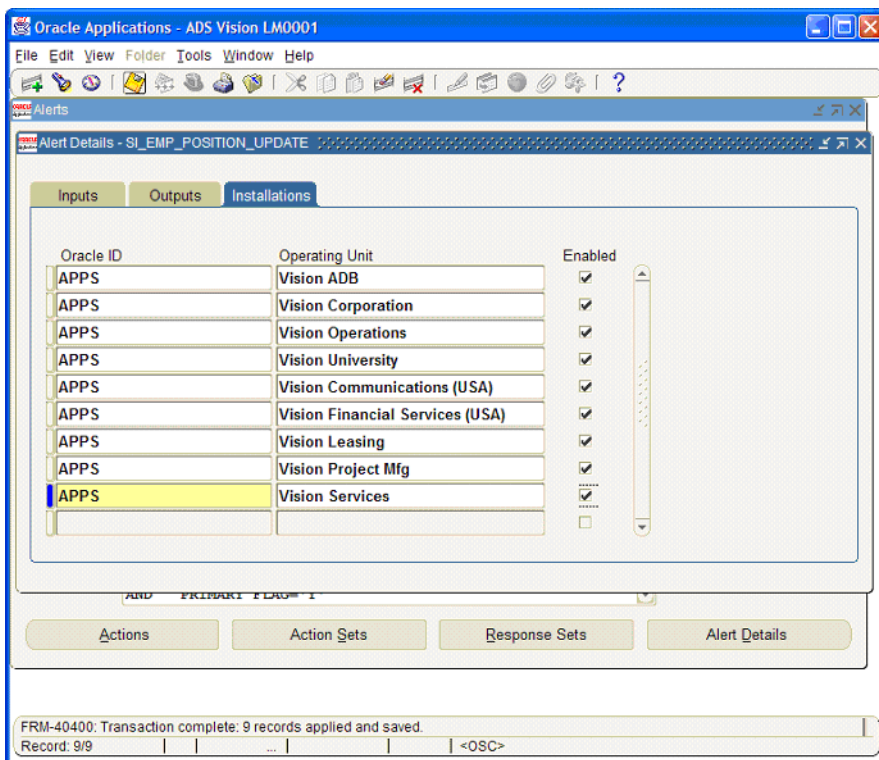
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 12 Enter the action name, such as `SENDSPLM`. Select **Detail** as the action level, then save the alert.
- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.

- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `ModifyEmployeeAction.sql` script. This populates the window with the content of the script.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to `/app/selectid`. Modify the directory name to specify the correct directory name. Also, modify the path to the `SIConfig.sql` script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as `SPMLACTION`, and save.
- 22 Click the **Members** tab.

- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.
- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).



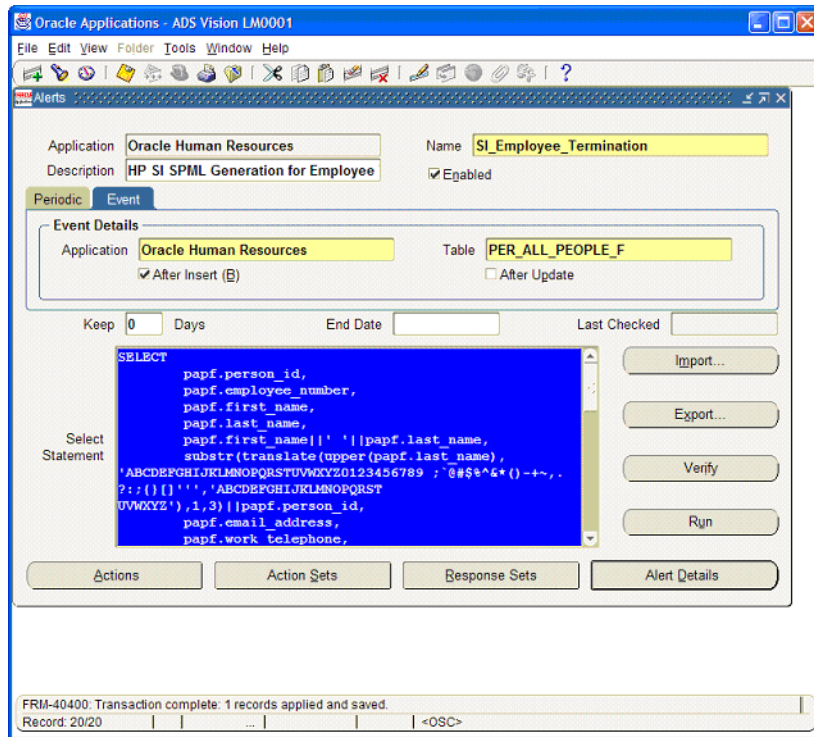
- 27 Save the alert. The alert is now enabled and active.

Defining an Employee Termination Alert

Complete the following steps to define an alert that is triggered when an employee is terminated in the Oracle Human Resources system:

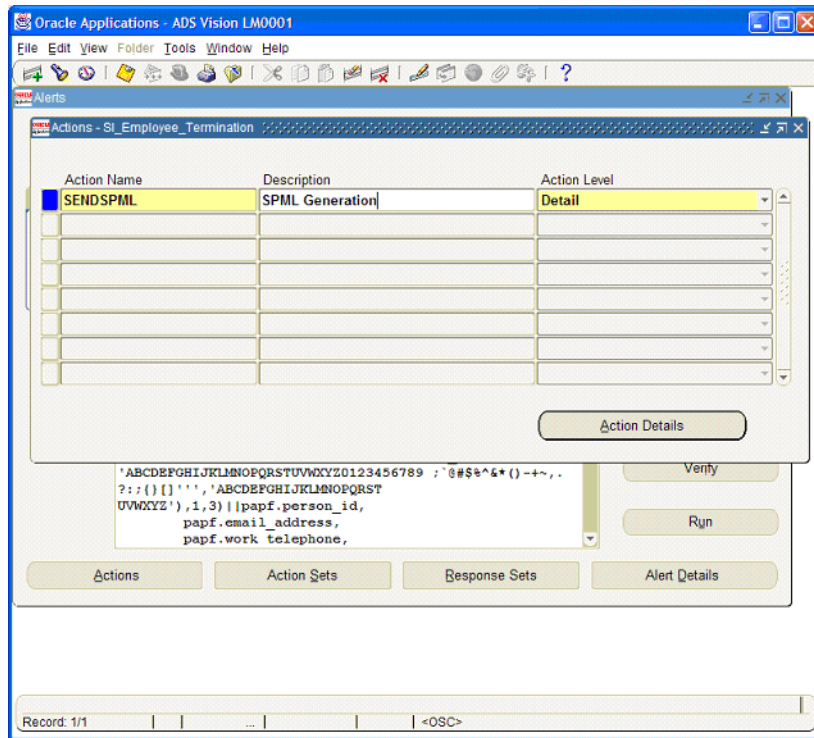
- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Event** tab to define the event alert.
- 3 Enter **Oracle Human Resources** in the Application field.
- 4 Enter **PER_ALL_PEOPLE_F** in the Table field.
- 5 Select the **After Insert** option. Make sure the **After Update** option is deselected.
- 6 Enter the alert name and description.
- 7 Save the alert.
- 8 Click the **Import** button on the right to import the Select Statement. A file upload window is displayed. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the `TerminateEmployeeSelect.sql` script.

- 9 Click **OK** on the file upload window.



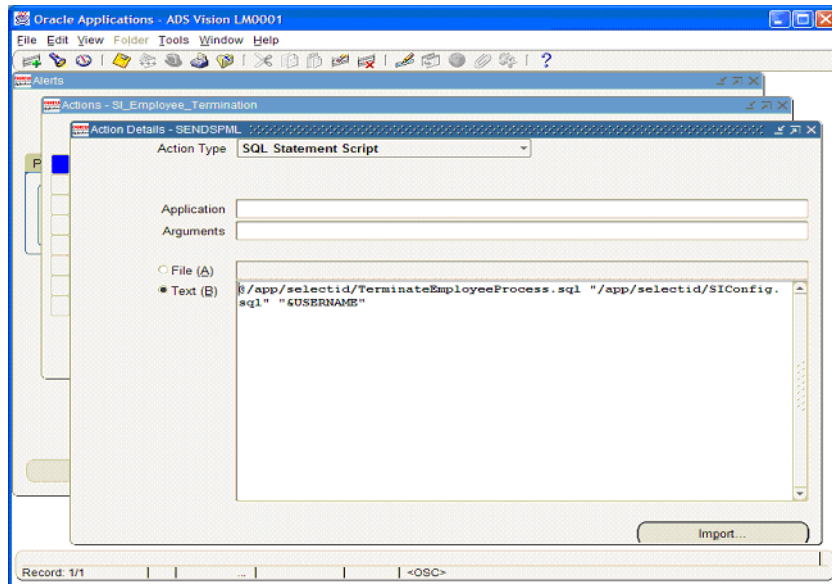
- 10 Verify the statement by clicking the **Verify** button on the Select Statement window, which is populated with the SQL script from the imported file.
- 11 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.

- 12 Enter the action name, such as **SENDSPML**. Select **Detail** as the action level, then save the alert.



- 13 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 14 Select **SQL Statement Script** for the Action Type.
- 15 Click **Text** to define the SQL statement in the window. Make sure that the Application and Arguments fields are empty.
- 16 Click the **Import** button to import the SQL action script. Browse to the Select Identity directory (created in [Step 2 on page 13](#)) and select the

TerminateEmployeeAction.sql script. This populates the window with the content of the script.



- 17 The action SQL script calls a processing script located in the installation directory. By default, the installation directory is set to /app/selectid. Modify the directory name to specify the correct directory name. Also, modify the path to the SIConfig.sql script to specify the correct installation directory.
- 18 Save the alert.
- 19 Close the Alert Details definition window and Action window and go to main alert definition window.
- 20 Click the **Action Sets** button.
- 21 Enter the action set name, such as SPMLACTION, and save.
- 22 Click the **Members** tab.
- 23 Choose the action defined above and save the alert.
- 24 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 25 In the alert details window, click the **Installations** tab.

- 26 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. It is important to specify all the operating units that the alert may be active (such as Vision Corporation, Vision Operations, and so on).
- 27 Save the alert. The alert is now enabled and active.

The SPMLProcess.sql script and SPML Messages

The event alerts insert action records into the `SIORAERP.USER_REQUESTS` table every time a user activity occurs. At regular intervals, the `SPMLProcess.sql` script can be run to process the individual user requests that were collected. `SPMLProcess.sql` performs following actions:

- Consolidates the user requests into `provision_requests` for each user. For example, multiple modify requests can be triggered for a user over time. The consolidation combines all the events into a single modify request.
- Performs retry processing. If an SPML message failed during processing, the `MAXRETRY` parameter in `SIConfig.sql` file specifies how many times SPML message is resent. The `RETRYINTVALMIN` specifies the number of minutes to wait before retry processing.
- Performs hung process processing. Resource-side errors can terminate the script before completing the SPML process. The parameter `HUNGINTVALMIN` specifies the number of minutes to wait before reprocessing the request.
- Processes the SPML message. Based on the user type, the `SPMLProcess.sql` script sends the appropriate attributes with `add`, `modify`, and `terminate` requests.

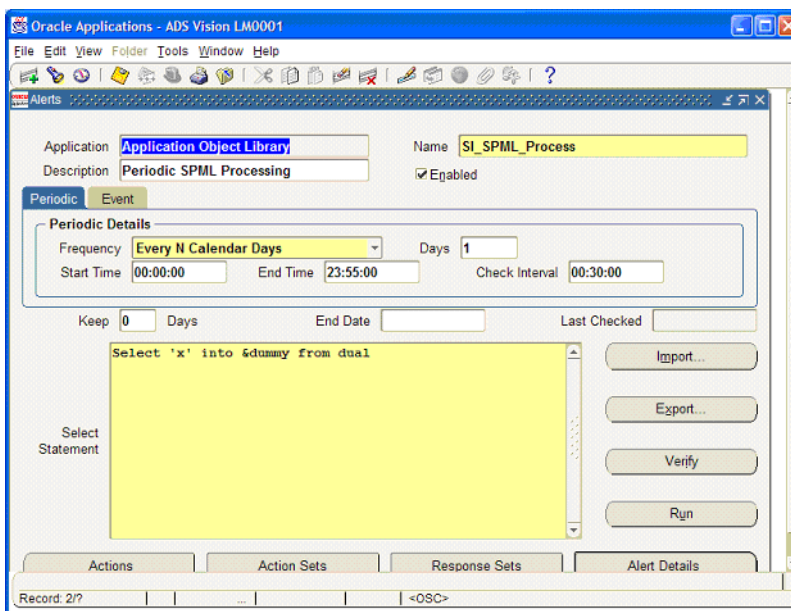
You can define a periodic alert, a concurrent program, or a job scheduler to process SPML messages. The following section describe how to define these.

Defining a Periodic Alert

A periodic alert can be defined to run the `SPMLProcess.sql` script at regular intervals. To define the period alert, complete these steps:

- 1 Display the alert definition window from the Oracle Application Alert Manager.
- 2 Click the **Periodic** tab to define the periodic alert.

- 3 Enter **Application Object Library** in the Application field.
- 4 Enter the alert name and description.
- 5 Save the alert.
- 6 Enter **Select 'x' into &dummy from dual** in the Select statement window and verify the statement.
- 7 Select **Every Day** as the frequency.
- 8 Enter **00:00:00** as the start time and **23:55:00** as the end time.
- 9 Enter **00:30:00** (every 30 minutes) as the check interval.



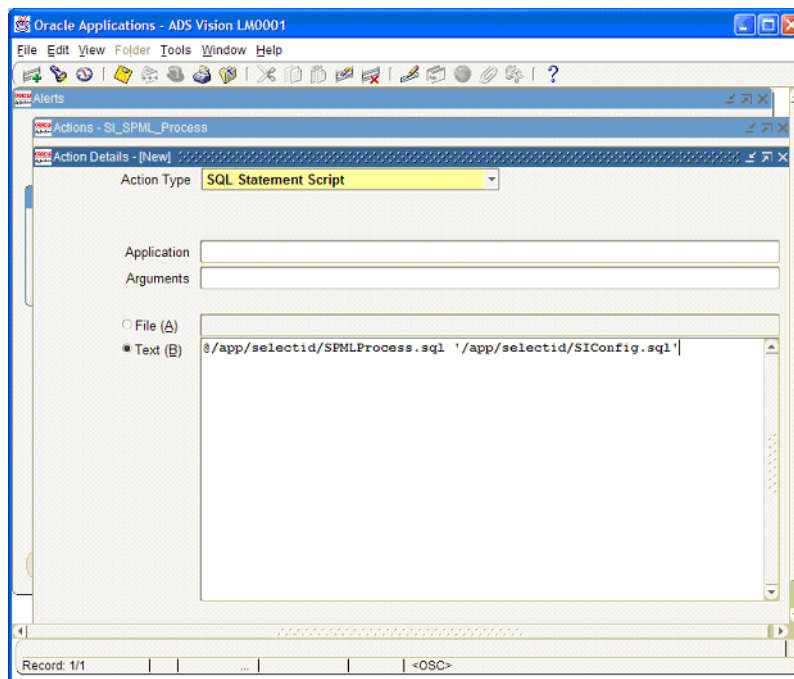
- 10 Define the action script by clicking the **Actions** button on the bottom left-hand side of the window.
- 11 Enter the action name, such as **SENDSFML**. Select **Detail** as the action level, then save the alert.
- 12 Click the **Action Details** button while the action is highlighted. The Action Detail definition window is displayed.
- 13 Select **SQL Statement Script** for the Action Type.

- 14 Click **Text** to define the SQL statement in the window. Make sure that the **Application** and **Arguments** fields are empty.
- 15 Enter `@SI Install Dir/SPMLProcess.sql 'SI_Install_Dir/SIConfig.sql'` in the text field. For example, for Windows, if the Select Identity installation directory is C:\si302, enter the following for Windows:

```
@c:\si302\SPMLProcess.sql 'c:\si302\SIConfig.sql'
```

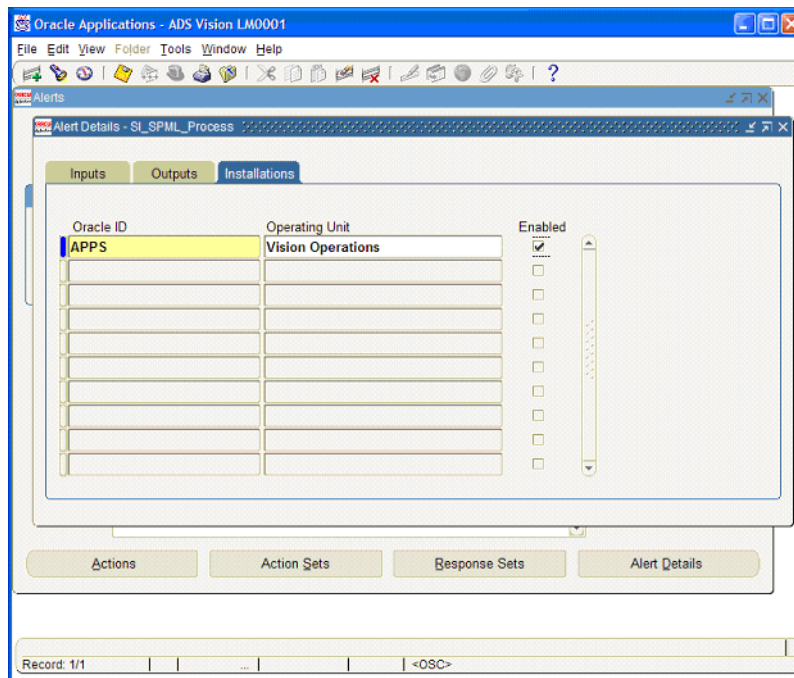
Enter the following for UNIX:

```
@/app/selectid/SPMLProcess.sql '/app/selectid/SIConfig.sql'
```



- 16 Save the alert.
- 17 Close the Alert Details definition window and Action window and go to main alert definition window.
- 18 Click the **Action Sets** button.
- 19 Enter the action set name, such as **SPMLACTION**, and save.

- 20 Click the **Members** tab.
- 21 Choose the action defined above and save the alert.
- 22 Close the Action Sets window and click on **Alert Details** to display the Alert Details window.
- 23 In the alert details window, click the **Installations** tab.
- 24 Enter the application user ID (**apps**) and the operating units (in case of multi-orgs) where the alert will be activated. For each installation, specify that the periodic alert will process `SPMLProcess.sql` again, and define the main installation for the sysadmin user only.



- 25 Make sure periodic alert scheduler is enabled in the Request →Schedule form from the Alert Manager menu.
- 26 Save the alert. The alert is now enabled and active.
- 27 Make sure the periodic alert is scheduled in the Request →Schedule window.

Defining a Concurrent Program

Instead of defining a periodic alert, you can define a concurrent request to process the SPML messages. A concurrent request provides an additional level of control and a printing option. To define a concurrent program, complete the following steps:

- 1 Modify the `sispml.sql` script to include the correct path to the agent script location. Here is an example:

```
@c:\si302\SPMLProcess.sql 'c:\si302\SIconfig.sql'
```

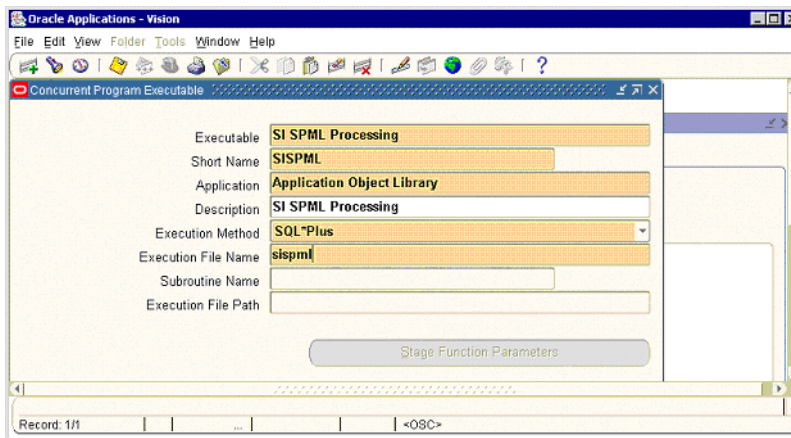
- 2 Copy the `sispml.sql` script to the `$FND_TOP/sql` (on UNIX) or `%FND_TOP%\sql` (on Windows) directory on the applicaiton server. Ensure that the file is readable by all. To set the environment variable, run the environment setup script in the Oracle Application home directory, as in these examples:

```
/app/oracle/visappl/VIS_alimosa.env (on UNIX)
```

```
f:\oracle\visappl\VIS_kod.cmd (on Windows)
```

- 3 Log on to Oracle Applications as SYSADMIN.
- 4 Select **Concurrent** → **Program** → **Executables**.
- 5 Enter the name and description of the custom program.
- 6 Enter **SISPML** as the short name for the custom program.
- 7 Enter **Application Object Library** in the Application field.
- 8 Select **SQL*Plus** for the execution method.
- 9 Enter **sispml** as the execution file name.

10 Save the executable definition.



- 11 Select **Concurrent** → **Program** → **Define**.
- 12 Enter a program name and description.
- 13 Enter **SISPML** for the short name.
- 14 Enter **Application Object Library** in the **Application** field.
- 15 Enter the **SISPML** as the executable name. This is the short name defined above. If different short name is used, enter that name.
- 16 Select the **User** in **SRS**, **Restart on System Failure**, **NLS Compliant**, **Save**, and **Print** options.

17 Save the program.

Oracle Applications - Vision

File Edit View Folder Tools Window Help

Concurrent Programs

Program: **SI SPML Processing** Enabled

Short Name: **SISPML**

Application: **Application Object Library**

Description: **HP SI SPML Processing for Application Reverse Agent**

Executable

Name: **SISPML** Options:

Method: **SQL*Plus** Priority:

Request

Type:

Incrementor:

MLS Function:

Use In SRS Allow Disabled Values

Run Alone Restart on System Failure

Enable Trace NLS Compliant

Output

Format: **Text**

Save (S) Print

Columns:

Rows:

Style:

Style Required

Printer:

Record: 1/1 | ... | <OBC>

18 Select the **Security** → **Responsibility** → **Request** form.

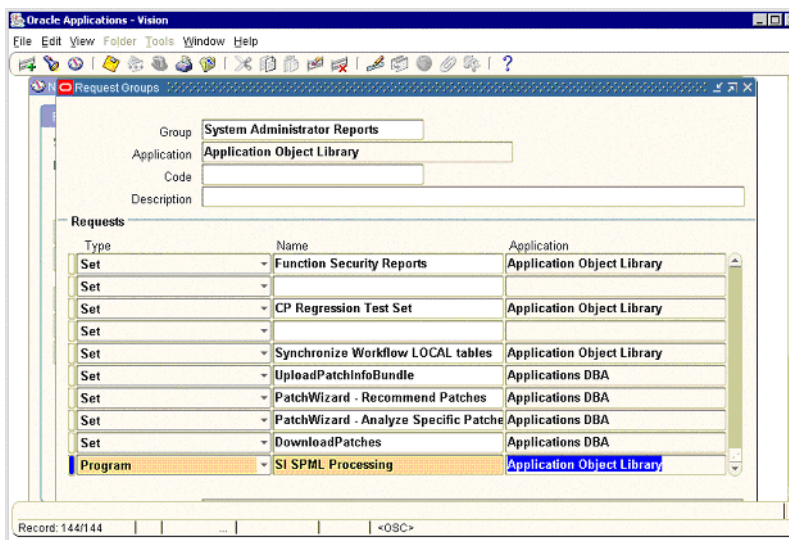
19 Find and choose the System Administrator Reports for the group.

20 At the end of the requests rows, add a new row by pressing TAB after the last field of last record.

21 Select **Program** as the type.

22 Enter the concurrent program defined for SPML processing above.

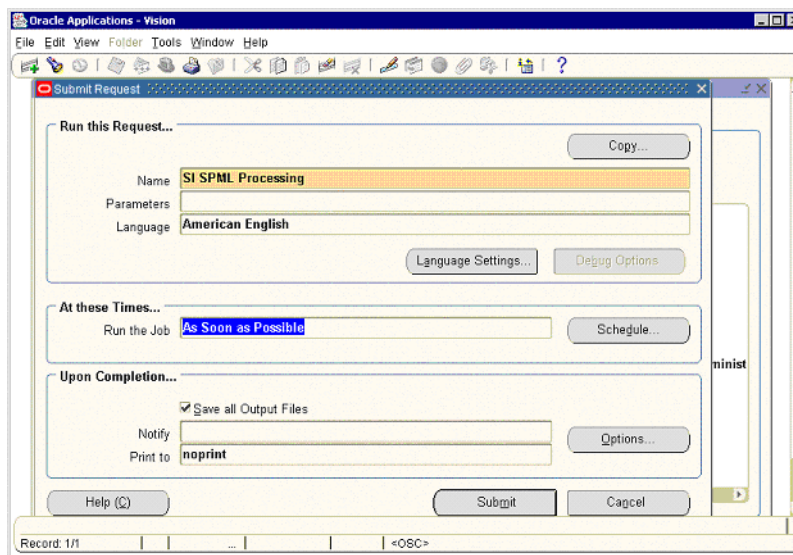
23 Save the form.



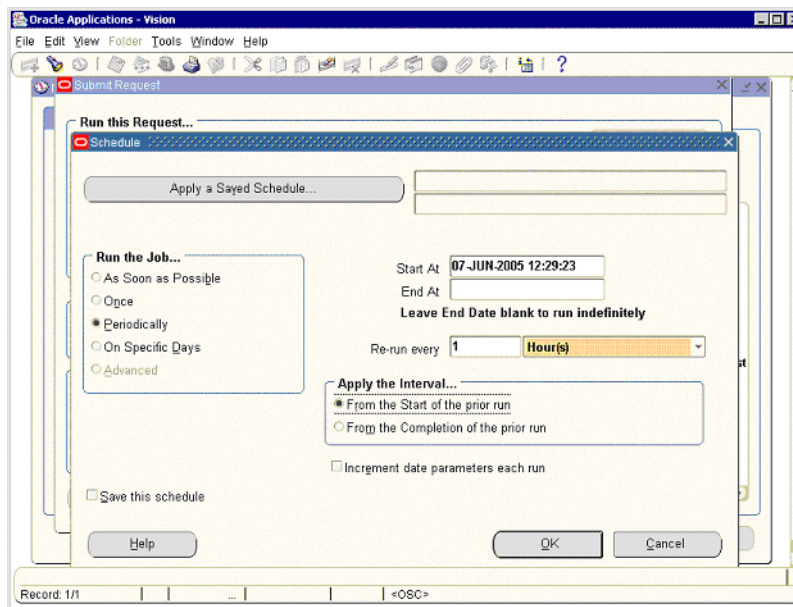
Now the SYSADMIN user can schedule the SPML process as concurrent request.

- 24 Select the **Requests** → **Run** form.
- 25 Select **Single Request**.
- 26 Choose the SPML processing concurrent program defined in above for the name.
- 27 For troubleshooting purposes, select **As Soon As Possible** for on-demand processing. For production mode, schedule for periodic execution.
- 28 Click **Schedule**.
- 29 Select **Periodically** in the Run the Job section.
- 30 Select the job execution interval.

31 Save the request.



32 Submit the request.



Defining a Job Scheduler

You can also define an OS job scheduler to process SPML messages. This is done outside of the Oracle Application, using the OS features. Because the script is run in the database, any job scheduler with access to the Oracle Database client can execute the job.

For example, you could use the Oracle Enterprise manager management console. To process SPML messages, run the `SPMLProcess.sql` script with `SIconfig.sql` as the parameter. If the job scheduler is external to the database and uses a client-based SQL*Plus tool, the `SPMLProcess.sql` and `SIconfig.sql` files need to be accessible from the SQL*Plus client.

Custom Attributes for Employee Modifications

To provision additional attributes for employees, several modifications need to be made to the agent:

- 1 The mapping file must be edited to include the additional attributes.
- 2 The event alert must be defined to monitor the attribute changes.
- 3 The `SPMLProcess.sql` script must be changed to incorporate the logic for additional attributes.

The following attributes are added in the example procedure below:

- `EMP_REVIEW_DATE`
- `EMP_REVIEW_RATING`

The following example describes the steps for adding employee attributes to track performance review dates and ratings:

- 1 Add the attributes to the mapping file (`ORAERPEMP-11-5-9.xml` for version 11.5.9 or `ORAERPEMP-11-5-10.xml` for version 11.5.10). See [Understanding the Mapping Files on page 85](#) for details about these files.

In the mapping file, the `<attributeDefinitionReference>` block and `<attributeDefinition>` block must be defined. Here are examples for the attributes listed above:

```
<attributeDefinitionReference name="ATTR_EMP_Review_Date"
required="false" concero:tafield="EMP_Review_Date"
concero:resfield="[x_emp_review_date] [EMP_REVIEW_DATE] [] [DATE]"
concero:init="true" />
```

```

<attributeDefinitionReference name="ATTR_EMP_Review_Rating"
required="false" concero:tafield="EMP_Review_Rating"
concero:resfield="[x_emp_review_rating] [EMP_REVIEW_RATING] [] [NUMBER]" concero:init="true" />

<attributeDefinition name="ATTR_EMP_Review_Date"
description="Employee Review Date" type="xsd:date">
  <properties>
    <attr name="minLength">
      <value>1</value>
    </attr>
    <attr name="maxLength">
      <value>64</value>
    </attr>
  </properties>
</attributeDefinition>

<attributeDefinition name="ATTR_EMP_Review_Rating"
description="Employee Review Rating" type="xsd:string">
  <properties>
    <attr name="minLength">
      <value>1</value>
    </attr>
    <attr name="maxLength">
      <value>64</value>
    </attr>
  </properties>
</attributeDefinition>

```

2 Modify oracle11ierp.xsl to add the attribute information for reverse provisioning. Note the attribute name is lowercase for the first line.

```

<xsl:when test="$ATTRNAME = 'emp_review_date'">
  <xsl:call-template name="AttributeBuilder">
    <xsl:with-param name="DSMLELEMENT" select="$DSMLELEMENT"/>
    <xsl:with-param name="ATTRNAME" select="'EMP_Review_Date'" />
  </xsl:call-template>
</xsl:when>

<xsl:when test="$ATTRNAME = 'emp_review_rating'">
  <xsl:call-template name="AttributeBuilder">
    <xsl:with-param name="DSMLELEMENT" select="$DSMLELEMENT"/>
    <xsl:with-param name="ATTRNAME"
      select="'EMP_Review_Rating'" />
    <xsl:with-param name="ATTRVALUE" select="$ATTRVALUE" />
  </xsl:call-template>
</xsl:when>

```

```

        <xsl:with-param name="MODIFYFLAG" select="$MODIFYFLAG" />
    </xsl:call-template>
</xsl:when>

```

- 3 **Modify the Select Identity resource and Service with the new mapping file and map the new attributes. See the *HP OpenView Select Identity Administrator's Guide* for details.**
- 4 **Modify the SIConfig.sql file to add the new attribute name and value container.**

```

/* Attribute name for employee review date */
SIEMPREVIEWDATE VARCHAR2(32) := 'EMP_Review_Date';

/* Attribute name for employee review rating */
SIEMPREVIEWRATING VARCHAR2(32) := 'EMP_Review_Rating';

/* Value container for review date. Format is YYYY-MM-DD.
 * TO_CHAR conversion of date value is nessary for SPML
 * transmission
 */
EMPREVIEWDATE VARCHAR2(32);

/* Value container for review rating. */
EMPREVIEWRATING VARCHAR2(32);

```

- 5 **Define a new event alert to monitor the performance review changes. The table used for employee performance review is PER_PERFORMANCE_REVIEWS. Create event alert on PER_PERFORMANCE_REVIEWS. The event alert should select the PERSON_ID. Refer to [Event Alert Definitions on page 18](#) for details on creating event alert for employees.**

Application Name: Human Resources

Table Name: PER_PERIODS_OF_SERVICE

Event to Monitor: After Update After Insert

Select SQL Script:

```

SELECT PERSON_ID
INTO &PERSON_ID
FROM PER_PERFORMANCE_REVIEWS
WHERE ROWID = :ROWID;

```

Action SQL Script:

```

INSERT INTO sioraerp.user_requests (request_num, ID, user_type,
action_type )
values (sioraerp.user_request_seq.nextval, &PERSON_ID, 'E', 'M');
COMMIT;

```

- 6** Modify the `SPMLProcess.sql` script to include the logic to retrieve the attribute values. Locate the following comment in the file:

```
/* Start Additional Attributes Here */
```

Insert the SQL code to retrieve the custom attribute values for the employee. The user ID is contained in `provision_req_rec.USER_ID` cursor variable.

Define the `MAX_REVIEW_DATE` variable in the declaration section in the top:

```
MAX_REVIEW_DATE Date;

/* Get the latest review date for the employee */
SELECT MAX(REVIEW_DATE)
INTO MAX_REVIEW_DATE
FROM _PERFORMANCE_REVIEWS
WHERE PERSON_ID = provision_req_rec.USER_ID;

SELECT TO_CHAR(PERFORMANCE_RATING)
INTO EMPREVIEWRATING
FROM _PERFORMANCE_REVIEWS
WHERE PERSON_ID= provision_req_rec.USER_ID
AND REVIEW_DATE = MAX_REVIEW_DATE;

/* Convert the date to string */
SELECT TO_CHAR(MAX_REVIEW_DATE, 'YYYY-MM-DD')
INTO EMPREVIEWDATE
FROM DUAL;

/* Assign the values to SPML attributes */
attab(SIEMPREVIEWDATE) := EMPREVIEWDATE;
attab(SIEMPREVIEWRATING) := EMPREVIEWRATING;
```

- 7** Test the new attributes. When performance review changes occur, the new event alert should fire. Check the Alert Request View window to make sure that the event alert executed. Then, check the contents of `sioraerp.user_requests` table by issuing the following SQL:

```
Select * from sioraerp.user_requests;
```

The user request table should contain the person ID of the employee review performed and 'M' for action type flag.

Run `SPMLProcess.sql` in SQL command window. It is recommended to run the script in the database server command window. Check the SPML message sent back to Select Identity. The SPML message can be found in the `sioraerp.provision_requests spml_message` column.

Secure Communication for Reverse Provisioning

The agent can communicate with the Select Identity Web Service over a secure communication channel (SSL) using the HTTPS protocol. To enable the SSL communication, perform the following steps:

- 1 Obtain a trusted certificate for the web server. Trusted certificates can be obtained from Certificate Authority such as Verisign or can be generated internally using the OpenSSL toolkit or Microsoft Certificate Authority (CA) Server. The following provides an example of how to generate a certificate:
 - a Set the environment for WebLogic by running `setenv.sh` (or `setenv.cmd`) on the WebLogic server domain.
 - b Run `keytool` to generate a keystore on the WebLogic server. Here is an example command:

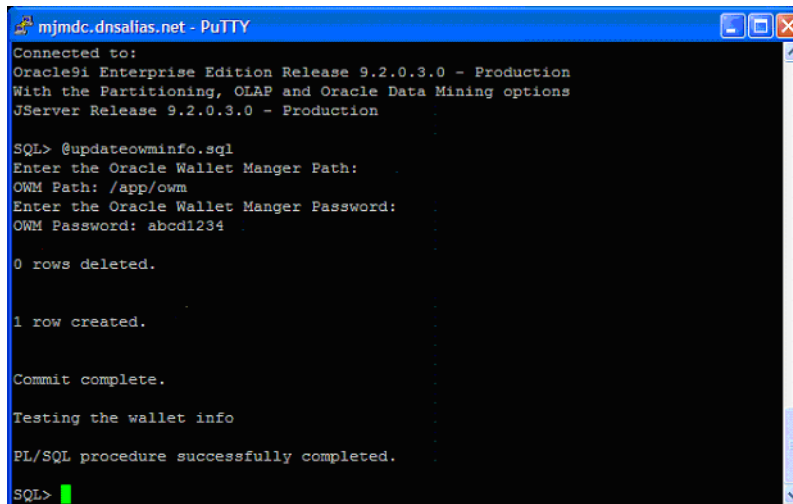
```
keytool -genkey -v -keyalg rsa -keysize 1024
-keypass abcd1234 -keystore helix.jks
-storepass abcd1234 -storetype jks
```
 - c Run `keytool` to generate the certificate request:

```
keytool -certreq -v -keyalg rsa -file sidemo.csr
-keypass abcd1234 -keystore helix.jks
-storepass abcd1234 -storetype jks
```
 - d Generate the certificate using the Microsoft CA server from the request generated.
 - e Export the certificate generated from MS CA server. Export the certificate in p7b format including the entire certificate chain.
 - f Export the root CA certificate in base 64 format.
 - g Export the server certificate.
 - h Copy the certificates to the WebLogic server.
 - i Run `keytool` to import the certificate:

```
keytool -import -v -trustcacerts -alias mykey
-keyalg rsa -file helix.p7b -keypass abcd1234
-keystore helix.jks -storepass abcd1234 -storetype jks
```
 - j Use the keystore to configure WebLogic SSL.

- k** Import the root CA certificate and server certificate in base 64 format to Oracle Wallet.
 - l** Copy the Oracle Wallet file (`ewallet.p12`) to the Oracle Application database server directory (`/app/owm`).
 - m** Configure the URL for secure WebLogic webs ervice in the `SIConfig.sql` file and run `updateowminfo.sql`.
- 2** Configure WebLogic to accept SSL communication with the certificate. A certificate keystore needs to be generated from the trusted certificate for use with WebLogic. Keytools can be used to generate the keystore.
 - 3** Specify the URL in `SIConfig.sql` (such as `https://weblogichost:7002/lmz/webservice`).
 - 4** To enable the Oracle 11i agent to use SSL communication, Oracle Wallet must be created. A wallet is a container that is used to store authentication and signing credentials, including private keys, certificates, and trusted certificates needed by SSL. In an Oracle environment, every entity that communicates over SSL must have a wallet containing an X.509 version 3 certificate, private key, and list of trusted certificates. Oracle Wallet manager is installed as part of Oracle 9i client tools. Launch Oracle Wallet Manager by selecting **Start** → **Programs** → **Oracle 92** → **Intergrated Management Tools** → **Wallet Manager**.
 - 5** Create a new wallet by clicking **New**.
 - 6** Enter the wallet password. This is password opens the wallet and is not associated with other passwords. This password must then be supplied to the agent to enable the agent to open the wallet.
 - 7** When prompted, click **No** to create a certificate request.
 - 8** Select **Operations** → **Import Trusted Certificate**.
 - 9** Import the trusted certificate and the root certificate of the certificate authority. It is important to import the full certificate chain.
 - 10** Save the wallet file in the wallet directory (such as `c:\owm`). The wallet file is called `ewallet.p12`.
 - 11** Create a wallet directory on the Oracle Application database server. The wallet command is issued by the database and the wallet directory must be accessible by the database server, not the application forms server. Only one wallet file can reside per wallet directory. Thus, if there is existing wallet directory, create a new one.
-

- 12 Copy the `ewallet.p12` file to the Oracle Applications database server wallet directory.
- 13 Go to the directory where the `SIConfig.sql` file resides.
- 14 Run `updateowminfo.sql` as the `sioraerp` or `apps` user in SQL Plus, which will prompt for the wallet directory and the wallet password. The script then tests wallet certificate functionality by calling HTTPS using the URL configured in `SIConfig.sql`.



```
mjmdc.dnsalias.net - PuTTY
Connected to:
Oracle9i Enterprise Edition Release 9.2.0.3.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
JServer Release 9.2.0.3.0 - Production

SQL> @updateowminfo.sql
Enter the Oracle Wallet Manger Path:
OWM Path: /app/owm
Enter the Oracle Wallet Manger Password:
OWM Password: abcd1234

0 rows deleted.

1 row created.

Commit complete.

Testing the wallet info

PL/SQL procedure successfully completed.

SQL>
```

If the wallet certificate is valid, the PL/SQL procedure will complete without any error. Address any errors that occur, such as an incorrect wallet path, incorrect password, or invalid certificate chain. (Invalid certificate chain usually means incorrect certificates or incomplete certificate chain.) Both the server certificate and CA root certificate must be imported into the wallet.

- 15 Modify `SIConfig.sql` to specify HTTPS as the protocol for the URL for the Web Service.

Configuring a Secure JDBC Connection Pool

To enable secure communication between Select Identity and the Oracle E-Business Suite, you must configure connection properties for the Oracle JDBC driver. You configure these properties in the WebLogic JDBC connection pool, in the Properties attribute. This attribute is available on the **JDBC Connection Pool** → **Configuration** → **General** tab on the Administration Console. You can set the following properties to enable encryption for the connection pool:

`oracle.net.encryption_client=ACCEPTED`

`oracle.net.encryption_types_client=RC4_256`

`oracle.net.crypto_checksum_client=ACCEPTED`

Here is a snapshot of an example configuration:

si302domain> JDBC Connection Pools> SecureOraERPConnectionPool

Connected to : mjmdc.dnsalias.net:7003 | You are logged in as : weblogic | Logout

Configuration | Target and Deploy | Monitoring | Control | Testing | Notes

General | Connections

This page allows you to define the general configuration of this JDBC connection pool.

Name: SecureOraERPConnectionPool
The name of this JDBC connection pool.

URL: jdbc:oracle:thin:@dbhelix.mjmdc.dn
The URL of the database to connect to. The format of the URL varies by JDBC driver.

Driver Classname: oracle.jdbc.OracleDriver
The full package name of JDBC driver class used to create the physical database connections in the connection pool. (Note that this driver class must be in the classpath of any server to which it is deployed.)

Properties:
user=apps
oracle.net.encryption_client=ACCEPTED
oracle.net.encryption_types_client=RC4_256
oracle.net.crypto_checksum_client=ACCEPTED
The list of properties passed to the JDBC driver that are used to create physical database connections. For example: server=dbserver1. List each property=value pair on a separate line.

Password: [REDACTED]
Confirm Password: [REDACTED]
The database account password used in the physical database connection.

Open String Password: [REDACTED]
Confirm Open String Password: [REDACTED]
The password for the XA open string. (If set, this value overrides a password in an open string in Properties.)

Apply

Configuring the Connector

After you deploy the connector on the application server, you must configure Select Identity to use the connector by deploying it in the Select Identity client. The following provides an overview of the procedures you must complete in order to deploy your connector. It also provides connector-specific information you must provide when configuring Select Identity to use the connector.

- 1 Register the connector with Select Identity by clicking the **Deploy New Connector** button on the Connectors home page. Complete this procedure as described in the “Connectors” chapter of the *HP OpenView Select Identity Administrator Guide*.

After you deploy the connector, the connector properties will look similar to this:

Connector Information	
* Connector Name:	Orat11
* Pool Name:	<input type="text" value="eis/ORAERP"/>
Mapper Available:	<input type="checkbox"/>

- 2 Deploy a resource that uses the newly created connector. On the Resources home page, click the **Deploy New Resource** button. Enter these values:

Field Name	Sample Values	Description
Resource Name	oracle11i	Name given to the resource. If you enabled reverse synchronization, this must be the same as the value provided for the urn:trulogica:conzero:2.0#resourceId attribute on the agent console.
Resource Type	Oracle 11i	The connector that was deployed in Step 1 on page 77 .
Authoritative Source	No	Whether this resource is a system that is considered to be the authoritative source for user data in your environment. Specify No if the connector is not enabled for reverse synchronization. Specify Yes if you want to add users through reverse synchronization. If the resource is not authoritative, the resource can only modify user entitlements during reverse synchronization.
Associate to Group	Selected	Whether the system uses the concept of groups. For the Oracle 11i connector, select this option.
DataSource Name	jdbc/oracle11i	The JDBC data source for the Oracle 11i database, if you are using a data source to connect to the database. If you are using WebSphere 5.1.1 as the application server, the JDBC data source for the Oracle database will not work; therefore, you must use the JDBC URL and driver to connect to the database.

Field Name	Sample Values	Description
URL	jdbc:oracle:thin:@kod: 1522:VIS	The JDBC URL to the Oracle database, if you are using a JDBC driver to connector to the database.
DriverClass Name	oracle.jdbc.Oracle Driver	<p>The class name of the Oracle JDBC driver, if you are using a JDBC driver to connector to the database.</p> <p>If you are using the WebLogic application server, use Oracle's Thin driver, which is listed in the drop down list of WebLogic. Both the data source and JDBC URL and driver connections can be used with this driver.</p> <p>If you are using WebSphere 5.1.1, download the latest Oracle Thin driver (ojdbc14.jar), version 10.1.0.4.0 or later.</p>
UserID	apps	The Oracle schema owner.
Password	apps	The password of the schema owner.
User Table Name	FND_USER	The Oracle database table where foundation user information is stored. Do not change this value.
Group Table Name	FND_ RESPONSIBILITY	The name of the Oracle table where responsibility information is stored. Do not change this value.

Field Name	Sample Values	Description
Link Table Name	FND_USER_RESP_GROUPS	The name of the Oracle table that maps users to responsibilities. Do not change this value.
Mapping File	ORAERP-11-5-9.xml	The name of the mapping file that maps Select Identity fields to Oracle fields. If you are provisioning in an Oracle Application server, specify ORAERP-11-5-9.xml or ORAERP-11-5-10.xml, depending on the version of Oracle. If you are provisioning in an Oracle Human Resources system, specify ORAERPEMP-11-5-9.xml or ORAERPEMP-11-5-10.xml, depending on the version of Oracle.

Complete the steps in this procedure as described in the “Resources” chapter of the *HP OpenView Select Identity Administrator Guide*. After you deploy the resource for the Oracle 11i connector, the Basic Info page of the resource properties will look similar to this:

The screenshot shows a web-based configuration interface for a resource. The title is "Resource Information". The fields are as follows:

- Resource Name:** Ora11iRes1
- Resource Description:** A large empty text area with a scroll bar.
- Resource Type:** Ora11i
- Authoritative Source:** Yes No
- Delete User:** Yes No
- Reconciliation Workflow:** ReconciliationDefaultProcess
- Resource Owner:** alisa

The Additional Info page will look similar to this:

Resource Information	
Resource Name:	Ora11iRes1
 Manage User	
Associate to Group:	<input checked="" type="checkbox"/>

The Access Info page will look similar to this:

Resource Access Information	
* Resource Name:	Ora11iRes1
Data Source Name:	<input type="text" value="jdbc/sint13"/> ?
URL:	<input type="text"/> ?
Driver Class Name:	<input type="text"/> ?
User ID:	<input type="text"/> ?
Password:	<input type="text"/> ?
* Is Employee:	<input type="text" value="No"/> ?
User Table Name:	<input type="text" value="FND_USER"/> ?
Group Table Name:	<input type="text" value="FND_RESPONSIBILITY"/> ?
Link Table Name:	<input type="text" value="FND_USER_RESP_GROUPS"/> ?
* Mapping File:	<input type="text" value="ORAERP11-5-10.xml"/> ? (View)

- 3 Create attributes that link Select Identity to the connector. For each mapping in the connector's mapping file, create an attribute using the Attributes capability on the Select Identity client. Refer to the "Attributes" chapter in the *HP OpenView Select Identity Administrator Guide* for more information.

If you create the attributes for the Oracle 11i connector for users on Oracle Application servers, the View Attributes page for the resource will look similar to this:

Name	Min Length	Max Length	Attribute Mapped To	Authoritative
Accesses	1	15	Accesses	N
CustomerID	1	15	CustomerID	N
Days	1	15	Days	N
Desc	1	240	Description	N
Email	1	240	Email	N
EmployeeID	1	15	EmployeeID	N
EndDate	1	64	EndDate	N
Fax	1	80	Fax	N
ICX_HR_PERSON_ID	1	15	ICX_HR_PERSON_ID	N
Owner	1	64	Owner	N
Password	1	64	Password	N
StartDate	1	64	StartDate	N
SupplierID	1	15	SupplierID	N
TO_PERSON_ID	1	15	TO_PERSON_ID	N
UserName	1	100	UserName	N
sihp3res_ENTITLEMENTS	1	255	sihp3res_ENTITLEMENTS	Y
sihp3res_KEY	1	255	sihp3res_KEY	Y

If you create attributes for provisioning employee data on Oracle Human Resources systems, the View Attributes pages for the resource will look similar to this:

Name	Min Length	Max Length	Attribute Mapped To	Authorative
Accesses	1	15	Accesses	N
Addr1	1	240	Addr1	N
Addr2	1	240	Addr2	N
City	1	30	City	N
CompanyName	1	240	CompanyName	N
Country	1	60	Country	N
CustomerID	1	15	CustomerID	N
DOB	1	64	DOB	N
Days	1	15	Days	N
Desc	1	240	Description	N
Email	1	240	Email	Y
EmployeeID	1	15	EmployeeID	N
EmployeeNumber	1	64	EmployeeNumber	N
EndDate	1	64	EndDate	N
Fax	1	60	Fax	N
FirstName	1	150	FirstName	N
Grade	1	240	Grade	N
HomePhone	1	64	HomePhone	N
ICX_HR_PERSON_ID	1	15	ICX_HR_PERSON_ID	N
Job	1	700	Job	N
LastName	1	150	LastName	N
Location	1	240	Location	N
ManagerFlag	1	10	ManagerFlag	N
OrganizationName	1	240	OrganizationName	N
Owner	1	64	Owner	N
Password	1	64	Password	Y
Position	1	240	Position	N
StartDate	1	64	StartDate	N
State	1	120	State	N
SupplierID	1	15	SupplierID	N
TO_PERSON_ID	1	15	TO_PERSON_ID	N
UserName	1	100	UserName	N
WorkPhone	1	64	WorkPhone	N
Zip	1	30	Zip	N
empsihp3res_ENTITLEMENTS	1	255	empsihp3res_ENTITLEMENTS	Y
empsihp3res_KEY	1	255	empsihp3res_KEY	Y

- 4 Create a Service that will use the newly created resource. To do so, click the **Deploy New Service** button on the Services home page. Complete this procedure as described in “Services” of the *HP OpenView Select Identity Administrator Guide*. You will reference your new resource created in [Step 2](#) while creating this service.

Keep the following in mind when creating the Service:

- Passwords cannot be modified when you modify user attributes. To modify a password, use the password reset function. During view creation for modification, remove the password from the field selection.
- The `EndDate` attribute is provided as a view-only attribute. The values for `EndDate` should not be set during an add or modify operation. Because `EndDate` is used to terminate a user in the Oracle Application, setting the `EndDate` attribute initiates user termination in the Oracle Application resource bypassing the Select Identity termination process. The user status will become out of synchronization between the resource and Select Identity. Use `EndDate` as a display-only attribute for the add and modify view.
- If you deployed the reverse synchronization agent, keep in mind that the Oracle 11i user name is not case sensitive, though the agent returns user name attributes in uppercase letters. The Select Identity configuration can force all user name attributes to uppercase or perform case insensitive comparisons during reverse synchronization.

Understanding the Mapping Files

The Oracle 11i connector is deployed with the following mapping files:

- `ORAERP-11-5-9.xml` or `ORAERP-11-5-10.xml`
- `ORAERPEMP-11-5-9.xml` or `ORAERPEMP-11-5-10.xml`
- `oracle11ierp.xsl`

The XML files map Select Identity user attributes to attributes on the Oracle server, which enables Select Identity to send account additions and modifications to Oracle. The `ORAERP-11-5-x.xml` mapping file enables Select Identity to provision user on an Oracle resource. The `ORAERPEMP-11-5-x.xml` file enables Select Identity to provision employees on an Oracle Human Resources system.

When you deploy a resource through the Select Identity Resources pages, you can review the XML files. You can create attributes that are specific to Select Identity through the Attributes pages on the Select Identity client. These attributes can be used to associate Select Identity user accounts with system resources by mapping them to the connector mapping file described in this chapter.

If you configured the agent to support reverse synchronization, you must also configure the `oracle11ierp.xsl` file, which provides a reverse mapping of the Select Identity and resource fields mapped in the XML file.

The mapping files do not need to be edited unless you want to map additional attributes to your resource. If attributes and values are not defined in this mapping file, they cannot be saved to the resource through Select Identity. The files are created in XML, according to SPML standards, and are bundled in a JAR file called `schema.jar`.

This chapter provides an explanation of the XML and XSL mapping files. The following sections are provided:

- [ORAERP-11-5-x.xml Mapping Information on page 86](#)
- [ORAERPMP-11-5-x.xml Mapping Information on page 89](#)
- [Elements in the XML Mapping Files on page 92](#)
- [Elements in the XSL Reverse Mapping File on page 96](#)

ORAERP-11-5-x.xml Mapping Information

The following are the attribute mappings supported for Oracle 11i. These are listed in the `ORAERP-11-5-x.xml` mapping file. You can add, modify, and delete attributes once you are familiar with the contents of this file.

The Select Identity resource attributes are editable. They reflect the identity information as seen in Select Identity. The physical resource attributes are literal attributes of user accounts on Oracle. These attributes cannot be changed. The user name attribute is not case sensitive in Oracle, though it is necessary to make the user name attribute all uppercase to be compatible with Oracle and Select Identity.

Select Identity Attribute				Oracle Resource Attribute
Name	Type	Min Length	Max Length	Name
UserName*	String	1	100	UserName
Description	String	1	240	Desc
customerID	String	1	15	CustomerId
supplierID	String	1	15	SupplierID

Select Identity Attribute				Oracle Resource Attribute
EmployeeID	String	1	15	EmployeeID
Email*	String	1	240	Email
Owner	String	1	64	Owner
Fax	String	1	80	Fax
PasswordLifespan Days	String	1	15	Days
Password Accesses	String	1	15	Accesses
StartDate	Date	1	64	StartDate
EndDate	Date	1	64	EndDate
Password*	String	1	64	Password
ICX_HR_PERSON_ID**	String	1	15	ICX_HR_PERSON_ID **
TO_PERSON_ID**	String	1	15	TO_PERSON_ID**

* Select Identity default attribute

** Oracle Application securing attribute

In the table above, some Oracle 11i attributes are securing attributes. These attributes are used by Oracle Self-Service Web Applications to allow visibility to specific users or responsibilities for rows (records) of data based on the specific data (attribute values) contained in the row. Because securing attributes are dynamic and differ from site to site, the specific securing attributes used for each site need to be contained in the XML mapping file.

Each Select Identity attribute name that maps to an Oracle Application securing attribute must match the Oracle Applications's ATTRIBUTE_CODE for reverse synchronization purposes (such as ICX_HR_PERSON_ID) . This

ATTRIBUTE_CODE is also placed in the **Column** field of the attribute in **ORAERP-11-5-x.xml** file. Here is an example:

```
<attributeDefinitionReference name="ATTR_ICX_HR_PERSON_ID"
required="false" concero:tafield="ICX_HR_PERSON_ID"
concero:resfield="[p_attribute_code] [ICX_HR_PERSON_ID]
[Oracle Self-Service Web Applications] [NUMBER] "
concero:init="true" />
```

```
<attributeDefinitionReference name="ATTR_TO_PERSON_ID"
required="false" concero:tafield="TO_PERSON_ID"
concero:resfield="[p_attribute_code] [TO_PERSON_ID]
[Oracle Self-Service Web Applications] [NUMBER] "
concero:init="true" />
```

The resfield attributes consists of four values:

[parameter name][COLUMN_NAME][Application Name][VALUE_TYPE]

where:

- **parameter name** — The name of the parameter used by Oracle's PL/SQL package. Standard attributes that map to Oracle Application's Foundation user attributes must specify the parameter name used by FND_USER packages. For securing attributes, parameter names will always be p_attribute_code.
- **COLUMN_NAME** — The name of the column that each attribute maps to Oracle's table. Standard attributes that map to Oracle Application's Foundation user attributes must specify the column name in FND_USER table. For securing attributes, specify ATTRIBUTE_CODE for each attribute.
- **Application Name** — Leave blank for standard attributes. For securing attributes, specify the Oracle Application Name for each securing attribute. The Application Name may differ for different releases of Oracle Application. The Application Name can be identified in the Oracle Application user definition window's securing attribute section. If the wrong Application Name is specified, actions against securing attributes will fail. For example, on Oracle App 11.5.9, the application name is Oracle Self-Service Web Applications. On Oracle App 11.5.10, the application name is Self-Service Web Applications.
- **VALUE_TYPE** — Specify one of the following: VARCHAR, DATE, or NUMBER.

ORAERPMP-11-5-x.xml Mapping Information

The following attributes are used for provisioning employees in the Oracle Human Resources system. These are listed in the `ORAERPMP-11-5-x.xml` mapping file. You can add, modify, or delete attributes once you are familiar with the contents of this file.

The Select Identity resource attributes are editable. They reflect the identity information as seen in Select Identity. The physical resource attributes are literal attributes of user accounts in the Oracle Human Resources system. These attributes cannot be changed. The user name attribute is not case sensitive in Oracle, though it is necessary to make the user name attribute all uppercase to be compatible with Oracle and Select Identity.

Select Identity Attribute					Oracle Resource Attribute
Name	Description	Type	Min Length	Max Length	Name
UserName*	User name derived from first three characters of last name plus the person ID	String	1	100	UserName
Description	Full name	String	1	240	Desc
customerID		String	1	15	CustomerId
supplierID		String	1	15	SupplierID
<i>EmployeeID</i>	Person ID	String	1	15	<i>EmployeeID</i>
Email*		String	1	240	Email
Owner		String	1	64	Owner
Fax		String	1	80	Fax
PasswordLifespan Days		String	1	15	Days

Select Identity Attribute					Oracle Resource Attribute
<i>Password Accesses</i>		String	1	15	<i>Accesses</i>
<i>StartDate</i>		Date	1	64	<i>StartDate</i>
<i>EndDate</i>	Termination date	Date	1	64	<i>EndDate</i>
<i>Password*</i>	Default set in XSL file	String	1	64	<i>Password</i>
<i>ICX_HR_PERSON_ID**</i>	Person ID	String	1	15	<i>ICX_HR_PERSON_ID**</i>
<i>TO_PERSON_ID**</i>	Person ID	String	1	15	<i>TO_PERSON_ID**</i>
<i>CompanyName</i>	GRE (business group name)	String	1	240	<i>Company Name</i>
<i>OrganizationName</i>	Organization (department) name	String	1	240	<i>Organization Name</i>
<i>FirstName</i>		String	1	150	<i>FirstName</i>
<i>LastName</i>		String	1	150	<i>LastName</i>
<i>EmployeeNumber</i>	Employee number for organization	String	1	64	<i>Employee Number</i>
<i>DOB</i>	Date of birth in this format: YYYY-MM-DD	Date	1	64	<i>DOB</i>
<i>Addr1</i>	Home address, line 1	String	1	240	<i>Addr1</i>

Select Identity Attribute					Oracle Resource Attribute
<i>Addr2</i>	Home address, line 2	String	1	240	<i>Addr2</i>
<i>City</i>	Home address city	String	1	30	<i>City</i>
<i>State</i>	Home address state	String	1	120	<i>State</i>
<i>Zip</i>	Home address postal code	String	1	30	<i>Zip</i>
<i>Country</i>	Home address country	String	1	60	<i>Country</i>
<i>Job</i>	Job name	String	1	700	<i>Job</i>
<i>Location</i>	Job location	String	1	240	<i>Location</i>
<i>Grade</i>	Pay grade	String	1	240	<i>Grade</i>
<i>Position</i>	Job Position	String	1	240	<i>Position</i>
<i>ManagerFlag</i>	Y for manager, N for employee	String	1	10	<i>ManagerFlag</i>
<i>WorkPhone**</i>		String	1	64	<i>WorkPhone</i>
<i>HomePhone</i>		String	1	64	<i>HomePhone</i>

* Select Identity default attribute

** Oracle Application securing attribute (see [page 87](#) for more information)

Italicized attributes are monitored for changes in the Oracle Human Resources system by the Oracle Alert system. Changes to the monitored employee attributes are communicated to Select Identity by the agent. Synchronizing additional attributes can be accomplished by adding the attribute information in the `ORAERPMP-11-5-x.xml` mapping file and the `oracle11ierp.xsl` reverse mapping file after creating the Oracle Alert to monitor the desired employee attributes.

Elements in the XML Mapping Files

Here is an explanation of the format of the XML mapping files.

- **<Schema>**, **<providerID>**, and **<schemaID>**

Provides standard elements for header information.

- **<objectClassDefinition>**

Defines the actions that can be performed on the specified object as defined by that name attribute (in the **<properties>** element block) and the Select Identity-to-resource field mappings for the object (in the **<memberAttributes>** block). For example, the object class definition for users defines that users can be created, read, updated, deleted, reset, and expired in Oracle 11i.

- **<properties>**

Defines the operations that are supported on the object. This can be used to control the operations that are performed through Select Identity. The following operations can be controlled:

- Create (CREATE)
- Read (READ)
- Update (UPDATE)
- Delete (DELETE)
- Enable (ENABLE)
- Disable (DISABLE)
- Reset password (RESET_PASSWORD)
- Expire password (EXPIRE_PASSWORD)
- Change password (CHANGE_PASSWORD)
- Assign entitlements (LINK)
- Unassign entitlements (UNLINK)
- Retrieve entitlements (GETALL)

The operation is assigned as the name of the <attr> element and access to the operation is assigned to a corresponding <value> element. You can set the values as follows:

- true — the operation is supported by the connector
- false — the operation is not supported by the connector and will throw a permission exception
- bypass — the operation is not supported by the connector but will not throw an exception; the operation is simply bypassed

Here is an example:

```
<objectClassDefinition name="User" description="Oracle ERP
User">
  <properties>
    <attr name="GETCHILDREN">
      <value>true</value>
    </attr>
    <attr name="DELETE">
      <value>true</value>
    </attr>
    <attr name="EXPIREPASSWORD">
      <value>>false</value>
    </attr>
    <attr name="GETALL">
      <value>true</value>
    </attr>
    ...
  </properties>
</objectClassDefinition>
```

- **<memberAttributes>**

Defines the attribute mappings. This element contains <attributeDefinitionReference> elements that describe the mapping for each attribute. Each <attributeDefinitionReference> can be followed by an <attributeDefinition> element that specifies details such as minimum length, maximum length, and so on.

Each <attributeDefinitionReference> element contains the following attributes:

- Name — the name of the reference.
- Required— if this attribute is required in the provisioning (set to true or false).

- **Concero:tafield** — the name of the Select Identity resource attribute. In general, the attribute assigned to **tafield** should be the same as the physical resource attribute, or at least the connector attribute. For example, it is recommended to have the following:

```
<attributeDefinitionReference name="FirstName"
required="false" concero:tafield="[givenname]"
concero:resfield="givenname" concero:init="true"
concero:isMulti="true"/>
```

instead of this:

```
<attributeDefinitionReference name="FirstName"
required="false" concero:tafield="[FirstName]"
concero:resfield="givenname" concero:init="true"
concero:isMulti="true"/>
```

- **Concero:resfield** — the name of the physical resource attribute from the resource schema. If the resource does not support an explicit schema (such as UNIX), this can be a tag field that indicates a resource attribute mapping.

Also, the attribute name may be case-sensitive; for example, if the attribute is defined in all uppercase letters on the resource, be sure to specify it in all uppercase letters here.

- **Concero:isKey** — An optional attribute that, when set to true, specifies that this is the key field to identify the object on the resource. Only one `<attributeDefinitionReference>` can be specified where `isKey="true"`. This key field does not need to be the same as the key field of the identity object in Select Identity.

Note that for a key field mapping where `isKey="true"` and `tafield` is not assigned the `UserName` attribute, `UserName` should not be used in any other mapping. That is, `UserName` can be assigned to `tafield` only in cases where it is mapped to the key field in the resource. Example:

```
<attributeDefinitionReference name="UserName"
required="true" concero:tafield="[UserName]"
concero:resfield="uid" concero:isKey="true"
concero:init="true"/>
```

- **Concero:init** — An optional attribute that identifies that the attribute is initialized with the value of the attribute passed in from Select Identity.

Here is an example:

```
<memberAttributes>
  <attributeDefinitionReference name="ATTR_UserName"
    required="true" concero:tafield="UserName"
    concero:resfield="[x_user_name] [USER_NAME] [] [VARCHAR]"
    concero:isKey="true" concero:init="true"/>
  ...
```

The interpretation of the mapping between the connector field (as specified by the `Concero:tafield` attribute) and the resource field (as specified by the `Concero:resfield` attribute) is determined by the connector. The Oracle 11i connector has code to interpret the mappings in one way, as follows:

- The connector attribute names are specified in `tafield`. The value of attribute `xyz` is taken from the `UserModel` during provisioning.
- Composite attributes can be specified in the Oracle 11i connector mapping file. To do this, specify `attr1 {xxxx} attr2` as the connector attribute. This specifies that the value of the `attr1` and `attr2` attributes should be combined with the string `xxxx` to form a mapping for the specified resource field. The Oracle 11i connector has code to handle these composite mappings.

You must specify static text (strings) in composite attributes within brackets (`{ }`). Also, if no string separates two connector attributes, you must add a space that is within brackets, like this: `attr1{ }attr2`.

- **<attributeDefinition>**

Defines the properties of each object's attribute. For example, the attribute definition for the Directory attribute defines that it must be between one and 50 characters in length and can contain the following letters, numbers, and characters: a-z, A-Z, 0-9, @, +, and a space.

Here is an example:

```
<attributeDefinition name="ATTR_ResponsibilityKey"
  description="Responsibility Key" type="xsd:string" >
  <properties>
    <attr name="minLength">
      <value>1</value>
    </attr>
    <attr name="maxLength">
      <value>128</value>
```

```

</attr>
<attr name="pattern">
  <value><![CDATA[[a-zA-Z0-9@]+]]> </value>
</attr>

</properties>
</attributeDefinition>

```

- **<concro:entitlementMappingDefinition>**

Defines how entitlements are mapped to users.

- **<concro:objectStatus>**

Defines how to assign status to a user.

- **<concro:relationshipDefinition>**

Defines how to create relationships between users.

Elements in the XSL Reverse Mapping File

If the agent is installed on the resource and you wish to enable reverse synchronization, you must configure the `oracle11ierp.xsl` file to map all attributes that are specified in the XML mapping file. The following XSL code examples explain how to configure the XSL file for any mapping file.

For each resource attribute, you must define a corresponding Select Identity attribute, which defines the attribute in Select Identity to which the resource attribute is mapped. The XSL code translates all of the resource attribute names to lowercase for easier comparison. Then, the resource attribute (`employeeid` in this example) is mapped to the Select Identity attribute `EmployeeID`.

```

<xsl:when test="$ATTRNAME = 'employeeid'">
  <xsl:call-template name="AttributeBuilder">
    <xsl:with-param name="DSMLELEMENT" select="$DSMLELEMENT"/>
    <xsl:with-param name="ATTRNAME" select="'EmployeeID'"/>
    <xsl:with-param name="ATTRVALUE" select="$ATTRVALUE"/>
    <xsl:with-param name="MODIFYFLAG" select="$MODIFYFLAG"/>
  </xsl:call-template>
</xsl:when>

```


Uninstalling the Connector

If you need to uninstall a connector from Select Identity, make sure that the following are performed:

- All resource dependencies are removed.
- The connector is deleted through the Connectors home page on the Select Identity client.

Uninstalling the Connector from WebLogic

Perform the following to delete a connector:

- 1 Log on to the WebLogic Server Console.
- 2 Navigate to *My_Domain* → **Deployments** → **Connector Modules**.
- 3 Click the delete icon next to the connector that you want to uninstall.
- 4 Click **Yes** to confirm the deletion.
- 5 Click **Continue**.

Uninstalling the Agent

To uninstall the Oracle 11i agent, perform following steps:

- 1 Deactivate all alerts defined during the agent installation. If alerts were fired, the alerts cannot be deleted.
- 2 Deactivate the Periodic Alerts for SPML processing if defined.
- 3 Deactivate the Concurrent Request processing for SPML processing if defined.
- 4 Log on to SQLPLUS as system account and issue following statements:

```
drop user sioraerp cascade;  
drop package apps.provision_service;
```



Troubleshooting

To troubleshoot agent issues, you can examine the `user_requests` and `provision_requests` tables. Also, the `SPMLProcess.sql` file can be run manually to produce debugging output.

- `User_requests` table

The `siroraerp.user_requests` table contains the output from each event alert. In the Alert History window in Oracle application, the result of the alert action can be examined. If the alert completed without any error, the contents of the `user_requests` table can be examined to ensure that proper records are inserted into the table. During this process, the periodic alert to run `SPMLProcess.sql` may need to be stopped because the script will remove all the entries from the `user_requests` table after updating the `provision_requests` table.

- `Provision_requests` table

The `provision_requests` table contains three columns:

- `spml_message` — contains the outgoing spml message.
- `response` — contains response from Select Identity.
- `error_msg` — contains any error message if the SPML message results in an error.

Also, the req_status field is set to -1 for error. The retry column contains the number of retries for the specific SPML message. To reprocess the user action, the req_status field can be set to 0.

The user_requests and termination_request tables are temporary and do not store historical information. The provision_requests table, however, does retain all SPML activities. The contents of provision_requests table can be used as an audit trail or for reporting purposes. The provision_requests table can also be purged. To purge the table, run following sql statement as the sioraerp user.

```
DELETE FROM PROVISION_REQUEST
WHERE REQ_STATUS IN ( -1, 2 );
COMMIT;
```

Finally, one of the easiest ways to debug the agent is to run the SPMLProcess.sql script manually in SQL Plus. In the Select Identity installation directory on the Oracle server, run SPMLProcess.sql script as the apps user. The script takes SICongfig.sql as an input parameter.