

iOpenMPDM 2.0 / Mural 2.0

Creating a Master Person Index Application

iOpenMPDM 2.0/Mural 2.0 Hand-On Tutorial Series

Authors: Charles Ye
Date: March 07, 2012

Creating a Master Person Index Application

1. Overview

The goal of a master person index application is to consolidate disparate sources into a centralized master person index database and produce a single version of the truth for master person information. This tutorial is designed to guide you to creating and running a master person index application using iOpen MPDM platform. There are two distinct perspectives that include the design environment and runtime environment as illustrated in Figure 1.

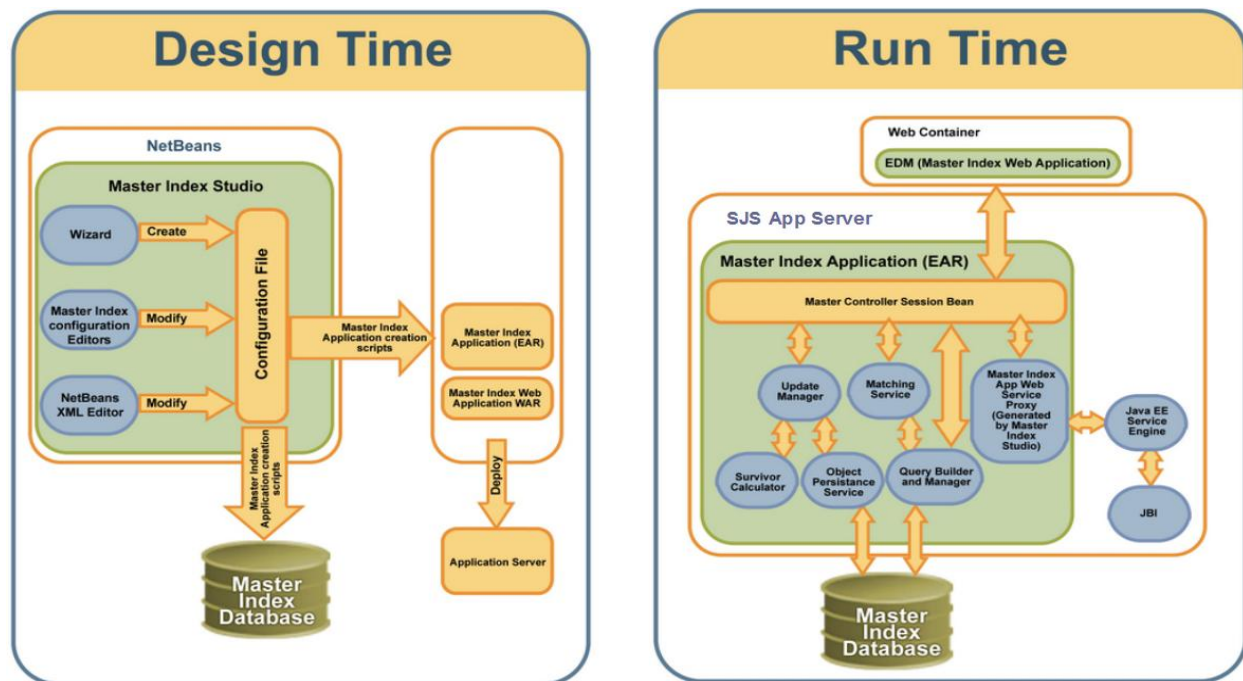


Figure 1 Design and Runtime Environments

The design environment is used for creating and configuring the master person index application. The runtime environment is used for deploying and running the master person index application. The tutorial describes building and testing procedures for a basic master person index application.

1. Install Mural 2.0 MDM NetBeans Plug-in
2. Create Master Person Index Project
3. Configure Matching Engine
4. Configure Standardization Engine
5. Define Search Block
6. Specify Filters

7. Build Master Person Index Application
8. Configure and Create Master Person Index Database
9. Configure Application Server
10. Deploy Master Person Index Application
11. Launch Web-based Master Index Data Manager (MIDM)
12. Test the Master Person Index Application

1.1 Required Components

For a complete set of supported components, please refer to <http://java.net/projects/mosaic>.

To exercise this tutorial, the following components are required:

1. Windows XP or Windows 7
Running on the following system environment:
RAM: 2 GB or above
FREE DISK: 8 GB or above
CPU: 1.5 GHZ +
2. NetBeans 7.0.1
You can download it from <http://netbeans.org/downloads>
3. GlassFish 3.1.1
You can download it from <http://glassfish.java.net/public/downloadsindex.html>
4. JDK 6 Update 29 or above (32 bit or 64bit)
You can download it from <http://www.oracle.com/technetwork/java/javase/downloads>
5. MySQL Server 5.1 or above
You can download it from <http://dev.mysql.com/downloads/mysql>
6. MySQL Connector/J 5.1.12 and over
You can download it from <http://dev.mysql.com/downloads/connector/j>
7. Mozilla Firefox 2.0 or above
You can download it from <http://www.mozilla.org>
8. Mural 2.0 MDM NetBeans Plug-in Module
You can download it from <http://java.net/projects/mosaic/downloads>

1.2 Prerequisites

You have to successfully install the required components on your computer before you are ready for exercising this tutorial:

1. NetBeans 7.0.1
2. GlassFish 3.0.1
3. JDK 6 Update 29 or above (32 bit or 64bit)

4. MySQL Server 5.1 or above
5. Mozilla Firefox 2.0 or above

1.3 References

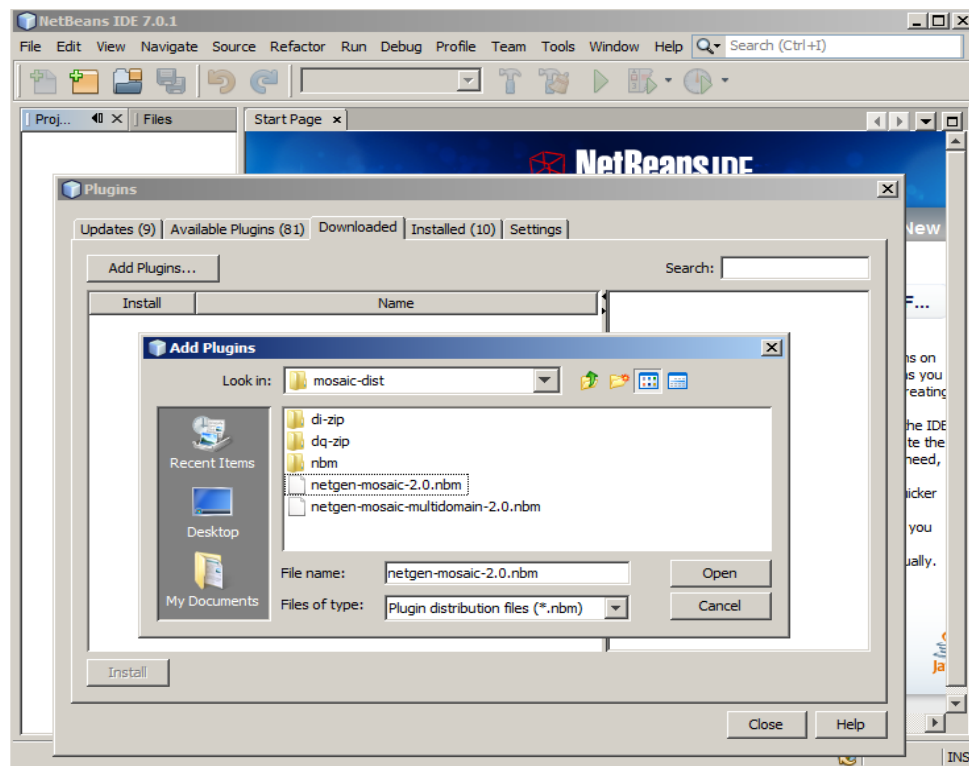
Mural 2.0 MDM Open Source Community

<http://java.net/projects/mosaic>

2. Install Mural 2.0 MDM NetBeans Plug-in

At this step you already have Mural 2.0 MDM NetBeans Plug-in mural-2.0.nbm file. Next is to install Mural 2.0 MDM NetBeans Plug-in module in the NetBeans 7.0.1 IDE. The following steps show how it is done. If a version of Mural 2.0 MDM NetBeans Plug-in already exists, it must be uninstalled by following the NetBeans instructions.

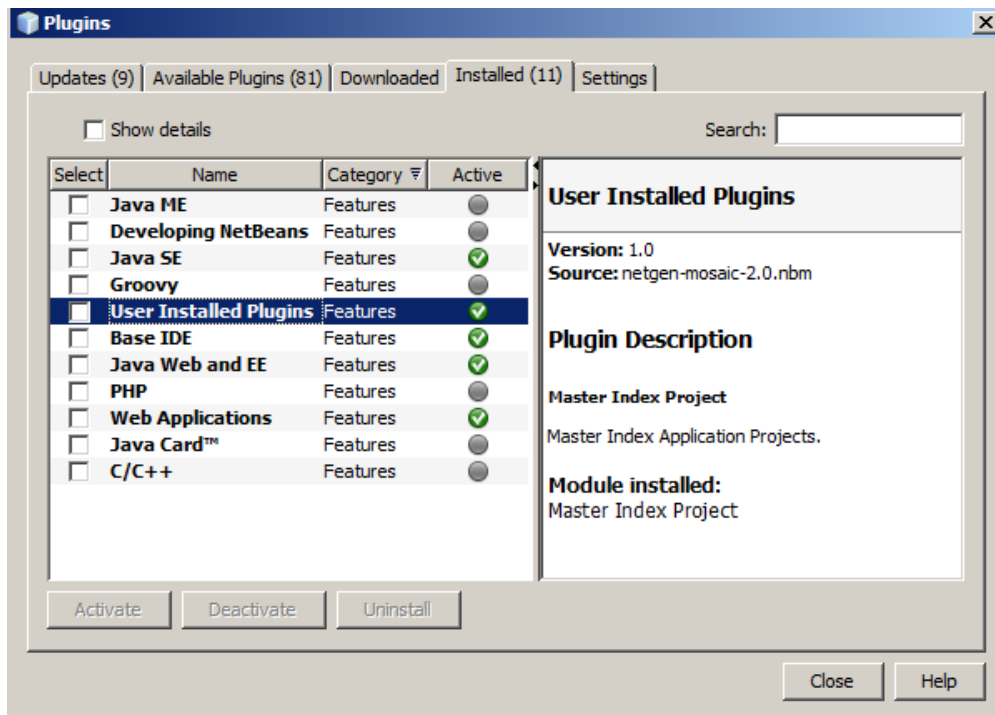
1. Start NetBeans 7.0.1.
2. Choose **Tools** on the menu bar, then choose **Plugins** menu option.
3. Choose **Downloaded** on the **Plugins** menu bar, then choose “**Add Plugins ...**” button.
4. Navigate to the directory where you downloaded mural-2.0.nbm file, then click **OK** button.



5. Click **Install** button, then click **Next** button on the pop-up window.

6. Accept the license agreement, then click **Install** button to install the Plug-in.
7. Click **Finish** button once the installation is completed.
8. Close **Plugins** window.

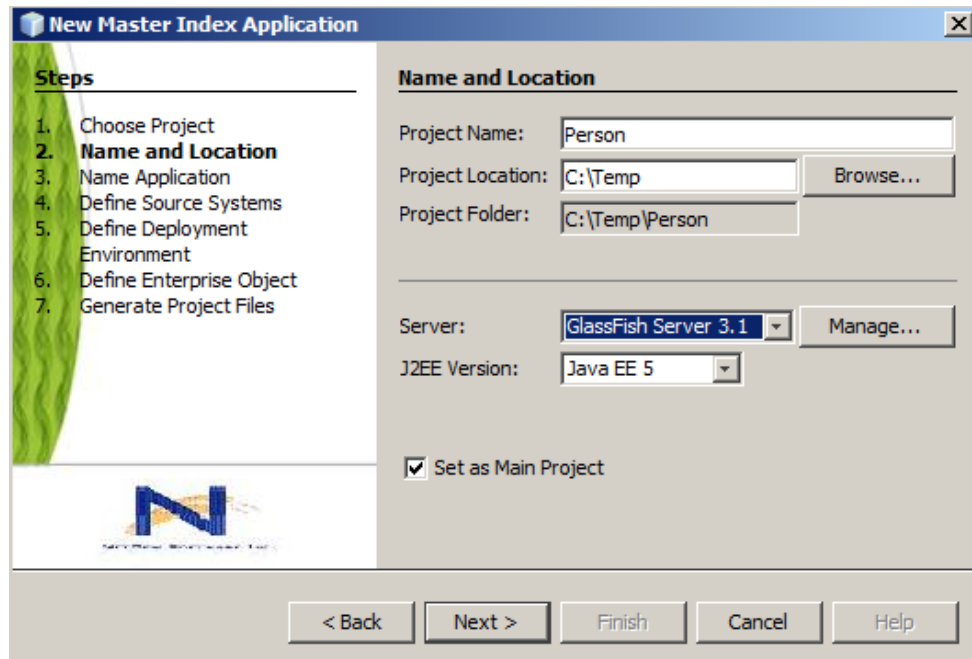
Now Mural 2.0 MDM NetBeans Plug-in shall be visible in Plugins dialog window. You are ready to start building a master person index application.



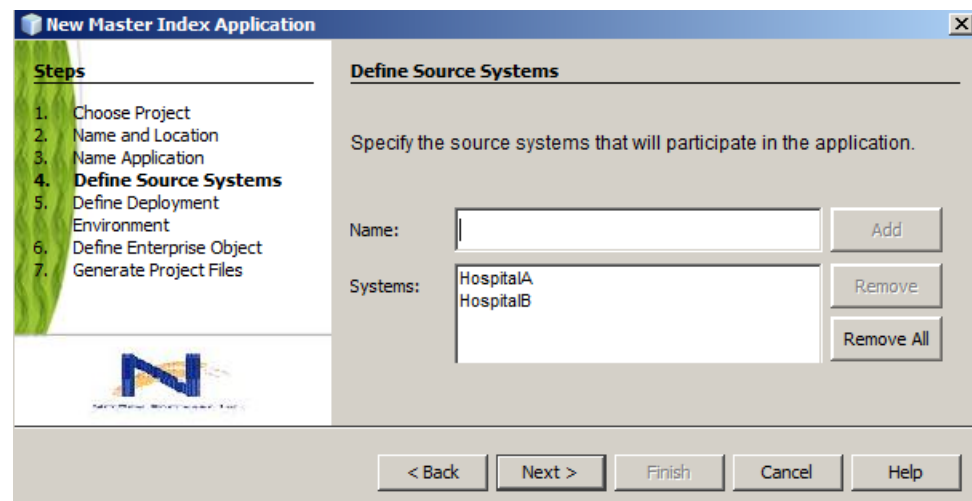
3. Create Master Person Index Project

The following steps will create a new generic-purpose Person master person index application.

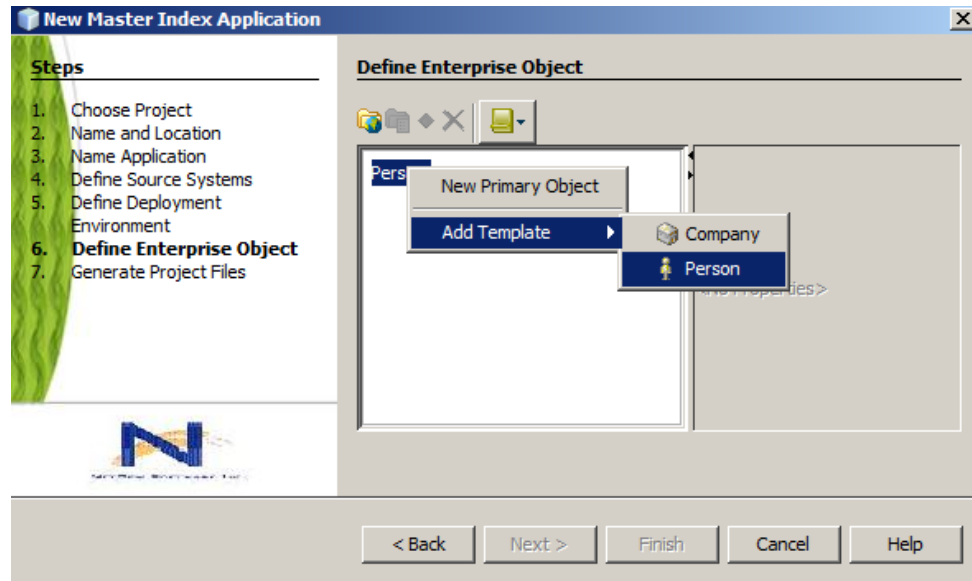
1. Start NetBeans 7.0.1 with Mural 2.0 MDM NetBeans Plug-in installed.
2. On the menu bar choose **File > New Project > MDM > Master Index Application**, click **Next** button.
3. Create a new project with the name of Person, choose **Server** type to GlassFish 3.1, and **J2EE Version** to Java EE 5. If GlassFish Server 3.1 is not listed in the pull-down menu, click **Manage..** button to add in GlassFish Server 3.1 following NetBeans instructions.



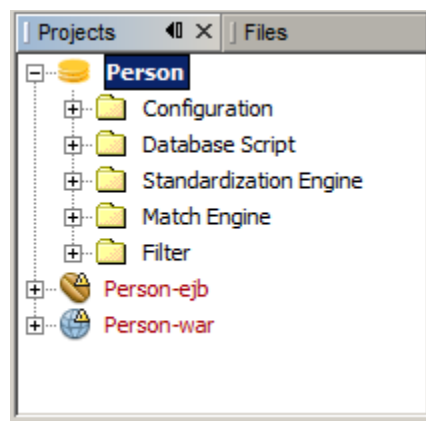
4. Click Next Button; Name Application as Person; Click Next button.
5. Specify HospitalA and HospitalB source systems that participate in the application; Click Next button.



6. Choose MySQL **Database** type; and keep the rest by default; then click **Next** Button.
7. Next step to create data object model using enterprise object definition wizard. Right click on Person; expand **Add Template** menu; chose **Person template**. You can expand Person object hierarchy to review the attributes and child objects of the Person template object model.



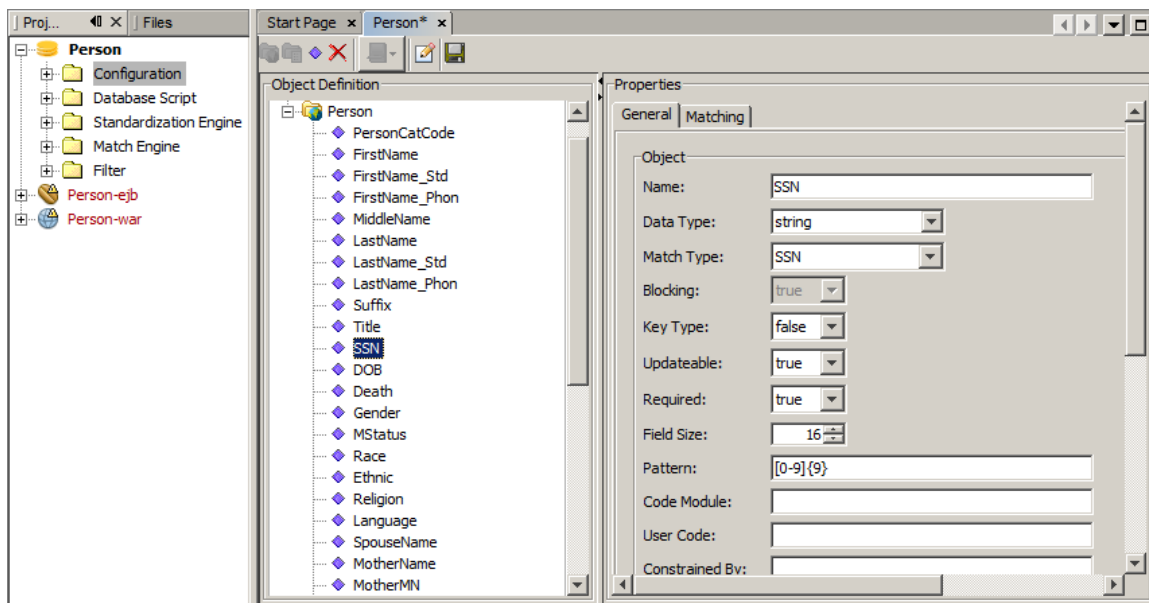
8. Click **Next** button and check “**Generate the remaining Master Index application files now**”; then click **Finish** button. The project wizard automatically generates all the project artifacts.
9. Now in the Projects panel Person Java EE enterprise application is generated, it contains lots of configuration files in different folders, an EJB module and a web module.



4. Configure Matching Engine

The Mural 2.0 match engine is highly configurable. You need to determine what Person attributes be used for matching records. In this tutorial, we use First Name, Last Name, SSN, DOB and Gender for Person record match.

1. In the Projects panel, right click on **Configuration** and select **Edit** to have Person object model detail.
2. Click on **SSN** in the **Person** object definition tree and set its **Match Type** property to **SSN**.

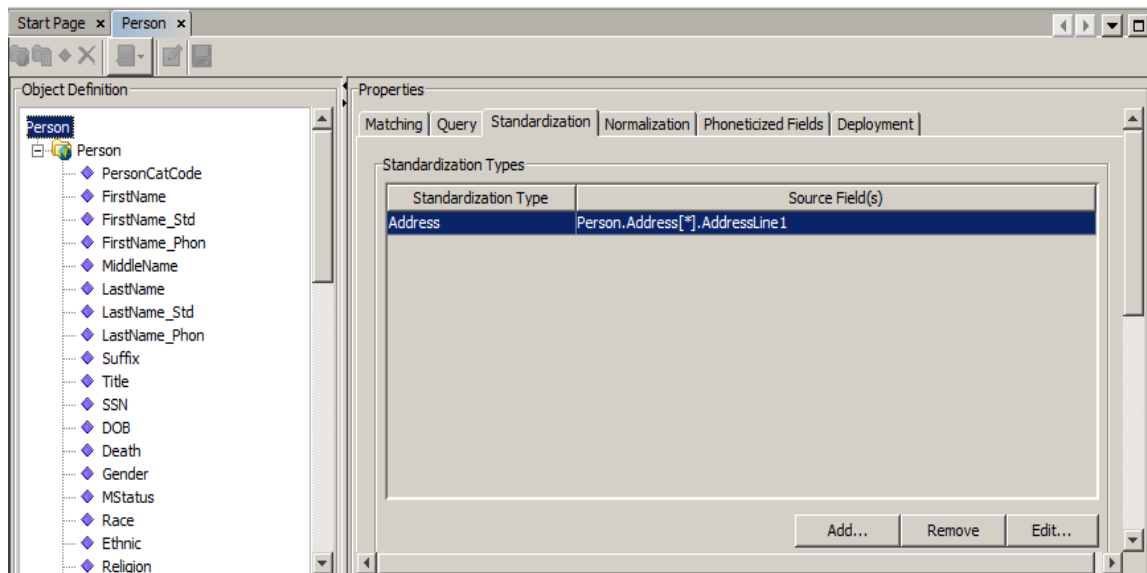


3. Select **DOB** and set its **Match Type** property to **DOB**.
4. Select **Gender** and set its **Match Type** property to **Gender**.
5. Expand **Address** child object and select **AddressLine1_StDir**; then set its **Match Type** property to **None**.
6. Expand **Address** child object and select **AddressLine1_StName**; then set its **Match Type** property to **None**.
7. Expand **Address** child object and select **AddressLine1_HouseNo**; then set its **Match Type** property to **None**.
8. Expand **Address** child object and select **AddressLine1_StType**; then set its **Match Type** property to **None**.
9. Save your changes.

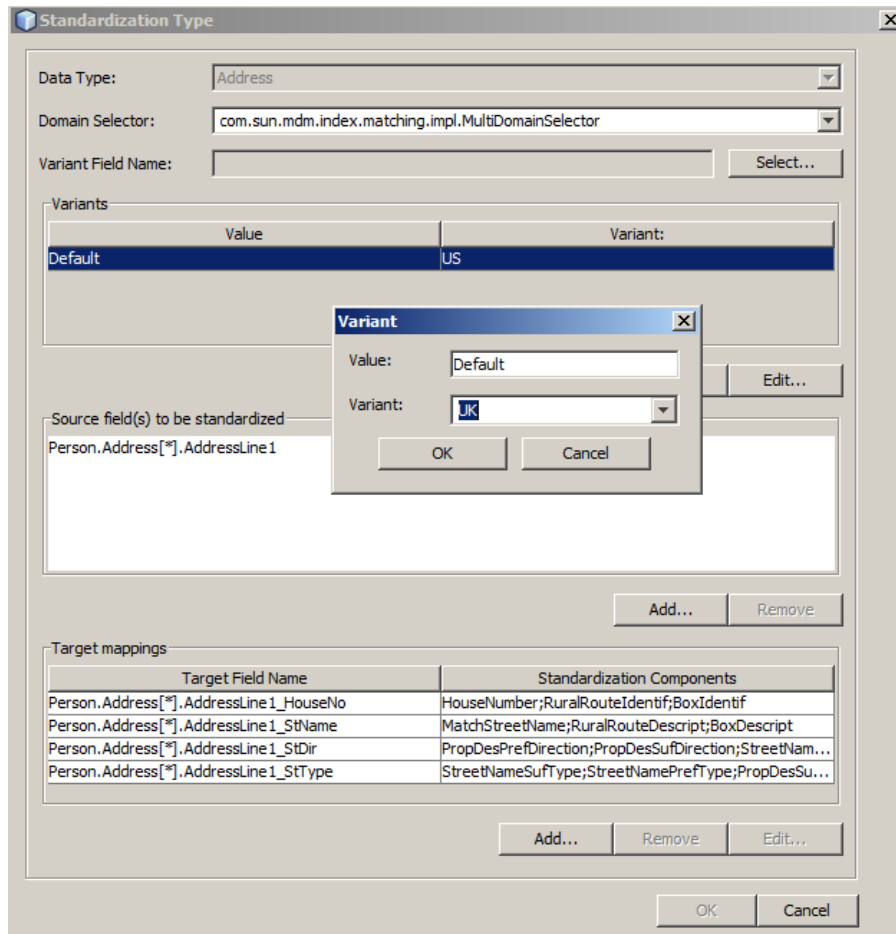
5. Configure Standardization Engine

The Mural 2.0 standardization engine is highly configurable to parse, normalize and phonetically encode attributes for improving matching accuracy and efficiency. The engine supports different languages and variants. The next step is to specify UK address variant to standardize Person addresses.

1. In the Projects panel, right click on **Configuration** and select **Edit** to have Person object model detail if the Person object model detail view is not displayed.
2. Click on the top **Person** node in the Object Definition tree and find the **Standardization** tab on the right panel.
3. Click on **Address** in Standardization Type panel and select **Edit** button to configure Address standardization parameters.



4. In the Standardization Type window, click on Default on Variants panel, then click on Edit button, choose UK variant.

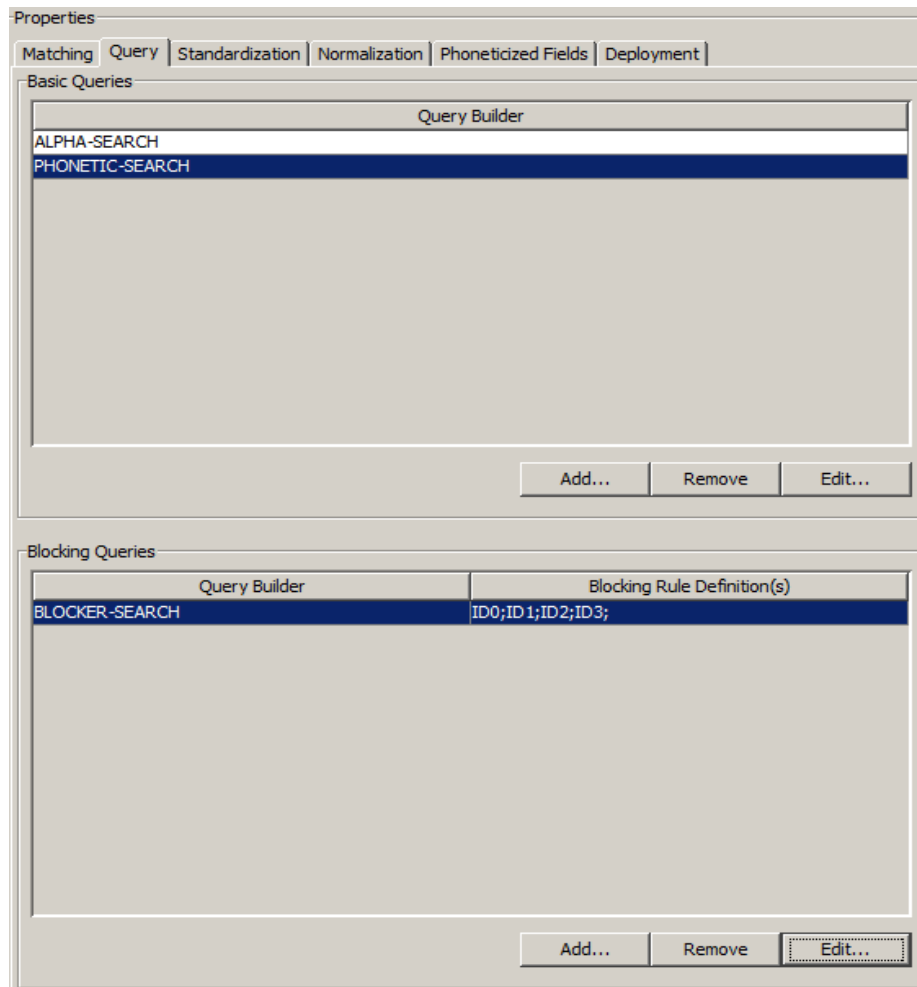


5. Click OK button and save your changes.

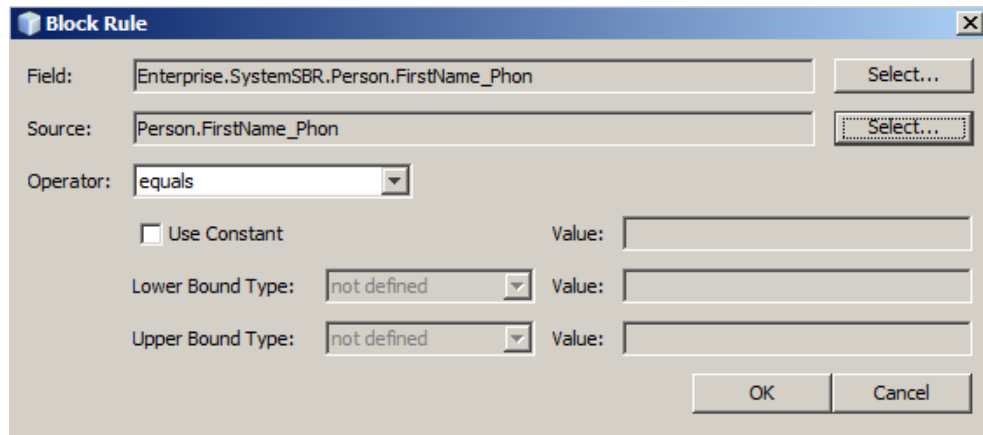
6. Define Search Block

Blocking procedure defines query criteria that retrieve a small set of possible candidates for matching an incoming record. In this tutorial, we define a new block combining with first name, DOB and gender.

1. In the Projects panel, right click on **Configuration** and select **Edit** to have Person object model detail.
2. Click on the top **Person** node in the Object Definition tree and find the **Query** tab on the right panel.
3. In the **Query** tab, scroll down to find the **Blocking Queries** panel. Click on the **BLOCKER-SEARCH** row and then click on **Edit** button.



4. In the **Blocking Query Builder** window, choose ID3 row, click **Remove** button to Address block.
5. In the **Blocking Query Builder** window, click **Add** button to add a new block definition.
6. In the Block Definition window, enter FN_DOB_GENDER for **Block Name** and select **Add** to define block rules for FN_DOB_GENDER block.
7. In the Block Rule pop-up window, click the **Select** button for Field to choose the FirstName_Phon field in the Person object model; click the **Select** button for Source select FirstName_Phon field in the Person object model.

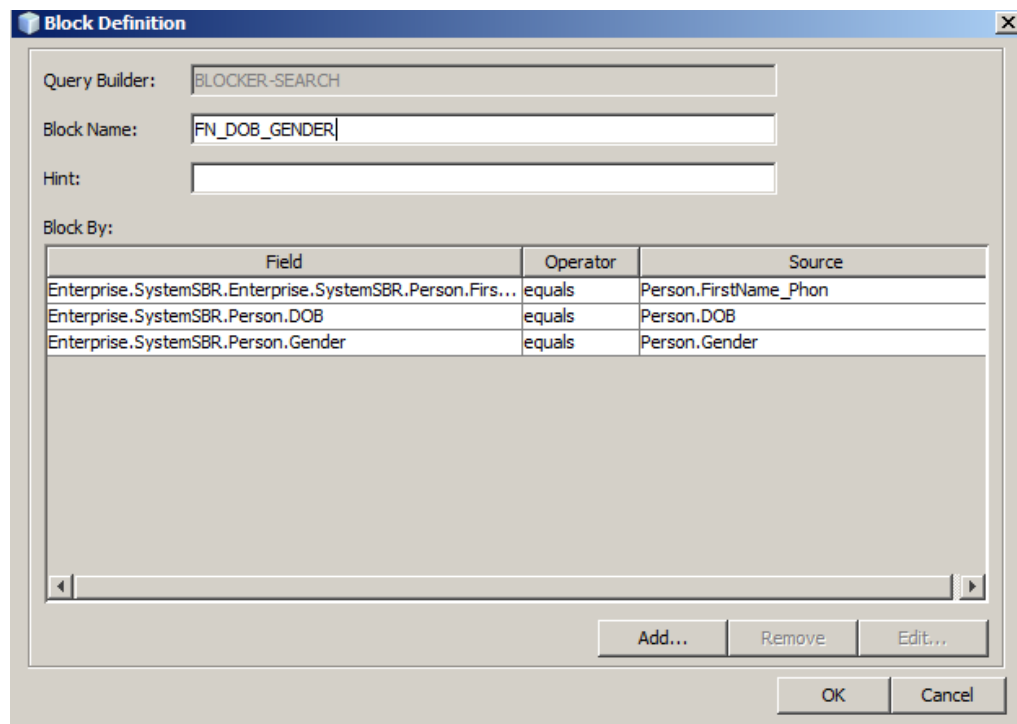


The Block Rule dialog box is shown with the following fields and values:

- Field: Enterprise.SystemSBR.Person.FirstName_Phon
- Source: Person.FirstName_Phon
- Operator: equals
- Use Constant: ☐
- Value: (empty)
- Lower Bound Type: not defined
- Value: (empty)
- Upper Bound Type: not defined
- Value: (empty)

Buttons: OK, Cancel

- In the **Block Definition** window, click **Add button** to repeat step 7 to add DOB and Gender for FN_DOB_GENDER block. Then click **OK** button to complete FN_DOB_GENDER block definition.



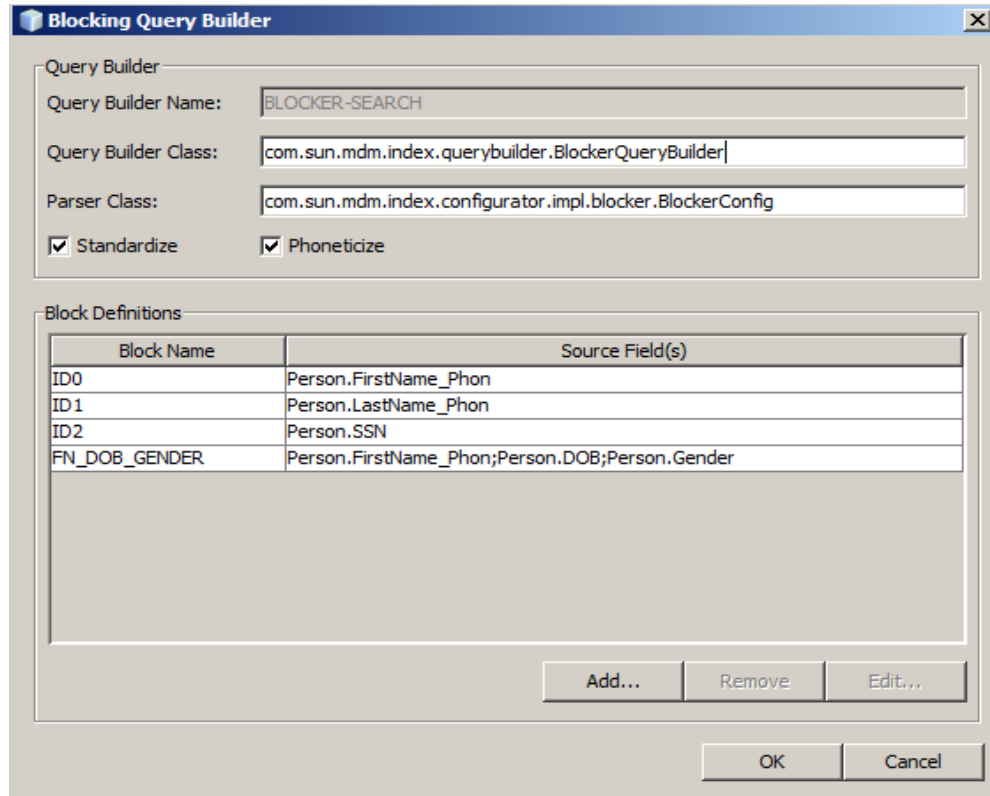
The Block Definition dialog box is shown with the following fields and values:

- Query Builder: BLOCKER-SEARCH
- Block Name: FN_DOB_GENDER
- Hint: (empty)
- Block By:

Field	Operator	Source
Enterprise.SystemSBR.Enterprise.SystemSBR.Person.Firs...	equals	Person.FirstName_Phon
Enterprise.SystemSBR.Person.DOB	equals	Person.DOB
Enterprise.SystemSBR.Person.Gender	equals	Person.Gender

Buttons: Add..., Remove, Edit..., OK, Cancel

- Click **OK** button on the **Block definition** window to complete BLOCKER-SEARCH definition. You have 4 blocks: ID0, ID1, ID2 and FN_DOB_GENDER.



The image shows a 'Blocking Query Builder' dialog box. It has a title bar with a close button. The main area is divided into two sections. The top section, 'Query Builder', contains four text fields: 'Query Builder Name' (BLOCKER-SEARCH), 'Query Builder Class' (com.sun.mdm.index.querybuilder.BlockerQueryBuilder), 'Parser Class' (com.sun.mdm.index.configurator.impl.blocker.BlockerConfig), and two checked checkboxes, 'Standardize' and 'Phoneticize'. The bottom section, 'Block Definitions', contains a table with two columns: 'Block Name' and 'Source Field(s)'. The table has four rows: ID0 (Person.FirstName_Phon), ID1 (Person.LastName_Phon), ID2 (Person.SSN), and FN_DOB_GENDER (Person.FirstName_Phon;Person.DOB;Person.Gender). Below the table are three buttons: 'Add...', 'Remove', and 'Edit...'. At the bottom right are 'OK' and 'Cancel' buttons.

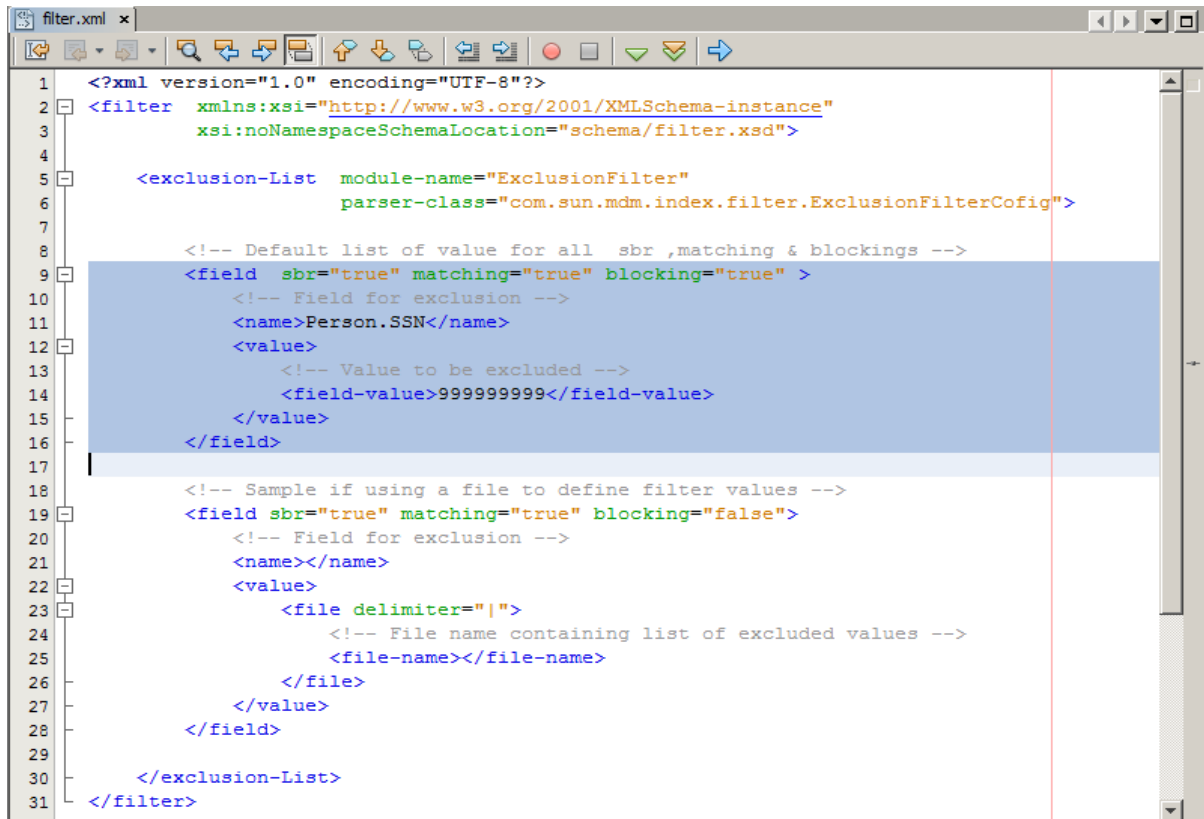
Block Name	Source Field(s)
ID0	Person.FirstName_Phon
ID1	Person.LastName_Phon
ID2	Person.SSN
FN_DOB_GENDER	Person.FirstName_Phon;Person.DOB;Person.Gender

10. Click **OK** button to finish Query Builder definition.

7. Specify Filters

The filter is used for excluding unwanted values during processing. The incoming records often contain default values when the actual values are unknown. For an example, “999-99-9999” or “000-00-000” is used for a social security number. “Baby” or “Girl” is used for the name of a newborn. In this tutorial, we specify a filter that ignores records with SSN value of 999999999.

1. In the Projects panel, expand Filter folder under Person project tree.
2. Double click on filter.xml to open the filter configuration file in the right side XML editor.
3. Edit the highlighted below to exclude Person.SSN value 999999999.



4. Save your changes. And close filter.xml.

8. Build Master Person Index Application

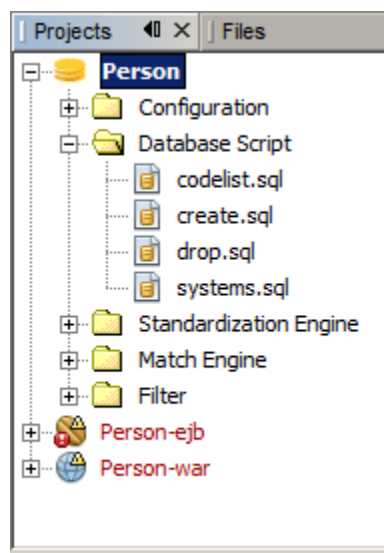
At this step we completed configuring master person index application. Now we are ready to generate the Person Java EE enterprise application archive file (EAR file) that can be deployed to the application server.

1. Save all your changes.
2. Right click on **Person** node under Person project tree; Choose **Build** in the pop-down menu.
3. Person.ear will be automatically created under the **dist** folder of Person project directory. If build is successful, you shall see a message like BUILD SUCCESSFUL (total time: 1 minute 45 seconds) in the Output window.

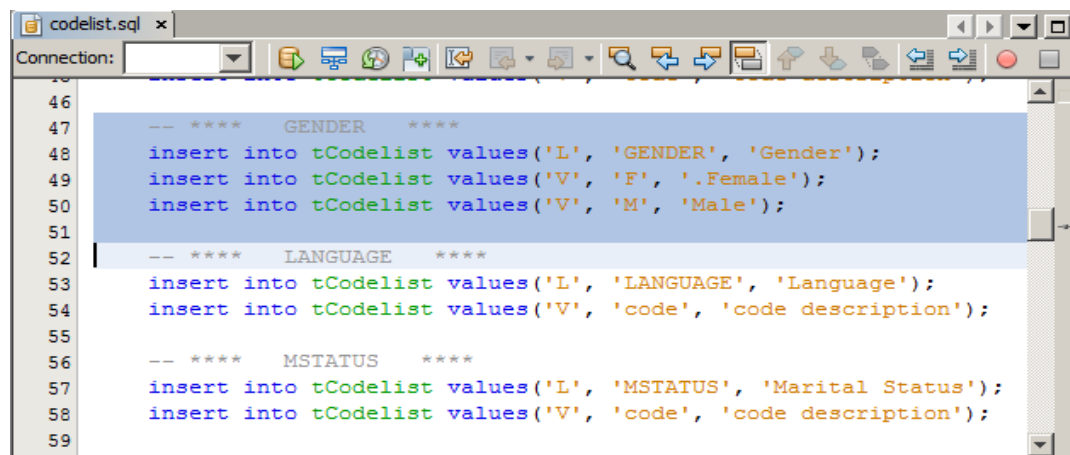
9. Configure and Create Master Person Index Database

The Mural 2.0 project wizard automatically creates a collection of database scripts when the Person master index application is built. In this tutorial, we will edit the database scripts before running these scripts.

1. In the Project panel, expand Person project and Database script folder. It shows a list of database scripts.

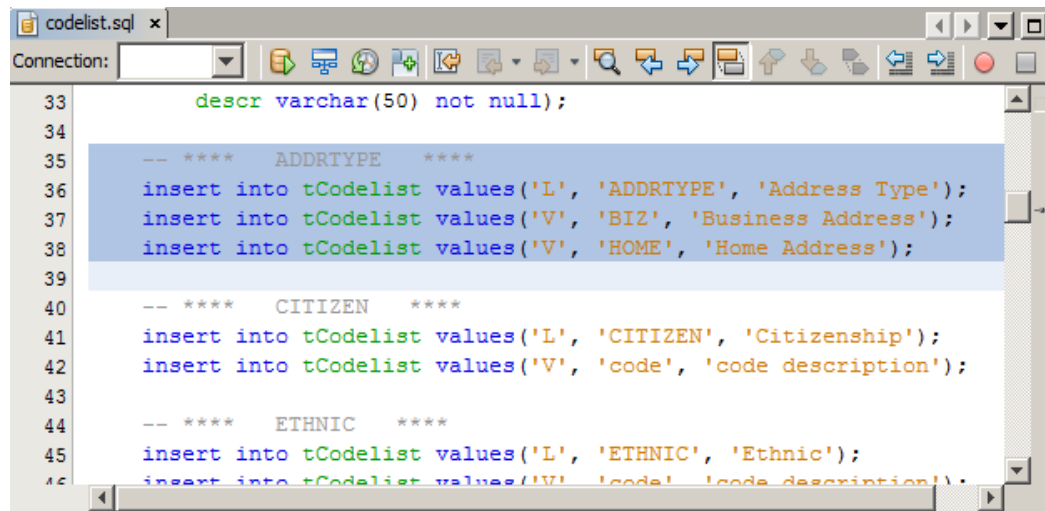


2. Double click on codelist.sql to open it in the text editor on the right panel.
3. Edit the GENDER section, add "F" and "M" entries for GENDER.

A screenshot of a text editor window titled 'codelist.sql'. The editor shows SQL code for creating a code list. The code is organized into sections: GENDER, LANGUAGE, and MSTATUS. The GENDER section is highlighted in blue. The code includes insert statements for 'L' (Language), 'V' (Value), and 'code' (code description) for each section. The GENDER section includes entries for 'F' (Female) and 'M' (Male).

```
46
47  -- **** GENDER ****
48  insert into tCodelist values('L', 'GENDER', 'Gender');
49  insert into tCodelist values('V', 'F', '.Female');
50  insert into tCodelist values('V', 'M', 'Male');
51
52  -- **** LANGUAGE ****
53  insert into tCodelist values('L', 'LANGUAGE', 'Language');
54  insert into tCodelist values('V', 'code', 'code description');
55
56  -- **** MSTATUS ****
57  insert into tCodelist values('L', 'MSTATUS', 'Marital Status');
58  insert into tCodelist values('V', 'code', 'code description');
59
```

4. Edit the ADDRTYPE section, add “BIZ” and “HOME” entries for ADDRTYPE.

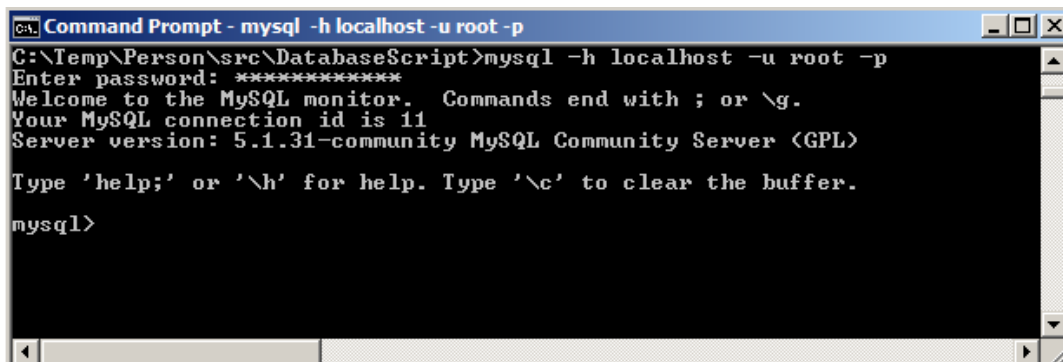


```

33      descr varchar(50) not null);
34
35      -- **** ADDRTYPE ****
36      insert into tCodelist values('L', 'ADDRTYPE', 'Address Type');
37      insert into tCodelist values('V', 'BIZ', 'Business Address');
38      insert into tCodelist values('V', 'HOME', 'Home Address');
39
40      -- **** CITIZEN ****
41      insert into tCodelist values('L', 'CITIZEN', 'Citizenship');
42      insert into tCodelist values('V', 'code', 'code description');
43
44      -- **** ETHNIC ****
45      insert into tCodelist values('L', 'ETHNIC', 'Ethnic');
46      insert into tCodelist values('V', 'code', 'code description');

```

5. Save all your changes and close codelist.sql window.
6. Ensure that your MySQL server installed and runs as a service.
7. Start a Command windows, change the current directory to the DatabaseScript folder of the Person project; then run mysql command as root user.



```

C:\Temp\Person\src\DatabaseScript>mysql -h localhost -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.1.31-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

```

8. Create a new database user person with password person by typing the command:
mysql>create user 'person' identified by 'person';
9. Grant all the privileges to the new user person by typing the command:
mysql>grant all privileges on *.* to person;
10. Create a new database person by typing the command:
mysql>create database person;
11. Quit and restart mysql command as person user; then switch to person database by typing the command:
mysql>use person;


```

C:\Temp\Person\src\DatabaseScript>mysql -h localhost -u person -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.1.31-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use person;
Database changed
mysql> _

```

12. Run database scripts by typing the command:

```

mysql>source create.sql;
mysql>source systems.sql;
mysql>source codelist.sql;

```

13. You can use any SQL console to connect to MySQL person database to review the master person index tables. We suggest you to choose Oracle SQL Developer or MySQL Administrator. The following picture lists all tables of the person schema by MySQL Administrator:

Table Name	Engine	R...	Data length	Index length	Update time
sbyn_address	InnoDB	0	16 kB	16 kB	
sbyn_addresssbr	InnoDB	0	16 kB	16 kB	
sbyn_alias	InnoDB	0	16 kB	16 kB	
sbyn_aliassbr	InnoDB	0	16 kB	16 kB	
sbyn_appl	InnoDB	1	16 kB	16 kB	
sbyn_assumedmatch	InnoDB	0	16 kB	32 kB	
sbyn_audit	InnoDB	0	16 kB	32 kB	
sbyn_common_detail	InnoDB	14	16 kB	16 kB	
sbyn_common_header	InnoDB	12	16 kB	32 kB	
sbyn_enterprise	InnoDB	0	16 kB	16 kB	
sbyn_merge	InnoDB	0	16 kB	48 kB	
sbyn_overwrite	InnoDB	0	16 kB	0 B	
sbyn_person	InnoDB	0	16 kB	16 kB	
sbyn_personsbr	InnoDB	0	16 kB	16 kB	
sbyn_phone	InnoDB	0	16 kB	16 kB	
sbyn_phonesbr	InnoDB	0	16 kB	16 kB	
sbyn_potentialduplicates	InnoDB	0	16 kB	48 kB	
sbyn_seq_table	InnoDB	17	16 kB	16 kB	
sbyn_systemobject	InnoDB	0	16 kB	0 B	
sbyn_systems	InnoDB	2	16 kB	0 B	
sbyn_systemsbr	InnoDB	0	16 kB	0 B	
sbyn_transaction	InnoDB	0	16 kB	80 kB	
sbyn_user_code	InnoDB	0	16 kB	0 B	

Num. of Tables: 23 Rows: 46 Data Len: 368 kB Index Len: 464 kB

Details >> Create Table Edit Table Maintenance Refresh

Now the master person index database is ready to be used by the master person index application.

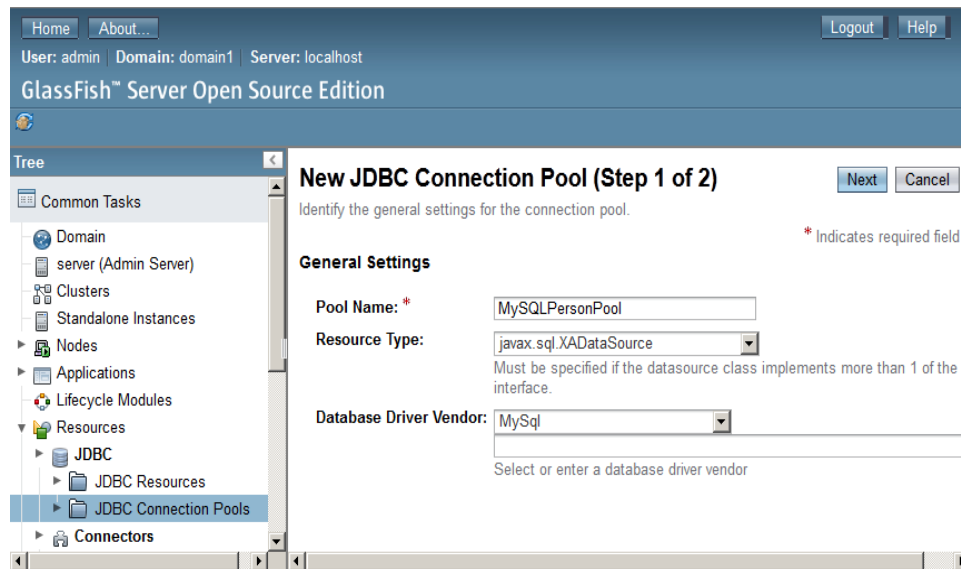
10. Configure Application Server

The application server needs to be configured before the master person index application is deployed. The configurations include:

- Install database driver
- Create connection pool
- Create JDBS resources
- Create JMS resource
- Setup user

In this tutorial, we use GlassFish 3.0.1 and MySQL 5.1.

1. Copy mysql-connector-java-5.1.12-bin.jar under glassfish/lib/endorsed folder.
2. Start GlassFish 3.1.1 application server by the command:
3. glassfish\bin\asadmin start-domain
4. Launch GlassFish Administration console using the URL: <http://localhost:4848>
5. Login with User Name admin and Password adminadmin
6. In the left-side panel, expand **Resources > JDBC > JDBC Connection Pools** and click **New** to create a new connection pool. Enter MySQLPersonPool for Pool Name; for Resource Type, choose javax.sql.XADataSource and for Database Vendor, select MySQL; click **Next** to continue the wizard.



7. In Additional Properties section, configure the following MySQLPersonPool properties
 - User: person
 - Password: person
 - URL: jdbc:mysql://localhost:3306/person

8. Leave the rest of properties default values. Click Finish button. Now MySQLPeronPool is created and listed in the Pools.
9. Click Ping button in MySQLPersonPool page, test the connection between your GlassFish and your MySQL database.

Edit JDBC Connection Pool Save Cancel

Modify an existing JDBC connection pool. A JDBC connection pool is a group of reusable connections for a particular database.

Load Defaults Flush Ping

* Indicates required field

General Settings

Pool Name: MySQLPersonPool

Resource Type: javax.sql.XADataSource
Must be specified if the datasource class implements more than 1 of the interface.

Datasource Classname: com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
Vendor-specific classname that implements the DataSource and/or XADataSource APIs

10. Now you need to create three JDBC resources. In the left-side panel, expand **Resources > JDBC > JDBC Resources** and click **New** to create a new JDBC resource. Specify jdbc/PersonDataSource for JNDI Name; Select MySQLPersonPool for Pool Name; Click OK button to complete. Repeat step 10 to create jdbc/PersonSequenceDataSource and jdbc/PersonReportDataSource.

New JDBC Resource OK Cancel

Specify a unique JNDI name that identifies the JDBC resource you want to create. The name must contain only alphanumeric, underscore, dash, or dot characters.

JNDI Name: * jdbc/PersonDataSource

Pool Name: MySQLPersonPool
Use the [JDBC Connection Pools](#) page to create new pools

Description:

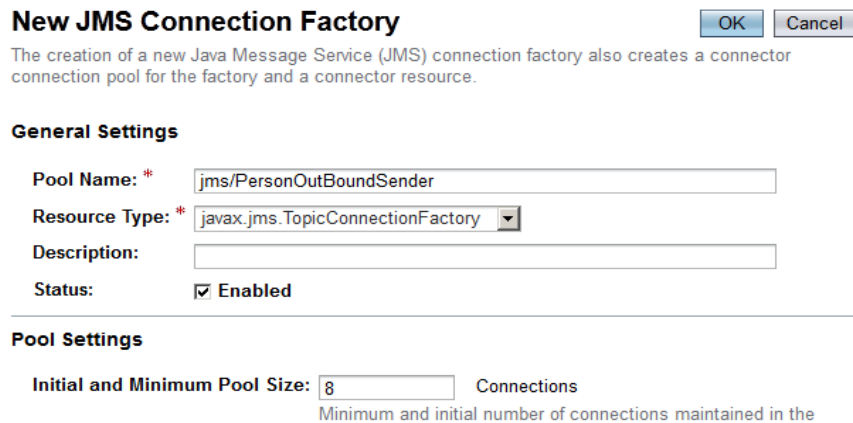
Status: ☒ Enabled

Additional Properties (0)

Add Property Delete Properties

Name	Value	Description:
No items found.		

11. In the left-side panel, expand **Resources > JMS Resources > Connection Factories** and click **New** to create a new JMS resource. Specify `jms/PersonOutBoundSender` for Pool Name; Select `javax.jms.TopicConnectionFactory` for Resource Type; Click OK button to complete.



New JMS Connection Factory OK Cancel

The creation of a new Java Message Service (JMS) connection factory also creates a connector connection pool for the factory and a connector resource.

General Settings

Pool Name: *

Resource Type: *

Description:

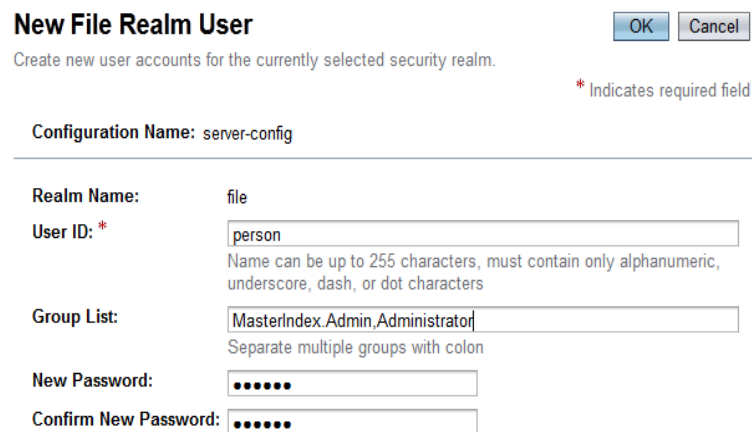
Status: ☒ Enabled

Pool Settings

Initial and Minimum Pool Size: Connections

Minimum and initial number of connections maintained in the

12. Now we need to setup security user. In the left-side panel, expand **Configurations > server-config > Security > Realms > file** and click New to add a new security user. Enter person for User ID; enter MasterIndex.Admin,Administrator for Group List; enter person for New Password; then click OK button for completion.



New File Realm User OK Cancel

Create new user accounts for the currently selected security realm.

* Indicates required field

Configuration Name: server-config

Realm Name: file

User ID: *

Name can be up to 255 characters, must contain only alphanumeric, underscore, dash, or dot characters

Group List:

Separate multiple groups with colon

New Password:

Confirm New Password:

Now the GlassFish is ready for the deployment of the master person index application.

11. Deploy Master Person Index Application

In section 8, you already generated the master person index application enterprise archive file `Person.ear`. Now you can deploy it to the GlassFish application server.

1. In the left-side panel, Select **Applications**; click **Deploy** button to deploy the application. Click **Browse** button to locate Person.ear; click **OK** button.

Deploy Applications or Modules

OK

Cancel

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

* Indicates required field

Location:

☒ Packaged File to Be Uploaded to the Server

Browse...

☐ Local Packaged File or Directory That Is Accessible from GlassFish Server

Browse Files...

Browse Folders...

Type: *

Enterprise Application

2. If deployed successfully, you will see the Person application listed.

Deployed Applications (1)				
	Name	Enabled	Engines	Action
<input type="checkbox"/>	Person	<input checked="" type="checkbox"/>	ear, ejb, webservices, web	Redeploy Reload

12. Launch the Master Index Data Manager (MIDM)

The master index data manager is web-based console for managing master index information. To access it,

1. Start Mozilla Firefox, enter the URL <http://localhost:8080/PersonMIDM>
2. Login with User Name person and Password person. The Person MIDM landing page is like:

The screenshot shows a Firefox browser window with the address bar displaying `localhost:5050/PersonMIDM/recorddetails.jsf`. The page title is "Master Index Web Application". The main content area features the "master index data manager" logo on the left and a "person Sign Out" link with a logo on the right. Below the logo, there is a navigation bar with buttons: "Dashboard", "Duplicate Records", "Record Details" (highlighted in blue), "Assumed Matches", "Transactions", and "Reports". Below this, there are two more buttons: "Source Record" and "Audit Log". The main search area is a yellow box containing a "Select the search Type:" dropdown menu with "Advanced Person Lookup (Phonetic)" selected. Below the dropdown, there are two sections: "Person" with fields for "First Name", "Last Name", and "SSN"; and "Person.Address" with fields for "Address Line1" and "City". At the bottom of the search area are "Search" and "Clear" buttons.

13. Test the Master Person Index Application

For testing, we enter three test records using Person MIDM web console.

1. Click **Source Record** tab; select Add tab to enter record.
2. Select HospitalA for **System**; Enter 000-000-0001 for **Local ID**; Click **Validate** button.
3. In the record entry form, enter
First Name: John
Last Name: Green
SSN: 111223333
DOB: 01/01/1960
Gender: Male
4. Scroll to the bottom of the page and click **Submit** button. You will see the following message like

The screenshot shows the 'Add' tab selected in the Person MIDM web console. The 'System' dropdown is set to 'HospitalA' and the 'Local ID' text box contains '000-000-0001'. A green message indicates: 'New enterprise object created with EUID: 0000000000'. Below the input fields are 'Validate' and 'Clear' buttons.

5. Enter another record. Select HospitalB for **System**; Enter 000-000-0001 for **Local ID**; Click **Validate** button.
6. In the record entry form, enter
First Name: Johnny (use nickname rather than John)
Last Name: Grene (different spelling)
SSN: 111232333 (transposition of two digits)
DOB: 01/10/1960 (transposition of two digits)
Gender: Male
7. Scroll to the bottom of the page and click **Submit** button. You will see the following message like

The screenshot shows the 'Add' tab selected in the Person MIDM web console. The 'System' dropdown is set to 'HospitalB' and the 'Local ID' text box contains '000-000-0001'. A green message indicates: 'Assumed match is found in EUID 0000000000'. Below the input fields are 'Validate' and 'Clear' buttons.

Based on matching configuration, the system determined that these two records are matched for the same person.

8. Enter another record. Select HospitalB for **System**; Enter 000-000-0002 for **Local ID**; Click **Validate** button.
9. In the record entry form, enter
First Name: Johnny (use nickname rather than John)
Last Name: Greene (different spelling)
SSN: 1112233555 (different from previous ones)
DOB: 01/12/1960 (different from previous ones)
Gender: Male
10. Scroll to the bottom of the page and click **Submit** button. You will see the following message like

The screenshot shows a web form with tabs 'View/Edit', 'Add', and 'Merge'. The 'Add' tab is active. It contains a 'System' dropdown menu set to 'HospitalB' and a 'Local ID' text box containing '000-000-0002'. Below these are 'Validate' and 'Clear' buttons. A green message bubble on the right states: 'New enterprise object created with EUID: 0000000001, which resulted in 1 potential duplicates'.

The system determines that this is a potential duplicate from previous two records you entered.

11. Click **Assumed Matches** tab and enter 0000000000 for EUID to display the matches.

The screenshot shows a search interface with fields for 'EUID' (set to '0000000000'), 'System', 'Local ID', 'Create Date From', 'To Create Date', 'Create Time From', and 'To Create Time'. There are 'Search' and 'Clear' buttons. Below the search fields is a table with the following data:

ID	EUID	Weight	System	Local ID	Create User	Create Date	First Name	Middle Name	Last Name
00000000000000000000000000000000	0000000000	30.981482	HospitalB	0000000001	person	03/08/2012	Johnny		

12. Click on EUID 0000000000 to show detail of the matches.

Assumed Matches Details				Back
Person Details	Main EUID	HospitalID	Add 03/08/2012	Update 03/08/2012
EUID	0000000000	0000000001	0000000000	0000000000
Status	Active	Active	Active	Active
Person Cat Code				
First Name	Johnny	Johnny	John	Johnny
Middle Name				
Last Name	Greene	Greene	Green	Greene
Suffix				
Title				
SSN	111-23-2333	111-23-2333	111-22-3333	111-23-2333
Date of Birth	01/10/1960	01/10/1960	01/01/1960	01/10/1960
Death				
Gender	Male	Male	Male	Male