**Lab 2: 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
   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
       2. Recon transformation
       3. 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.

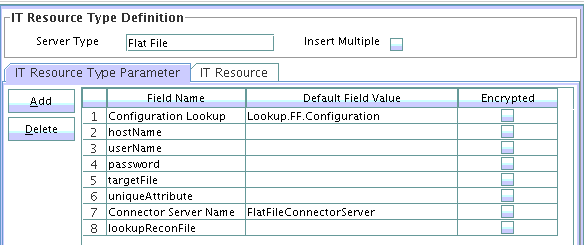
1. **Contents**
   1. Creating common Metadata for provisioning and reconciliation
      1. IT Resource type definition

**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**
* **IT Resource Type Definition** should look like below.

****

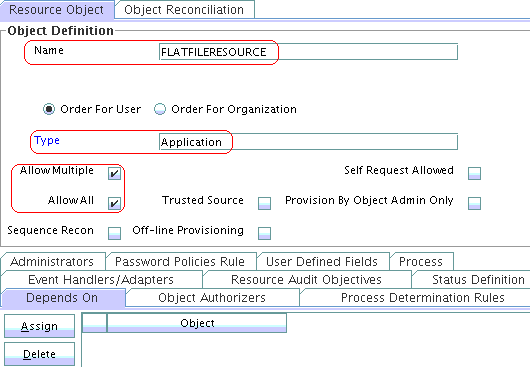
2.1.2 Resource Object

**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**
* Check **Allow Multiple** and **Allow All**
* Click **Save**

****

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

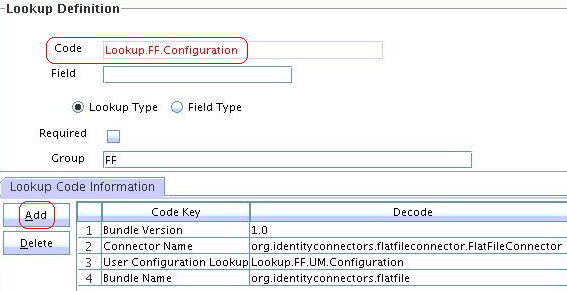
2.1.3 Configuration Lookup

**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



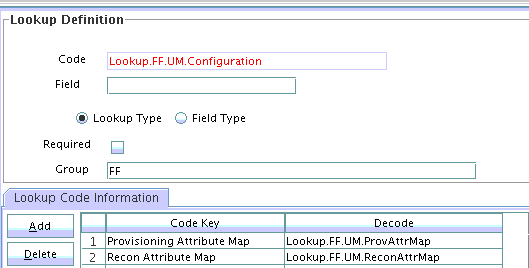
2.1.4 Object Configuration Lookup

**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.

****

2.1.5 Process Form

2.1.5.1 Child Form

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

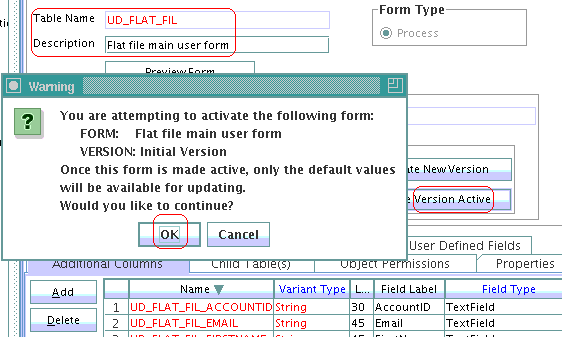
2.1.5.2 Parent Form

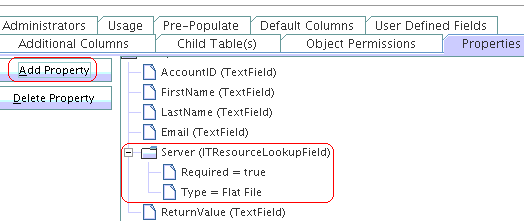
**Purpose**

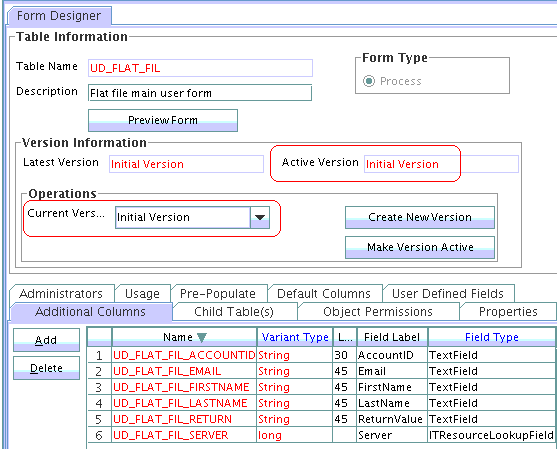
This step includes creating the parent process form.

**Steps**

* Open **Form Designer** under **Development Tools**
* Enter **UD\_FLAT\_FIL** 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 **Properties** tab
  + Add below Properties for **Server(ITResourceLookupField)**
    - **Required = true**
    - **Type = Flat File**
* Click **Make Version Active** and click **Ok** for the prompt
* Click **Save**







2.1.6 Adapters – Parent form

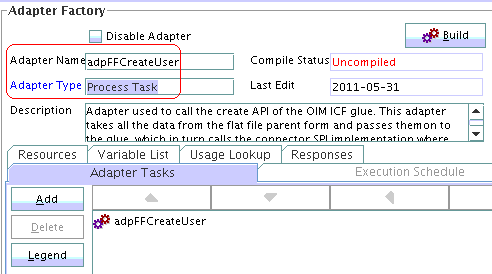
2.1.6.1 Create adapter

**Purpose**

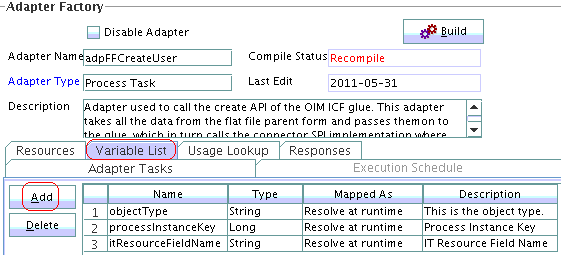
This step includes creating adapter for create provisioning operation. This adapter is responsible to take all the data from parent form and call the OIM ICF glue (icf-oim-intg.jar).

**Steps**

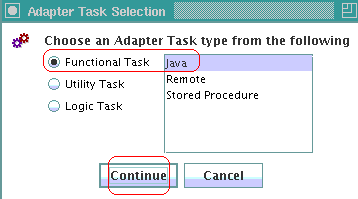
* Under **Development tools**, click **Adapter Factory**
* Enter **adpFFCreateUser** for **Adapter Name**
* Enter Process Task for **Adapter Type**
* Enter ‘Adapter used to call the create API of the OIM ICF glue. This adapter takes all the data from the flat file parent form and passes them on to the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

****

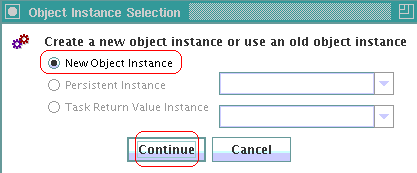
* Click **Variable List** tab
* Click **Add** and add the following variables
  + **Name – objectType, Type – String, Map To – Resolve at runtime**
  + **Name – processInstanceKey, Type – Long, Map To – Resolve at runtime**
  + **Name – itResourceFieldName, Type – String, Map to – Resolve at runtime**

****

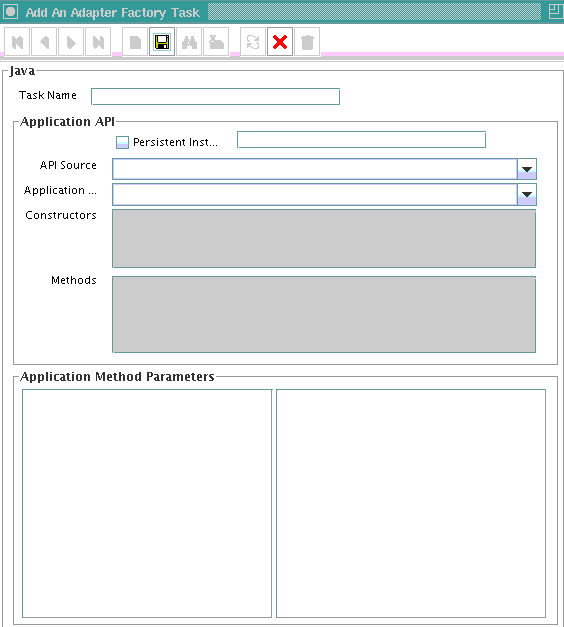
* Click **Adapter Tasks** tab
* Select adpFFCreateUser and click **Add** to see below pop up



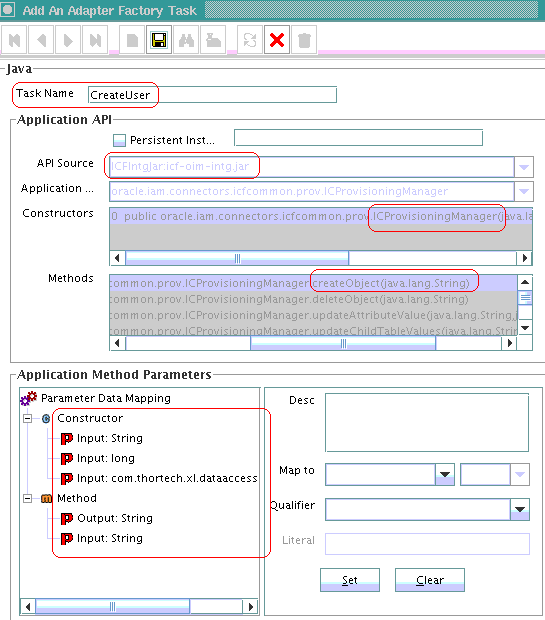
* Click **Continue** to see next screen as shown below

****

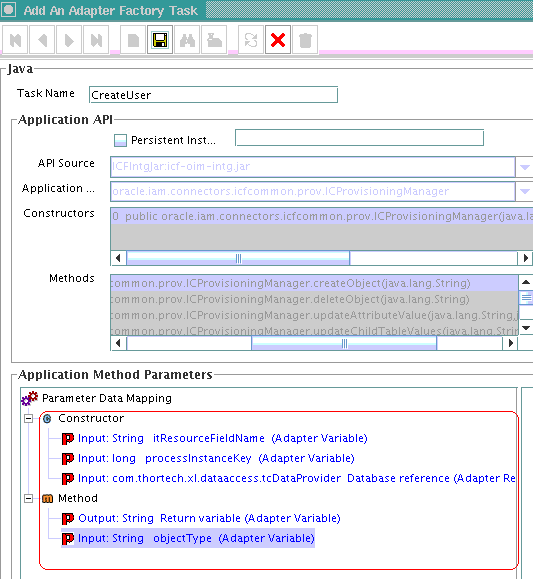
* Click **Continue** to see next screen as shown below



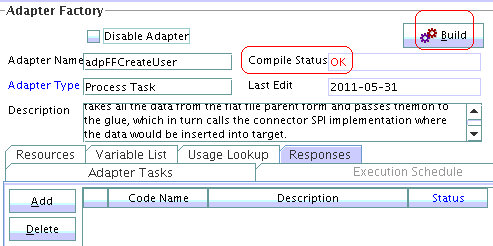
* Enter CreateUser for **TaskName**
* Select **ICFIntgJar:icf-oim-intg.jar** from the dropdown for **API Source**
* Select **oracle.iam.connectors.icfcommon.prov.ICProvisioningManager** from the dropdown for **Application…**
* Select **createObject** for Methods
* Click **Save** to see the below screen



* 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** to **Adapter References, Database Reference**
  + Method
    - **Output: String** to **Adapter Variables, Return Variable**
    - **Input: String** to **Adapter Variables, objectType**

****

* Click **Build** to see the C**ompile Status Ok**
* Click **Save**

****

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**

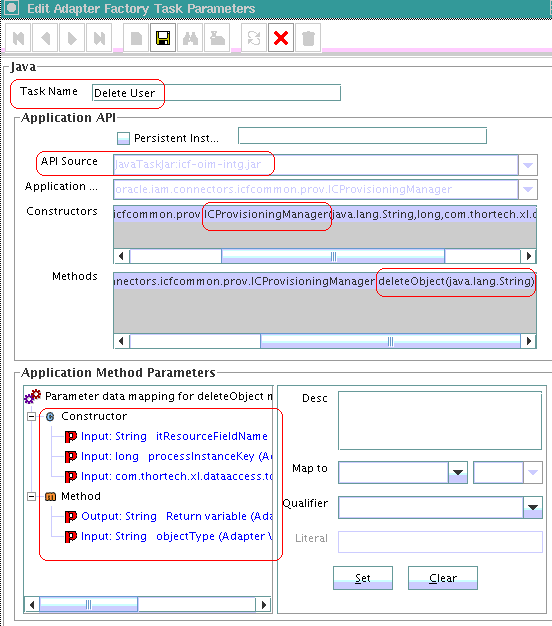
* Under **Development tools,** click **Adapter Factory**
* Enter **adpFFDeleteUser** for **Adapter Name**
* Enter **Process Task** for **Adapter Type**
* Enter ‘Adapter used to call the delete API of the OIM ICF glue. which in turn calls the connector SPI implementation where the data would be deleted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

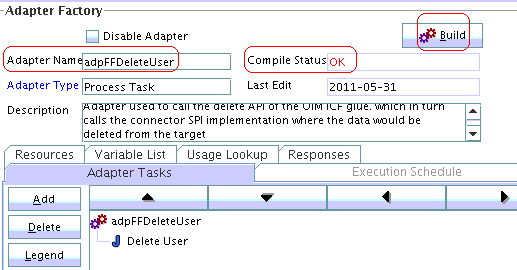
The steps to create the **Adapter Variables** are same as shown in section 2.1.6.1. Please follow the same.

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

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

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





2.1.6.3 Enable adapter

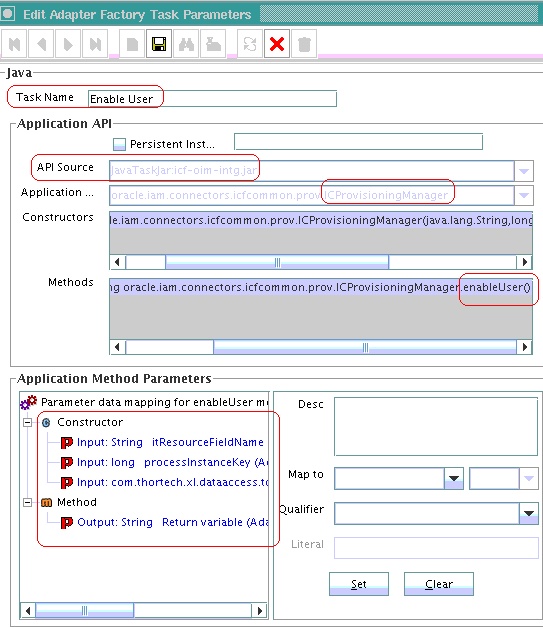
**Purpose**

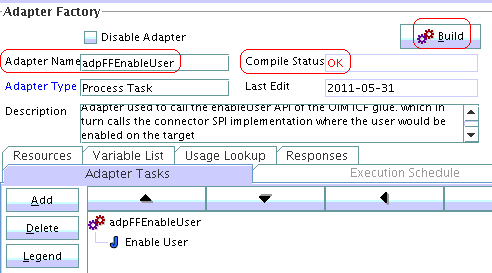
The steps include creating an adapter for enable operation. This adapter calls the **enableUser** API of the ICF OIM glue.

**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, Map To – Resolve at runtime**
  + **Name – itResourceFieldName, Type – String, Map to – Resolve at runtime**
* Create the **Adapter Task** calling the **enableUser** API of the **ICProvisioningManager.**
* **Build** and **Save** the adapter.

Final screen shots should look like below:-





2.1.6.4 Disable Adapter

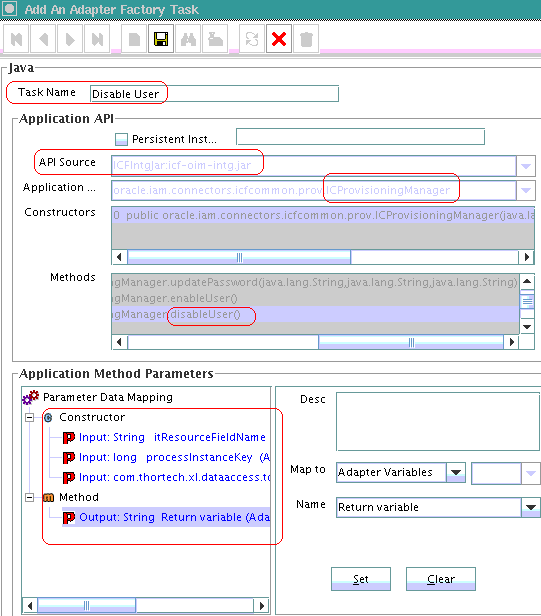
**Purpose**

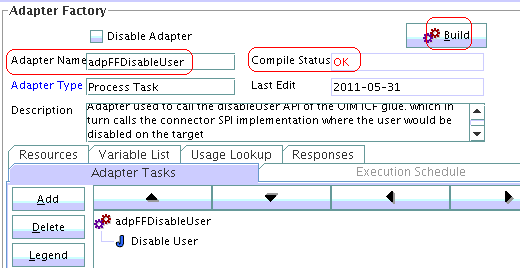
This section includes creating an adapter for disable operation. This adapter is calls the disableUser API of the OIM ICF glue.

**Steps**

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

Final screen shots should like below:-





2.1.6.5 Update adapter

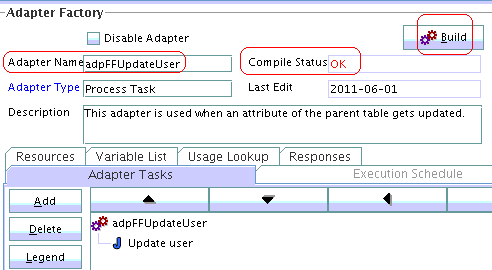
**Purpose**

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, Map To – Resolve at runtime**
* **Name – itResourceFieldName, Type – String, Map to – Resolve at runtime**
* **Name – objectType, Type – String, Map to – Resolve at runtime**
* **Name –** **attrFieldName, Type – String, Map to – Resolve at runtime**
* Create the **Adapter Task** calling the **updateAttributeValue** API of the **ICProvisioningManager.**
* **Build** and **Save** the adapter.

****

****

2.1.7 Process Definition and Process tasks – Parent form

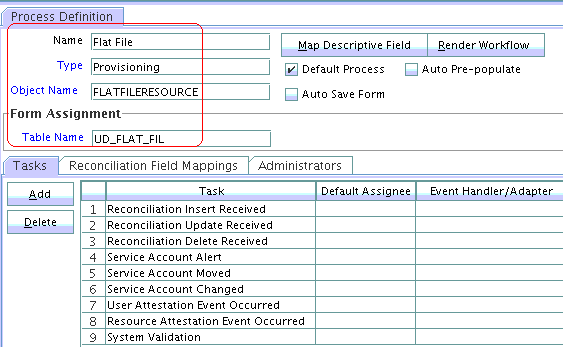
2.1.7.1 Flat File Process Definition

**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**

****

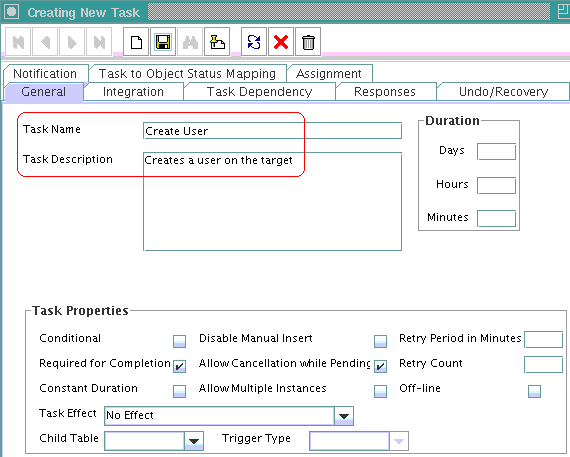
2.1.7.2 Create Process Task

**Purpose**

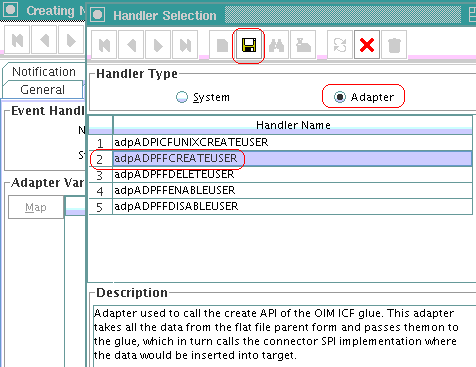
This section includes steps to create a process task for create operation.

**Steps**

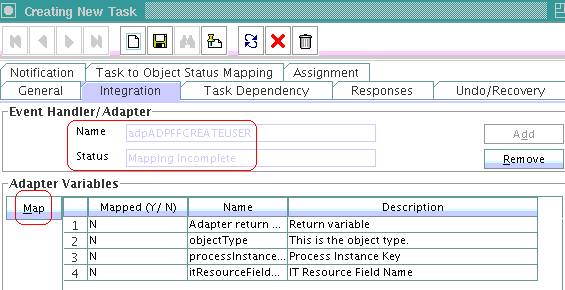
* Open **Process Definition** under **Process Management**
* Search for **Flat File**
* Under **Tasks** tab, click **Add** to see the pop up as shown below:-
* Enter **Create User** as **Task Name**
* Enter **Creates a user on the target** for **Task Description**
* Click **Save**



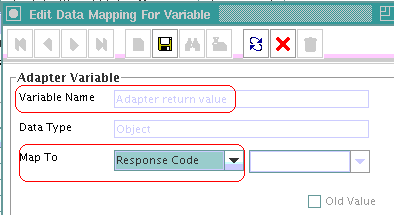
* Select **Integration** tab, click **Add** to see **Hanlder Selection** screen, select **Adapter** to see a list of adapters available as shown below. Select adpADPFFCREATEUSER



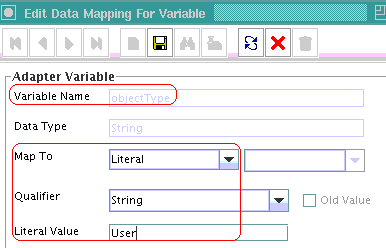
* Click **Save** and **Ok**



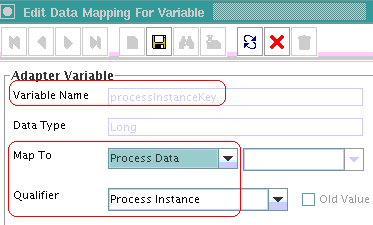
* Select each **variable** and map them by clicking **Map** on each of the variable as shown below
  + Adapter Return Variable – Response Code



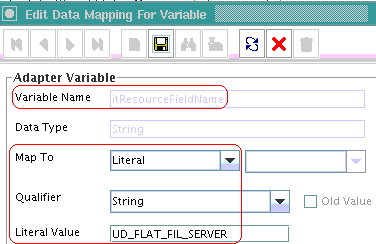
* + Object Type – [Literal:String] User (Name of the object type)



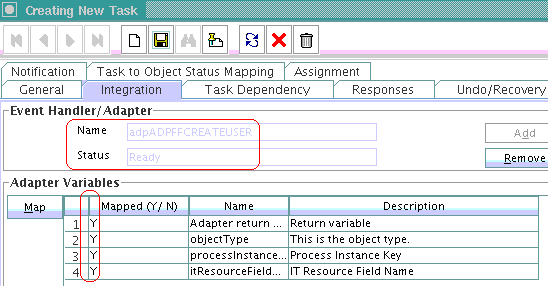
* + Process Instance Key – [Process Data] Process Instance



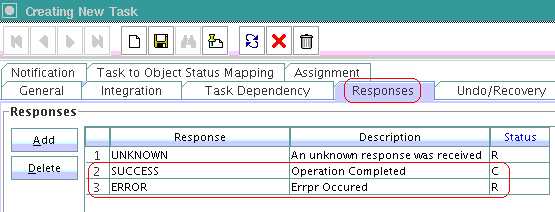
* + IT Resource Field Name – [Literal:String] UD\_FLAT\_FIL\_SERVER (Form field name that contains the IT resource information)

. 

* + Screen shot should look like below



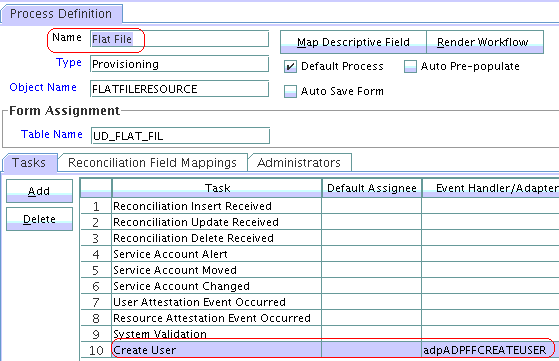
* Click on **Responses** tab and add 2 more responses as shown below:-



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



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



2.1.7.3 Delete Process Task

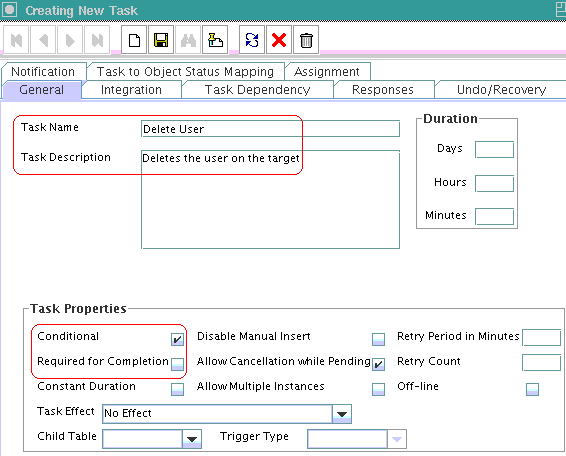
**Purpose**

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

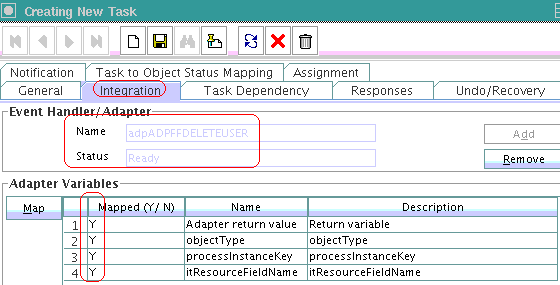
**Steps**

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**

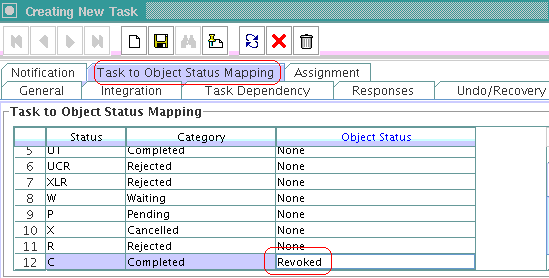
****

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

****

**Note: Map the variables accordingly**

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

****

* Open and update ‘Create User’ task with ‘Delete User’ task being the Undo task of ‘Create User’ task.

2.1.7.4 Enable Process Task

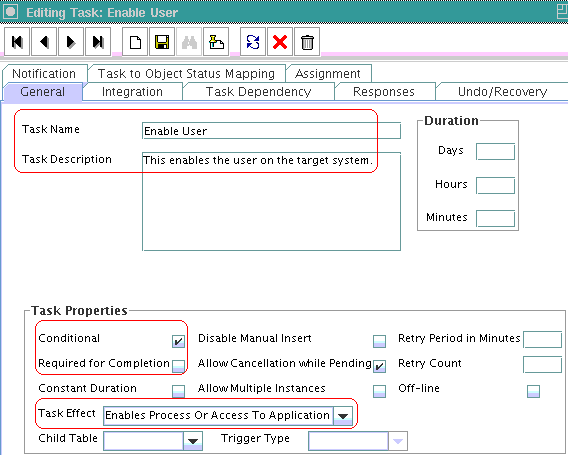
**Purpose**

This includes creation of a process task for enable operation.

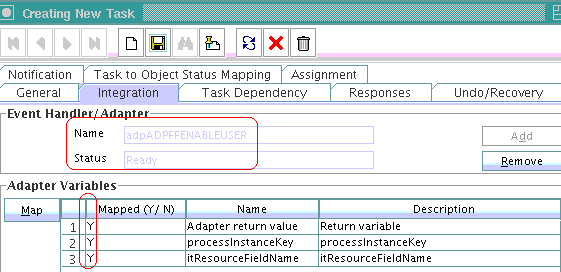
**Steps**

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

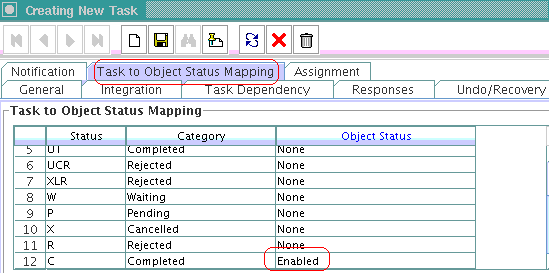
* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Task Effect** should be **Enables Process or Access to application**



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



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



2.1.7.5 Disable Process Task

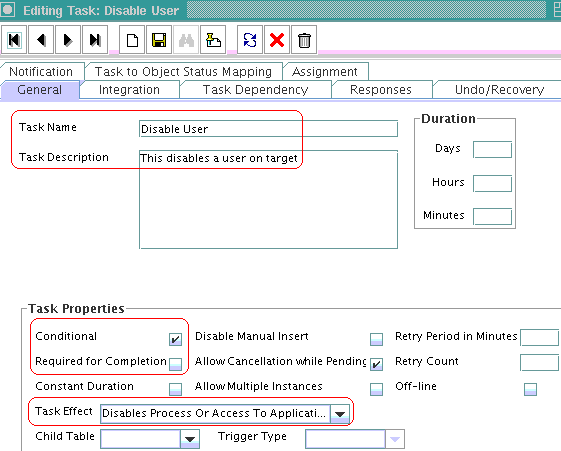
**Purpose**

This includes creation of a process task for disable operation.

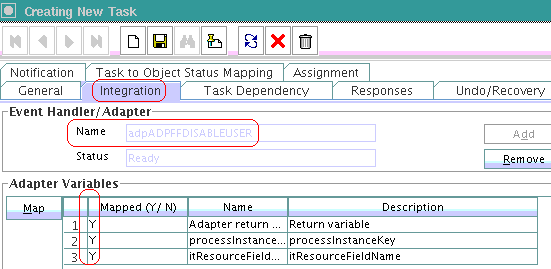
**Steps**

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

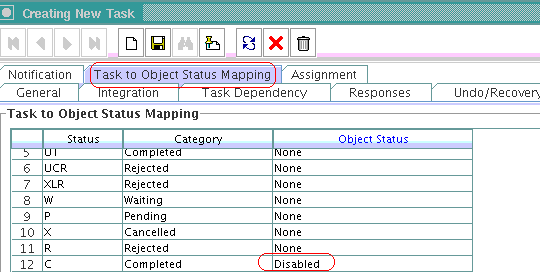
* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **Task Effect** should be **Disables Process or access to application**



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



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



2.1.7.6 Update process task

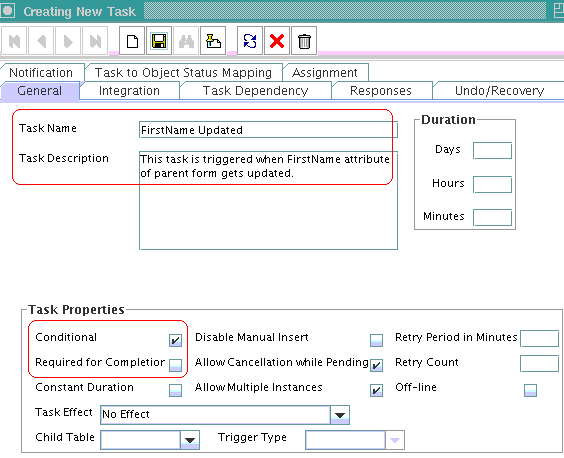
**Purpose**

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 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 is map
  + **attrFieldName** to **FirstName**
* Click **Save**

2.2 Creating provisioning Meta data

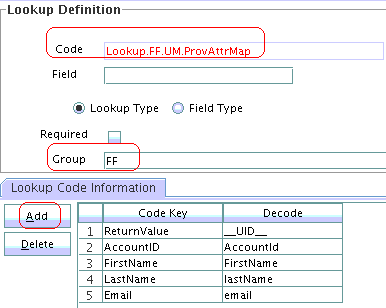
2.2.1 Provisioning attribute map lookup

**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**

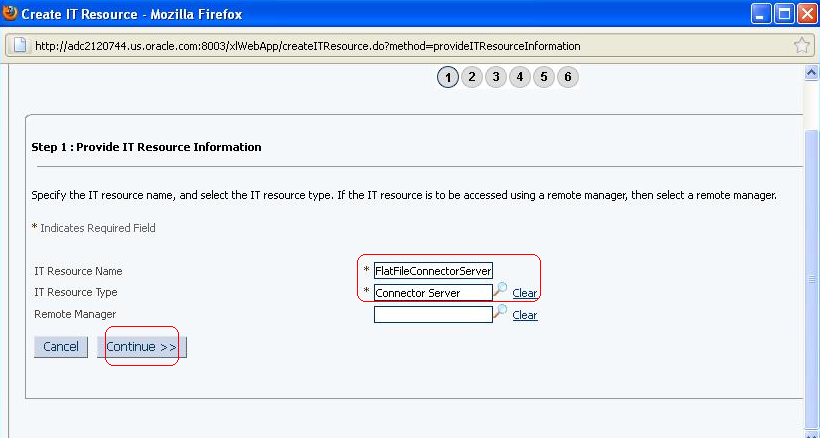
This step includes creating an IT resource of type connector server in OIM.

**Steps**

* Log in to the Administrative and User Console by using the **xelsysadm** account.
* Navigate to the **Advanced Administration console**
* Click on **Create IT Resource**



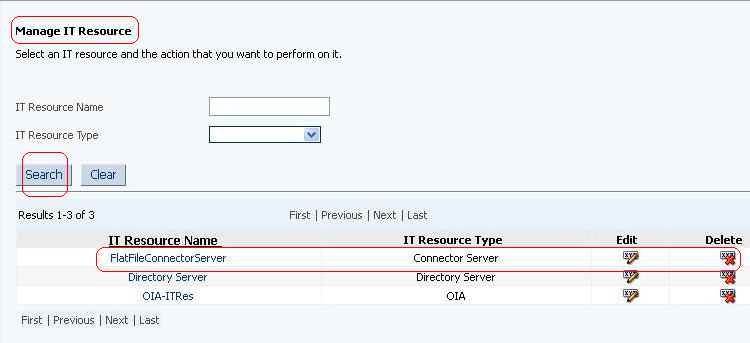
* Enter **FlatFileConnectorServer** for **IT Resource Name**
* Select **Connector Server** for **IT Resource Type**
* Click **Continue >>**



* Enter the **IP address** of the Host where connector server is running
* Enter **Key**
* Enter **Port**
* Click **Continue >>**



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



2.2.3 Flat file IT resource

**Purpose**

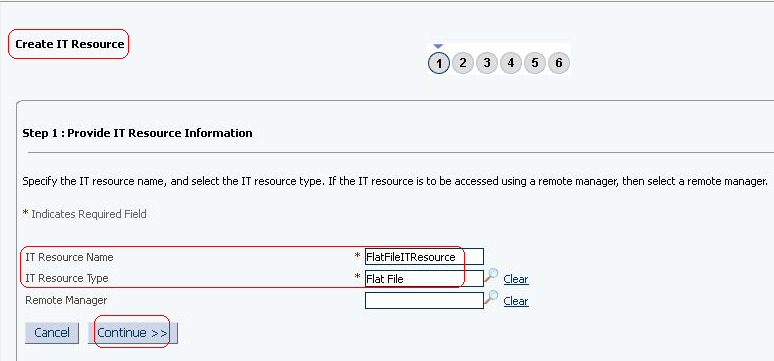
This step involves creating an IT resource of type flat file in OIM.

**Steps**

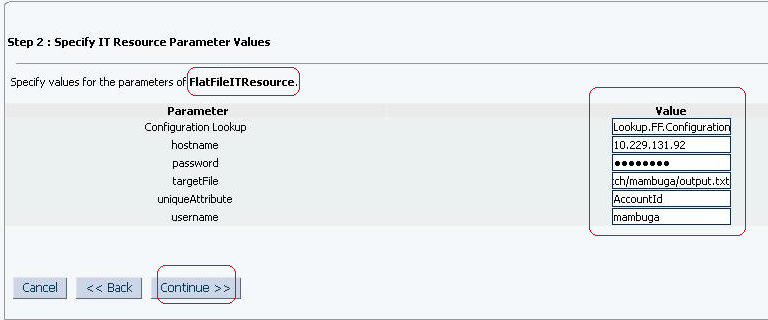
* Log in to the Administrative and User Console by using the **xelsysadm** account.
* Navigate to the **Advanced Administration console**
* Click on **Create IT Resource**



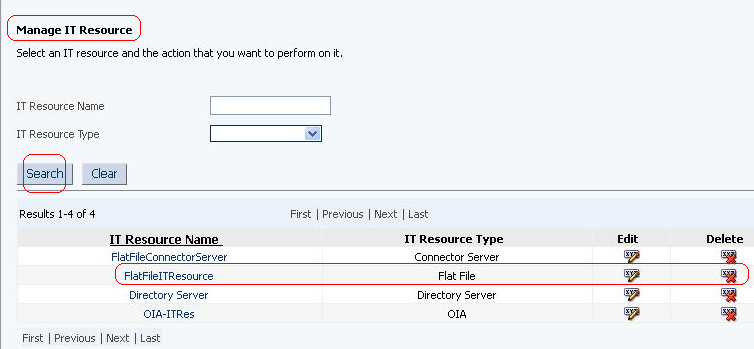
* Enter **FlatFileITResource** for **IT Resource Name**
* Select **Flat File** for **IT Resource Type**
* Click **Continue >>**



* Enter IP address of the target machine for hostname
* Enter password to be used to connect to target system
* Enter targetFile to be used for prov/recon operations
* Enter Unique attribute as **AccountId**
* Enter username to be used to connect to target machine.
* Enter Connector Server Name as **FlatFileConnectorServer**



* Search IT resources on **Manage IT resource** screen, it should look like below:-



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 set up look up recon. It would read different roles available on the target system and updates the lookup. This 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 populate the lookup used in child form.

**Steps**

* Create a new task xml with entry as shown below and save it as FFLookupRecon.xml

<?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>

* Upload this new xml in MDS using weblogicImportMetadata.sh available at $DW\_HOME/server/bin/. Follow below steps
  + Execute sh weblogicImportMetadata.sh
  + Enter weblogic username
  + Enter weblogic password
  + Enter server URL for example t3://localhost:8003, where 8003 is the port where OIM is running.
* Purge the cache by executing PurgeCache.sh all. Steps are outlined below.
  + Execute sh PurgeCache.sh all
  + Enter OIM admin username
  + Enter OIM admin password
  + Enter servive URL, for example t3://localhost:8003)

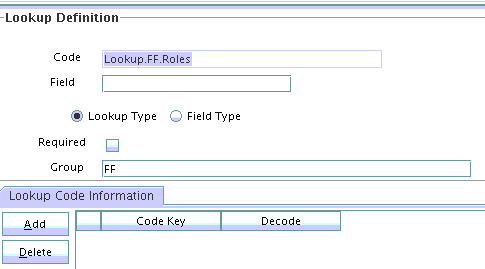
2.3.2 Create a new lookup

**Purpose**

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

**Steps**

* Log in 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**
* Under **Additional Columns,** add
  + **Name = UD\_FLAT\_CH\_ROLES, Variant Type = String, Length = 30, Field Label = Roles, Field Type = Lookup Fied and order = 1**
* Under **Properties** tab, select **Roles (Lookup field)** and **Add Property**
  + **Lookup Code = Lookup.FF.Roles**
* Click **Make Version Active** and accept the pop up

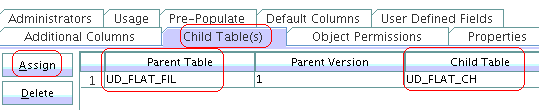
2.3.4 Update Parent Form (created in 2.1.5.2)

**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
      * Save
      * Select tab **Child Table(S),** click **Assign**
      * Select **UD\_FLAT\_CH,** click **Ok**
      * **Make Version Active** and click **Save**

****

2.4 Adapters – Child form

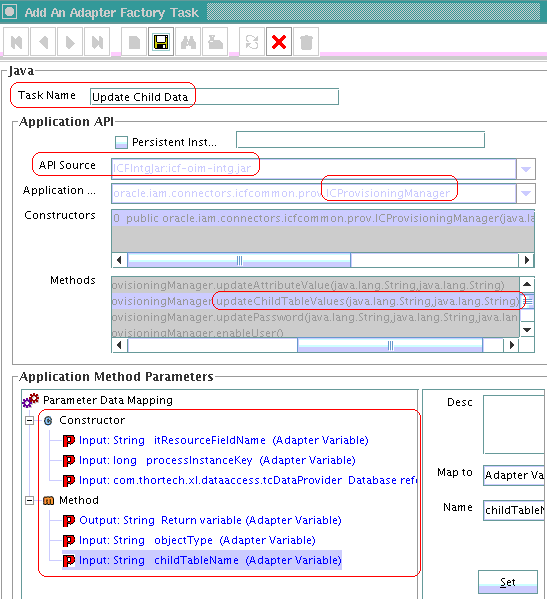
2.4.1 Create adapter for child table.

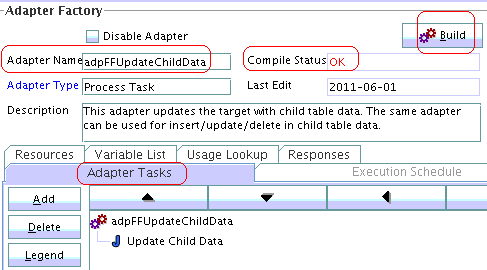
**Purpose**

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

**Steps**

* Under **Development tools,** click **Adapter Factory**
* Enter **adpFFUpdateChildData** for **Adapter Name**
* Enter **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
  + **Name – childTableName, Type – String, Map to – Resolve at runtime**
* **Build** the adapter and **Save.**
* The final screen shot of delete adapter should look like below:-





2.5 Process Tasks – Child table

2.5.1 Create Process Task for child table.

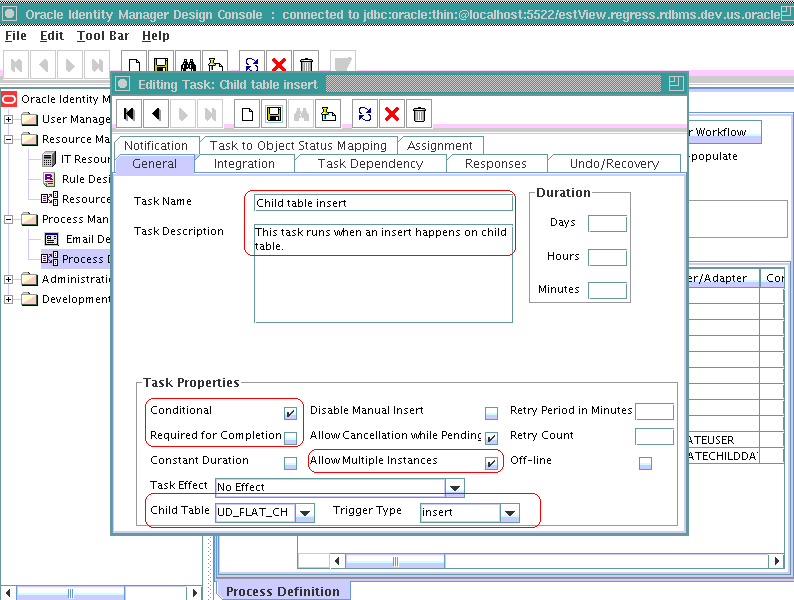
**Purpose**

This is step includes 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 except the following:-

* **Task Properties**
  + **Conditional** should be **checked**
  + **Required for Completion** should be **un-checked**
  + **UD\_FLAT\_CH** should be selected from the **Child Table** dropdown
  + **Trigger Type** should be **insert**
  + **Allow Multiple Instances** should be checked.

****

* Under **Integration** tab **Event Handler/Adapter** should be **adpADPFFUPDATECHILDTABLEDATA.**
* Mapping of adapter variables are same except for **childTableName.** This should be mapped to a String literal = **UD\_FLAT\_CH**

2.5.2 Update Process Task for child table

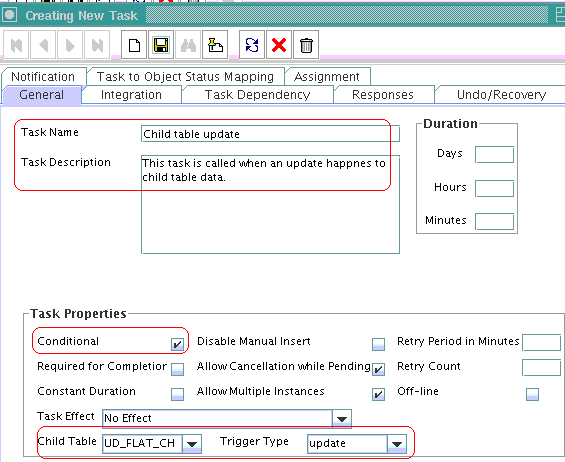
**Purpose**

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

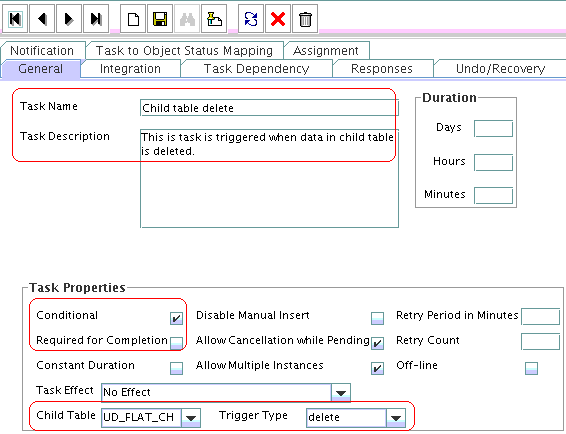
**Purpose**

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

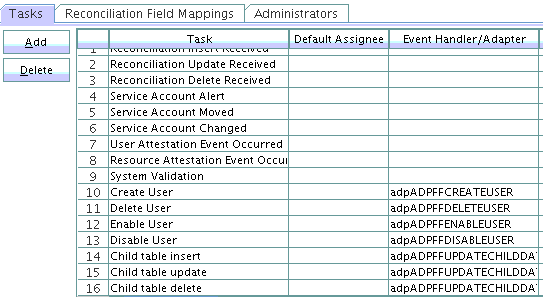
**Steps**

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

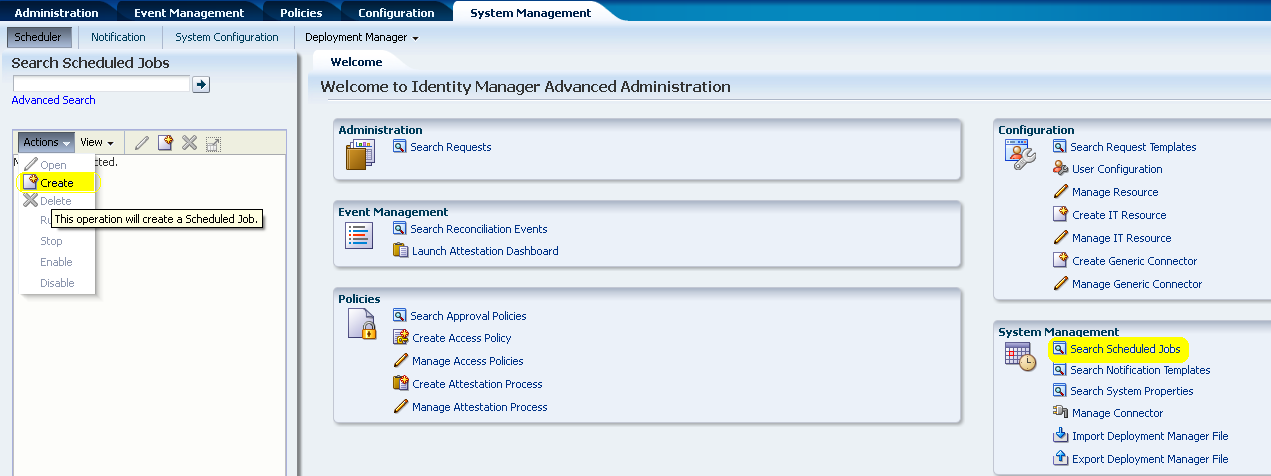
2.6.1 Create a scheduled job and run

**Purpose**

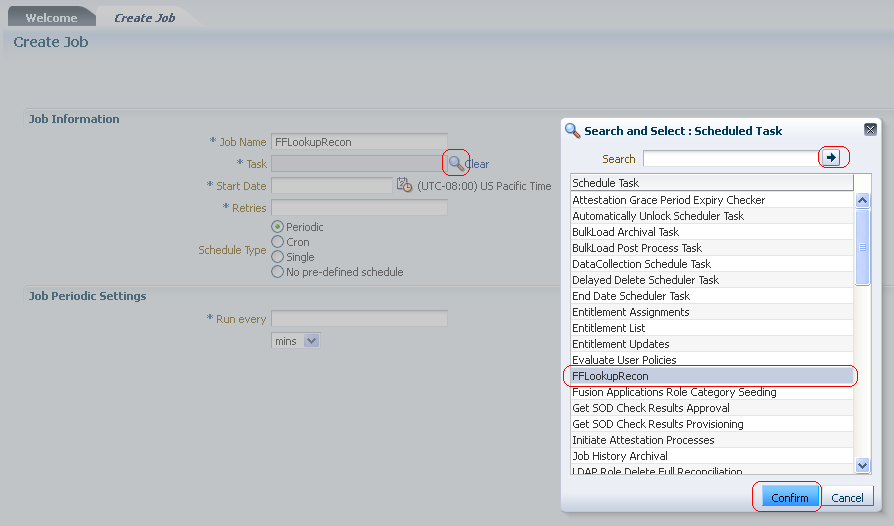
This step includes creating a scheduled job, which when run updates the lookup **Lookup.FF.Roles** created in 2.3.2. This lookup data would be used in the child form created in 2.3.3

**Steps**

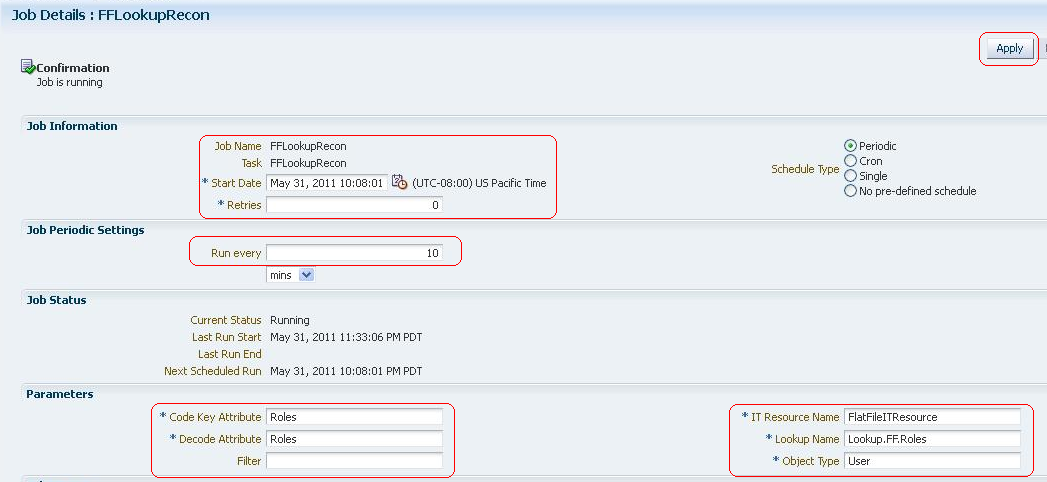
* Log in to the Administrative and User Console by using the **xelsysadm** account.
* Navigate to the **Advanced Administration console**
* Click on **Search Scheduled Jobs** and click on **Create** on from the **Actions** dropdown



* Enter **Job Name** as **FFLookupRecon**
* Click on **Task/Implementation** icon to see pop up **Search and Select: Scheduled Task,** click on **Search.**
* Select **FFLookupRecon,** click **Confirm**



* Fill information as shown below and click **Apply**

****



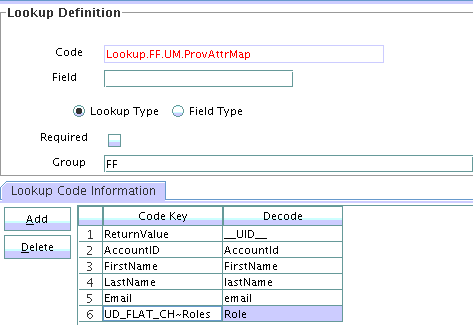
2.6.2 Update Provisioning attribute map lookup to include child form field label

**Purpose**

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

**Steps**

* 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**



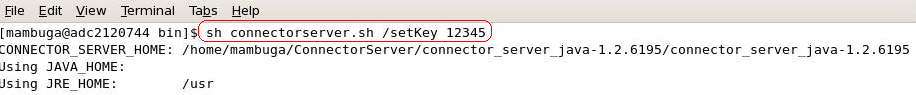
2.7 Setup remote connector server

**Purpose**

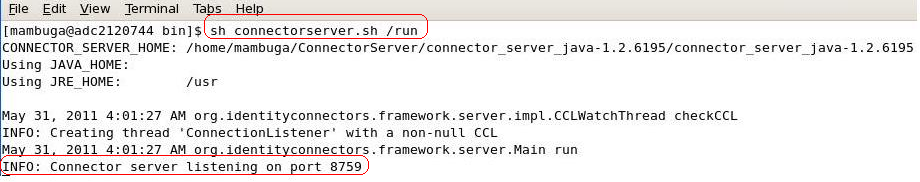
This step includes starting a remote connector server. ICF flat file bundle would be placed on this machine instead of OIM machine. Once the connector server is running, we could create an IT resource in OIM.

**Steps**

* Navigate to the bin directory of the connector server.
* Set a key as shown below



* Run the Connector server as shown below



Running 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** to run the scheduled job.
* Login to design console and search for the lookup **Lookup.FF.Roles** and verify that roles have been populated.

2.8 Test provisioning

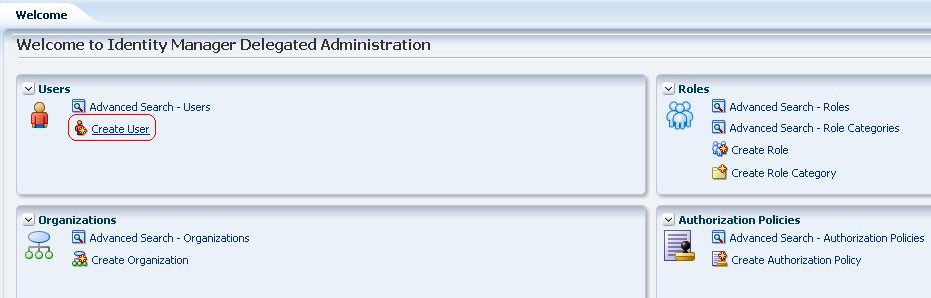
2.8.1 Create User (with child table data)

**Purpose**

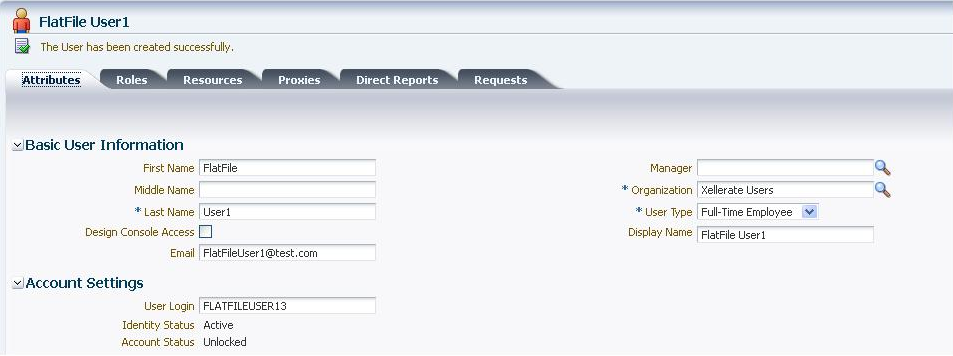
Steps include creating a user on the target flat file to test the flat file bundle code and the corresponding OIM metadata.

**Steps**

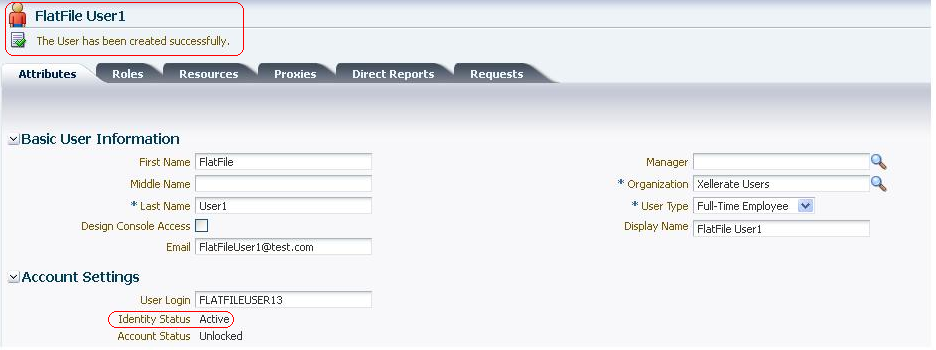
* Log into OIM administration console.
* Click on **Create user** link



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



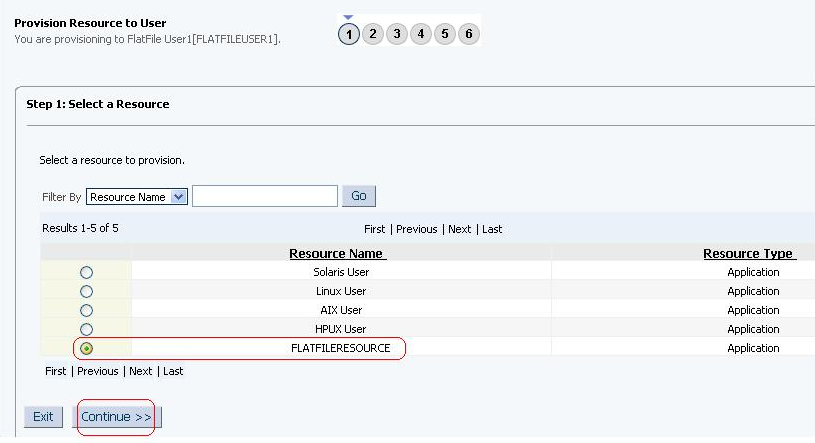
* Check the confirmation user has been created and click on **Resources** tab



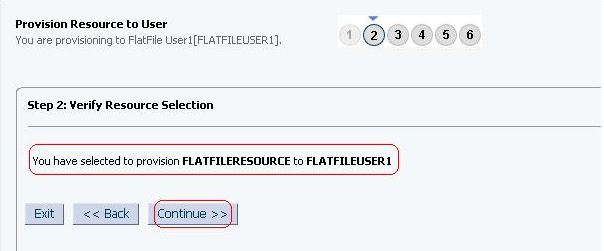
* Click on **Add**



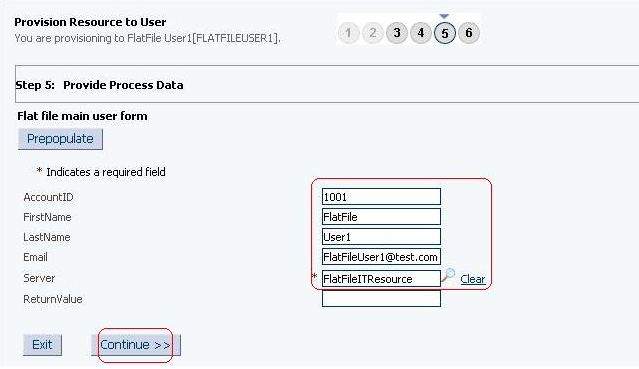
* Select **Resource Name – FLATFILERESOURCE**. Click **Continue**



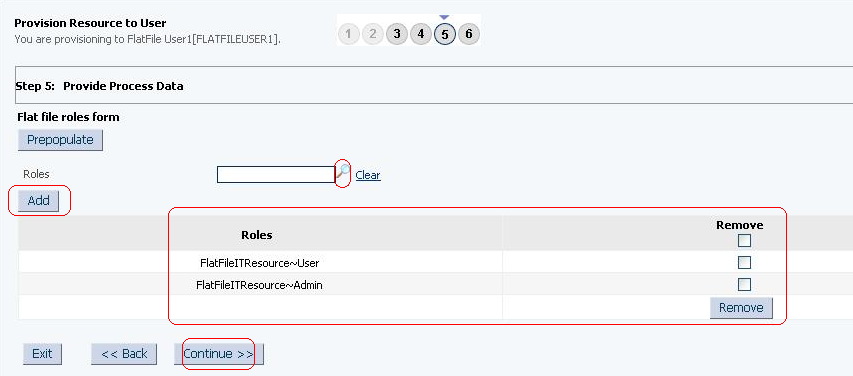
* Verify and Click **Continue**



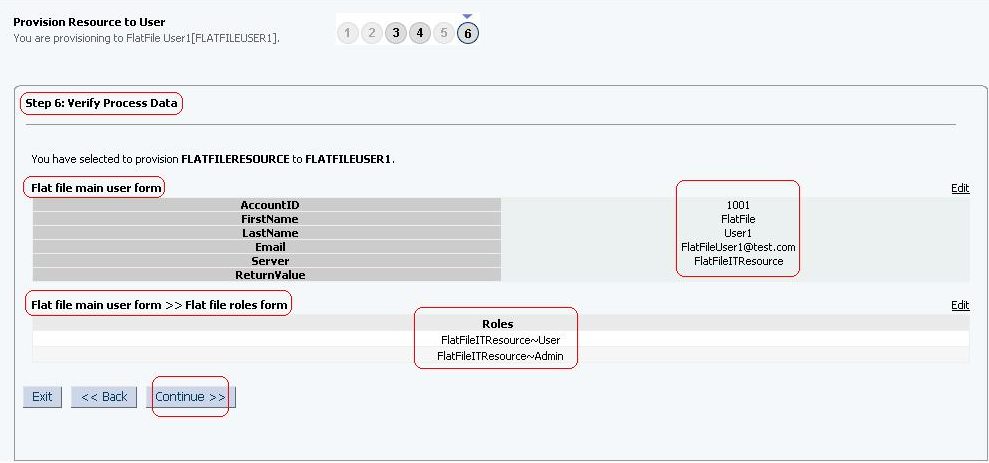
* Enter appropriate user information on the **Flat file main user form,** click **Continue**



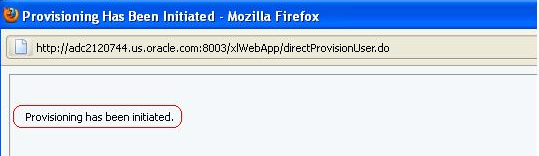
* Select roles and **Add** them as shown below, click **Continue**



* Verify and click **Continue** to see the create process happening.



* Verify to see the final screen



* Go back to user screen and click on **Refresh** button to see the status **Provisioned**



* Open the target file to see data being inserted as shown below



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**

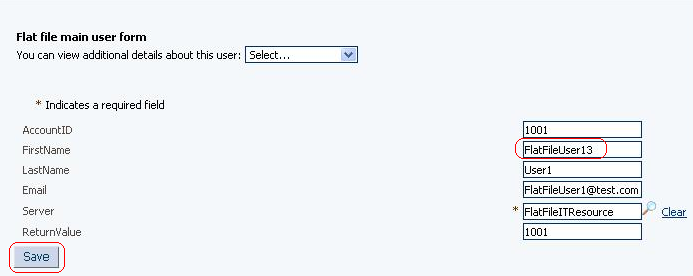
* On the same screen, select **Resource Name – FLATFILERESOURCE** and click on **Open**



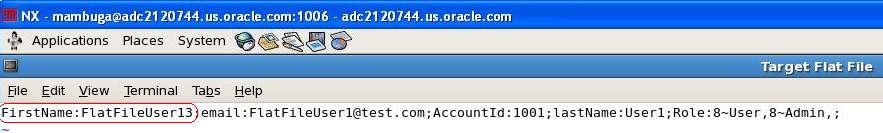
* Below screen shows the existing user information on target flat file.



* Update **FirstName** from **FlatFile** to **FlatFileUser13** and click on **Save**



* Open target flat file to see the updated **FirstName**

c

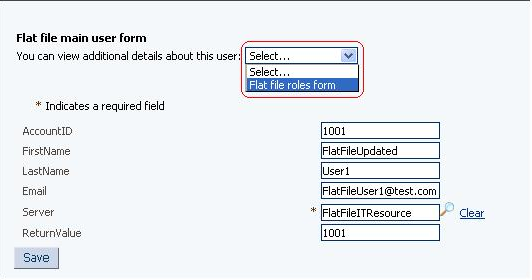
2.8.2.2 Child table data

**Purpose**

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

**Steps**

* Open the **Resource Details Form,** select **Flat file roles form** from the dropdown as shown below

****

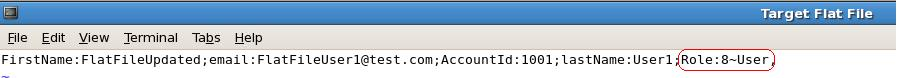
* Screen below displays existing details, check **Remove** for **Admin** role and click on **Remove** button

****

* Clicking **Remove** displays below screen, verify **Admin Role** being removed.

****

* Open target flat file to see, **Admin** role being removed



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.

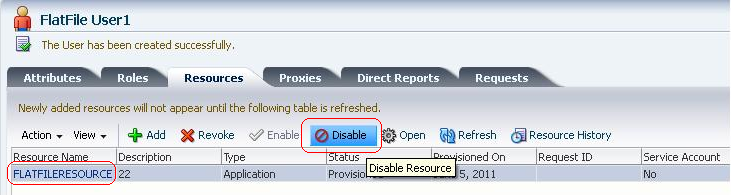
2.8.3 Disable user

**Purpose**

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

**Steps**

* On the **Resource Details** screen, select **Resource Name – FLATFILERESOURCE** and click on **Disable**



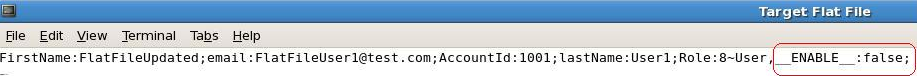
* Click **Ok** to confirm



* Verify the **Status** to see **Disabled**



* Open the target flat file to see status 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

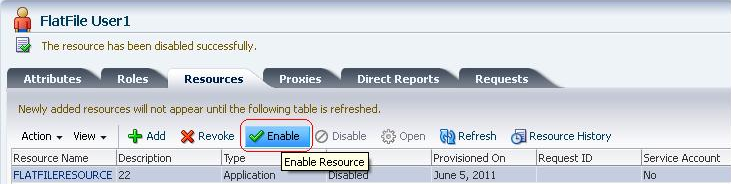
2.8.4 Enable User

**Purpose**

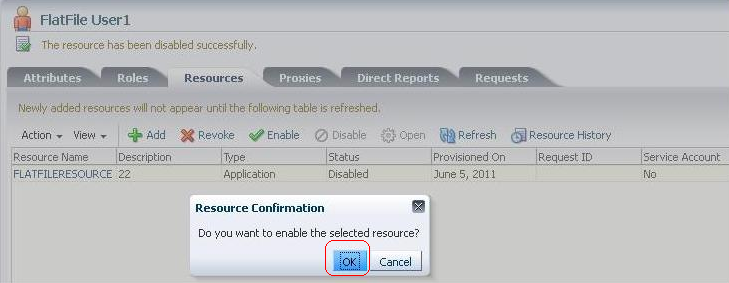
This section includes steps which enables the user on the target flat file.

**Steps**

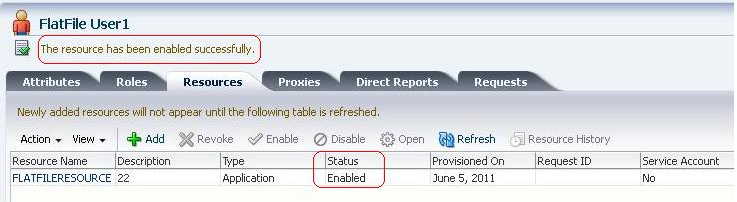
* On the **Resource Details** screen, select **Resource Name – FLATFILERESOURCE** and click on **Enable**



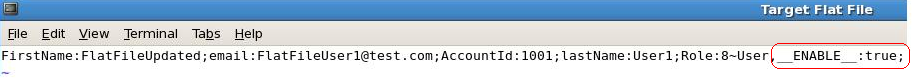
* Click **Ok** to confirm



* Verify the **Status** – **Enabled**



* Open the flat file to see the status



**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**

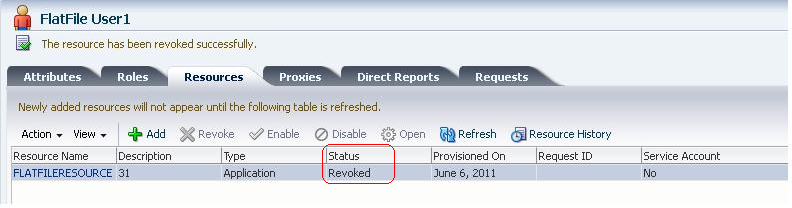
* On the **Resource Details** screen, select **Resource Name – FLATFILERESOURCE** and click on **Revoke**



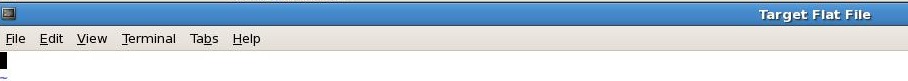
* Click **Ok** to confirm



* Verify the **Status – Revoked**



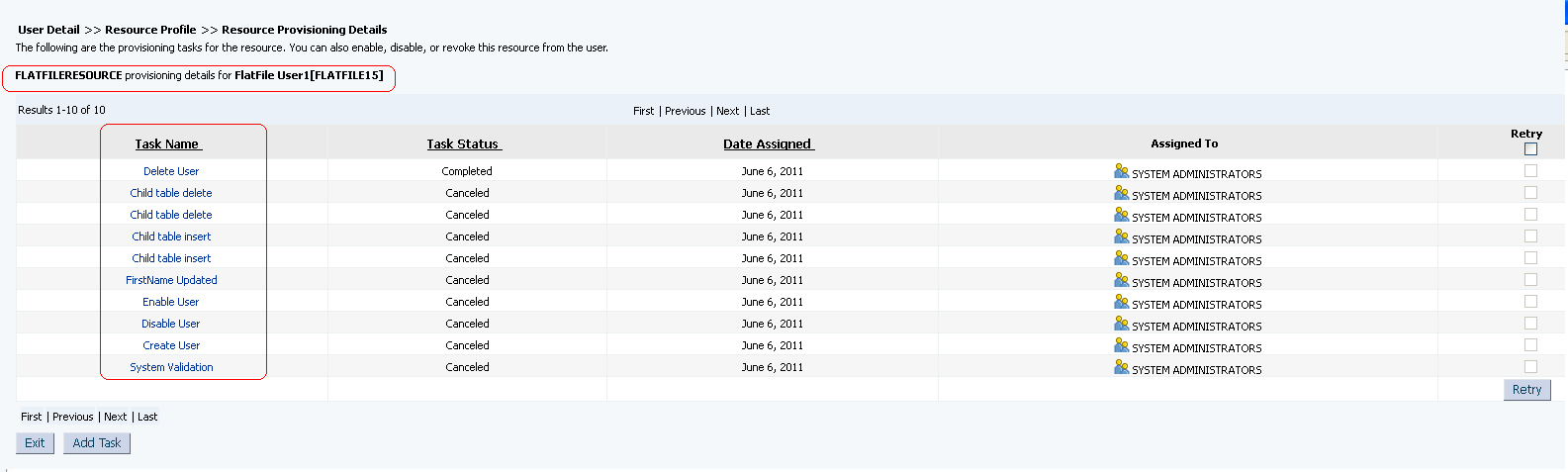
* Open the flat file to see entry has been deleted



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

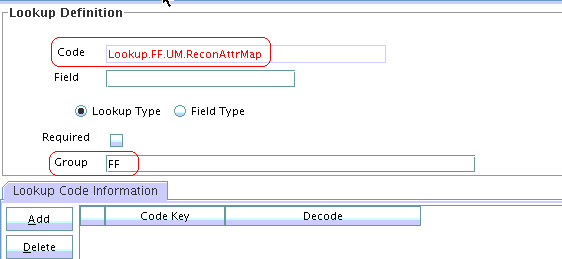
2.9.1.1 Reconciliation attribute map

**Purpose**

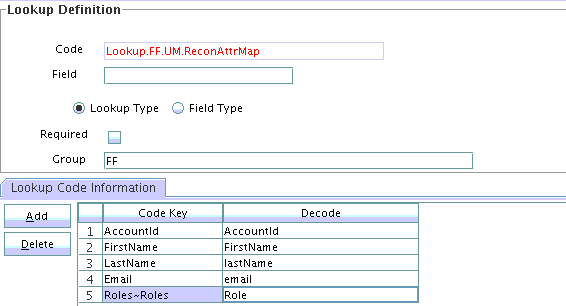
This section includes steps to creating the reconciliation attribute map. This map is referred by the OIM ICF glue during reconciliation.

**Steps**

* Login to OIM design console, Click on **Lookup Definition** under **Administration.**
* Enter **Lookup.FF.UM.ReconAttrMap** for **Code** and **FF** for **Group**
* Click **Save**



* Enter values as shown below



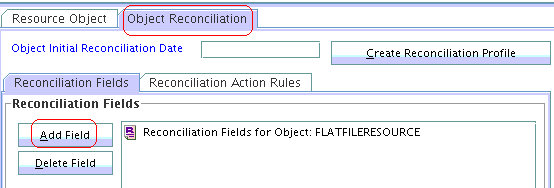
2.9.1.2 Reconciliation Profile

**Purpose**

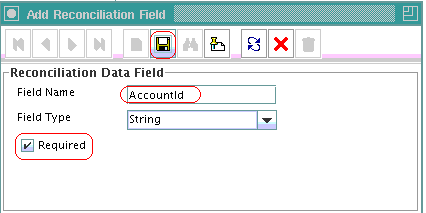
Create reconciliation profile

**Steps**

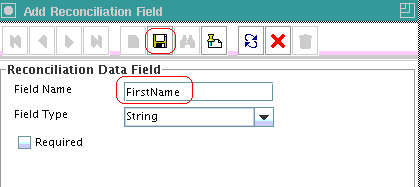
* Open **Resource Object** under **Resource Management**
* Search for **FLATFILERESOURCE** and click on **Object Reconciliation** tab

****

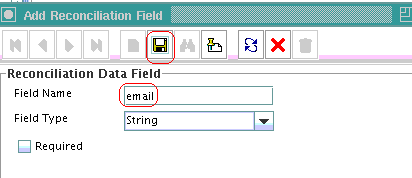
* Click **Add Field** and add **AccountId** and make it as **Required** and **Save**

****

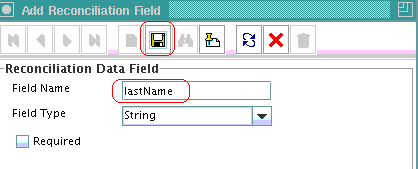
* Add **FirstName** and **Save**

****

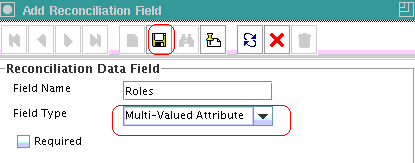
* Add **email** and **Save**



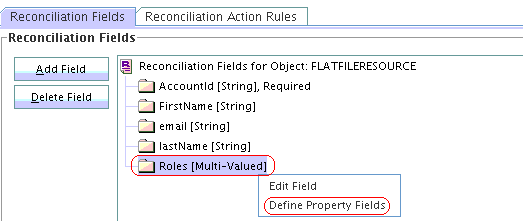
* Add **lastName** and **Save**



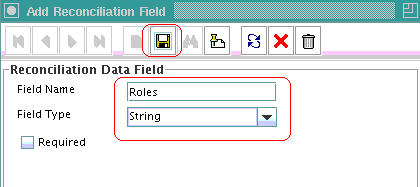
* Add **Roles** and **Save**



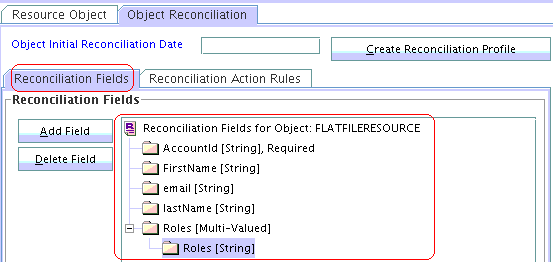
* Right click on **Roles** and select **Define Property Field**



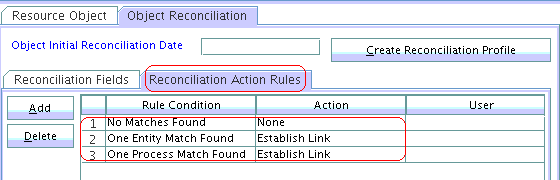
* Enter **Roles** as show



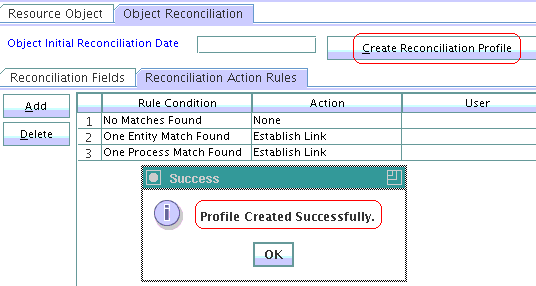
* Reconciliation Fields should look like below



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

****

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



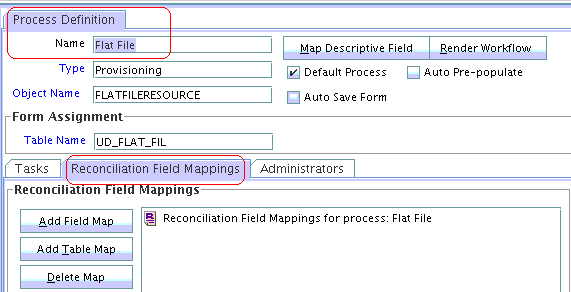
2.9.1.3 Reconciliation Mapping

**Purpose**

Create mapping required for reconciliation

**Steps**

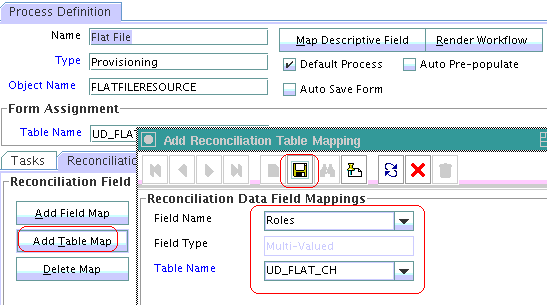
* Open **Process Definition** under **Process Management**
* Search for **Flat File** and click **Reconciliation Field Mappings**



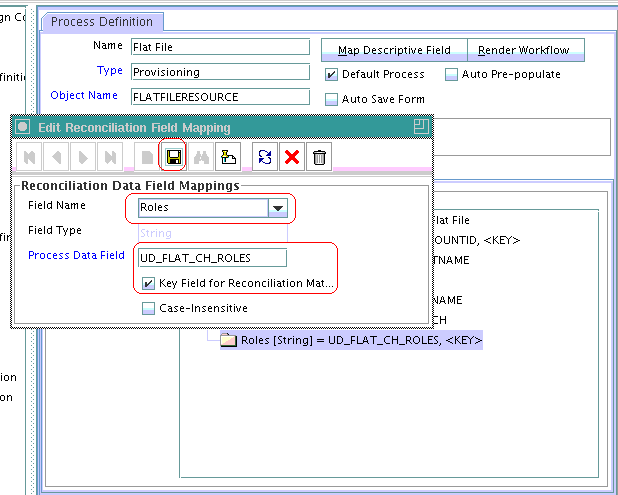
* Add **Field Mapping** for **AccountId, FirstName, lastName** and **email** as shown below.



* 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**

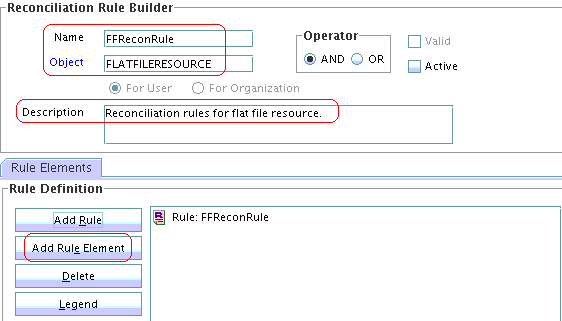
2.9.1.4 Reconciliation Rule

**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,

2.9.1.5 Create Scheduled task and update MDS

**Purpose**

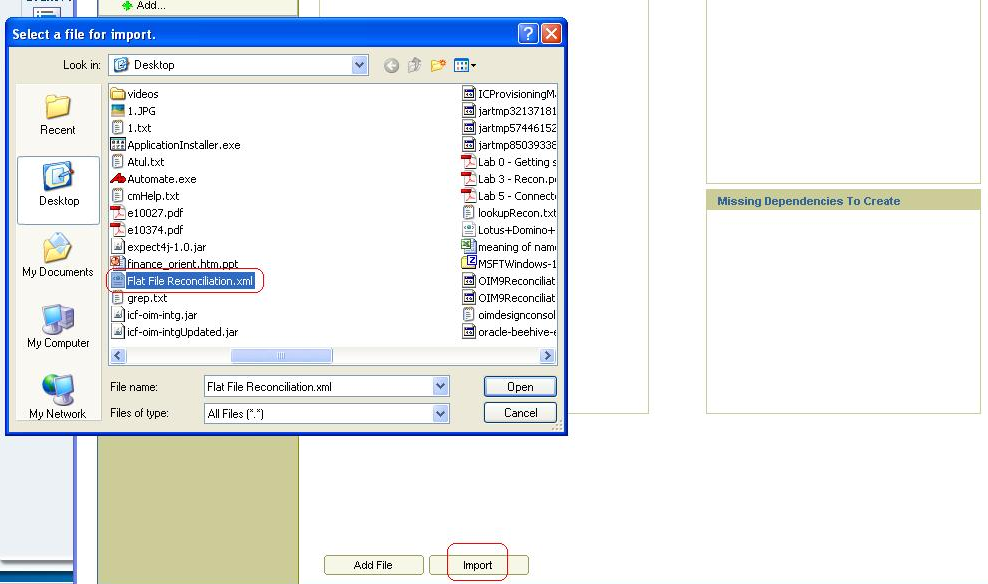
This section includes steps to create a scheduled task used by reconciliation.

**Steps**

* Login into OIM Admin console, click on **Import Deployment Manager File**



* Select **Flat File Reconciliation** and click import



* Complete the wizard.

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 PurgeCacahe utility.

2.10 Test reconciliation

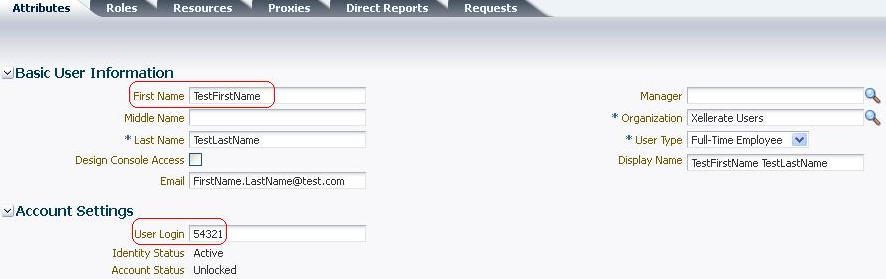
2.10.1 Full/Incremental reconciliation

**Purpose**

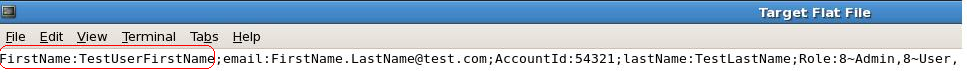
Test Full/Incremental reconciliation

**Steps**

* For testing we assume the following user has been provisioned and corresponding **FirstName** has been updated to **TestUserFirstName**

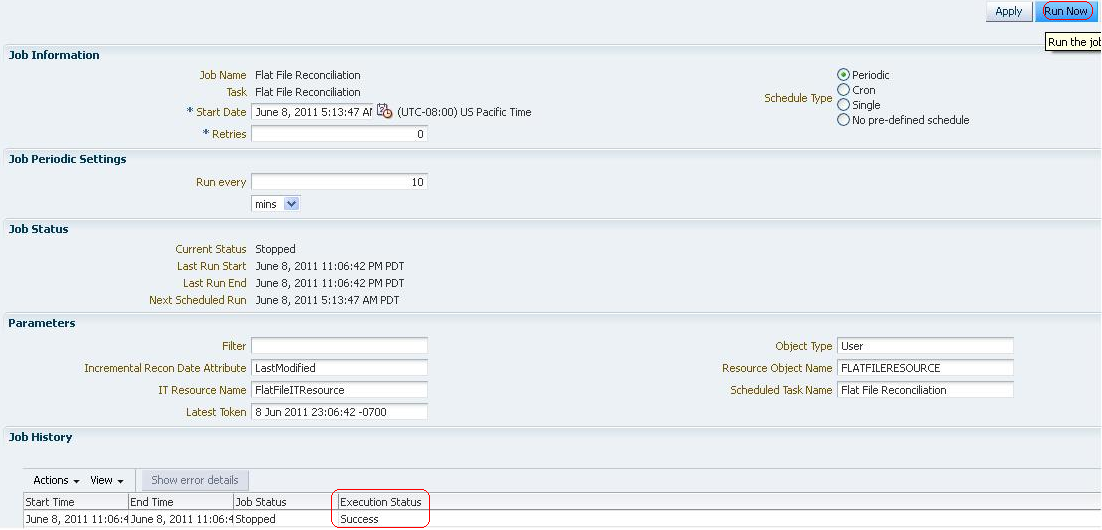
****

* Target flat file looks like below



**Note: for testing purpose, we have directly updated the FirstName to TestUserFirstName**

* Search for the scheduled job **Flat File Reconciliation** and click run to success message.

****

* Verify the reconciliation events being generated
* Click **Event Management** and search for **Reconciliation** to see new events being generated.



2.12 Extending the functionality of connector

2.12.1 Provisioning validation

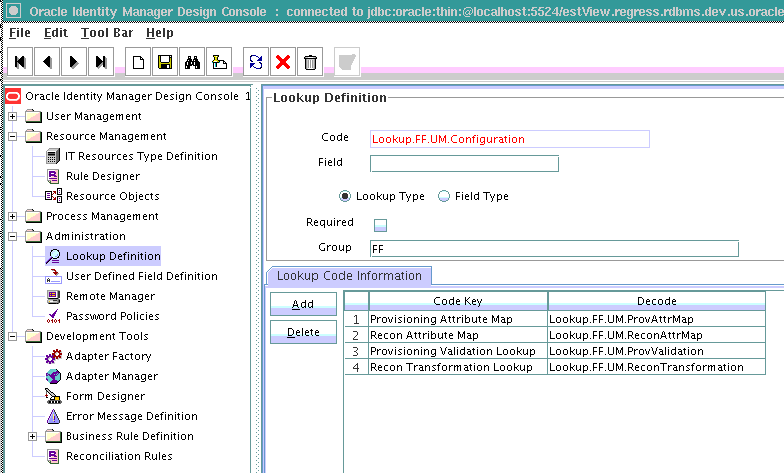
**Purpose**

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

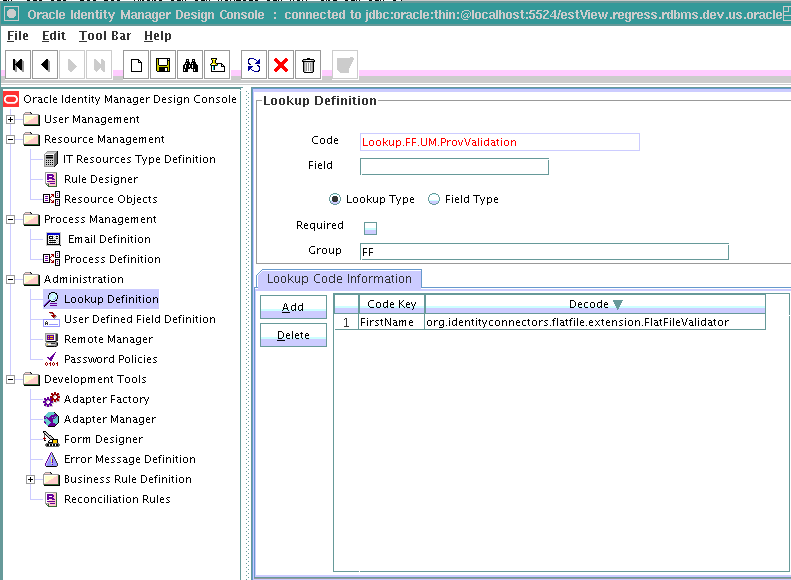
**Steps**

* 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**
* 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.

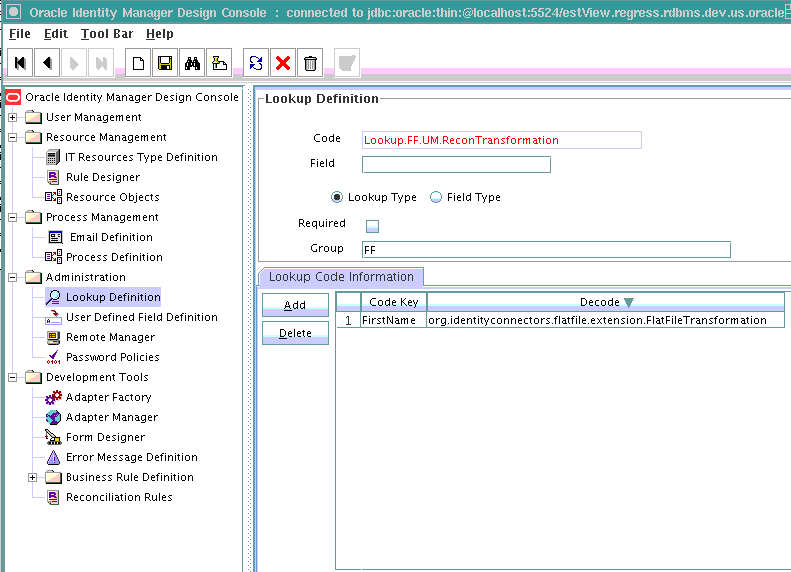
Object configuration lookup updated with validation and transformation lookup.



Provisioning Validation Lookup



Recon Transformation Lookup



* Update the OIM database with the **FlatFileValidationAndTransformation.jar** as created in lab1 using UpdateJars.sh script.
* PurgeCache
* Perform provisioning and recon operation in OIM to see this logic being called.

2.12.3 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 MANIFEST file appropriately. It should look like below

ConnectorBundle-FrameworkVersion: 1.0

ConnectorBundle-Name: org.identityconnectors.flatfile

***ConnectorBundle-Version: 1.1***

* 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 both the connector versions 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 advanced administration UI.
* 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.

This creates a connector install package.