**OIM Metadata for flat file Connector**

1. Introduction
2. Contents
   1. Creating common Metadata for provisioning and reconciliation
      1. IT resource type definition
      2. Resource object
      3. Configuration lookup
      4. Object configuration lookup
      5. Process form
         1. Child form
         2. Parent form
      6. Adapters – Parent form
         1. Create adapter
         2. Delete adapter
         3. Enable adapter
         4. Disable adapter
         5. Update adapter
      7. Process definition and process tasks – Parent form
         1. Flat File process definition
         2. Create process task
         3. Delete process task
         4. Enable process task
         5. Disable process task
         6. Update Process task
   2. Creating Provisioning meta data
      1. Provisioning attribute map lookup
      2. IT resource of type connector server
      3. Flat file IT resource
      4. Application Instance
   3. Setup lookup recon for provisioning
      1. Create a new xml with name FFLookupRecon.xml and update MDS
      2. Create a new lookup
      3. Create Child form
      4. Update Parent form created in 2.1.5.2
   4. Adapters – Child form
      1. Create adapter for child table.
   5. Process tasks – Child table
      1. Create Process Task for child table.
      2. Update Process Task for child table
      3. Delete Process Task for child table
   6. Scheduled job for lookup recon
      1. Create a scheduled job
      2. Update Provisioning attribute map lookup to include child form field label
   7. Setup remote connector server
   8. Test provisioning
      1. Create user (with child table data)
      2. Update user details
         1. Primary for data
         2. Child table data
      3. Disable user
      4. Enable user
      5. Revoke user
   9. Creating Reconciliation meta data
      1. Full/Incremental reconciliation
         1. Reconciliation attribute map
         2. Reconciliation profile
         3. Reconciliation mapping
         4. Reconciliation rule
         5. Create a task xml and update MDS
         6. Create a scheduled job
      2. Lookup recon
      3. Role recon
      4. Org recon
   10. Test reconciliation
       1. Full/Incremental reconciliation
   11. Run scheduled job as created in section 2.4.1.6
   12. Extending the functionality of connector
       1. Provisioning validation and reconciliation transformation
          1. Testing the provisioning validation
          2. Testing the reconciliation transformation
       2. CLASSPATH isolation
   13. Packaging the flat file connector
3. **Introduction**

Lab stories

As part of Phase 1 development of Connector developers at ACME Capital have created Connector Bundle and tested all ICF Contracts like Create User, Modify User, Enable User, Disable user, Delete User etc independently.

As part of Phase 2 objectives to achieve an end to end solution objective they need to invoke user life cycle operations on this connector bundle from OIM. For that they need to develop OIM Metadata which will capture user lifecycle inputs generated in OIM and invoke appropriate operations on Connector Bundle.

In this lab, we would be creating necessary OIM artifacts for the flat file connector developed in lab1.

**Remark: In order to speed up the system, you can stop the WebLogic Admin and SOA servers**

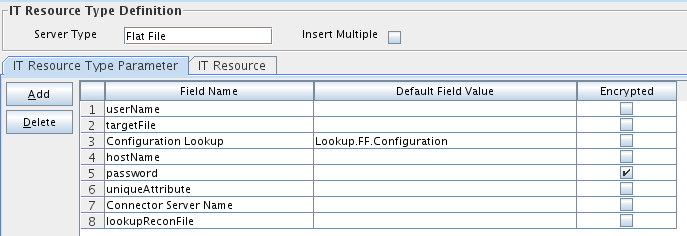
1. **Contents**
   1. Creating common Metadata for provisioning and reconciliation
      1. IT Resource type definition

**Purpose**

This step includes creating an **IT resource type definition** of type ‘Flat File’.

**Steps**

* Login to OIM design console
* Click on **IT Resource type definition** under **Resource management**.
* Enter ‘Flat File’ as server type and click **Save**
* Under **IT Resource Type Parameter**, click **Add**
* Add the following:
  + **Field Name = Configuration Lookup** and **Default Field value = Lookup.FF.Configuration**. Main configuration lookup (see section 2.1.3) containing the ICF bundle information.
  + **Field Name = hostName**. Target host name.
  + **Field Name = userName**. User name used to connect to target system
  + **Field Name = password,** Select **Encrypted**. Password for the above user
  + **Field Name = targetFile**, Flat file on target system
  + **Field Name = uniqueAttribute,** Unique attribute
  + **Field Name = Connector Server Name,** Connector server
  + **Field Name =** **lookupReconFile,** Lookup recon file
* Click **Save** (click Yes to confirm the encryption of the password)
* **IT Resource Type Definition** should look like below.



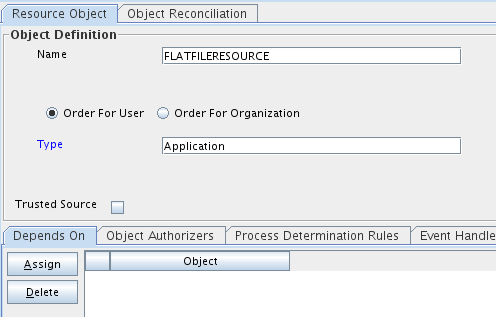
2.1.2 Resource Object

**Purpose**

This step includes creating a resource object called ‘FLATFILERESOURCE’

**Steps**

* Click **Resource Objects** under **Resource Management**
* Enter ‘FLATFILERESOURCE’ as **Name**
* Enter **Application** for **Type** (double-click the field to select in the lookup window)



* Click **Save**

Note: In this lab, we would not be considering the target as trusted source.

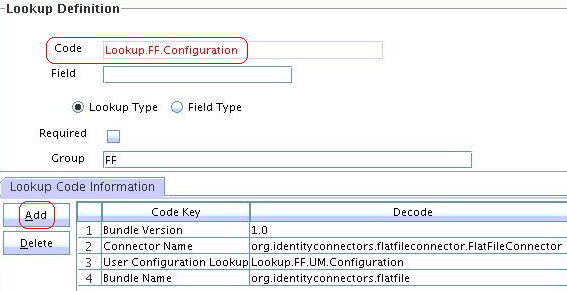
2.1.3 Configuration Lookup

**Purpose**

This step includes creating a configuration lookup. This lookup contains information regarding the connector bundle like name, version, connector name etc… This lookup is used by the ICF to load the connector at run time.

**Steps**

* Click on **Lookup Definition** under **Administration**
* Enter **Lookup.FF.Configuration** for **Code**
* Enter **FF** for **Group**
* Click **Save**
* Under **Lookup Code Information** enter the following:
  + **Code Key = Bundle Name, Decode =** **org.identityconnectors.flatfile.** Bundle name which contains the connector SPI implementation.
  + **Code Key = Connector Name, Decode = org.identityconnectors.flatfileconnector.FlatFileConnector.** Fully qualified name of the connector implementation
  + **Code Key = User Configuration Lookup, Decode = Lookup.FF.UM.Configuration.** User configuration lookup containing the information about the provisioning attribute map and reconciliation attribute map (defined later).
  + **Code Key = Bundle Version, Decode = 1.0.** Bundle version



* Click **Save**

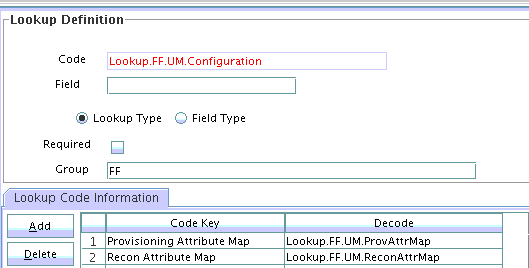
2.1.4 Object Configuration Lookup

**Purpose**

This step includes creating object configuration lookup. This lookup contains information regarding provisioning attribute map, recon attribute map etc… This information will be used by the OIM ICF glue (icf-oim-intg.jar) during provisioning and reconciliation operations.

**Steps**

* Click on **Lookup Definition** under **Administration**
* Enter **Lookup.FF.UM.Configuration** for **Code**
* Enter **FF** for **Group**
* Click **Save**
* Under **Lookup code information** enter the following:-
  + **Code Key = Provisioning Attribute Map, Decode = Lookup.FF.UM.ProvAttrMap.** Contains mapping of OIM fields with the target fields and is used during provisioning operation.
  + **Code Key = Recon Attribute Map, Decode = Lookup.FF.UM.ReconAttrMap.** Contains mapping of OIM fields with the target fields and is used during reconciliation operation.

****

* Click on **Save**

2.1.5 Process Form

2.1.5.1 Child Form

Child form uses lookup reconciliation and hence the steps to create child form are included in the **section 2.3.3**

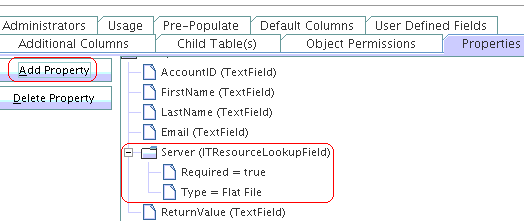
2.1.5.2 Parent Form

**Purpose**

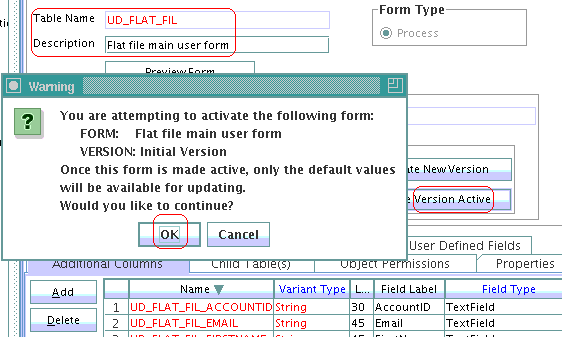
This step includes creating the parent process form.

**Steps**

* Open **Form Designer** under **Development Tools**
* Enter **UD\_FLAT\_FIL** (you don’t have to enter UD) for **Table Name**
* Enter **Flat file main user form** for **Description**
* Click **Save**
* Under **Additional Columns** click **Add** to add columns as stated below:-
  + **Name = UD\_FLAT\_FIL\_ACCOUNTID, Variant Type = String, Length = 30, Field Label = AccountID, Field Type = TextField** and **Order = 1.**
  + **Name = UD\_FLAT\_FIL\_Email, Variant Type = String, Length = 45, Field Label = Email, Field Type = TextField** and **Order = 4**
  + **Name = UD\_FLAT\_FIL\_FIRSTNAME, Variant Type = String, Length = 45, Field Label = FirstName, Field Type = TextField** and **Order = 2**
  + **Name = UD\_FLAT\_FIL\_LASTNAME, Variant Type = String, Length = 45, Field Label = LastName, Field Type = TextField** and **Order = 3**
  + **Name = UD\_FLAT\_FIL\_RETURN, Variant Type = String, Length = 45, Field Label = ReturnValue, Field Type = TextField** and **Order = 6**
  + **Name = UD\_FLAT\_FIL\_SERVER, Variant Type = long, Field Label = Server, Field Type = ITResourceLookupField** and **Order = 5**
* Click **Save**
* Click **Properties** tab
  + Add below Properties for **Server (ITResourceLookupField)**
    - **Required = true**
    - **Type = Flat File**



* Click **Save**
* Click **Make Version Active** and click **Ok** for the prompt





2.1.6 Adapters – Parent form

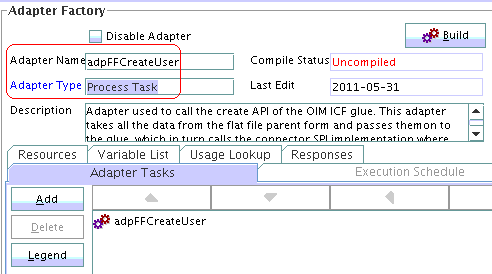
2.1.6.1 Create adapter

**Purpose**

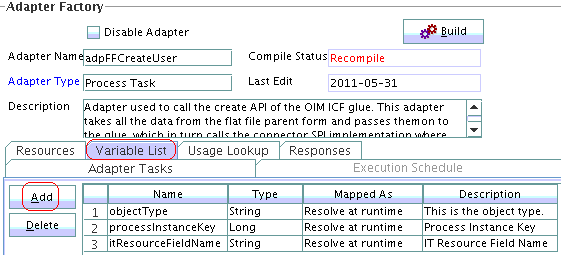
This step includes creating adapter for create provisioning operation. This adapter is responsible to take all the data from parent form and call the OIM ICF glue (icf-oim-intg.jar).

**Steps**

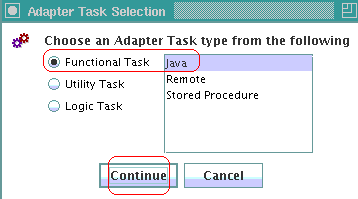
* Under **Development tools**, click **Adapter Factory**
* Enter **adpFFCreateUser** for **Adapter Name**
* Enter **Process Task** for **Adapter Type**
* Enter ‘Adapter used to call the create API of the OIM ICF glue. This adapter takes all the data from the flat file parent form and passes them on to the glue, which in turn calls the connector SPI implementation where the data would be inserted into target’ for **Description**
* Click **Save**

****

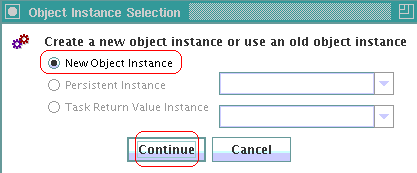
* Click **Variable List** tab
* Click **Add** and add the following variables
  + **Name – objectType, Type – String, Description – This is the object type, Map To – Resolve at runtime**
  + **Name – processInstanceKey, Type – Long, Description – Process Instance Key, Map To – Resolve at runtime**
  + **Name – itResourceFieldName, Type – String, Description – IT Resource Field Name, Map to – Resolve at runtime**

****

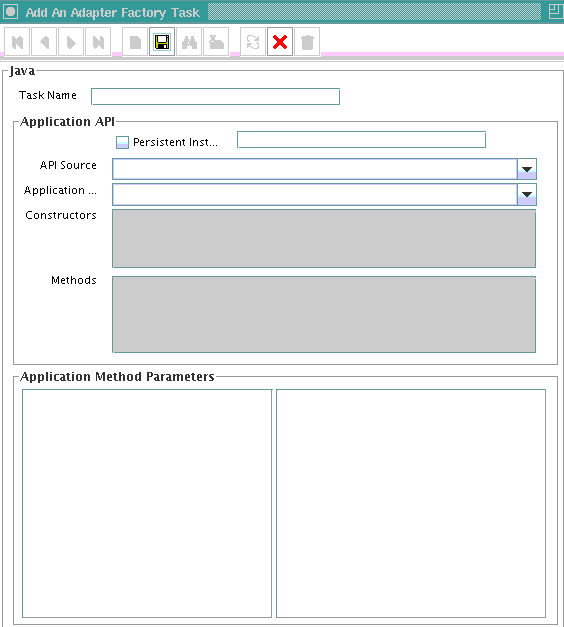
* Click **Adapter Tasks** tab
* Select **adpFFCreateUser** and click **Add** to see below pop up



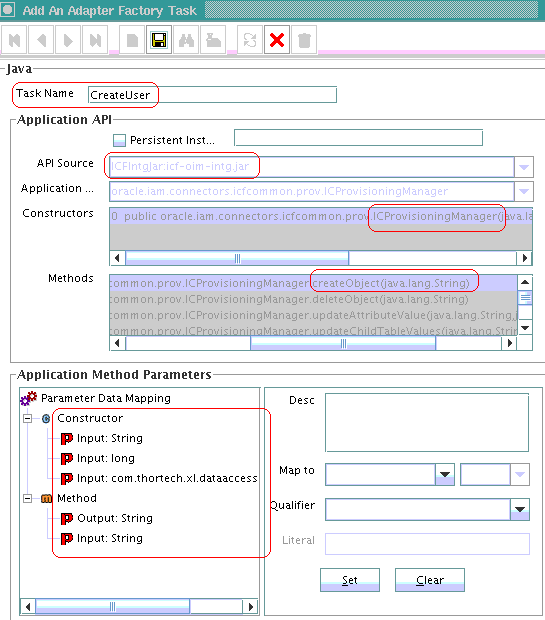
* Click **Continue** to see next screen as shown below

****

* Click **Continue** to see next screen as shown below



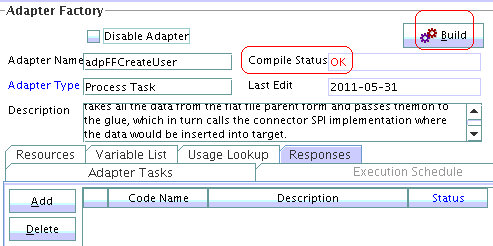
* Enter **CreateUser** for **TaskName**
* Select **ICFIntgJar:icf-oim-intg.jar** from the dropdown for **API Source**
* Select **oracle.iam.connectors.icfcommon.prov.ICProvisioningManager** from the dropdown for **Application…**
* Select **createObject** for Methods
* Click **Save** to see the below screen



* Update **Application Method Parameters** in the same order as below:
* Constructor
  + **Input: String** to **Adapter Variables**, **itResourceFieldName**
  + **Input: long** to **Adapter Variables**, **processInstanceKey**
  + **Input: com.thortech.xl.dataacess.tcDataProvider** to **Adapter References, Database reference**
* Method
  + **Output: String** to **Adapter Variables, Return Variable**
  + **Input: String** to **Adapter Variables, objectType**

****

* ClickOn **Save** and close the CreateUser Adapter Factory Task.
* Click **Save** again on the Adapter Factory screen
* Click **Build** to see the C**ompile Status Ok**

****

2.1.6.2 Delete adapter

**Purpose**

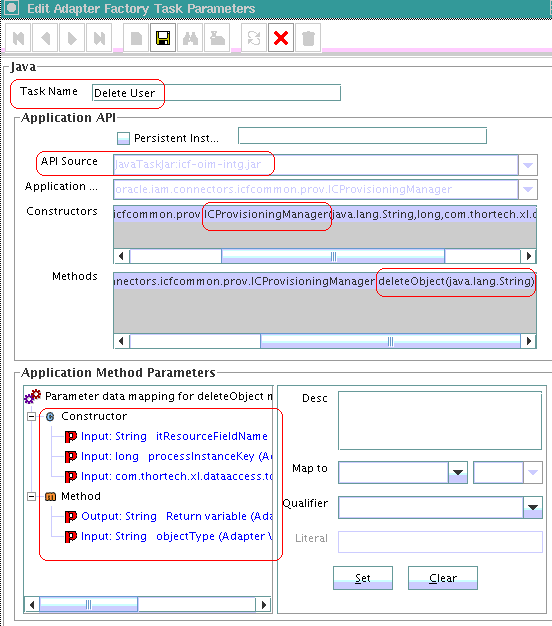
This step includes creating an adapter for delete operation. This adapter is responsible to take the UID of the user being delete and pass it on to ICF OIM glue.

**Steps**

* Under **Development tools,** click **Adapter Factory**
* Enter **adpFFDeleteUser** for **Adapter Name**
* Enter **Process Task** for **Adapter Type**
* Enter ‘Adapter used to call the delete API of the OIM ICF glue. which in turn calls the connector SPI implementation where the data would be deleted from the target’ for **Description**
* Click **Save**

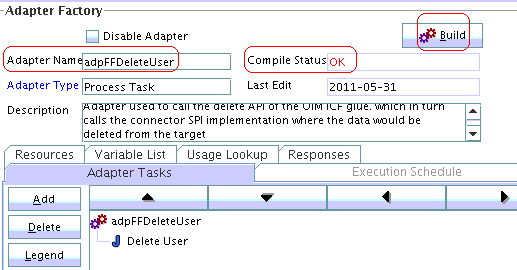
The steps to create the **Adapter Variables** are same as shown in section 2.1.6.1. Please follow the same.

The **Adapter Task - named Delete User -** should call the **deleteObject** API of the **ICProvisioningManager.** The rest of steps is the same. Please follow them as shown in section 2.1.6.1



**Save** the task and the adapter and **Build** the adapter**.**

The final screen shot of delete adapter should look like below:



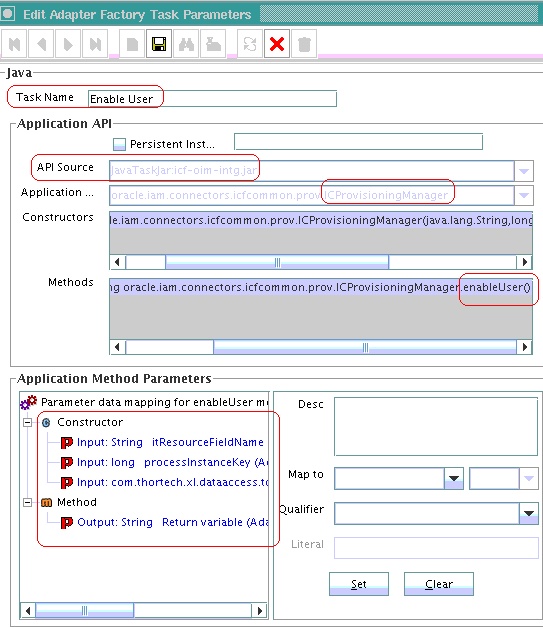
2.1.6.3 Enable adapter

**Purpose**

The steps include creating an adapter for enable operation. This adapter calls the **enableUser** API of the ICF OIM glue.

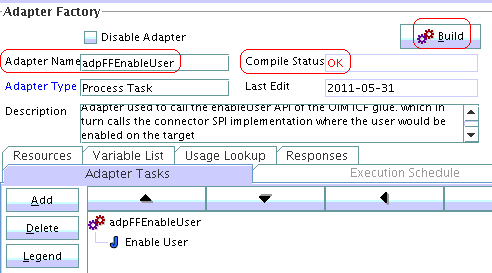
**Steps**

* Under **Development tools,** click **Adapter Factory**
* Enter **adpFFEnableUser** for **Adapter Name**
* Enter **Process Task** for **Adapter Type**
* Enter ‘Adapter used to call the enableUser API of the OIM ICF glue. which in turn calls the connector SPI implementation where the user would be enabled on the target’ for **Description**
* Click **Save**
* Create the following **Adapter Variables**
  + **Name – processInstanceKey, Type – Long, Description - Process Instance Key, Map To – Resolve at runtime**
  + **Name – itResourceFieldName, Type – String, Description - IT Resource Field Name, Map to – Resolve at runtime**
* Create the **Adapter Task - named Enable User -** calling the **enableUser** API of the **ICProvisioningManager** Application.



* **Save** the task and the adapter and **Build** the adapter.

Final screen shot should look like below:



2.1.6.4 Disable Adapter

**Purpose**

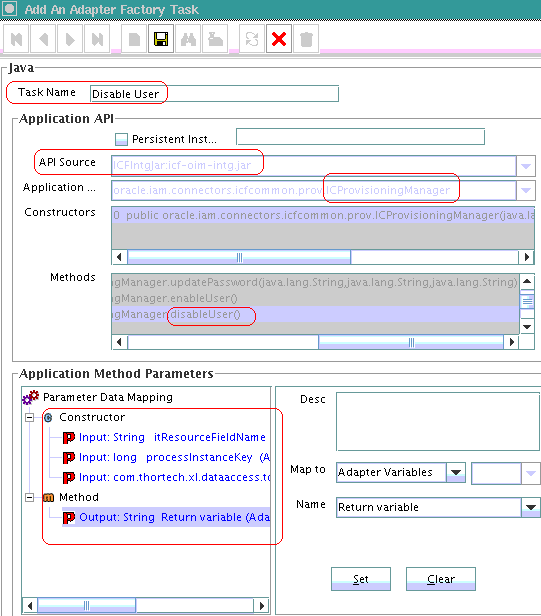
This section includes creating an adapter for disable operation. This adapter calls the disableUser API of the OIM ICF glue.

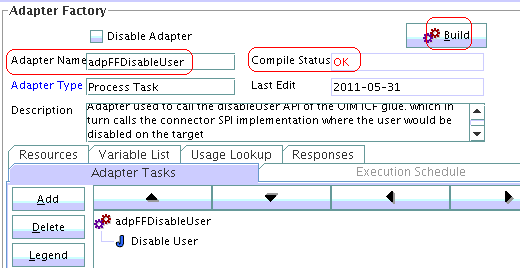
**Steps**

Steps are same as shown in section 2.1.6.3 except the adapter task calls the **disableUser** API of **ICProvisioningManager**

His name has to be **adpFFDisableUser**

Final screen shots should like below:





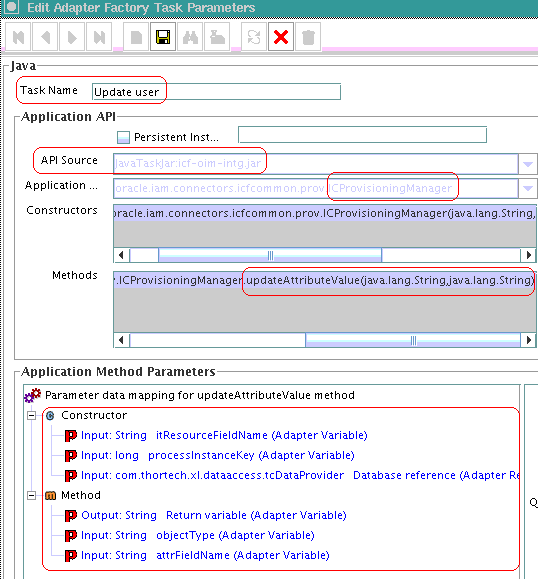
2.1.6.5 Update adapter

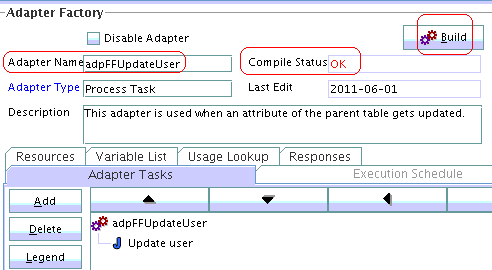
**Purpose**

This section includes steps to create an adapter to be used when an update happens to parent form field value.

**Steps**

* Under **Development tools,** click **Adapter Factory**
* Enter **adpFFUpdateUser** for **Adapter Name**
* Enter **Process Task** for **Adapter Type**
* Enter ‘This adapter is used when an attribute of the parent table gets updated.’ for **Description**
* Click **Save**
* Create the following **Adapter Variables**
* **Name – processInstanceKey, Type – Long, Description – Process Instance Key, Map To – Resolve at runtime**
* **Name – itResourceFieldName, Type – String, Description – IT Resource Field Name, Map to – Resolve at runtime**
* **Name – objectType, Type – String, Description – Object Type, Map to – Resolve at runtime**
* **Name –** **attrFieldName, Type – String, Description – Attribute Field Name, Map to – Resolve at runtime**
* Create the **Adapter Task – named Update User -** calling the **updateAttributeValue** (with 2 strings parameters) API of the **ICProvisioningManager** Application.
* **Save** the task and the adapter and **Build** the adapter.

****

****

2.1.7 Process Definition and Process tasks – Parent form

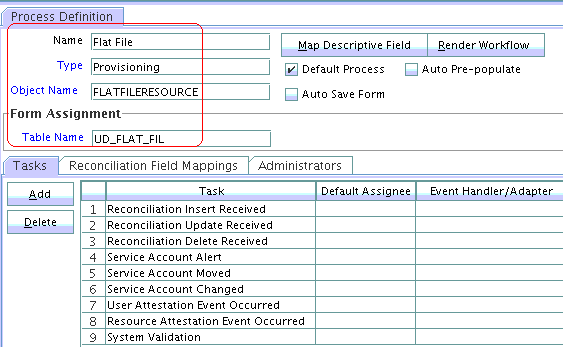
2.1.7.1 Flat File Process Definition

**Purpose**

This section includes creating a process definition for Flat File.

**Steps**

* Open **Process Definition** under **Process Management**
* Enter 'Flat File' as **Name**
* Select **Provisioning** for **Type**
* Select **FLATFILERESOURCE** for **Object Name**
* Select **UD\_FLAT\_FIL** for **Table Name**
* Click **Save**

****

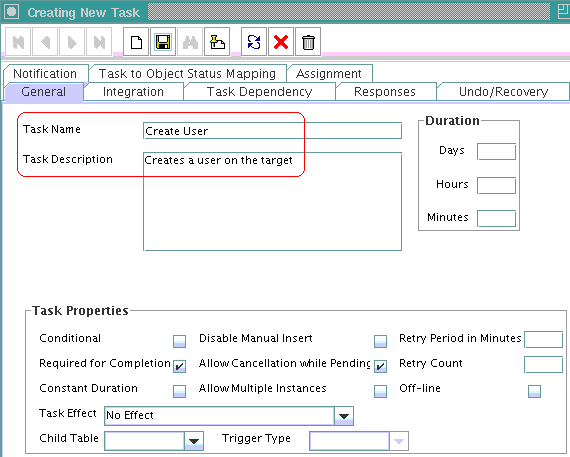
2.1.7.2 Create Process Task

**Purpose**

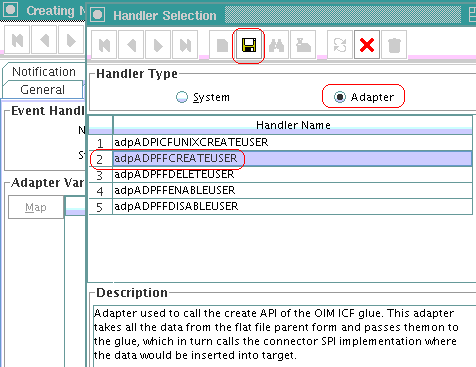
This section includes steps to create a process task for create operation.

**Steps**

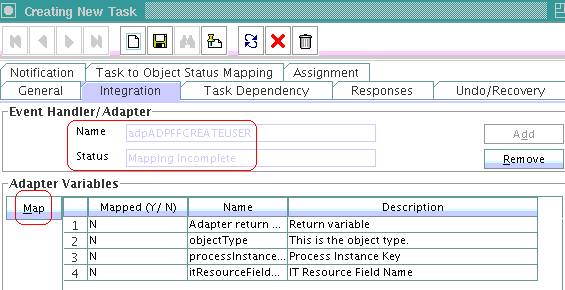
* Under **Tasks** tab (of 'Flat File' Process Definition), click **Add** to open 'Creating New Task' dialog box
* Enter **Create User** as **Task Name**
* Enter **Creates a user on the target** for **Task Description**
* Click **Save**



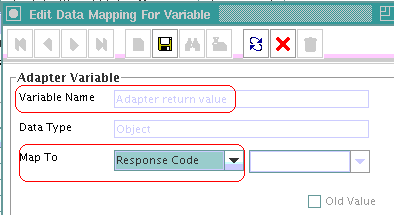
* Select **Integration** tab, click **Add** to see **Handler Selection** screen, select **Adapter** to see a list of adapters available as shown below. Select adpADPFFCREATEUSER



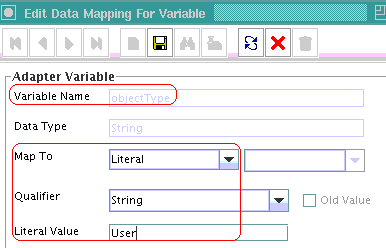
* Click **Save** and **Ok**



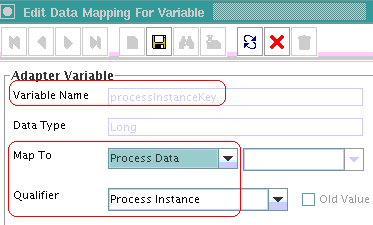
* Select each **variable** and map them by clicking **Map** on each of the variable as shown below
* Adapter Return Variable – Response Code



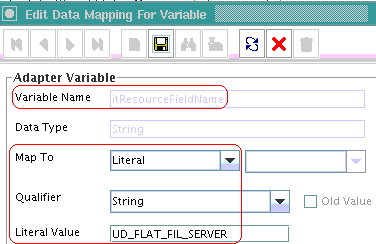
* Object Type – [Literal:String] User (Name of the object type)



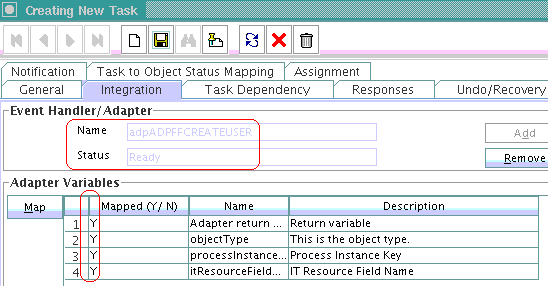
* Process Instance Key – [Process Data] Process Instance



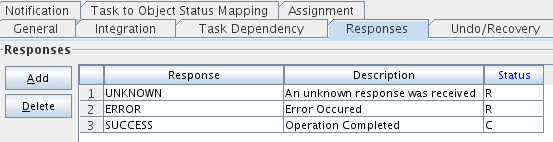
* IT Resource Field Name – [Literal:String] UD\_FLAT\_FIL\_SERVER (Form field name that contains the IT resource information)

.

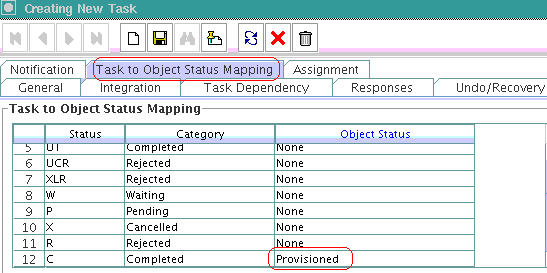
Screen shot should look like below



* Click on **Responses** tab and add 2 more responses as shown below:

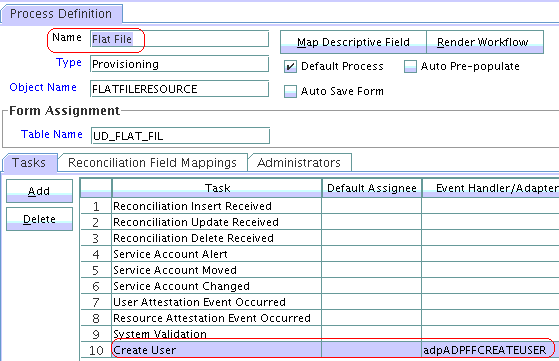


* Click on **Task to Object Status Mapping** and update the **Object Status** for **Status ‘C’** as shown below:



* Click **Save** and **close** the New Task dialog box.

Flat File **Process Definition** should look like below with the new task **Create User**



2.1.7.3 Delete Process Task

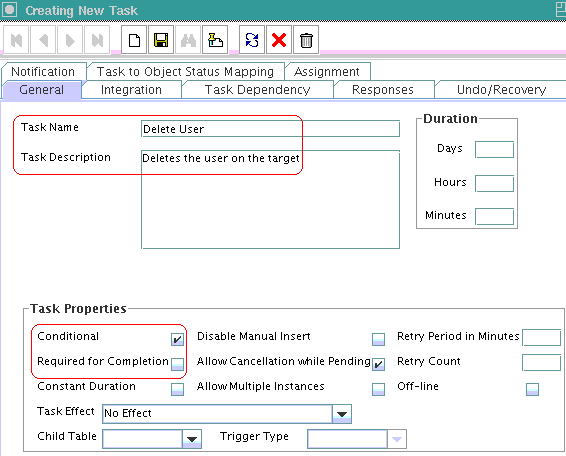
**Purpose**

The step includes creation of a process task for delete operation. This process task is conditional and is **undo** task of **create user task.**

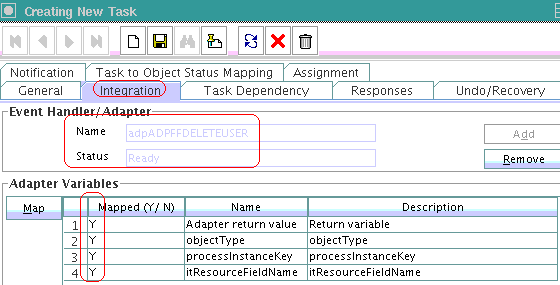
**Steps**

The steps to be followed are exactly the same as in section 2.1.7.2 except the following:

* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**

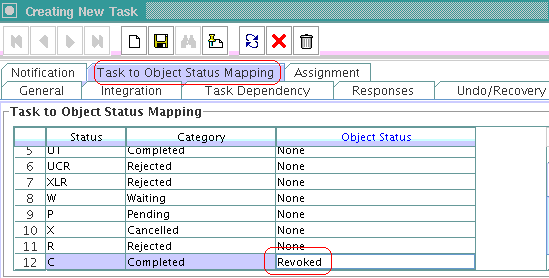
****

* Under **Integration** tab **Event Handler/Adapter** should be **adpADPFFDELETEUSER**

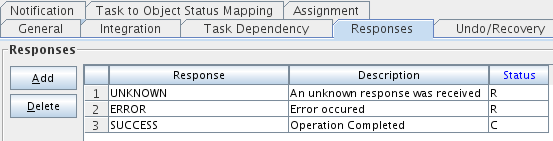
****

**Note: Map the variables accordingly**

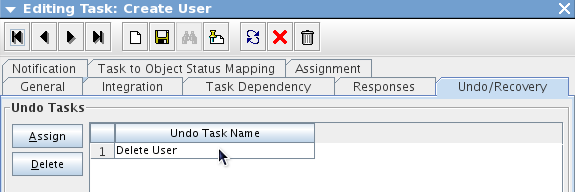
* **Object Status** should be **Revoked** (under tab **Task to Object Status Mapping**)

****

* Add also the Responses:



* **Save** and **close** the New Task dialog box
* Back to the Process Definition, double-click the Create User task (row number) and update the task with ‘Delete User’ task being the Undo task of ‘Create User’ task.



* + Save and close the Create User task; save also the Process Definition

2.1.7.4 Enable Process Task

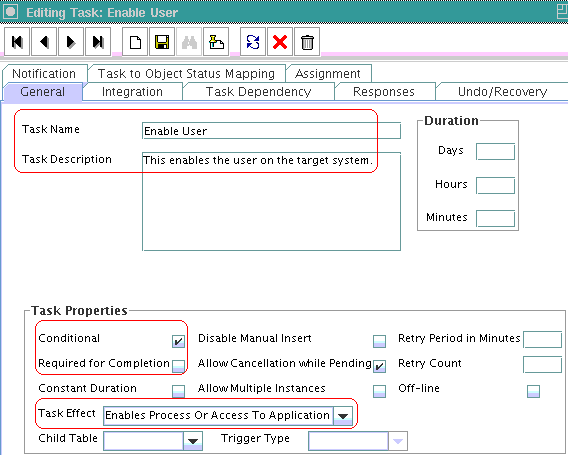
**Purpose**

This includes creation of a process task for enable operation.

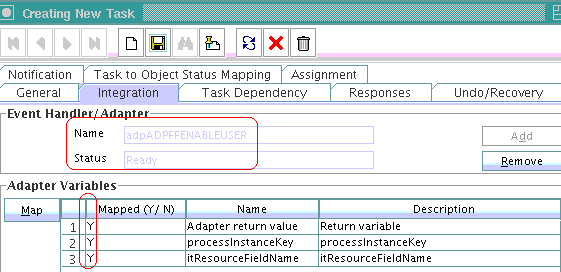
**Steps**

The steps to be followed are exactly the same as in section 2.1.7.2 except the following:

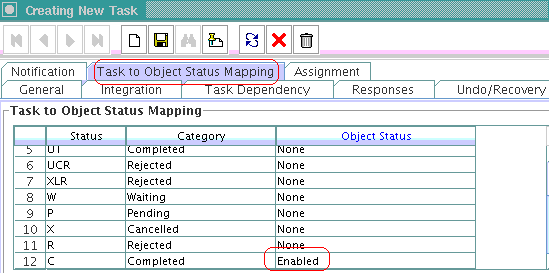
* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Task Effect** should be **Enables Process or Access to application**



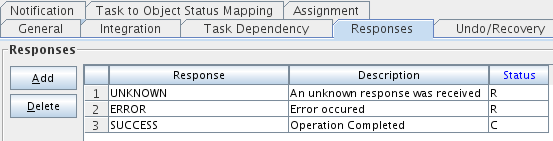
* Under **Integration** tab **Event Handler/Adapter** should be **adpADPFFENABLEUSER**



* **Object Status** should be **Enabled.** This under tab **Task to Object Status Mapping**



* Add also the responses:



2.1.7.5 Disable Process Task

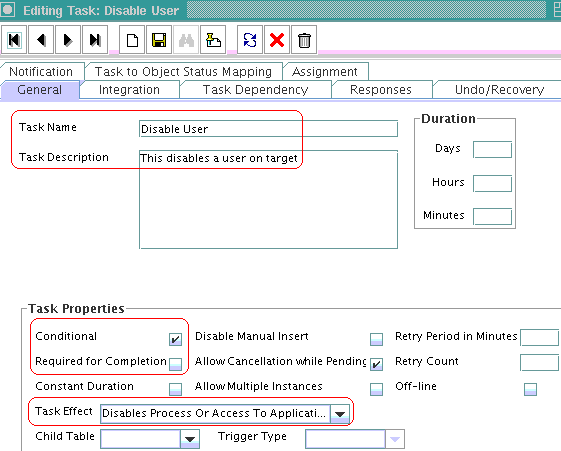
**Purpose**

This includes creation of a process task for disable operation.

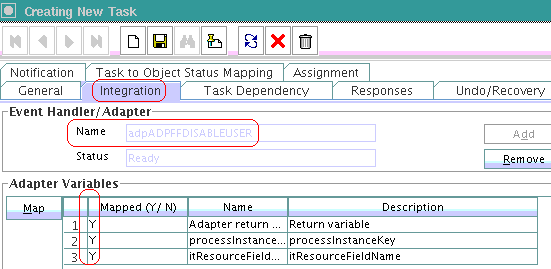
**Steps**

The steps to be followed are exactly the same as in section 2.1.7.4 except the following:

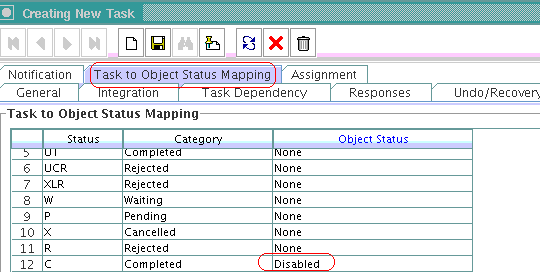
* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Task Effect** should be **Disables Process or access to application**



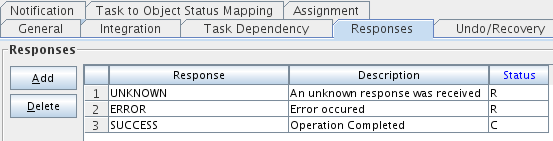
* Under **Integration** tab **Event Handler/Adapter** should be **adpADPFFDISABLEUSER**



* **Object Status** should be **Disabled.** This under tab **Task to Object Status Mapping**



* Also update the responses:



2.1.7.6 Update process task

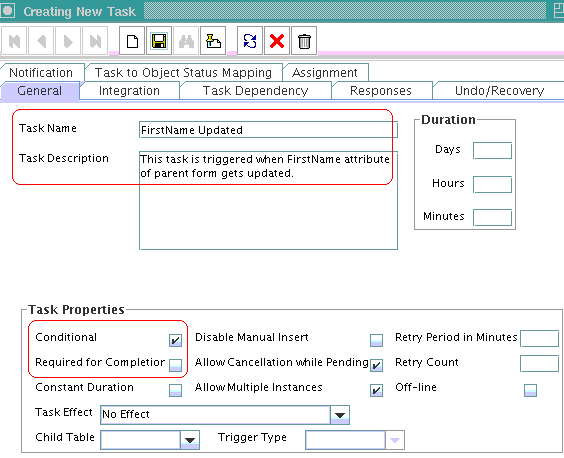
**Purpose**

This includes creation of a process task for updating First Name in target system.

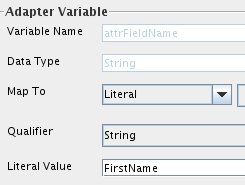
**Steps**

The steps to be followed are exactly the same as in section 2.1.7.4 except the following:

* **Task Name** shouldbe **FirstName Updated**
* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Allow Multiple Instances** should be **checked**

****

* **Event Hanlder/Adapter** should be **adpADPFFUPDATEUSER**
* Map variables accordingly, only change this map
  + **attrFieldName** to **Literal** (String) -> **FirstName**

****

* Configure Task to Object Status Mapping: C – Completed - **Provisioned**
* Add the two responses:
  + SUCCESS – Operation completed – C
  + ERROR – Error occurred - R
* Click **Save**

2.2 Creating provisioning Meta data

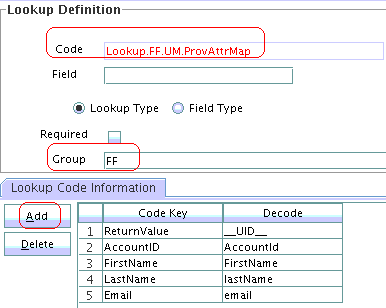
2.2.1 Provisioning attribute map lookup

**Purpose**

This step includes creating a provisioning attribute map lookup, this lookup is used by the ICF OIM glue during provisioning operations. This contains mapping between OIM attributes with target attributes.

**Steps**

* Under **Administration** click **Lookup Definition**
* Enter **Lookup.FF.UM.ProvAttrMap** for **Code**
* Enter **FF** for **Group**
* Click **Save**
* Under **Lookup Code Information,** enter the following:
  + **Code Key = ReturnValue, Decode =** **\_\_UID\_\_.** UID returned by ICF
  + **Code Key = AccountID, Decode = AccountId.** Account ID as represented in flat file.
  + **Code Key = FirstName, Decode = FirstName.** First name as represented in flat file.
  + **Code Key = LastName, Decode = lastName.** Last name as represented in flat file.
  + **Code Key = Email, Decode = email.** Email as represented in flat file.
* Click **Save**



2.2.2 IT resource of type connector server

**Purpose**

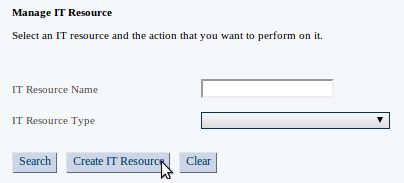
This step includes creating an IT resource of type connector server in OIM.

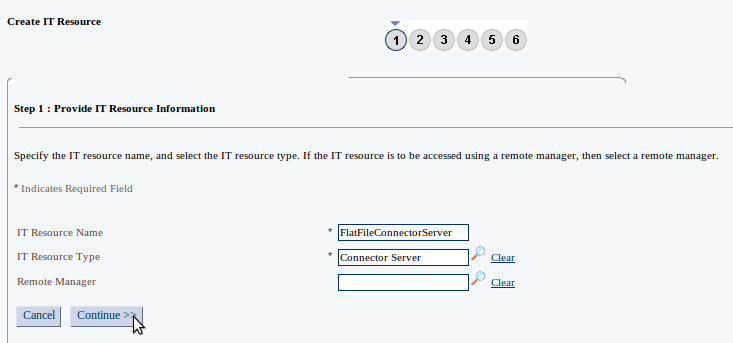
**Steps**

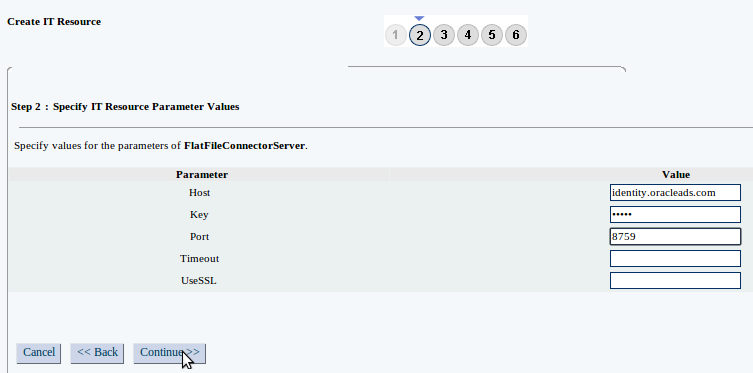
* Login to the System Administration web console by using the **xelsysadm** account.
* Click on **Configuration -> IT Resource**

****

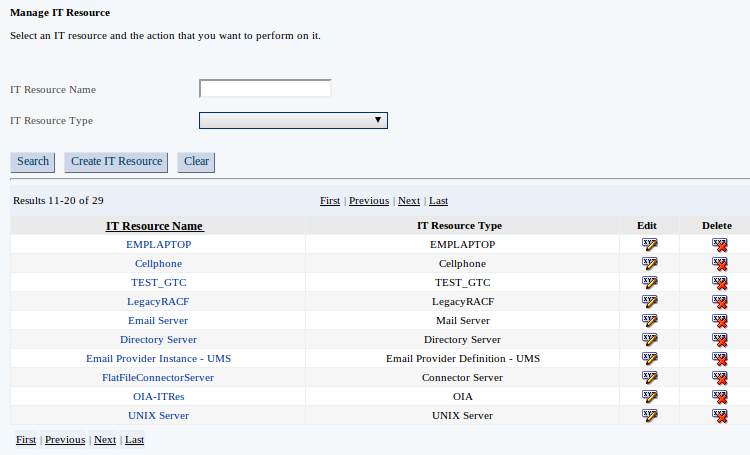
* Click on **Create IT Resource**

****

* Enter **FlatFileConnectorServer** for **IT Resource Name**
* Select **Connector Server** for **IT Resource Type**
* Click **Continue >>**
* Enter the **host name:**  **identity.oracleads.com** of the Host where connector server is running
* Enter **Key: 12345** – remember the creation of the key in lab 17 (sh connectorserver.sh /setKey 12345)
* Enter **Port: 8759**
* Click **Continue >>**



* Complete the wizard by clicking **Continue >>/ Finish**
* Search for the newly created IT resource under **IT Resource.** Screen should look like below



2.2.3 Flat file IT resource

**Purpose**

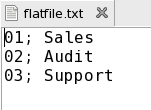
This step involves creating an IT resource of type flat file in OIM.

Before creating the IT Resource with the Sysadmin console, create a text file called **flatfile.txt** in the folder of your choice (i.e. /app/Middleware/Oracle\_IDM1) with the following content:

**01; Sales**

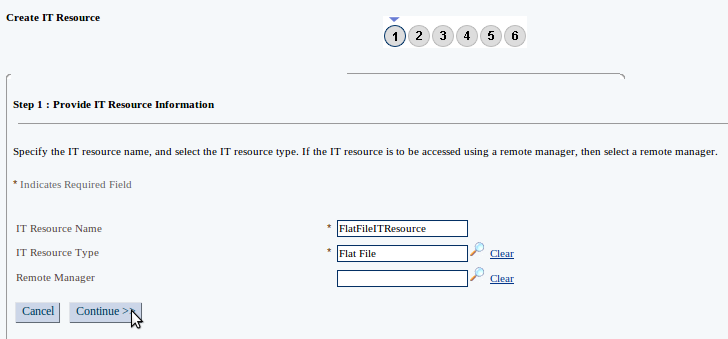
**02; Audit**

**03; Support**



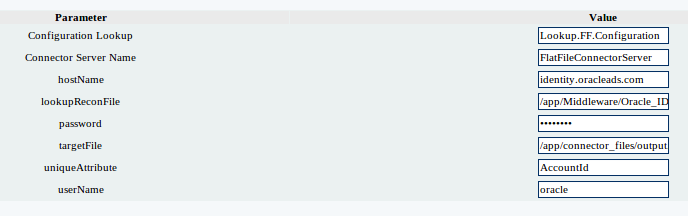
**Steps**

* Login to the System Administration web console by using the **xelsysadm** account.
* Click on **Configuration** -> **IT Resource**
* Click on **Create IT Resource**
* Enter **FlatFileITResource** for **IT Resource Name**
* Select **Flat File** for **IT Resource Type**
* Click **Continue >>**

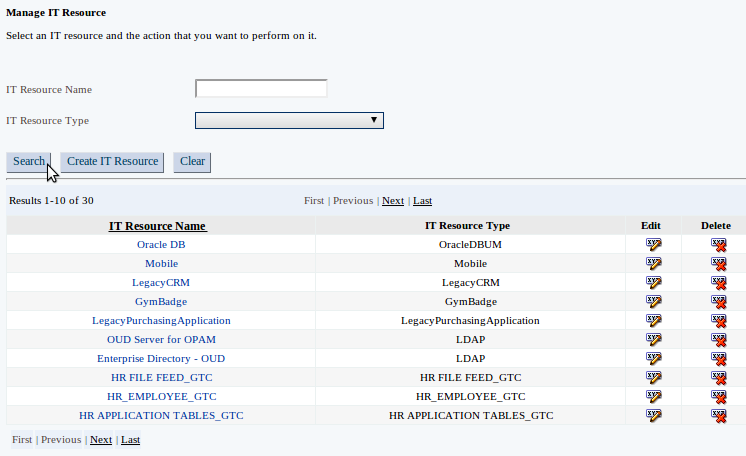


**Parameters:**

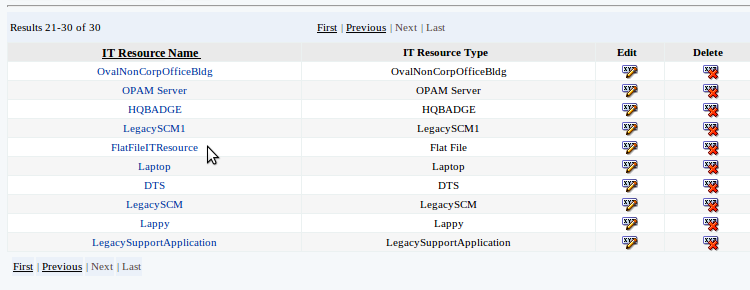
* Connector Server Name: **FlatFileConnectorServer**
* HostName of the target machine: **identity.oracleads.com**
* LookupReconFile: path of flatfile.txt (i.e. /app/Middleware/Oracle\_IDM1/flatfile.txt)
* Password to be used to connect to target system: **Oracle123**
* TargetFile to be used for prov/recon operations: **/app/connector\_files/output.txt**
* Unique attribute: **AccountId**
* Username to be used to connect to target machine: **oracle**



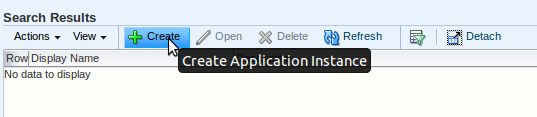
* Click **Continue** 3 more times and then **Finish**
* Search IT resources on **Manage IT resource** screen clicking on **Configuration** -> **IT Resource**, it should look like below:



* Click on **Next** 2 times to double-check that FlatFileITResource has been created:



* + 1. Create an Application Instance for FlatFileITResource.
* Still in The OIM System Administration web console click on **Configuration** -> **Application Instances**
* Click on **Create**



* Enter the following parameters:

Name: **FlatFileApplication**

Display Name: **Flat File Application**

Description: **Flat File Application**

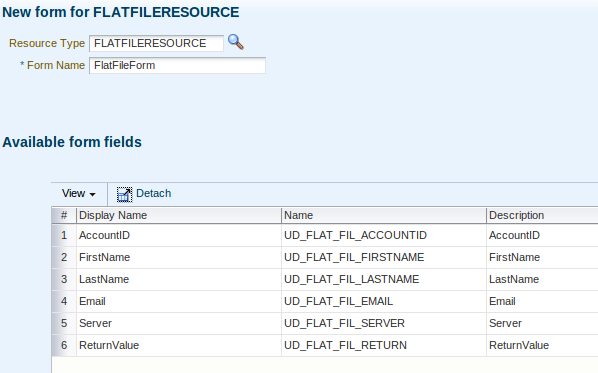
Resource Object: **FLATFILERESOURCE**

IT Resource Instance: **FlatFileITResource**

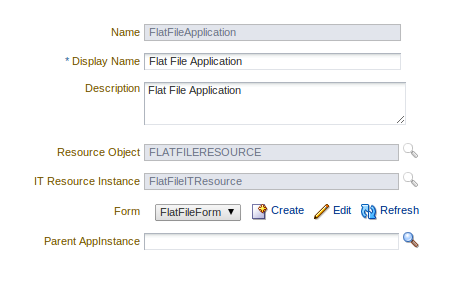
* Click on **Save and close** the Flat File Application and Application Instances tabs

Create an associated Form but before **create a sandbox (Sandboxes** -> **Create Sandbox)**

* Edit the application (Applications Instances -> Search -> Flat File Application -> Open -> Attributes tab) and click on **Create** close to Form.
* Name the form **FlatFileForm** and click on **Create**



* Back to the Flat File Application, click on **Refresh** and select this new form. Click on **Apply**



Don't forget to **publish the sandbox** (close all tabs before).

2.3 Setup lookup recon required for provisioning

2.3.1 Create a new xml with name **FFLookupRecon.xml** and update MDS

**Purpose**

This includes steps to setup lookup recon. It would read different roles available on the target system and update the lookup. These lookup values would be used in the child form during provisioning.

2.3.1 Create a task xml and update MDS

**Purpose**

This is required to populate the lookup used in child form.

**Steps**

* Create a new task xml with entry as shown below and save it as **FFLookupRecon.xml** in **/tmp/mds/db** (create folders if needed)

<?xml version='1.0' encoding='UTF-8'?>

<scheduledTasks xmlns="http://xmlns.oracle.com/oim/scheduler">

<task>

<name>FFLookupRecon</name>

<class>oracle.iam.connectors.icfcommon.recon.LookupReconTask</class>

<description>Gets all the roles as specified in the target machine </description>

<retry>0</retry>

<parameters>

<string-param required="true" encrypted="false" helpText="IT Resource Name">IT Resource Name</string-param>

<string-param required="true" encrypted="false" helpText="Object Type">Object Type</string-param>

<string-param required="true" encrypted="false" helpText="Lookup Name">Lookup Name</string-param>

<string-param required="true" encrypted="false" helpText="Code Key Attribute">Code Key Attribute</string-param>

<string-param required="true" encrypted="false" helpText="Decode Attribute">Decode Attribute</string-param>

<string-param required="false" encrypted="false" helpText="Filter">Filter</string-param>

</parameters>

</task>

</scheduledTasks>

* Change in /app/Middleware/Oracle\_IDM1/server/bin, **weblogic.properties** entries for the import:
  + **wls\_servername=oim\_server1**
  + **application\_name=OIMMetadata**
  + **metadata\_from\_loc=/tmp/mds**
  + **metadata\_files=/db/FFLookupRecon.xml**
* **Start WebLogic Admin Server**
* Upload this new xml in MDS using weblogicImportMetadata.sh available at $DW\_HOME/server/bin/ (/app/Middleware/Oracle\_IDM1/server/bin). Follow below steps
  + **cd /app/Middleware/Oracle\_IDM1/server/bin**
  + **sh weblogicImportMetadata.sh**
  + Enter weblogic username (weblogic)
  + Enter weblogic password (Oracle123)
  + Enter server URL: t3://identity.oracleads.com:7001 (7001 the port where the Admin server is running).

Successful message: **End of importing metadata script ...**

* + Stay in the same directory
* Purge the cache by executing PurgeCache.sh all. Steps are outlined below.
  + Set the application server variable: **export APP\_SERVER=weblogic**
  + Set the domain home: **export DOMAIN\_HOME=/app/Middleware/user\_projects/domains/iam\_domain/**
  + Execute **sh PurgeCache.sh all**
  + Enter OIM admin username (xelsysadm)
  + Enter OIM admin password (Oracle123)
  + Enter OIM URL: **t3://identity.oracleads.com:14000**

Successful message: **Purging the cache categories:[all] is successful**

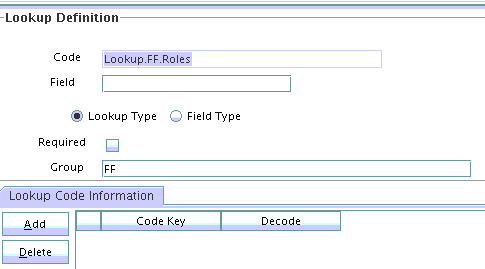
2.3.2 Create a new lookup

**Purpose**

This step includes creating a look up for roles in OIM design console. The values would be used by the child form.

**Steps**

* Login to Design console
* Under **Administration,** click **Lookup Definition**
* Enter **Lookup.FF.Roles** for **Code**
* Enter **FF** for **Group**
* Click **Save**



2.3.3 Create Child form

**Purpose**

This includes steps to create a child form, this child form would be used during provisioning and its data would be coming through via lookup recon.

**Steps**

* Under **Development Tools,** click **Form Designer**
* Enter **UD\_FLAT\_CH** for **Table Name**
* Enter **Flat file roles form** for **Description**
* Click **Save**
* Under **Additional Columns,** add
  + **Name = UD\_FLAT\_CH\_ROLES, Variant Type = String, Length = 30, Field Label = Roles, Field Type = Lookup Field and order = 1**
* Click **Save**
* Under **Properties** tab, select **Roles (Lookup field)** and **Add Property**
  + **Lookup Code = Lookup.FF.Roles**
* **Save** the form
* Click **Make Version Active** and accept the pop up

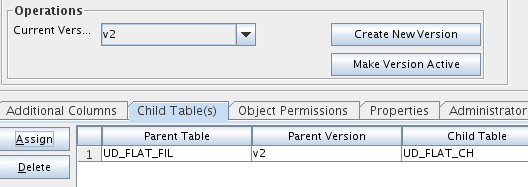
2.3.4 Update Parent Form (created in 2.1.5.2)

**Purpose**

This step includes updating existing parent form to include newly created child form as its child.

**Steps**

* + - * Searchfor **UD\_FLAT\_FIL,** in **Form Designer.**
      * Click on **Create New Version**
      * Enter appropriate version number (v2 for example)
      * **Save** (the dialog box should blink) and **close** the “Create a new version” dialog box
      * Make sure that the new version is selected as Current Version
      * Select tab **Child Table(s),** click **Assign**
      * Select **UD\_FLAT\_CH,** click **Ok**

****

* Click on **Save** and on **Make Version Active**

2.4 Adapters – Child form

2.4.1 Create adapter for child table.

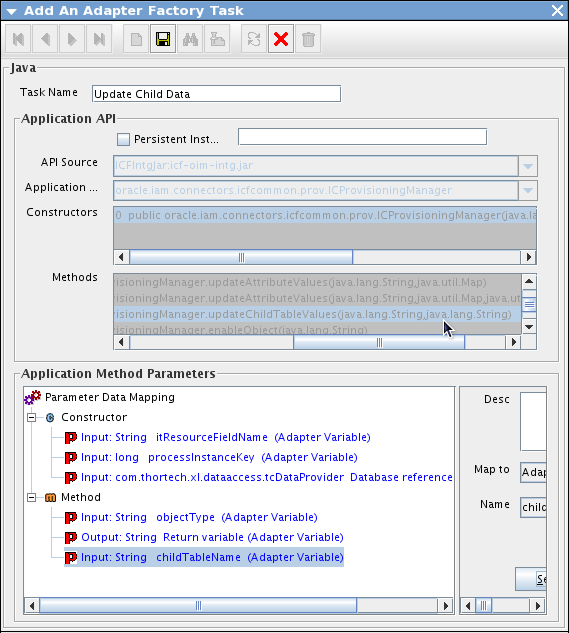
**Purpose**

This step includes creating an adapter for the above child table created. This adapter would be calling the **updateChildTableValues** API of the **ICProvisioningManager** Application.

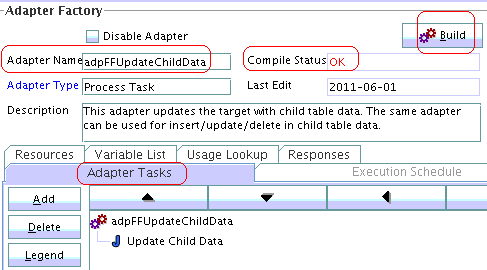
**Steps**

* Under **Development Tools,** click **Adapter Factory**
* Enter **adpFFUpdateChildData** for **Adapter Name**
* Select **Process Task** for **Adapter Type**
* Enter ‘This adapter updates the target with child table data. The same adapter can be used for insert/update/delete in child table data.’ for **Description**
* Click **Save**
* The **Adapter Task** should call the **updateChildTableValues** API of the **ICProvisioningManager.** Steps are same as shown in section 2.1.6.1 except that this adapter needs another variable as stated below
* For your information, the variables created in section 2.1.6.1 are:
  + **Name – objectType, Type – String, Description – This is the object type, Map To – Resolve at runtime**
  + **Name – processInstanceKey, Type – Long, Description – Process Instance Key, Map To – Resolve at runtime**
  + **Name – itResourceFieldName, Type – String, Description – IT Resource Field Name, Map to – Resolve at runtime**
* The added variable is:
  + **Name – childTableName, Type – String, Description – Child Table Name, Map to – Resolve at runtime**
* Add an Adapter Task named **Update Child Data** (be careful to take the right method for the updateChildTableValues (notice the final s): the signature should be with **2** **Strings)**.

Its final screenshot should look like:



* **Save** and **Build** the adapter**.**



2.5 Process Tasks – Child table

2.5.1 Create Process Task for child table.

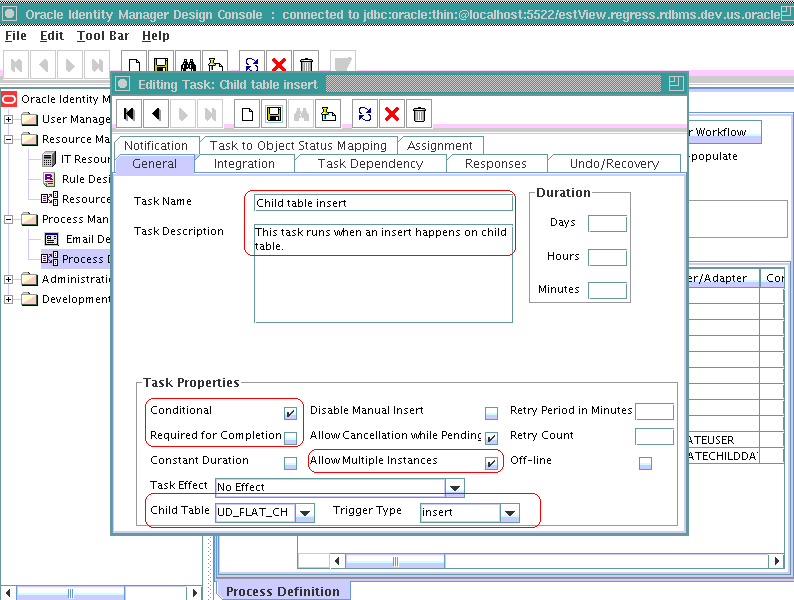
**Purpose**

These steps include creating a process task when an insert happens on child table.

**Steps**

The steps to be followed are exactly the same as in section 2.1.7.2 (search **Flat File** in Process Definition) except the following:

* **Task Properties**
  + Task Name: **Child table insert**
  + Task Description: **This task runs when an insert happens on child table**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Allow Multiple Instances** should be checked.
  + **UD\_FLAT\_CH** should be selected from the **Child Table** dropdown
  + **Trigger Type** should be **insert**

****

* Click **Save**
* Under **Integration** tab **Event Handler/Adapter** should be **adpADPFFUPDATECHILDDATA.**
* Mapping of adapter variables are same except for **childTableName.** This should be mapped to a String literal = **UD\_FLAT\_CH**
* For your information, the other mappings are the following:
  + Adapter return value – Response Code
  + objectType – [Literal:String] User
  + processInstanceKey – [Process Data] Process Instance
  + itResourceFieldName – [Literal:String] UD\_FLAT\_FIL\_SERVER
* Configure Task to Object Status Mapping: C – Completed - **Provisioned**
* Add the two responses:
  + SUCCESS – Operation completed – C
  + ERROR – Error occurred – R
* **Save**

2.5.2 Update Process Task for child table

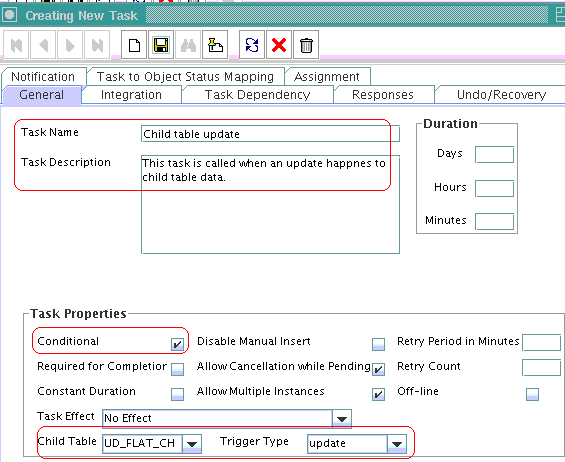
**Purpose**

This step includes creation of process task when an update happens to child table data

**Steps**

Steps are exactly the same except for the below:

* **Trigger Type** should be **update**



2.5.3 Delete Process Task for child table

**Purpose**

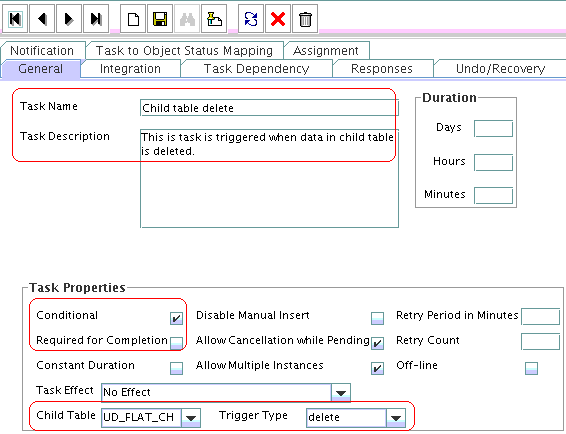
This step includes creation of process task when a delete happens to child table data

**Steps**

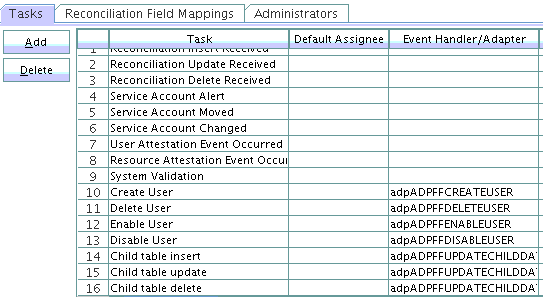
Steps are exactly the same except for the below:

* **Trigger Type** should be **delete**

Process tasks should look like below



The **Process Task** for **Flat File** should look like below



2.6 Scheduled Job for lookup recon

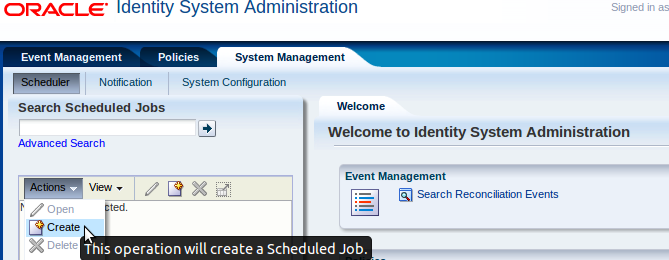
2.6.1 Create a scheduled job and run

**Purpose**

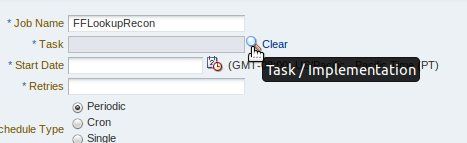
This step includes creating a scheduled job, which when run updates the lookup **Lookup.FF.Roles** created in 2.3.2. This lookup data would be used in the child form created in 2.3.3

**Steps**

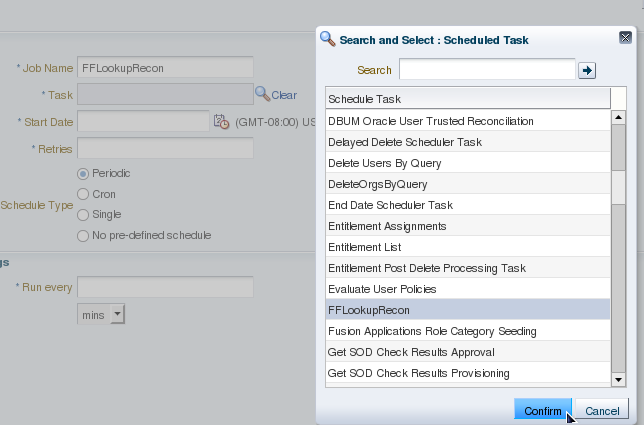
* Log in to the System Administration web console by using the **xelsysadm** account.
* Click on **System Management** -> **Scheduler**
* Click on **Create** from the **Actions** dropdown



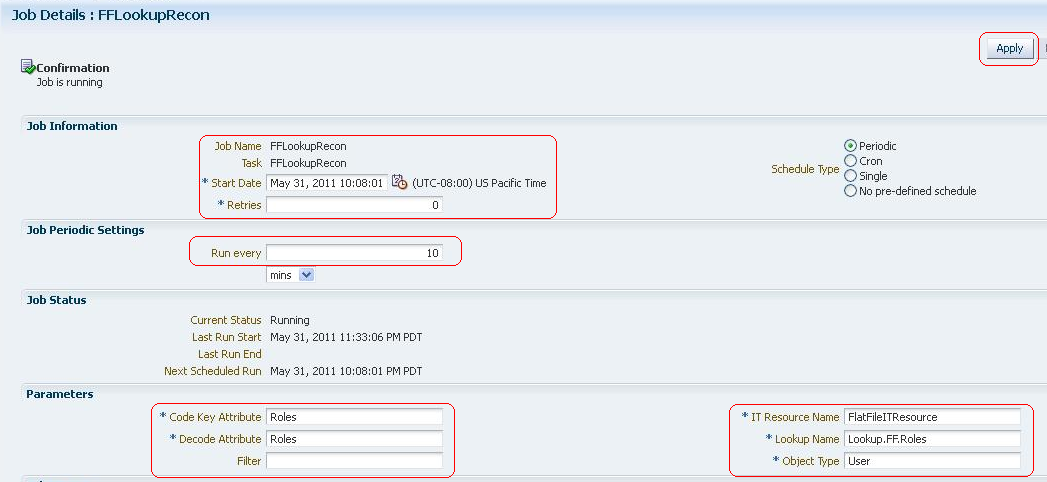
* Enter **Job Name** as **FFLookupRecon**
* Click on **Task/implementation** icon



* “Search and Select: Scheduled Task” pops up**,** click on **Search arrow.**
* Select **FFLookupRecon,** click **Confirm**



* Fill information as shown below (enter the current date for the start date) and click **Apply**

****

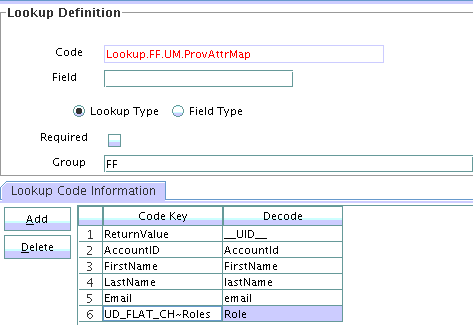
2.6.2 Update Provisioning attribute map lookup to include child form field label

**Purpose**

This section includes updating the provisioning attribute map as created in section. ICF uses this lookup for attributes to be provisioned.

**Steps**

* Using the Design Console, search for **Lookup.FF.UM.ProvAttrMap**.
* Add **Code Key = UD\_FLAT\_CH~Roles, Decode = Role.** ICFunderstands this format **Code =** **CHILD\_FORM~CHILD\_FROM\_FIELD\_LABEL** and **Decode = Target\_Attribute**
* Click **Save**



2.7 Setup remote connector server

**Purpose**

This step includes starting a remote connector server. ICF flat file bundle would be placed on this machine instead of OIM machine. Once the connector server is running, we could create an IT resource in OIM.

**Steps**

* Navigate to the bin directory of the connector server (/app/Connector\_Server\_111150/connector\_server\_java-1.2.6195/bin)
* Start the Connector Server: **sh connectorserver.sh /run**
* Run the lookup recon to get all roles from the target server
* Search and open the scheduled job **FFLookupRecon** as created in section 2.6
* Click **Run Now** to run the scheduled job.
* Close the schedule job window and back to the Sysadmin web console, click on **Configuration** -> **Lookups**. Search for the lookup **Lookup.FF.Roles** and verify that roles have been populated (the codes numbers could differ).



2.8 Test provisioning

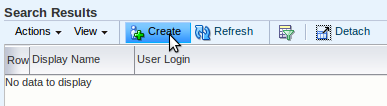
2.8.1 Create User (with child table data)

**Purpose**

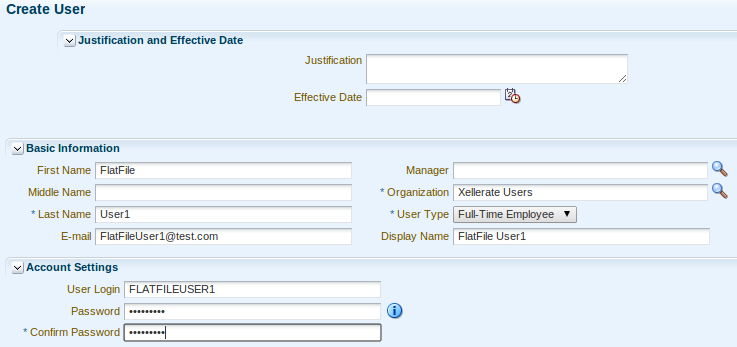
Steps include creating a user on the target flat file to test the flat file bundle code and the corresponding OIM metadata.

**Steps**

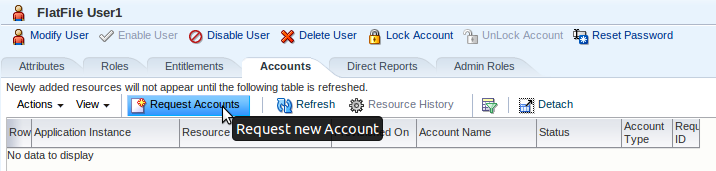
* Log into OIM Identity Self Service web console.
* Click on **Users** and on **Create** link



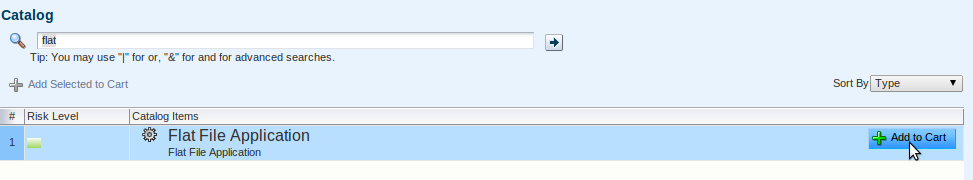
* Enter details as shown below and click on **Submit**



* You should see a confirmation message that the user has been created.
* Click on **Accounts** tab and on **Request Accounts**



* Enter **Flat** in the Catalog search, click on the **right arrow** and on **Add to Cart**



* Click on **Checkout**



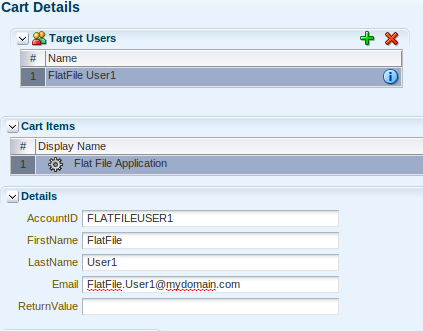
* + **Enter :**

Account ID: **FLATFILEUSER1**

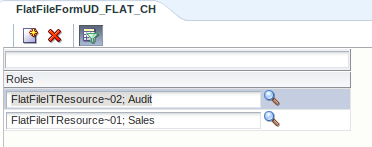
First Name: **FlatFile**

Last Name: **User1**

Email: **FlatFile.User1@mydomain.com**



* + Add also some roles:



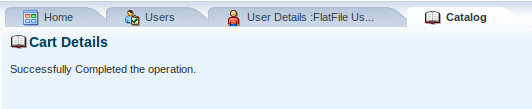
* + - Click on **Ready to submit**

Notice that the status has changed

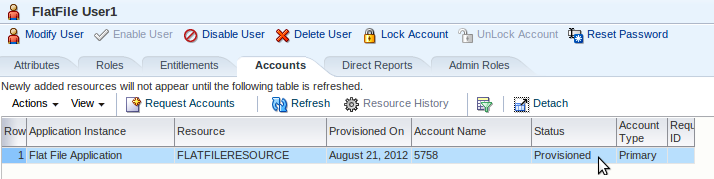


* + - and click on **Submit**

You should see a successful message:



* Clicking back to the **User Details** tab, click on **Refresh** to get the status of the provisioning process



* Double check the result file (**output.txt**):

**FirstName:FlatFile;email:FlatFile.User1@mydomain.com;AccountId:FLATFILEUSER1;lastName:User1;Role:490~01; Sales,**

**Michel: issue with the roles: only one has been added instead of two**

**Manju: This is because of R2 bug. BUG 14317800 - CHILD TABLE PROVISIONING IS NOT WORKING PROPERLY**

With these above steps we have verified the following

* Parent table data being inserted into target flat file
* Child table data being inserted into target flat file
* Child table data being fetched from **Lookup recon**

2.8.2 Update user details

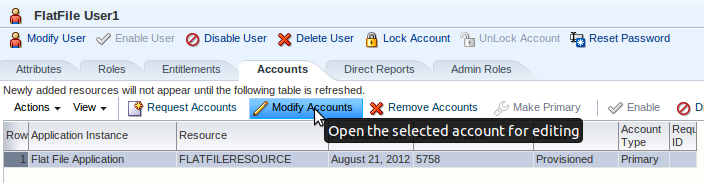
2.8.2.1 Primary form data

**Purpose**

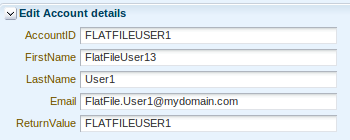
This section updates the **First Name** belonging to parent form of the newly added resource and verifies the same has been updated on the target flat file.

**Steps**

* On the same screen, select **Resource – FLATFILERESOURCE** and click on **Modify Accounts**



* Update **FirstName** from **FlatFile** to **FlatFileUser13** and click on **Ready to submit and then Submit.**



* Check in output.txt that the result is correct:

FirstName:**FlatFileUser13**;email:FlatFile.User1@mydomain.com;AccountId:FLATFILEUSER1;lastName:User1;

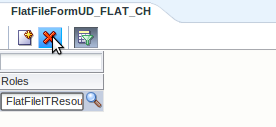
* + - 1. Child table data – delete role

**Purpose**

This section removes the **Role - Sales** belonging to child form of the newly added resource and verifies the same has been updated on the target flat file.

**Steps**

* Select the FLATFILERESOURCE and click on Modify Accounts
* At the bottom of the screen, **select the role** and click on **delete**



* Click on **Ready to submit** and **Submit**.
* Open target flat file to see, **Sales** role being removed

**Michel: not the case (note: no new entry in resource history)**

**Manju: Bug 14335634 - DELETE CHILD DATA IS NOT WORKING PROPERLY**

With these above steps we have verified the following

* Updates to parent table data being propagated to target flat file
* Updates to child table data being propagated to target flat file.

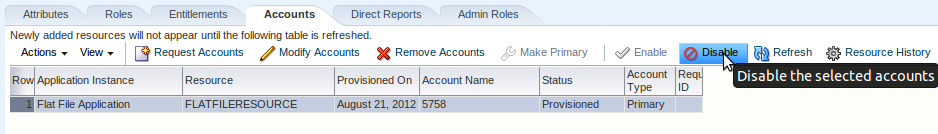
2.8.3 Disable user

**Purpose**

This section includes steps which disables the user on the target flat file.

**Steps**

* On the User **Details / Accounts** screen, select **Resource – FLATFILERESOURCE** and click on **Disable**



* Click **Submit** to confirm
* **Refresh** and verify the **Status** to see **Disabled**



* Open the target flat file to see status enable = false

**FirstName:FlatFile;email:FlatFile.User1@mydomain.com;AccountId:FLATFILEUSER1;lastName:User1;Role:490~01; Sales,\_\_ENABLE\_\_:false;**

**Note: -** When we disable the user on OIM like above, the executeQuery() API gets called on the bundle code with an attribute **\_\_ENABLE\_\_ = false.** Based on this condition we can call the disable target API (if target supports), in this lab, for simplicity, we are just appending it to target flat file.

With these above steps we have verified the following

* Disable a user on the target flat file

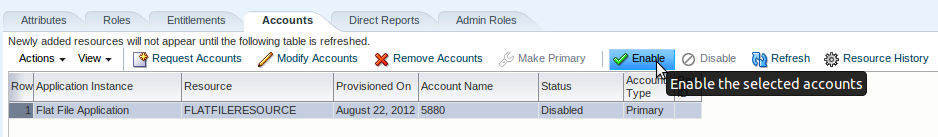
2.8.4 Enable User

**Purpose**

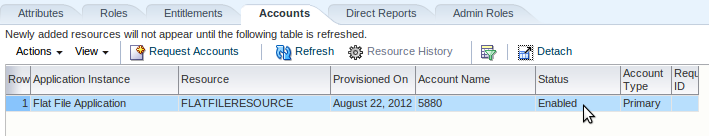
This section includes steps which enables the user on the target flat file.

**Steps**

* On the **User Details / Accounts** screen,select **Resource – FLATFILERESOURCE** and click on **Enable**



* Click on **Submit**
* **Refresh** and verify the Status to see **Enabled**



* Open the target flat file to see status enable = true

**FirstName:FlatFile;email:FlatFile.User1@mydomain.com;AccountId:FLATFILEUSER1;lastName:User1;Role:490~01; Sales,\_\_ENABLE\_\_:true;**

**Note: -** When we enable the user on OIM like above, the executeQuery() API gets called on the bundle code with an attribute **\_\_ENABLE\_\_ = true.** Based on this condition we can call the enable target API (if target supports), in this lab, for simplicity, we are just appending it to target flat file.

With these above steps we have verified the following

* Enable a user on the target flat file

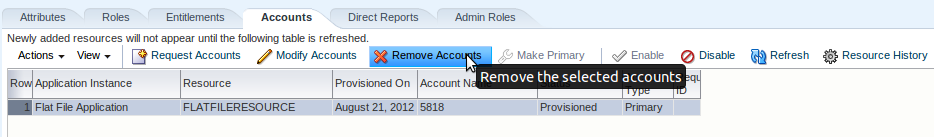
2.8.5 Revoke User

**Purpose**

Steps include deleting a user

**Steps**

* On the **User Details / Accounts** screen,select **Resource – FLATFILERESOURCE** and click on **Remove Accounts**



* Click **Submit** to confirm
* Verify the **Status**

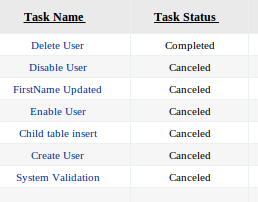


* Open the flat file to see entry has been deleted

With these above steps we have verified the following

* Revoking the resource, i.e. details from target flat file has been deleted.

The Resource history looks like below. To see the history, select the resource and click **Resource History**



2.9 Creating Reconciliation metadata

2.9.1 Full/Incremental reconciliation

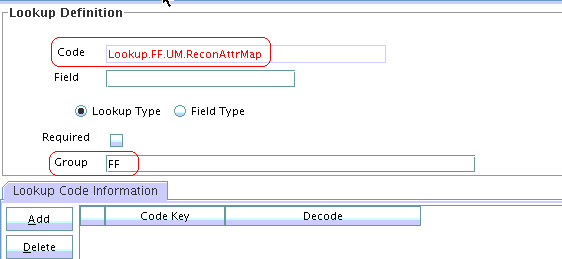
2.9.1.1 Reconciliation attribute map

**Purpose**

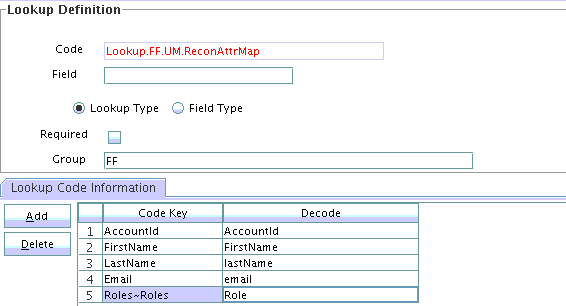
This section includes steps to creating the reconciliation attribute map. This map is referred by the OIM ICF glue during reconciliation.

**Steps**

* Login to OIM design console (you can use also the Sysadmin web console), Click on **Lookup Definition** under **Administration.**
* Enter **Lookup.FF.UM.ReconAttrMap** for **Code** and **FF** for **Group**
* Click **Save**



* Enter values as shown below and **Save**



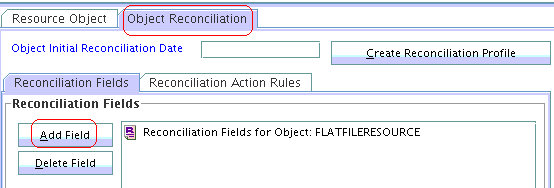
2.9.1.2 Reconciliation Profile

**Purpose**

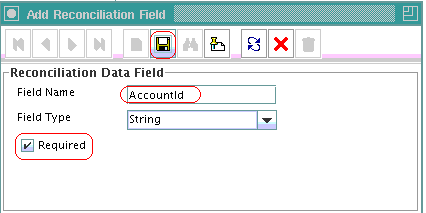
Create reconciliation profile

**Steps**

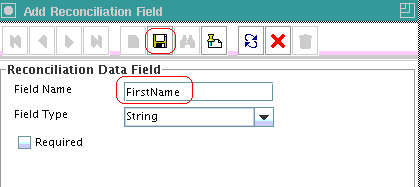
* Open **Resource Object** under **Resource Management**
* Search for **FLATFILERESOURCE** and click on **Object Reconciliation** tab

****

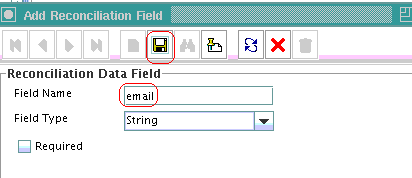
* Click **Add Field** and add **AccountId** Stringand make it as **Required** and **Save**

****

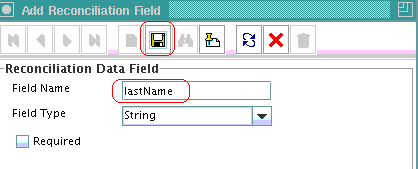
* Add **FirstName** and **Save**

****

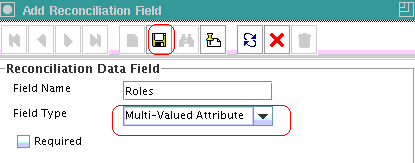
* Add **email** and **Save**



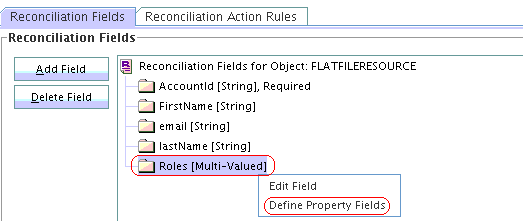
* Add **lastName** and **Save**



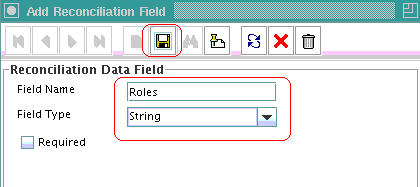
* Add **Roles** and **Save**



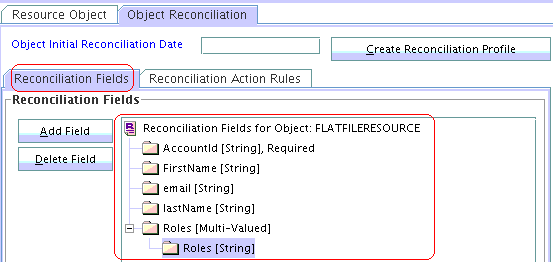
* Right click on **Roles** (**Multi-Valued**)and select **Define Property Field** (right-click the field)



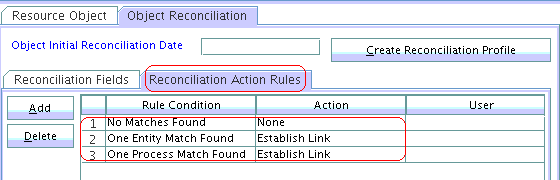
* Enter **Roles** as show



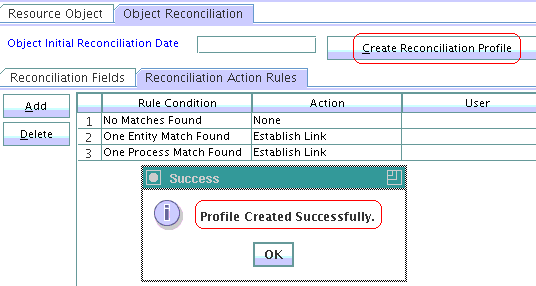
* Reconciliation Fields should look like below (the order of the fields maybe different)



* Click tab **Reconciliation Action Rules** and enter below action rules

****

* Click **Create Reconciliation Profile** to see the confirmation.



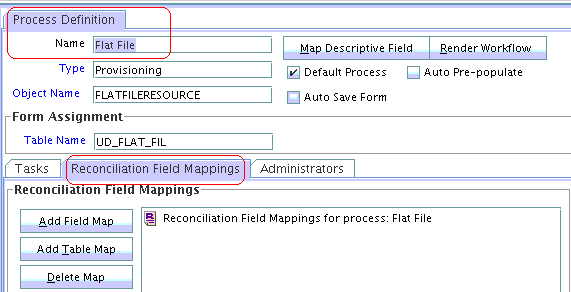
2.9.1.3 Reconciliation Mapping

**Purpose**

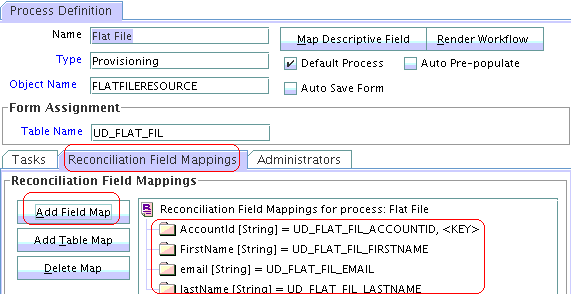
Create mapping required for reconciliation

**Steps**

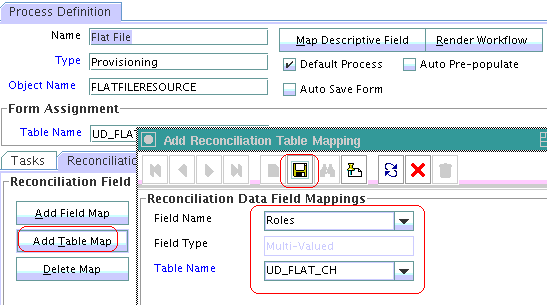
* Open **Process Definition** under **Process Management**
* Search for **Flat File** and click **Reconciliation Field Mappings**



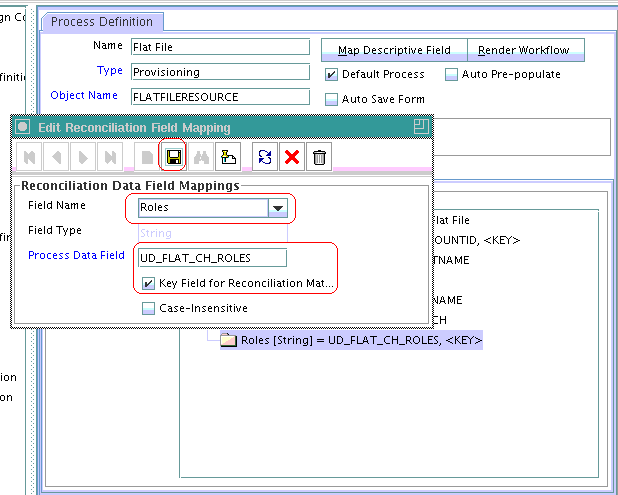
* Add **Field Mapping** for **AccountId** (Key field)**, FirstName, lastName** and **email** as shown below.



* Click **Add Table Map** and Select Roles as shown below. Click **Save**



* Right click on the newly created field name '**Roles**', click on '**Define property field map**' and select data as shown, select **Key Field**



* **Save** the **Process Definition**

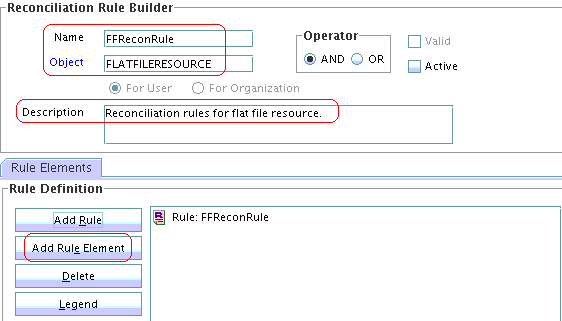
2.9.1.4 Reconciliation Rule

**Purpose**

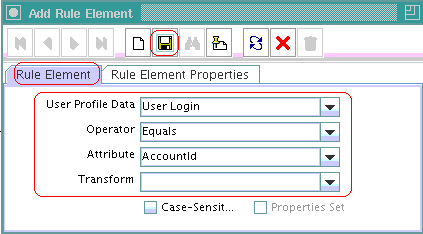
Create reconciliation rule

**Steps**

* Open **Reconciliation Rules** under **Development Tools**
* Enter data as shown and click **Save,** and click **Add Rule Element**

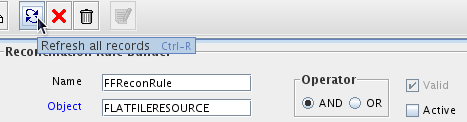


* Enter data as shown below



* Make Recon rule active and save

Note: if you get an error uncheck **Active** and **Refresh** the rule:



Notice that Valid is checked

Check **Active** and **Save**

2.9.1.5 Create Scheduled task and update MDS

**Purpose**

This section includes steps to create a scheduled task used by reconciliation.

**Steps**

* Create FlatFileReconciliation.xml in /app/Middleware/Oracle\_IDM1) with the following content:

<?xml version="1.0" encoding="utf-8"?>

<xl-ddm-data version="1.1.0.4" user="XELSYSADM" database="jdbc:oracle:thin:@adc2190969.us.oracle.com:5525/oimdb.regress.rdbms.dev.us.oracle.com" exported-date="1324042906836" description="FlatFileReconciliation.xml">

<ScheduleTask name="Flat File Reconciliation">

<TSK\_UPDATE>1274258546000</TSK\_UPDATE>

<TSK\_DISABLE>1</TSK\_DISABLE>

<TSK\_FREQTYPE>ONCE</TSK\_FREQTYPE>

<TSK\_LAST\_START\_TIME>1274258548000</TSK\_LAST\_START\_TIME>

<TSK\_LAST\_STOP\_TIME>1274258555000</TSK\_LAST\_STOP\_TIME>

<TSK\_CLASSNAME>oracle.iam.connectors.icfcommon.recon.SearchReconTask</TSK\_CLASSNAME>

<TSK\_STOP>0</TSK\_STOP>

<TSK\_START\_TIME>1274258548000</TSK\_START\_TIME>

<TSK\_RETRY\_COUNT>0</TSK\_RETRY\_COUNT>

<TSK\_STATUS>INACTIVE</TSK\_STATUS>

<ScheduleTaskParam name="Scheduled Task Name">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

<TSA\_VALUE>Flat File Reconciliation</TSA\_VALUE>

</ScheduleTaskParam>

<ScheduleTaskParam name="IT Resource Name">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

<TSA\_VALUE>FF IT Resource</TSA\_VALUE>

</ScheduleTaskParam>

<ScheduleTaskParam name="Resource Object Name">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

<TSA\_VALUE>FF User</TSA\_VALUE>

</ScheduleTaskParam>

<ScheduleTaskParam name="Object Type">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

<TSA\_VALUE>User</TSA\_VALUE>

</ScheduleTaskParam>

<ScheduleTaskParam name="Incremental Recon Attribute">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

<TSA\_VALUE>LastModified</TSA\_VALUE>

</ScheduleTaskParam>

<ScheduleTaskParam name="Latest Token">

<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

</ScheduleTaskParam>

<ScheduleTaskParam name="Filter">

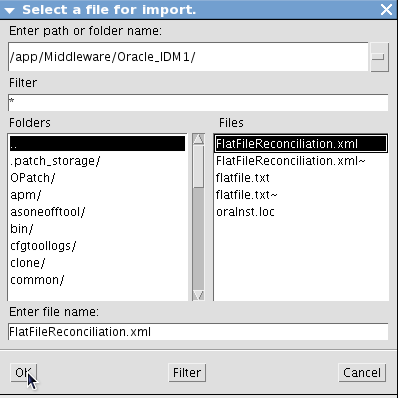
<TSA\_UPDATE>1274190657000</TSA\_UPDATE>

</ScheduleTaskParam>

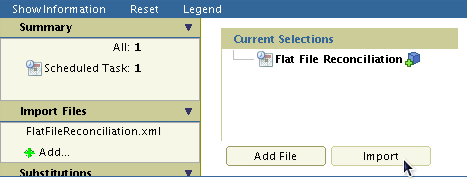
</ScheduleTask>

</xl-ddm-data>

* Login into OIM Sysadmin web console, click on **System Management** -> **Import** (you may have to check “Start import anyway (get lock by force)” and click on “Get Lock”)
* Select the xml file that you have created, click **OK** and on **Add File**



* Click on Import.



2.9.2 Lookup recon

See section 2.3

2.9.3 Role recon – This should be same as having Full/Incremental reconciliation, except for the fact that ObjectClass would be different. This should SearchReconTask of OIM ICF glue. See section 2.9.1

2.9.4 Org recon – This should be same as having Full/Incremental reconciliation, except for the fact that ObjectClass would be different. This should SearchReconTask of OIM ICF glue. See section 2.9.1

Re-create the reconciliation profile and clear cache by following below steps:

* Open design console
* Open Resource Objects under Resource Management.
* Search for FLATFILERESOURCE
* Go to tab Object Reconciliation.
* Click on Create Reconciliation Profile
* Clear cache using PurgeCache utility:
* Set the application server variable: **export APP\_SERVER=weblogic**
* Set the domain home: **export DOMAIN\_HOME=/app/Middleware/user\_projects/domains/iam\_domain/**
* **cd /app/Middleware/Oracle\_IDM1/server/bin**
* Execute **sh PurgeCache.sh all**
* Enter OIM admin username (xelsysadm)
* Enter OIM admin password (Oracle123)
* Enter OIM URL: **t3://identity.oracleads.com:14000**

2.10 Test reconciliation

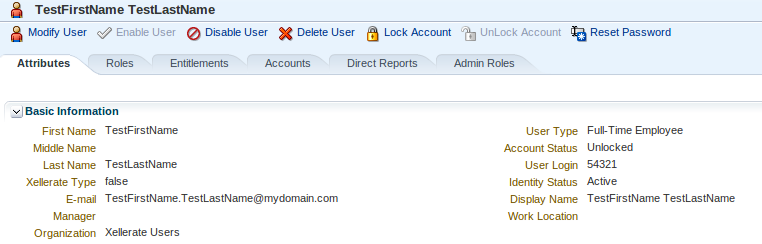
2.10.1 Full/Incremental reconciliation

**Purpose**

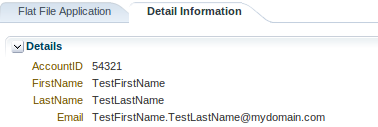
Test Full/Incremental reconciliation

**Steps**

* For testing we assume the following user has been provisioned:



Account data:



Content of Output.txt :

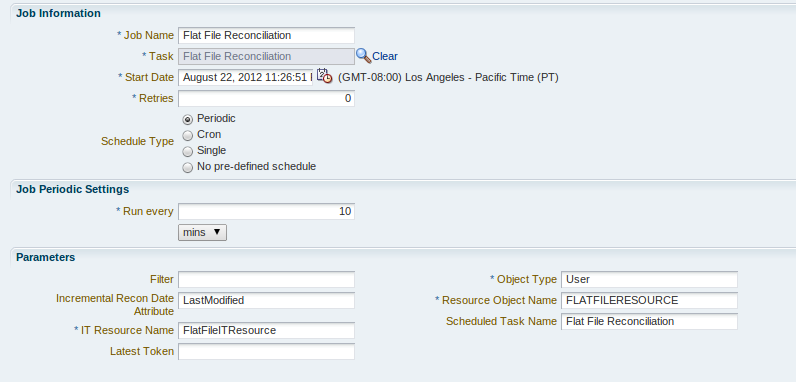
**FirstName:TestFirstName;email:TestFirstName.TestLastName@mydomain.com;AccountId:54321;lastName:TestLastName;**

* Now let's change the value in output.txt :

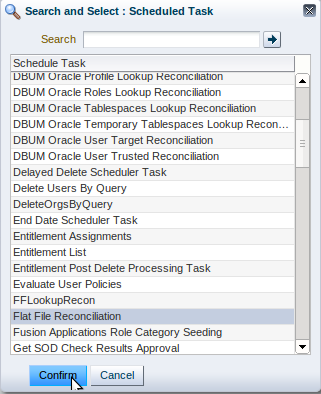
FirstName:**TestUserFirstName**;email:TestFirstName.TestLastName@mydomain.com;AccountId:54321;lastName:TestLastName;

Note: for testing purpose, we have directly updated the FirstName to TestUserFirstName. Make sure the content of the file Output.txt is only above and does not have any roles in it. This would be sufficient for demo purpose.

* Using Sysadmin web console, create a schedule job Flat File Reconciliation:



Don’t forget to select Flat File Reconciliation scheduled task:

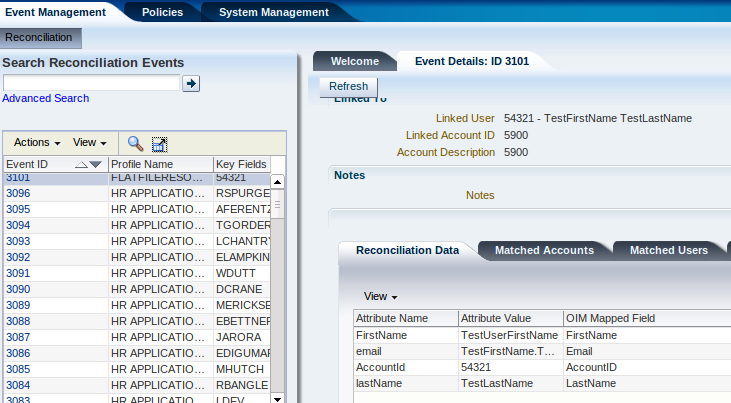


* Run the schedule job and wait for the job to finish (you may have an error message that you can ignore).

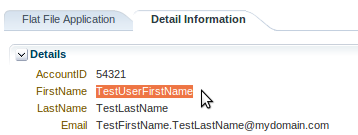
**Michel: many error messages in OIM terminal but it works!**

**Manju: This could be because the task xml initially was in 11g format. I have updated the task.xml ( section 2.9.1.5 ) to be in 9x format.**

* Verify the reconciliation events being generated
* Click **Event Management** and search for **Reconciliation** to see new events being generated. Look for the Key field 54321 and notice the new value for the FirstName been collected



* Finally have a look to the account attributes for the related user:



2.12 Extending the functionality of connector

2.12.1 Provisioning validation and reconciliation transformation

**Purpose**

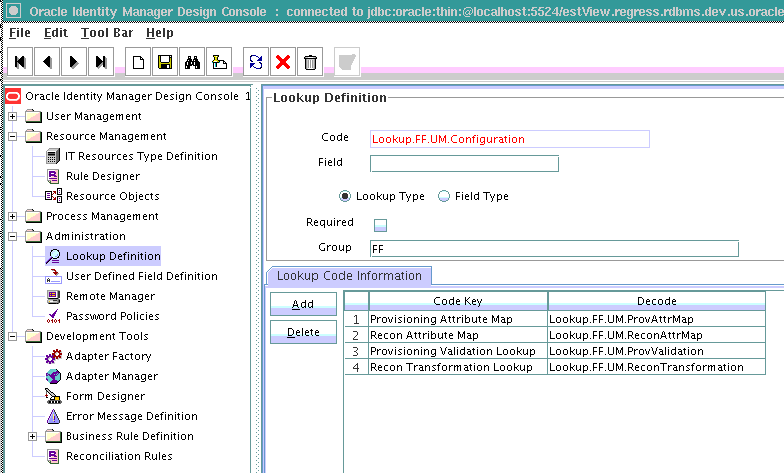
This section includes steps to extend the functionality of the connector by including the validation and transformation logic.

**Steps**

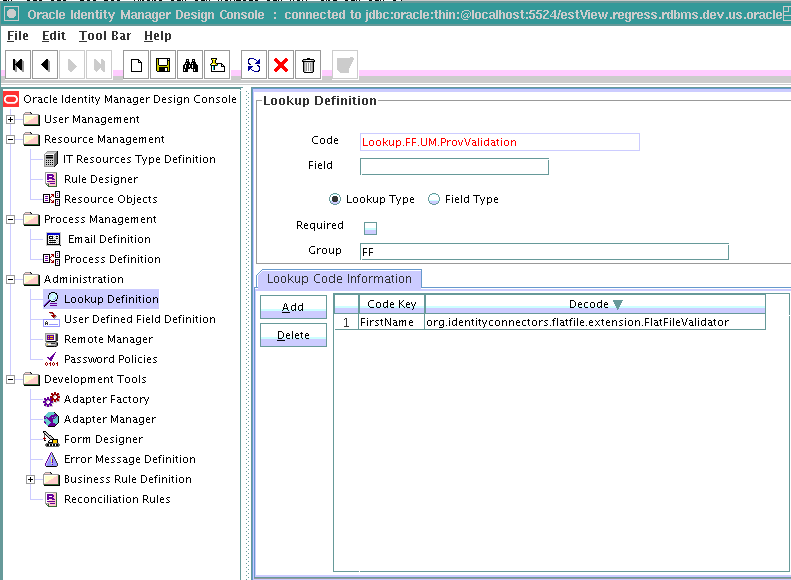
* Login to OIM design console, Open Lookup Definition under Administration
* Search for **Lookup.FF.UM.Configuration**
* Enter **Code Key = Provisioning Validation Lookup** and **Decode = Lookup.FF.UM.ProvValidation**
* Enter **Code Key = Recon Transformation Lookup** and **Decode = Lookup.FF.UM.ReconTransformation**
* Create a new lookup **Lookup.FF.UM.ProvValidation** and enter **Code Key = FirstName** and **Decode = org.identityconnectors.flatfile.extension.FlatFileValidator**
  + Code Key represents the form filed label on which validation needs to be done.
  + Decode represents the Java class where validation logic is present.

* Create a new lookup **Lookup.FF.UM.ReconTransformation** and enter **Code Key = FirstName** and **Decode = org.identityconnectors.flatfile.extension.FlatFileTransformation**
  + Code Key represents the form filed label on which transformation needs to be done.
  + Decode represents the Java class where transformation logic is present.

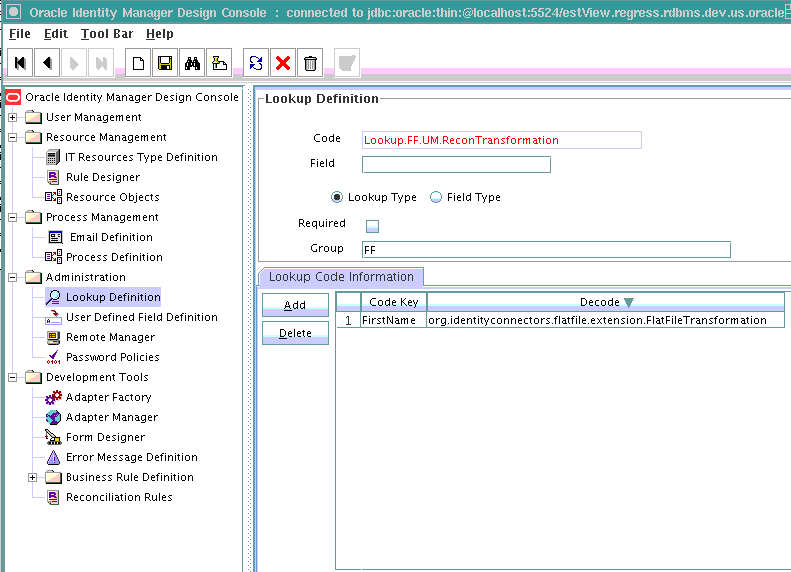
Object configuration lookup updated with validation and transformation lookup.



Provisioning Validation Lookup



Recon Transformation Lookup



* Update the OIM database with the **FlatFileValidationAndTransformation.jar** as created in lab1 using UpdateJars.sh script:
* cd /app/Middleware/Oracle\_IDM1/server/bin
* export APP\_SERVER=weblogic
* export DOMAIN\_HOME=/app/Middleware/user\_projects/domains/iam\_domain/
* Enter
* Xellerate admin username: xelsysadm
* Admin password: Oracle123
* Server URL: t3://identity.oracleads.com:14000
* Context: weblogic.jndi.WLInitialContextFactory
* Type 4 (ICFBundle)
* Enter jar path: /app/home/oracle/FlatFileValidationAndTransformation.jar

**Don't forget to push the jar there from your JDeveloper working directory**

* Answer n not to load another jar

The result should be: Upload jar executed successfully

* **Restart OIM**

Perform provisioning and reconciliation operation in OIM to see this logic being called.

2.12.1.1 Testing the provisioning validation

* Request an account for a new user
* Result:

Running CREATEUSER

Target Class = oracle.iam.connectors.icfcommon.prov.ICProvisioningManager

**Inside validate method...**

2.12.1.2 Testing the reconciliation transformation

* Change the FirstName value in output.txt and run Flat File Reconciliation scheduled task
* Result:

**Inside flat file recon transformation**

**Returning same value for attribute FirstName**

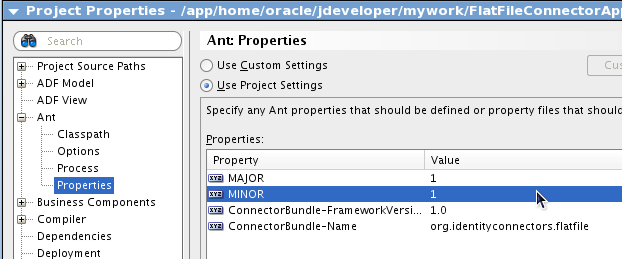
2.12.2 CLASSPATH isolation

**Purpose**

To show ICF eliminates CLASSPATH issues when using two different versions of same connector.

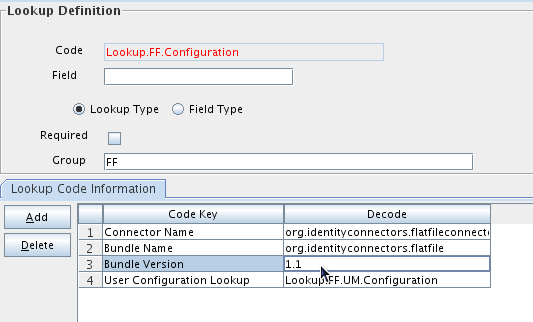
**Steps**

* Create another bundle version; let’s say **org.identityconnectors.flatfile-1.1.jar** *(1.1 version)*.
* Update the Ant Property MINOR value to 1



Notice the change in MANIFEST.XML when you run Ant (right-click on build.xml -> Run Ant Target -> All): ConnectorBundle-Version: 1.1 and also in Resources the new jar org.identityconnectors.flatfile-1.1.jar

* Add this JAR into bundles directory of the remote connector server and re-start it.
* Create all the above OIM metadata as created from section 2.1 to 2.12.3 except for below change.
  + Section 2.1.3 describes how to create configuration lookup, here we will need to provide ***1.1***  for ***Code Key = Bundle Version***



* Clear cache

Test by using the new connector version for provisioning and reconciliation operations.

2.13 Packaging the flat file connector

**Purpose**

This section involves creating a flat file connector installer package.

**Steps**

* Create a directory FlatFile in any suitable location
* Create a sub directory called configuration
* Create a new xml by name FlatFile-CI.xml and enter the following information.

<?xml version="1.0" encoding="UTF-8"?>

<!--

~ Copyright (c) 2011, Oracle and/or its affiliates. All rights reserved.

-->

<connector orderid="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:noNamespaceSchemaLocation=".\ConnectorSchema.xsd">

<connector-name>Flat File Connector</connector-name>

<connector-version>1.0</connector-version>

<filecopy>

<destination folder="bundles">

<file>org.identityconnectors.flatfile-1.0.jar</file>

</destination>

</filecopy>

<configuration>

<source folder="xml">

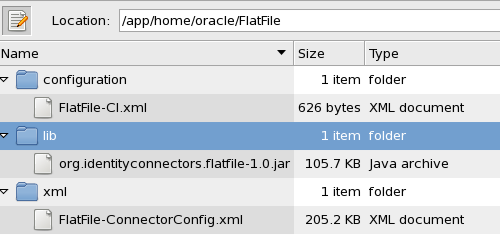
<file>FlatFile-ConnectorConfig.xml</file>

</source>

</configuration>

</connector>

* Save and close
* Create another subdirectory called xml under main directory FlatFile
* Login to OIM Sysadmin web console.
* Click on Export Deployment Manager File
* Select all the OIM artifacts (Adapters, Forms etc…) created in this lab and export it to FlatFile-ConnectorConfig.xml
* Save the exported file in the xml directory.
* Create another sub directory lib and copy org.identityconnectors.flatfile-1.0.jar into it.



Create a zip file of the FlatFile directory to finalize the creation of the FlatFile connector install package.